8

4

¥

[原创]某企鹅xxx-base分析

杰克王 6

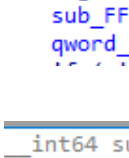
极客

6天前

2404

系统回调部分

系统回调注册：



快乐周末开始了

```
__int64 __fastcall g_InitRegisterProcessAndThread(__int64 (*a1)(void))
{
  NTSTATUS v2; // edi
  void *v3; // rax
  _UNICODE_STRING DestinationString; // [rsp+20h] [rbp-29h] BYREF
  struct _OB_CALLBACK_REGISTRATION CallbackRegistration; // [rsp+30h] [rbp-19h] BYREF
  OB_OPERATION_REGISTRATION v7[2]; // [rsp+60h] [rbp+17h] BYREF

  if ( !byte_FFFFF80737848550 )
  {
    dword_FFFFF80737848604 = 327808;
    memset(&unk_FFFFF80737848558, 0, 0x20ui64);
    memset(&unk_FFFFF80737848578, 0, 0x20ui64);
    memset(&unk_FFFFF80737848598, 0, 0x20ui64);
    memset(&unk_FFFFF807378485B8, 0, 0x20ui64);
    v7[0].ObjectType = PsProcessType;
    v7[0].Operations = 3;
    v7[0].PreOperation = g_PreProcess;
    v7[0].PostOperation = g_PostProcess;
    v7[1].ObjectType = PsThreadType;
    v7[1].PreOperation = g_PreThread;
    v7[1].Operations = 3;
    v7[1].PostOperation = g_PostThread;
    RtlInitUnicodeString(&DestinationString, L"366666");
    CallbackRegistration.RegistrationContext = 0i64;
    *(_DWORD *)&CallbackRegistration.Version = 131328;
    CallbackRegistration.OperationRegistration = v7;
    CallbackRegistration.Altitude = DestinationString;
    v2 = ObRegisterCallbacks_0(&CallbackRegistration, &RegistrationHandle);
    if ( v2 < 0 )
      goto LABEL_7;
    sub_FFFFF8073777F790((__int64)g_StartRoutineHid0, (__int64)&byte_FFFFF80737848551);
    qword_FFFFF807378485D8 = v3;
```

```
__int64 sub_FFFFF8073778336C()
{
  char v0; // al
  unsigned int v1; // ebx

  v0 = byte_FFFFF80737848210;
  if ( !byte_FFFFF80737848210 )
  {
    if ( PsSetLoadImageNotifyRoutine_0((PLOAD_IMAGE_NOTIFY_ROUTINE)g_ImageNotifyRoutine) < 0 )
    {
      v0 = byte_FFFFF80737848210;
      v1 = 0xC00000EF;
      goto LABEL_6;
    }
    v0 = 1;
    byte_FFFFF80737848210 = 1;
  }
  v1 = 0;
LABEL_6:
  if ( !v0 )
    sub_FFFFF807377833C0();
  return v1;
}
```

```
NTSTATUS sub_FFFFF8073777F620()
{
  NTSTATUS result; // eax

  if ( !byte_FFFFF807378396C0 )
  {
    result = PsSetCreateThreadNotifyRoutine_0((PCREATE_THREAD_NOTIFY_ROUTINE)g_ThreadNotifyRoutine);
    if ( result < 0 )
      return result;
    byte_FFFFF807378396C0 = 1;
  }
  return 0;
}
```

```
__int64 CreateProcessNotifyRoutineEx()
{
  char v0; // al
  NTSTATUS ProcessNotifyRoutine; // ebx

  v0 = byte_FFFFF80737848330;
  if ( byte_FFFFF80737848330 )
  {
    ProcessNotifyRoutine = 0;
  }
  else
  {
    ProcessNotifyRoutine = PsSetCreateProcessNotifyRoutineEx_0(g_ProcessNotifyRoutineEx, 0);
    if ( ProcessNotifyRoutine < 0 )
    {
      v0 = byte_FFFFF80737848330;
    }
    else
    {
      v0 = 1;
      ProcessNotifyRoutine = 0;
      byte_FFFFF80737848330 = 1;
    }
  }
  if ( !v0 )
    sub_FFFFF80737783BB0();
  return (unsigned int)ProcessNotifyRoutine;
}
```

这里样本会选择将系统回调拿到的信息先放到全局然后KeSetEvent通知其他地方处理

```
__int64 __fastcall g_PostThreadTask(unsigned int a1, const void *a2)
{
  __int64 result; // rax
  unsigned int v3; // [rsp+20h] [rbp-38h]
  volatile signed __int32 *v4; // [rsp+28h] [rbp-30h]
  void *v5; // [rsp+30h] [rbp-28h]

  v4 = 0i64;
  v3 = 0;
  v5 = 0i64;
  if ( !byte_FFFFF8073780295A || a1 )
  {
    if ( byte_FFFFF8073780295A && a1 == 1 )
    {
      v4 = (volatile signed __int32 *)&unk_FFFFF807378032D8;
      result = (__int64)&unk_FFFFF80737802EA0;
      v5 = &unk_FFFFF80737802EA0;
      v3 = 1080;
    }
    else if ( byte_FFFFF8073780295A && a1 == 2 )
    {
      v4 = (volatile signed __int32 *)&unk_FFFFF807378032F8;
      result = (__int64)&unk_FFFFF807378032E0;
      v5 = &unk_FFFFF807378032E0;
      v3 = 24;
    }
    else if ( byte_FFFFF8073780295A && a1 == 3 )
    {
      v4 = (volatile signed __int32 *)&unk_FFFFF80737803318;
      result = (__int64)&unk_FFFFF80737803300;
      v5 = &unk_FFFFF80737803300;
      v3 = 24;
    }
    else if ( byte_FFFFF8073780295A && a1 == 5 )
    {
      v4 = (volatile signed __int32 *)&unk_FFFFF80737803540;
      result = (__int64)&unk_FFFFF80737803320;
      v5 = &unk_FFFFF80737803320;
      v3 = 544;
    }
    else
    {
      result = (unsigned __int8)byte_FFFFF8073780295A;
      if ( byte_FFFFF8073780295A && a1 == 4 )
      {
        v4 = (volatile signed __int32 *)&unk_FFFFF80737803544;
00001E36 g_PostThreadTask:42 (FFFFF80737762B36)
```

```
__int64 __fastcall sub_FFFFF80737762960(unsigned int a1)
{
  __int64 result; // rax
  int i; // [rsp+20h] [rbp-18h]

  if ( a1 < 6 )
  {
    for ( i = 0; i < 4; ++i )
    {
      if ( *((_BYTE *)&unk_FFFFF80737802960 + 64 * (__int64)i) )
      {
        if ( *((_QWORD *)&unk_FFFFF80737802960 + 8 * (__int64)i + a1 + 2) )
          KeSetEvent(*((PRKEVENT *)&unk_FFFFF80737802960 + 8 * (__int64)i + a1 + 2), 0, 0);
      }
      result = (unsigned int)(i + 1);
    }
  }
  return result;
}
```

系统回调部分基本都是一些常规操作，所以这里就不赘述了。

驱动线程部分

快也搜这个创建线程

1.线程创建部分

用ark观察有3个线程的起始地址为SeSetAuditParameter函数内部的jmp rcx,

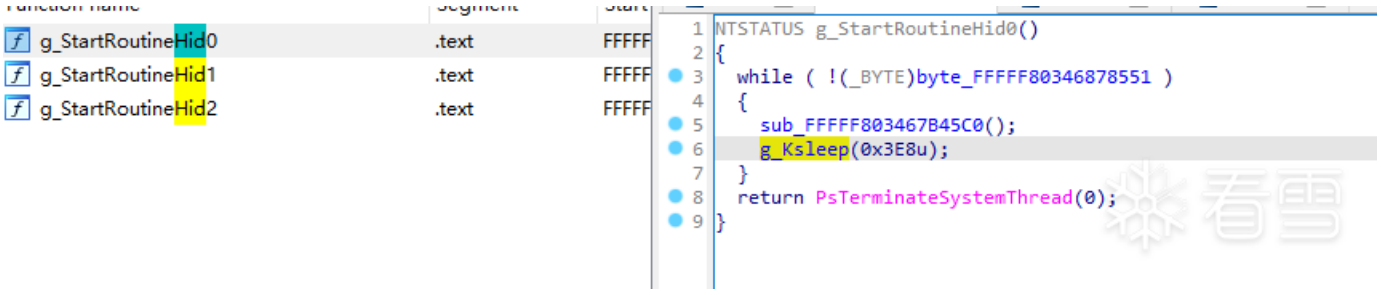
初始化线程地址：在SeSetAuditParameter内部搜0xff 0xe1

```
_BYTE *g_GetThreadAddres_forSeSetAuditParameter()
{
  __int64 v0; // rdi
  char *SystemAddress; // rax
  _BYTE *v2; // rbx
  _BYTE *v3; // rsi

  v0 = 0i64;
  if ( !sub_FFFFF803467B13C0() )
  {
    SystemAddress = (char *)g_GetSystemAddress(L"SeSetAuditParameter");
    if ( SystemAddress )
    {
      v2 = SystemAddress + 16;
      v3 = SystemAddress + 528;
      while ( v2 != v3 )
      {
        if ( MmIsAddressValid_1(v2) && MmIsAddressValid_1(v2 + 1) && *v2 == 0xFF && v2[1] == 0xE1 )
          return v2;
        ++v2;
      }
    }
  }
  return (_BYTE *)v0;
}
```

这里具体也可以参考大表哥的帖子，我按表哥帖子，结果发现那部分代码已经被vm了。

```
1 | 2.反手hook创建线程得到真实的线程地址。
```



g_StartRoutineHid0内容：

```
v2 = 0i64;
RtlInitUnicodeString(&DestinationString, L"\\Driver");
ObjectAttributes.Length = 48;
ObjectAttributes.ObjectName = &DestinationString;
ObjectAttributes.RootDirectory = 0i64;
ObjectAttributes.Attributes = 576;
*(_OWORD *)&ObjectAttributes.SecurityDescriptor = 0i64;
v3 = ZwOpenDirectoryObject(&DirectoryHandle, 1u, &ObjectAttributes);
if ( v3 >= 0 )
{
  v3 = ObReferenceObjectByHandle_1(DirectoryHandle, 0, 0i64, 0, &Object, 0i64);
  if ( v3 >= 0 )
  {
    v5 = (PVOID **)Object;
    LOBYTE(v3) = MmIsAddressValid_1(Object);
    if ( (_BYTE)v3 )
    {
      v6 = 37i64;
      do
      {
        for ( i = *v5; i; i = (PVOID *)*i )
        {
          LOBYTE(v3) = MmIsAddressValid_1(i);
          if ( !(_BYTE)v3 )
            break;
          v8 = i[1];
          if ( !v8 )
```

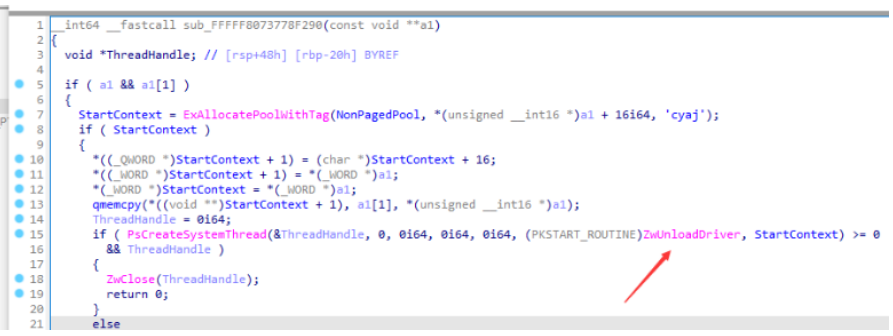
g_StartRoutineHid1内容：

```
NTSTATUS g_StartRoutineHid1()
{
  int v1; // [rsp+20h] [rbp-18h]

  v1 = 0;
  while ( dword_FFFFF80737804310 )
  {
    v1 = 0;
LABEL_6:
    g_Ksleep(1000u);
    if ( (_BYTE)g_flag )                    // 等待90秒 g_flag 还不初始化 则蓝屏
      return PsTerminateSystemThread(0);
  }
  if ( (unsigned int)++v1 < 90 )
    goto LABEL_6;
  sub_FFFFF80737765A60();                   // 卸载驱动 卸载失败 则蓝屏
  return PsTerminateSystemThread(0);
}
```

```
__int64 __fastcall sub_FFFFF80737765A90(unsigned int a1)
{
  __int64 result; // rax

  BugCheckParameter1 = a1;
  result = sub_FFFFF8073778F290((const void **)&BugCheckParameter3);
  if ( (_DWORD)result )
    KeBugCheckEx(0x1010003u, BugCheckParameter1, (int)result, (ULONG_P
  return result;
}
```



```
__int64 __fastcall sub_FFFFF8073778F290(const void **a1)
{
  void *ThreadHandle; // [rsp+48h] [rbp-20h] BYREF

  if ( a1 && a1[1] )
  {
    StartContext = ExAllocatePoolWithTag(NonPagedPool, *(unsigned __int16 *)a1 + 16i64, 'cyaj');
    if ( StartContext )
    {
      *((_QWORD *)StartContext + 1) = (char *)StartContext + 16;
      *((_WORD *)StartContext + 1) = *(_WORD *)a1;
      *(_WORD *)StartContext = *(_WORD *)a1;
      qmemcpy(*((void **)StartContext + 1), a1[1], *(unsigned __int16 *)a1);
      ThreadHandle = 0i64;
      if ( PsCreateSystemThread(&ThreadHandle, 0, 0i64, 0i64, 0i64, (PKSTART_ROUTINE)ZwUnloadDriver, StartContext) >= 0
        && ThreadHandle )
      {
        ZwClose(ThreadHandle);
        return 0;
      }
      else
```

g_StartRoutineHid2内容：

```
NTSTATUS g_StartRoutineHid2()
{
  ULONG v0; // edx
  unsigned int a1; // [rsp+24h] [rbp-24h]
  unsigned int a1_4; // [rsp+28h] [rbp-20h]
  int i; // [rsp+2Ch] [rbp-1Ch]
  int j; // [rsp+30h] [rbp-18h]
  int v6; // [rsp+34h] [rbp-14h] BYREF
  PIRP process; // [rsp+38h] [rbp-10h]

  do
  {
    for ( a1 = 16; a1 < 0x40000; a1 += 4 )
    {
      process = (PIRP)g_GetProcessObj((HANDLE)a1);
      if ( process )
      {
        v6 = 0;
        if ( sub_FFFFF80737787C34((PRKPROCESS)process, &v6) )// 查 \\TX_SSO_SHARE_INFO_SIZE \\TENPROTECT3_SHARE_DATA_ \\TENPROTECT6_SHARE_DATA_
          sub_FFFFF80737766FB0((struct _KPROCESS *)process, v6, a1);// 则加入 全局保护结构
        WdmlibIoValidateDeviceIoControlAccess_1(process, v0);
      }
    }
    for ( i = 0; i < 10 && !(_BYTE)byte_FFFFF80737839468; ++i )
      g_Ksleep(0x3E8u);
    if ( (_BYTE)byte_FFFFF80737839468 )
      break;
    for ( a1_4 = 16; a1_4 < 0x40000; a1_4 += 4 )
      EnumProcessHandleTable((void *)a1_4, (__int64)g_CheckProcessHandlesCallback, 0i64);// 遍历句柄表
    for ( j = 0; j < 10 && !(_BYTE)byte_FFFFF80737839468; ++j )
      g_Ksleep(0x3E8u);
  }
  while ( !(_BYTE)byte_FFFFF80737839468 );
  return PsTerminateSystemThread(0);
}
```

句柄表遍历


```
char __fastcall g_CheckProcessHandlesCallback(
        void *rcx0,
        __int64 a2,
        __int64 a3,
        __int64 a4,
        struct _KPROCESS *a5,
        int a6,
        int a7,
        __int64 a8,
        __int64 a9)
{
  ULONG v9; // edx
  PEPROCESS process; // [rsp+30h] [rbp-38h]
  HANDLE ProcessId; // [rsp+48h] [rbp-20h]

  if ( rcx0 )
  {
    if ( a2 )
    {
      if ( a5 )
      {
        if ( a3 )
        {
          if ( a8 )
          {
            if ( a9 )
            {
              if ( (POBJECT_TYPE *)sub_FFFFF80737781AB8((__int64)a5) == PsProcessType )
              {
                process = g_GetProcessObj(rcx0);
                if ( process )
                {
                  WdmlibIoValidateDeviceIoControlAccess_1((PIRP)process, v9);
                  if ( !g_IsProtectProcess(process) )
                  {
                    if ( g_IsProtectProcess(a5) )
                    {
                      ProcessId = PsGetProcessId(a5);
                      sub_FFFFF80737766540(ProcessId);
                    }
                  }
```

这个是在线程里边的

```
_int64 __fastcall sub_FFFFF80737772C00(
        __int64 a1,
        __int64 a2,
        __int64 a3,
        __int64 a4,
        PETHREAD Thread,
        int a6,
        int a7,
        __int64 a8,
        __int64 a9,
        __int64 a10)

_DWORD *v11; // [rsp+28h] [rbp-20h]

if ( a1 && a2 && Thread && a3 && a8 && a9 && a10 )
{
  if ( *(_QWORD *)(a10 + 16) )
  {
    ++**(_DWORD **)(a10 + 16);
  }
  else if ( *(_DWORD *)(a10 + 12) < *(_DWORD *)(a10 + 8) )
  {
    v11 = (_DWORD *)(*(_QWORD *)a10 + 1068i64 * *(unsigned int *)(a10 + 12));
    *v11 = a1;
    v11[1] = a2;
    v11[2] = sub_FFFFF80737772AF0((__int64)Thread);
    v11[3] = *(_DWORD *)(a3 + 8);
    if ( v11[2] == 12 )
    {
      v11[265] = (unsigned int)PsGetThreadId(Thread);
    }
    else if ( v11[2] == 11 )
    {
      v11[266] = (unsigned int)PsGetProcessId_1(Thread);
      g_GetProcessFileInfomation_1(Thread, (char *)v11 + 538, 0x20Au);
    }
    ++*(_DWORD *)(a10 + 12);
  }
}
return 0i64;
```

Pseudocode-BK

```
__int64 __fastcall sub_FFFFF80737772AF0(__int64 a1)
{
  unsigned int v2; // [rsp+20h] [rbp-18h]
  POBJECT_TYPE *v3; // [rsp+28h] [rbp-10h]

  v2 = 0;
  if ( a1 )
  {
    v3 = (POBJECT_TYPE *)sub_FFFFF80737781AB8(a1);
    if ( v3 == IoFileObjectType )
    {
      return 2;
    }
    else if ( v3 == ExEventObjectType )
    {
      return 3;
    }
    else if ( v3 == ExSemaphoreObjectType )
    {
      return 6;
    }
    else if ( v3 == PsProcessType )
    {
      return 11;
    }
    else if ( v3 == PsThreadType )
    {
      return 12;
    }
    else if ( v3 == PsJobType )
    {
      return 13;
    }
    else if ( v3 == SeTokenObjectType )
    {
      return 14;
    }
    else
    {
      return 25;
    }
  }
}
```

这个是3环接口使用的

部分信息查询

我就查一下怎么了


```
__int64 sub_FFFFF8073778C650()
{
  unsigned int v0; // ebx
  unsigned int tid; // esi
  PETHREAD ThreadObj; // rax
  IRP *v3; // rdi
  ULONG v4; // edx
  unsigned int v5; // ebp
  IRP *v6; // rsi
  ULONG v7; // edx
  __int64 v9; // [rsp+40h] [rbp+8h] BYREF

  v0 = 0;
  v9 = 0i64;
  tid = 8;
  while ( 1 )
  {
    ThreadObj = g_GetThreadObj((HANDLE)tid);
    v3 = (IRP *)ThreadObj;
    if ( ThreadObj )
      break;
LABEL_6:
    tid += 4;
    if ( tid > 0x40000 )
      goto LABEL_7;
  }
  if ( (int)g_GetThreadStartAddress(ThreadObj, &v9) < 0 || (unsigned __int64)(v9 - 1) > 0x7FFFFFFFFFFDi64 )
  {
    WdmlibIoValidateDeviceIoControlAccess_1(v3, v4);
    v3 = 0i64;
    goto LABEL_6;
  }
LABEL_7:
  if ( v3 )
```

```
  __int64 v18; // [rsp+A0h] [rbp+18h]
  PSYSTEM_PROCESS_INFORMATION P; // [rsp+A8h] [rbp+20h] BYREF

  if ( a1 )
  {
    v18 = a3;
    P = 0i64;
    if ( a2 )
    {
      v17 = 0;
      v5 = g_QuerySystemInformation(SystemProcessInformation, &P, &v17);
      v6 = P;
      if ( v5 >= 0 )
      {
        if ( !P )
          return;
        if ( v17 )
        {
          v7 = P;
          if ( P->UniqueProcessId == a1 )
          {
LABEL_9:
            NumberOfThreads = v7->NumberOfThreads;
            if ( NumberOfThreads )
            {
              p_HardFaultCount = &v7[1].HardFaultCount;
              v11 = NumberOfThreads;
              do
              {
                v12 = (void *)*((_QWORD *)p_HardFaultCount + 4);
                v13 = *(_QWORD *)p_HardFaultCount;
                v14 = p_HardFaultCount[13];
                v15 = p_HardFaultCount[14];
                v16[0] = 0i64;
                g_GethThreadTodwStartAddres(v12, v16);
                a2((__int64)a1, (__int64)v12, v16[0], v13, v14, v15, v18);
                p_HardFaultCount += 20;
                --v11;
              }
              while ( v11 );
              v6 = P;
            }
```

```
_int64 __fastcall sub_FFFFF8073777FB28(_QWORD *a1)
{
  int SystemInformation; // ebx
  _DWORD *v3; // rax

  if ( a1 )
  {
    SystemInformation = g_QuerySystemInformation(SystemModuleInformation, a1, 0i64);
    if ( SystemInformation >= 0 )
    {
      v3 = (_DWORD *)*a1;
      if ( !*a1 || v3 == (_DWORD *)-8i64 )
        return 0x8000001A;
      else
        sub_FFFFF80737780000((__int64)(v3 + 2), 0, *v3 - 1);
    }
  }
  else
  {
    return 0xC00000EF;
  }
  return (unsigned int)SystemInformation;
}
```

```
  if ( a1 )
  {
    P = 0i64;
    if ( (int)g_QuerySystemInformation(SystemProcessInformation, &P, &v9) < 0 )
    {
LABEL_11:
      if ( P )
        ExFreePoolWithTag(P, 0);
    }
    else
    {
      v4 = P;
      if ( P )
      {
        while ( 1 )
        {
          ProcessObj = (IRP *)g_GetProcessObj(v4->UniqueProcessId);
          if ( ProcessObj )
          {
            sub_FFFFF807377B0F00(v8, 0i64, 0x20Aui64);
            if ( (int)g_GetProcessFileInfomation_0((struct _KPROCESS *)ProcessObj, v8, 0x208u) >= 0
              && !a1(v4->UniqueProcessId, v4->InheritedFromUniqueProcessId, (char *)v8, v4->NumberOfThreads, a2) )
            {
              WdmlibIoValidateDeviceIoControlAccess_1(ProcessObj, v6);
              goto LABEL_11;
            }
            WdmlibIoValidateDeviceIoControlAccess_1(ProcessObj, v6);
          }
          NextEntryOffset = v4->NextEntryOffset;
          if ( !(_DWORD)NextEntryOffset )
            goto LABEL_11;
          v4 = (PSYSTEM_PROCESS_INFORMATION)((char *)v4 + NextEntryOffset);
        }
      }
    }
  }
```

```
__int64 __fastcall sub_FFFFF807377899B0(_QWORD *a1)
{
  unsigned int v1; // ebx
  int v3; // eax
  _DWORD *v4; // rcx
  ULONG v6; // [rsp+38h] [rbp+10h] BYREF
  PVOID P; // [rsp+40h] [rbp+18h] BYREF

  v1 = 0;
  v6 = 0;
  P = 0i64;
  v3 = g_QuerySystemInformation(SystemModuleInformation, &P, &v6);
  v4 = P;
  if ( v3 >= 0 )
  {
    if ( !P )
      return v1;
    if ( a1 )
      *a1 = *((_QWORD *)P + 3);
    v1 = v4[8];
  }
  if ( v4 )
    ExFreePoolWithTag(v4, 0);
  return v1;
}
```

```
_int64 __fastcall sub_FFFFF80737771B10(_QWORD *a1, _DWORD *a2)

NTSTATUS SystemInformation_1; // [rsp+20h] [rbp-48h]
ULONG NumberOfBytes[3]; // [rsp+24h] [rbp-44h] BYREF
int SystemInformation[8]; // [rsp+30h] [rbp-38h] BYREF

memset(NumberOfBytes, 0, sizeof(NumberOfBytes));
if ( a1 && a2 )
{
  SystemInformation[0] = 0;
  memset(&SystemInformation[2], 0, 0x18ui64);
  SystemInformation_1 = ZwQuerySystemInformation_1(SystemBigPoolInformation, SystemInformation, 0x20u, NumberOfBytes);
  if ( !SystemInformation_1 || SystemInformation_1 == -1073741820 )
  {
    NumberOfBytes[0] += 4096;
    *(_QWORD *)&NumberOfBytes[1] = ExAllocatePoolWithTag(NonPagedPool, NumberOfBytes[0], 0x6B737074u);
    if ( *(_QWORD *)&NumberOfBytes[1] )
    {
      g_ZeroMem(*(void **)&NumberOfBytes[1], NumberOfBytes[0]);
      SystemInformation_1 = ZwQuerySystemInformation_1(
                              SystemBigPoolInformation,
                              *(PVOID *)&NumberOfBytes[1],
                              NumberOfBytes[0],
                              NumberOfBytes);
      if ( !SystemInformation_1 )
      {
        *a1 = *(_QWORD *)&NumberOfBytes[1];
        *a2 = NumberOfBytes[0];
      }
    }
    else
    {
      SystemInformation_1 = -1073741664;
    }
  }
}
else
{
  SystemInformation_1 = -1073741811;
```

```
_int64 __fastcall sub_FFFFF80737772750(void *a1)

unsigned int v2; // [rsp+20h] [rbp-238h]
wchar_t Str1[264]; // [rsp+30h] [rbp-228h] BYREF

v2 = 0;
if ( a1 )
{
  memset(Str1, 0, 520);
  if ( QueryObject(a1, ObjectTypeInformation, Str1, 520) )
  {
    if ( !wcsncmp(Str1, L"KeyedEvent", 0xAui64) )
    {
      return 21;
    }
    else if ( !wcsncmp(Str1, L"Key", 3ui64) )
    {
      return 1;
    }
    else if ( !wcsncmp(Str1, L"File", 4ui64) )
    {
      return 2;
    }
    else if ( !wcsncmp(Str1, L"Event", 5ui64) )
    {
      return 3;
    }
    else if ( !wcsncmp(Str1, L"Timer", 5ui64) )
    {
      return 20;
    }
    else if ( !wcsncmp(Str1, L"Token", 5ui64) )
    {
      return 14;
    }
    else if ( !wcsncmp(Str1, L"Mutant", 6ui64) )
    {
      return 4;
```

页目录初始化

```
__int64 g_InitializeSystemSpace()
{
  unsigned __int64 v0; // rbx
  unsigned __int64 cr3; // rbx
  __int64 result; // rax
  int v3; // ecx
  __int64 v4; // r8
  __int64 v5; // rdx
  __int64 v6; // rbx

  v0 = __readcr3();
  cr3 = v0 & 0xFFFFFFFFFFFFF000ui64;
  result = (__int64)MmGetVirtualForPhysical((PHYSICAL_ADDRESS)cr3);
  v3 = 0;
  v4 = result;
  if ( result )
  {
    v5 = 0i64;
    v6 = cr3 & 0xFFFFFFFFF000i64;
    while ( 1 )
    {
      result = *(_QWORD *)(v4 + 8 * v5) & 0xFFFFFFFFF000i64;
      if ( result == v6 )
        break;
      ++v3;
      if ( ++v5 >= 512 )
        return result;
    }
    PteBase = (v3 + 0x1FFFE00i64) << 39;
    PdeBase = ((v3 + 0x1FFFE00i64) << 39) + ((__int64)v3 << 30);
    result = PdeBase + ((__int64)v3 << 21);
    PpeBase = result;
    PxeBase = result + ((__int64)v3 << 12);
  }
  return result;
}
```

驱动回调部分

回调接口部分的校验：

```
v10 = a4;
if ( rcx0 )
{
  if ( a2 >= 0x18ui64 )
  {
    if ( a3 )
    {
      sub_FFFFF80737774190(rcx0, a2);
      if ( (int)abs32(sub_FFFFF8073777FB90() - *(_DWORD *)(rcx0 + 4)) <= 10000 )
      {
        hThread = g_GetCurrentThread();
        hProcess = IoThreadToProcess_0(hThread);
        if ( g_IsProtectProcess(hProcess) )
        {
          if ( (int)g_DispatchCall(*(_DWORD *)rcx0, rcx0 + 24, a2 - 24, a3, v10) >= 0 )
          {
            if ( a3 )
            {
              sub_FFFFF80737774A10(a3, v10);
              *a5 = v10;
            }
          }
        }
```

驱动的接口函数

```
{
  __int64 (__fastcall *v6)(); // [rsp+28h] [rbp-20h]

  v6 = 0i64
  if ( a1
  {
    if ( a1
    {
      if ( a1 >
      {
        if ( a1
        {
          switch ( a1 )
          {
            case 0x80002814:
              v6 = (__int64 (__fastcall *)())sub_FFFFF80737770120;
              break;
            case 0x80002818:
              v6 = (__int64 (__fastcall *)())sub_FFFFF807377703D0;
              break;
            case 0x8000281C:
              v6 = (__int64 (__fastcall *)())sub_FFFFF8073776F1E0;
              break;
            case 0x80002820:
              v6 = (__int64 (__fastcall *)())sub_FFFFF8073776F690;
              break;
          }
        }
        else
        {
          switch ( a1 )
          {
            cas
              v6 = (__int64 (__fastcall *)())sub_FFFFF8073776F510;
              bre
            cas
              v6 = (__int64 (__fastcall *)())sub_FFFFF80737770550;
              bre
            cas
              v6 = (__int64 (__fastcall *)())sub_FFFFF8073776F900;
              bre
          case
            v6 = (__int64 (__fastcall *)())sub_FFFFF8073776F3F0;
            break;
          case
```

理论上还是不要偷它回调了，虽然楼主试过了。

关于一些奥里给检查

```
__int64 v1; // rbx
__int64 v3; // rax
_DWORD *v4; // rsi

v1 = 0i64;
if ( a1 )
{
  if ( [illegible] 64i64) && *(_WORD *)a1 == 0x5A4D )
  {
    v3 = *(int *)(a1 + 60);
    if ( (int)v3 <= 1024 )
    {
      v4 = (_DWORD *)(a1 + v3);
      if ( [illegible]a1 + v3, 248i64) && *v4 == 0x4550 )
        return v4;
    }
  }
}
return (_DWORD *)v1;
```

```
__int64 v2; // rdi
__int64 v5; // rax
unsigned int v6; // ecx
__int64 v7; // r14
__int64 v8; // r12
__int64 v9; // r13
__int64 v10; // r15
unsigned int v11; // ecx
unsigned int i; // esi
char *v13; // rbp
BOOLEAN (__stdcall **v14)(PVOID); // rax
_BYTE *v15; // rax
unsigned __int16 v16; // [rsp+60h] [rbp+8h]
char *v17; // [rsp+68h] [rbp+10h]
unsigned int v18; // [rsp+70h] [rbp+18h]

v17 = (char *)a2;
v2 = 0i64;
if ( !a1 )
  return 0i64;
if ( *(_WORD *)a1 != 0x5A4D )
  return 0i64;
v5 = *(int *)(a1 + 60);
if ( *(_DWORD *)(v5 + a1) != 0x4550 )
  return 0i64;
if ( *(_WORD *)(v5 + a1 + 24) == 0x20B )
  v6 = *(_DWORD *)(v5 + a1 + 136);
else
  v6 = *(_DWORD *)(v5 + a1 + 120);
v7 = v6;
v8 = a1 + *(unsigned int *)(v6 + a1 + 36);
v9 = a1 + *(unsigned int *)(v6 + a1 + 32);
v18 = *(_DWORD *)(v6 + a1 + 20);
v10 = a1 + *(unsigned int *)(v6 + a1 + 28);
v11 = v18;
for ( i = 0; i < v11; ++i )
{
  if ( a2 > 0xFFFF )
  {
    if ( i >= *(_DWORD *)(v7 + a1 + 24) )
      return 0i64;
    v16 = *(_WORD *)(v8 + 2i64 * i);
```

快抹PE头

关于一部分标志检查


给我看看

```
NTSTATUS KdDisableDebugger_0(void)
{
  if ( KdDisableDebugger )
    return KdDisableDebugger();
  else
    return 0xC0000002;
}
```

```
BOOLEAN KdRefreshDebuggerNotPresent_0(void)
{
  BOOLEAN result; // al

  result = 1;
  if ( KdRefreshDebuggerNotPresent )
    return KdRefreshDebuggerNotPresent() != 0;
  return result;
}
```

```
  v2[17] = 0;
  SystemRoutineName.Length = 34;
  SystemRoutineName.MaximumLength = 36;
  SystemRoutineName.Buffer = (PWSTR)v2;
  g_KdDebuggerEnabled = (__int64)MmGetSystemRoutineAddress(&SystemRoutineName);// KdDebuggerEnabled
}
if ( !g_KdDebuggerNotPresent )
{


  v1.MaximumLength = 42;
  v1.Buffer = (PWSTR)v3;
  g_KdDebuggerNotPresent = (__int64)MmGetSystemRoutineAddress(&v1);// KdDebuggerNotPresent
}
```

```
__int64 g_GetProcessDebugPortOffset()
{
  unsigned int v0; // ebx
  char *SystemAddress; // rax
  unsigned int v2; // eax

  v0 = 0;
  SystemAddress = (char *)g_GetSystemAddress(L"PsGetProcessDebugPort");
  if ( SystemAddress )
  {
    v2 = *(_DWORD *)(SystemAddress + 3);
    if ( v2 - 0xA1 > 0x1F5E )
      return 0;
    return v2;
  }
```

参考

https://bbs.pediy.com/thread-260331.htm

https://bbs.pediy.com/thread-254276.htm


表哥牛逼

【公告】看雪·众安 2021 KCTF秋季赛 【最受欢迎战队奖】评选开始！

收藏 · 8　点赞 · 4　打赏　分享

最新回复 (16)


如斯咩咩咩 5天前　2 楼　0

浩哥牛逼

极客


killleer 5天前　3 楼　0

你们可做个人吧。。。

极客

最新回复 (16)

8

4

¥

Names
ace-game.sys
ACE-GAME.sys
nKgZdX
ACE-GAME64
iBiJpFnPjRu
uRcPdAxKhQnPoQbCfUvR
sOyKwEyLrXaHIC
sKeCkSaJyGyFzOcDIE
eKoDgVtNlUaYsRq
yBmXlBaDdH
fGpVhBbRmOeJxZgHdD
yFsOpMiJiV
jZfDvOgZcFnGiFfWkWj
xGuJeY
gAzPqCuAv
kNxLlQdKID
hGaMpUbZuPf
bMnCjVkCtSb
xYyDkFkFfCoVvJfZsLxV
zIzBmTeYeOiTwTcFtYvTh
tTpShYdO
tElNjErFdEuCcTi
fNmWjMiQwYqZyH
nMhSrFyNzNtGxW
mUuUmZwCk
lFnlxFcXhEt
wLqQiWnWpG
vZdYkX
ace-base.sys
kRkBwldEpDsGcBjW
zQdDbO

最新回复 (16)

8

4

¥

Files Written

C:\ACE-GAME.sys

C:\ACE-BASE.sys

C:\mfc140d.dll

Files Deleted

C:\ACE-GAME.sys

C:\ACE-BASE.sys

C:\mfc140d.dll

00:00	System:4:60	SetValueKey	[<HKLM>\SYSTEM\CURRENTCONTROLSET\ENUM\ROOT\LEGACY_QZELVBZ\0000] 'Class' = 'LegacyDriver'
00:00	System:4:60	SetValueKey	[<HKLM>\SYSTEM\CURRENTCONTROLSET\ENUM\ROOT\LEGACY_QZELVBZ\0000] 'ClassGUID' = '{8ECC055D-047F-11D1-A537-0000F8753ED1}'
00:00	System:4:60	SetValueKey	[<HKLM>\SYSTEM\CURRENTCONTROLSET\ENUM\ROOT\LEGACY_QZELVBZ\0000] 'DeviceDesc' = 'qzelvbz'
00:00	System:4:60	SetValueKey	[<HKLM>\SYSTEM\CURRENTCONTROLSET\SERVICES\qzelvbz\Enum] '0' = 'Root\LEGACY_QZELVBZ\0000'
00:00	System:4:60	SetValueKey	[<HKLM>\SYSTEM\CURRENTCONTROLSET\SERVICES\qzelvbz\Enum] 'Count' = '00000001'
00:00	System:4:60	SetValueKey	[<HKLM>\SYSTEM\CURRENTCONTROLSET\SERVICES\qzelvbz\Enum] 'NextInstance' = '00000001'
00:00	System:4:60	SetValueKey	[<HKLM>\SYSTEM\CURRENTCONTROLSET\ENUM\ROOT\LEGACY_QZELVBZ\0000\Control] 'ActiveService' = 'qzelvbz'
00:00	System:4:60	SetValueKey	[<HKLM>\System\CurrentControlSet\Services\qzelvbz\Enum] 'Count' = '00000001'
00:00	System:4:60	SetValueKey	[<HKLM>\System\CurrentControlSet\Services\qzelvbz\Enum] 'NextInstance' = '00000001'
00:00	System:4:60	WriteFile	'<DRIVERS>\ace-game.sys' Offset = 0x0 Length = 752768
00:00	System:4:60	SetValueKey	[<HKLM>\System\CurrentControlSet\Services\ACE-GAME] 'Type' = '00000001'
00:00	System:4:60	SetValueKey	[<HKLM>\System\CurrentControlSet\Services\ACE-GAME] 'ErrorControl' = '00000001'
00:00	System:4:60	SetValueKey	[<HKLM>\System\CurrentControlSet\Services\ACE-GAME] 'Start' = '00000001'
00:00	System:4:60	SetValueKey	[<HKLM>\System\CurrentControlSet\Services\ACE-GAME] 'WOW64' = '00000001'
00:00	System:4:60	SetValueKey	[<HKLM>\System\CurrentControlSet\Services\ACE-GAME] 'ImagePath' = '<DRIVERS>\ACE-GAME.sys'
00:00	System:4:60	SetValueKey	[<HKLM>\System\CurrentControlSet\Services\ACE-GAME] 'DisplayName' = 'ACE-GAME'
00:00	System:4:60	DriverLoad	DriverServiceName = '\Registry\Machine\System\CurrentControlSet\Services\ACE-GAME'

LuciferAda 5天前 4 楼 0

浩哥牛逼！！！！！！！！

极客

L0x1c 3 4天前 5 楼 0

浩哥牛逼！！

专家

值得怀疑 4天前 6 楼 0

> killleer 你们可做个人吧。。。

最后一个图是什么工具？？

极客

killleer 4天前 7 楼 0

> 值得怀疑 最后一个图是什么工具？？

大蜘蛛的沙箱，好像得企业申请，有人给这一类上传到大蜘蛛沙箱去了，所以vt那里能看到大蜘蛛沙箱的结果

极客

hixhi 4天前 8 楼 0

配图怪。图片比文章有意思了，这样可不行呀，老铁。

极客

最新回复 (16)

tutuj 9 4天前 9 楼 0

牛逼，表情包偷一手

大侠

杰克王 6 3天前 10 楼 0

> killleer 你们可做个人吧。。。

懂了 马上去看其他驱动

极客

杰克王 6 3天前 11 楼 0

> tutuj 牛逼，表情包偷一手

快 给我点表情包 快用完了

极客

杰克王 6 3天前 12 楼 0

> hixhi [em_85]配图怪。图片比文章有意思了，这样可不行呀，老铁。

老铁 我也不知道写啥子 但是代码很清楚了

极客

杰克王 6 3天前 13 楼 0

> L0x1c 浩哥牛逼！！

元神牛逼

极客

杰克王 6 3天前 14 楼 0

> LuciferAda 浩哥牛逼！！！！！！！！

牛逼

极客

hixhi 6 3天前 15 楼 0

> 杰克王 老铁 我也不知道写啥子 但是代码很清楚了

配图抢了代码的风头

极客

killleer 6 3天前 16 楼 0

> 杰克王 [em_13]懂了 马上去看其他驱动

危楼主最好做个人

极客

yirucandy 5 4 14分钟前 17 楼 0

牛逼牛逼，学习了！

专家