

在前面的博文《驱动开发：win10内核枚举SSDT表基址》中已经教大家如何寻找 SSDT 表基址了，找到后我们可根据序号获取到指定 SSDT 函数的原始地址，而如果需要输出所有 SSDT 表信息，则可以定义字符串列表，以此循环调用 GetSSDTFunctionAddress() 函数得到，当然在此期间也可以调用系统提供的 MmGetSystemRoutineAddress() 函数顺便把当前地址拿到，并通过循环方式得到完整的SSDT列表。

进程	驱动模块	内核层	内核钩子	应用层钩子	设置	监控	启动信息	注册表	服务	文件	网络	调试引擎
SSDT	Shadow SSDT	内核钩子	系统中断表	Object钩子								

索引	函数名	原始函数地址	当前函数地址	当前函数地址所在模块
0	NtAccessCheck	0xFFFFF8036EB12340	0xFFFFF8036EB12340	C:\Windows\system32\ntoskrnl.exe
1	NtWorkerFactoryWorkerReady	0xFFFFF8036EB1C3D0	0xFFFFF8036EB1C3D0	C:\Windows\system32\ntoskrnl.exe
2	NtAcceptConnectPort	0xFFFFF8036F0DE450	0xFFFFF8036F0DE450	C:\Windows\system32\ntoskrnl.exe
3	NtMapUserPhysicalPagesScatter	0xFFFFF8036F2992F0	0xFFFFF8036F2992F0	C:\Windows\system32\ntoskrnl.exe
4	NtWaitForSingleObject	0xFFFFF8036EFF3B50	0xFFFFF8036EFF3B50	C:\Windows\system32\ntoskrnl.exe
5	NtCallbackReturn	0xFFFFF8036EBC4E10	0xFFFFF8036EBC4E10	C:\Windows\system32\ntoskrnl.exe
6	NtReadFile	0xFFFFF8036EFE5220	0xFFFFF8036EFE5220	C:\Windows\system32\ntoskrnl.exe
7	NtDeviceIoControlFile	0xFFFFF8036EFE8770	0xFFFFF8036EFE8770	C:\Windows\system32\ntoskrnl.exe

调用 MmGetSystemRoutineAddress() 得到当前地址很容易实现，只需要将函数名字符串通过 RtlInitUnicodeString() 格式化一下即可。

```
// 署名权
// right to sign one's name on a piece of work
// PowerBy: LyShark
// Email: me@lyshark.com

#include <ntifs.h>

VOID UnDriver(PDRIVER_OBJECT driver)
{
    DbgPrint(("驱动程序卸载成功! \n"));
}

NTSTATUS DriverEntry(PDRIVER_OBJECT DriverObject, PUNICODE_STRING RegistryPath)
{
    DbgPrint("hello lyshark.com \n");

    // 获取SSDT起源地址
    UNICODE_STRING unicode;
    RtlInitUnicodeString(&unicode, L"NtOpenFile");
    PULONGLONG source_address = MmGetSystemRoutineAddress(&unicode);
    DbgPrint("[LyShark] NtOpenFile起源地址 = %p \n", source_address);

    DriverObject->DriverUnload = UnDriver;
    return STATUS_SUCCESS;
}
```

代码获得 NtOpenFile 这个函数的内存地址，输出效果如下所示：

索引	函数名	原始函数地址	钩子类型	当前函数地址
51	NtOpenFile	0xFFFFF8044A4304D0	-	0xFFFFF8044A4304D0
52	NtDelayExecution	0xFFFFF8044A3EB160	-	0xFFFFF8044A3EB160

DebugView on \\DESKTOP-B53PAV1 (local)

#	Time	Debug Print
13	30.23935318	674531250 - STORMINI: StorNVMe - POWER: IDLE
14	30.63413048	678437500 - STORMINI: StorNVMe - POWER: ACTIVE
15	30.78094482	hello lyshark.com
16	30.78575706	[LyShark] NtOpenFile起源地址 = FFFFFF8044A4304D0
17	31.90192413	688437500 - STORMINI: StorNVMe - POWER: IDLE
18	31.90888977	688437500 - STORMINI: StorNVMe - POWER: ACTIVE
19	32.46394730	驱动程序卸载成功!

根据上一章节的内容扩展，枚举完整SSDT表我们可以这样来实现。

```
// 署名权
// right to sign one's name on a piece of work
// PowerBy: LyShark
// Email: me@lyshark.com

#include <ntifs.h>
#pragma intrinsic(__readmsr)

typedef struct _SYSTEM_SERVICE_TABLE
{
    PVOID      ServiceTableBase;
    PVOID      ServiceCounterTableBase;
    ULONGLONG  NumberOfServices;
    PVOID      ParamTableBase;
} SYSTEM_SERVICE_TABLE, *PSYSTEM_SERVICE_TABLE;

ULONGLONG ssdt_base_aadress;
PSYSTEM_SERVICE_TABLE KeServiceDescriptorTable;

typedef UINT64(__fastcall *SCFN)(UINT64, UINT64);
SCFN scfn;

// 解密算法
VOID DecodeSSDT()
{
    UCHAR strShellCode[36] =
    "\x48\x8B\xC1\x4C\x8D\x12\x8B\xF8\xC1\xEF\x07\x83\xE7\x20\x4E\x8B\x14\x17\x4D\x63\x1C\x82\x49\x8B\xC3\x49\xC1\xFB\x04\x4D\x03\xD3\x49\x8B\xC2\xC3";
    /*
    48:8BC1          | mov rax,rcx          |
rcx=index
    4C:8D12          | lea r10,qword ptr ds:[rdx] |
rdx=ssdt
    8BF8            | mov edi,eax          |
    C1EF 07         | shr edi,7            |
    83E7 20         | and edi,20          |
    4E:8B1417       | mov r10,qword ptr ds:[rdi+r10] |
    4D:631C82       | movsxd r11,dword ptr ds:[r10+rax*4] |
    */
}
```

```

49:8BC3          | mov rax,r11          |
49:C1FB 04      | sar r11,4            |
4D:03D3          | add r10,r11          |
49:8BC2          | mov rax,r10          |
C3              | ret                  |
*/
scfn = ExAllocatePool(NonPagedPool, 36);
memcpy(scfn, strShellCode, 36);
}

// 获取 KeServiceDescriptorTable 首地址
ULONGLONG GetKeServiceDescriptorTable()
{
    // 设置起始位置
    PCHAR StartSearchAddress = (PCHAR)__readmsr(0xC0000082) - 0x1806FE;

    // 设置结束位置
    PCHAR EndSearchAddress = StartSearchAddress + 0x8192;
    // DbgPrint("扫描起始地址: %p --> 扫描结束地址: %p \n", StartSearchAddress,
    EndSearchAddress);

    PCHAR ByteCode = NULL;

    UCHAR OpCodeA = 0, OpCodeB = 0, OpCodeC = 0;
    ULONGLONG addr = 0;
    ULONG templong = 0;

    for (ByteCode = StartSearchAddress; ByteCode < EndSearchAddress; ByteCode++)
    {
        // 使用MmIsAddressValid()函数检查地址是否有页面错误
        if (MmIsAddressValid(ByteCode) && MmIsAddressValid(ByteCode + 1) &&
        MmIsAddressValid(ByteCode + 2))
        {
            OpCodeA = *ByteCode;
            OpCodeB = *(ByteCode + 1);
            OpCodeC = *(ByteCode + 2);

            // 对比特征值 寻找 nt!KeServiceDescriptorTable 函数地址
            // LyShark.com
            // 4c 8d 15 e5 9e 3b 00 lea r10,[nt!KeServiceDescriptorTable
            (fffff802`64da4880)]
            // 4c 8d 1d de 20 3a 00 lea r11,[nt!KeServiceDescriptorTableShadow
            (fffff802`64d8ca80)]
            if (OpCodeA == 0x4c && OpCodeB == 0x8d && OpCodeC == 0x15)
            {
                // 获取高位地址fffff802
                memcpy(&templong, ByteCode + 3, 4);

                // 与低位64da4880地址相加得到完整地址
                addr = (ULONGLONG)templong + (ULONGLONG)ByteCode + 7;
                return addr;
            }
        }
    }
    return 0;
}

```

```

}

// 得到函数相对偏移地址
ULONG GetOffsetAddress(ULONGLONG FuncAddr)
{
    ULONG dwtmp = 0;
    PULONG ServiceTableBase = NULL;
    if (KeServiceDescriptorTable == NULL)
    {
        KeServiceDescriptorTable =
        (PSYSTEM_SERVICE_TABLE)GetKeServiceDescriptorTable();
    }
    ServiceTableBase = (PULONG)KeServiceDescriptorTable->ServiceTableBase;
    dwtmp = (ULONG)(FuncAddr - (ULONGLONG)ServiceTableBase);
    return dwtmp << 4;
}

// 根据序号得到函数地址
ULONGLONG GetSSDTFunctionAddress(ULONGLONG NtApiIndex)
{
    ULONGLONG ret = 0;
    if (ssdt_base_address == 0)
    {
        // 得到ssdt基地址
        ssdt_base_address = GetKeServiceDescriptorTable();
    }
    if (scfn == NULL)
    {
        DecodeSSDT();
    }
    ret = scfn(NtApiIndex, ssdt_base_address);
    return ret;
}

// 查询函数系统地址
ULONG_PTR QueryFunctionSystemAddress(PWCHAR name)
{
    UNICODE_STRING na;
    ULONG_PTR address;
    RtlInitUnicodeString(&na, name);
    address = (ULONG_PTR)MmGetSystemRoutineAddress(&na);
    return address;
}

VOID UnDriver(PDRIVER_OBJECT driver)
{
    DbgPrint(("驱动程序卸载成功! \n"));
}

NTSTATUS DriverEntry(PDRIVER_OBJECT DriverObject, PUNICODE_STRING RegistryPath)
{
    DbgPrint("hello lyshark.com \n");
}

```

```

char *SSDT[464] = { "NtAccessCheck", "NtWorkerFactoryWorkerReady",
"NtAcceptConnectPort", "NtMapUserPhysicalPagesScatter", "NtWaitForSingleObject",
"NtCallbackReturn", "NtReadFile", "NtDeviceIoControlFile", "NtWriteFile",
"NtRemoveIoCompletion", "NtReleaseSemaphore", "NtReplyWaitReceivePort",
"NtReplyPort", "NtSetInformationThread", "NtSetEvent", "NtClose",
"NtQueryObject", "NtQueryInformationFile", "NtOpenKey", "NtEnumerateValueKey",
"NtFindAtom", "NtQueryDefaultLocale", "NtQueryKey", "NtQueryValueKey",
"NtAllocateVirtualMemory", "NtQueryInformationProcess",
"NtWaitForMultipleObjects32", "NtWriteFileGather", "NtSetInformationProcess",
"NtCreateKey", "NtFreeVirtualMemory", "NtImpersonateClientOfPort",
"NtReleaseMutant", "NtQueryInformationToken", "NtRequestWaitReplyPort",
"NtQueryVirtualMemory", "NtOpenThreadToken", "NtQueryInformationThread",
"NtOpenProcess", "NtSetInformationFile", "NtMapViewOfSection",
"NtAccessCheckAndAuditAlarm", "NtUnmapViewOfSection",
"NtReplyWaitReceivePortEx", "NtTerminateProcess", "NtSetEventBoostPriority",
"NtReadFileScatter", "NtOpenThreadTokenEx", "NtOpenProcessTokenEx",
"NtQueryPerformanceCounter", "NtEnumerateKey", "NtOpenFile", "NtDelayExecution",
"NtQueryDirectoryFile", "NtQuerySystemInformation", "NtOpenSection",
"NtQueryTimer", "NtFsControlFile", "NtWriteVirtualMemory",
"NtCloseObjectAuditAlarm", "NtDuplicateObject", "NtQueryAttributesFile",
"NtClearEvent", "NtReadVirtualMemory", "NtOpenEvent", "NtAdjustPrivilegesToken",
"NtDuplicateToken", "NtContinue", "NtQueryDefaultUILanguage",
"NtQueueApcThread", "NtYieldExecution", "NtAddAtom", "NtCreateEvent",
"NtQueryVolumeInformationFile", "NtCreateSection", "NtFlushBuffersFile",
"NtApphelpCacheControl", "NtCreateProcessEx", "NtCreateThread",
"NtIsProcessInJob", "NtProtectVirtualMemory", "NtQuerySection",
"NtResumeThread", "NtTerminateThread", "NtReadRequestData", "NtCreateFile",
"NtQueryEvent", "NtWriteRequestData", "NtOpenDirectoryObject",
"NtAccessCheckByTypeAndAuditAlarm", "NtQuerySystemTime",
"NtWaitForMultipleObjects", "NtSetInformationObject", "NtCancelIoFile",
"NtTraceEvent", "NtPowerInformation", "NtSetValueKey", "NtCancelTimer",
"NtSetTimer", "NtAccessCheckByType", "NtAccessCheckByTypeResultList",
"NtAccessCheckByTypeResultListAndAuditAlarm",
"NtAccessCheckByTypeResultListAndAuditAlarmByHandle",
"NtAcquireProcessActivityReference", "NtAddAtomEx", "NtAddBootEntry",
"NtAddDriverEntry", "NtAdjustGroupsToken", "NtAdjustTokenClaimsAndDeviceGroups",
"NtAlertResumeThread", "NtAlertThread", "NtAlertThreadByThreadId",
"NtAllocateLocallyUniqueId", "NtAllocateReserveObject",
"NtAllocateUserPhysicalPages", "NtAllocateUuids", "NtAllocateVirtualMemoryEx",
"NtAlpcAcceptConnectPort", "NtAlpcCancelMessage", "NtAlpcConnectPort",
"NtAlpcConnectPortEx", "NtAlpcCreatePort", "NtAlpcCreatePortSection",
"NtAlpcCreateResourceReserve", "NtAlpcCreateSectionView",
"NtAlpcCreateSecurityContext", "NtAlpcDeletePortSection",
"NtAlpcDeleteResourceReserve", "NtAlpcDeleteSectionView",
"NtAlpcDeleteSecurityContext", "NtAlpcDisconnectPort",
"NtAlpcImpersonateClientContainerOfPort", "NtAlpcImpersonateClientOfPort",
"NtAlpcOpenSenderProcess", "NtAlpcOpenSenderThread", "NtAlpcQueryInformation",
"NtAlpcQueryInformationMessage", "NtAlpcRevokeSecurityContext",
"NtAlpcSendWaitReceivePort", "NtAlpcSetInformation", "NtAreMappedFilesTheSame",
"NtAssignProcessToJobObject", "NtAssociateWaitCompletionPacket",
"NtCallEnclave", "NtCancelIoFileEx", "NtCancelSynchronousIoFile",
"NtCancelTimer2", "NtCancelWaitCompletionPacket", "NtCommitComplete",
"NtCommitEnlistment", "NtCommitRegistryTransaction", "NtCommitTransaction",
"NtCompactKeys", "NtCompareObjects", "NtCompareSigningLevels",
"NtCompareTokens", "ArbPreprocessEntry", "NtCompressKey", "NtConnectPort",

```

"NtConvertBetweenAuxiliaryCounterAndPerformanceCounter", "ArbAddReserved",
"NtCreateDebugObject", "NtCreateDirectoryObject", "NtCreateDirectoryObjectEx",
"NtCreateEnclave", "NtCreateEnlistment", "NtCreateEventPair", "NtCreateIRTimer",
"NtCreateIoCompletion", "NtCreateJobObject", "ArbAddReserved",
"NtCreateKeyTransacted", "NtCreateKeyedEvent", "NtCreateLowBoxToken",
"NtCreateMailslotFile", "NtCreateMutant", "NtCreateNamedPipeFile",
"NtCreatePagingFile", "NtCreatePartition", "NtCreatePort",
"NtCreatePrivateNamespace", "NtCreateProcess", "NtCreateProfile",
"NtCreateProfileEx", "NtCreateRegistryTransaction", "NtCreateResourceManager",
"NtCreateSectionEx", "NtCreateSemaphore", "NtCreateSymbolicLinkObject",
"NtCreateThreadEx", "NtCreateTimer", "NtCreateTimer2", "NtCreateToken",
"NtCreateTokenEx", "NtCreateTransaction", "NtCreateTransactionManager",
"NtCreateUserProcess", "NtCreateWaitCompletionPacket", "NtCreateWaitablePort",
"NtCreateWnfStateName", "NtCreateWorkerFactory", "NtDebugActiveProcess",
"NtDebugContinue", "NtDeleteAtom", "NtDeleteBootEntry", "NtDeleteDriverEntry",
"NtDeleteFile", "NtDeleteKey", "NtDeleteObjectAuditAlarm",
"NtDeletePrivateNamespace", "NtDeleteValueKey", "NtDeleteWnfStateData",
"NtDeleteWnfStateName", "NtDisableLastKnownGood", "NtDisplayString",
"NtDrawText", "NtEnableLastKnownGood", "NtEnumerateBootEntries",
"NtEnumerateDriverEntries", "NtEnumeratesSystemEnvironmentValuesEx",
"NtEnumerateTransactionObject", "NtExtendSection", "NtFilterBootOption",
"NtFilterToken", "NtFilterTokenEx", "NtFlushBuffersFileEx",
"NtFlushInstallUILanguage", "ArbPreprocessEntry", "NtFlushKey",
"NtFlushProcessWriteBuffers", "NtFlushVirtualMemory", "NtFlushWriteBuffer",
"NtFreeUserPhysicalPages", "NtFreezeRegistry", "NtFreezeTransactions",
"NtGetCachedSigningLevel", "NtGetCompleteWnfStateSubscription",
"NtGetContextThread", "NtGetCurrentProcessorNumber",
"NtGetCurrentProcessorNumberEx", "NtGetDevicePowerState",
"NtGetMUIRegistryInfo", "NtGetNextProcess", "NtGetNextThread",
"NtGetNlsSectionPtr", "NtGetNotificationResourceManager", "NtGetWriteWatch",
"NtImpersonateAnonymousToken", "NtImpersonateThread", "NtInitializeEnclave",
"NtInitializeNlsFiles", "NtInitializeRegistry", "NtInitiatePowerAction",
"NtIsSystemResumeAutomatic", "NtIsUILanguageComitted", "NtListenPort",
"NtLoadDriver", "NtLoadEnclaveData", "NtLoadKey", "NtLoadKey2", "NtLoadKeyEx",
"NtLockFile", "NtLockProductActivationKeys", "NtLockRegistryKey",
"NtLockVirtualMemory", "NtMakePermanentObject", "NtMakeTemporaryObject",
"NtManageHotPatch", "NtManagePartition", "NtMapCMFModule",
"NtMapUserPhysicalPages", "NtMapViewOfSectionEx", "NtModifyBootEntry",
"NtModifyDriverEntry", "NtNotifyChangedDirectoryFile",
"NtNotifyChangedDirectoryFileEx", "NtNotifyChangeKey",
"NtNotifyChangeMultipleKeys", "NtNotifyChangeSession", "NtOpenEnlistment",
"NtOpenEventPair", "NtOpenIoCompletion", "NtOpenJobObject", "NtOpenKeyEx",
"NtOpenKeyTransacted", "NtOpenKeyTransactedEx", "NtOpenKeyedEvent",
"NtOpenMutant", "NtOpenObjectAuditAlarm", "NtOpenPartition",
"NtOpenPrivateNamespace", "NtOpenProcessToken", "NtOpenRegistryTransaction",
"NtOpenResourceManager", "NtOpenSemaphore", "NtOpenSession",
"NtOpenSymbolicLinkObject", "NtOpenThread", "NtOpenTimer", "NtOpenTransaction",
"NtOpenTransactionManager", "NtPlugPlayControl", "NtPrePrepareComplete",
"NtPrePrepareEnlistment", "NtPrepareComplete", "NtPrepareEnlistment",
"NtPrivilegeCheck", "NtPrivilegeObjectAuditAlarm",
"NtPrivilegedServiceAuditAlarm", "NtPropagationComplete", "NtPropagationFailed",
"NtPulseEvent", "NtQueryAuxiliaryCounterFrequency", "NtQueryBootEntryOrder",
"NtQueryBootOptions", "NtQueryDebugFilterState", "NtQueryDirectoryFileEx",
"NtQueryDirectoryObject", "NtQueryDriverEntryOrder", "NtQueryEaFile",
"NtQueryFullAttributesFile", "NtQueryInformationAtom",

```

"NtQueryInformationByName", "NtQueryInformationEnlistment",
"NtQueryInformationJobObject", "NtQueryInformationPort",
"NtQueryInformationResourceManager", "NtQueryInformationTransaction",
"NtQueryInformationTransactionManager", "NtQueryInformationWorkerFactory",
"NtQueryInstallUILanguage", "NtQueryIntervalProfile", "NtQueryIoCompletion",
"NtQueryLicenseValue", "NtQueryMultipleValueKey", "NtQueryMutant",
"NtQueryOpenSubKeys", "NtQueryOpenSubKeysEx",
"CmpCleanupHigherLayerKcbCachesPreCallback", "NtQueryQuotaInformationFile",
"NtQuerySecurityAttributesToken", "NtQuerySecurityObject",
"NtQuerySecurityPolicy", "NtQuerySemaphore", "NtQuerySymbolicLinkObject",
"NtQuerySystemEnvironmentValue", "NtQuerySystemEnvironmentValueEx",
"NtQuerySystemInformationEx", "NtQueryTimerResolution", "NtQueryWnfStateData",
"NtQueryWnfStateNameInformation", "NtQueueApcThreadEx", "NtRaiseException",
"NtRaiseHardError", "NtReadOnlyEnlistment", "NtRecoverEnlistment",
"NtRecoverResourceManager", "NtRecoverTransactionManager",
"NtRegisterProtocolAddressInformation", "NtRegisterThreadTerminatePort",
"NtReleaseKeyedEvent", "NtReleaseWorkerFactoryWorker", "NtRemoveIoCompletionEx",
"NtRemoveProcessDebug", "NtRenameKey", "NtRenameTransactionManager",
"NtReplaceKey", "NtReplacePartitionUnit", "NtReplyWaitReplyPort",
"NtRequestPort", "NtResetEvent", "NtResetWriteWatch", "NtRestoreKey",
"NtResumeProcess", "NtRevertContainerImpersonation", "NtRollbackComplete",
"NtRollbackEnlistment", "NtRollbackRegistryTransaction",
"NtRollbackTransaction", "NtRollforwardTransactionManager", "NtSaveKey",
"NtSaveKeyEx", "NtSaveMergedKeys", "NtSecureConnectPort", "NtSerializeBoot",
"NtSetBootEntryOrder", "NtSetBootOptions", "NtSetCachedSigningLevel",
"NtSetCachedSigningLevel2", "NtSetContextThread", "NtSetDebugFilterState",
"NtSetDefaultHardErrorPort", "NtSetDefaultLocale", "NtSetDefaultUILanguage",
"NtSetDriverEntryOrder", "NtSetEaFile", "NtSetHighEventPair",
"NtSetHighWaitLowEventPair", "NtSetIRTimer", "NtSetInformationDebugObject",
"NtSetInformationEnlistment", "NtSetInformationJobObject",
"NtSetInformationKey", "NtSetInformationResourceManager",
"NtSetInformationSymbolicLink", "NtSetInformationToken",
"NtSetInformationTransaction", "NtSetInformationTransactionManager",
"NtSetInformationVirtualMemory", "NtSetInformationWorkerFactory",
"NtSetIntervalProfile", "NtSetIoCompletion", "NtSetIoCompletionEx",
"BvgaSetVirtualFrameBuffer", "NtSetLowEventPair", "NtSetLowWaitHighEventPair",
"NtSetQuotaInformationFile", "NtSetSecurityObject",
"NtSetSystemEnvironmentValue", "NtSetSystemEnvironmentValueEx",
"NtSetSystemInformation", "NtSetSystemPowerState", "NtSetSystemTime",
"NtSetThreadExecutionState", "NtSetTimer2", "NtSetTimerEx",
"NtSetTimerResolution", "NtSetUuidSeed", "NtSetVolumeInformationFile",
"NtSetWnfProcessNotificationEvent", "NtShutdownSystem",
"NtShutdownWorkerFactory", "NtSignalAndWaitForSingleObject",
"NtSinglePhaseReject", "NtStartProfile", "NtStopProfile",
"NtSubscribeWnfStateChange", "NtSuspendProcess", "NtSuspendThread",
"NtSystemDebugControl", "NtTerminateEnclave", "NtTerminateJobObject",
"NtTestAlert", "NtThawRegistry", "NtThawTransactions", "NtTraceControl",
"NtTranslateFilePath", "NtUmsThreadYield", "NtUnloadDriver", "NtUnloadKey",
"NtUnloadKey2", "NtUnloadKeyEx", "NtUnlockFile", "NtUnlockVirtualMemory",
"NtUnmapViewOfSectionEx", "NtUnsubscribeWnfStateChange", "NtUpdateWnfStateData",
"NtVdmControl", "NtWaitForAlertByThreadId", "NtWaitForDebugEvent",
"NtWaitForKeyedEvent", "NtWaitForWorkViaWorkerFactory", "NtWaitHighEventPair",
"NtWaitLowEventPair" };

```

```

for (size_t lyshark = 0; lyshark < 464; lyshark++)

```



```

{
    // 获取起源地址
    ANSI_STRING ansi = { 0 };
    UNICODE_STRING unicode = { 0 };

    ULONGLONG ssdt_address = GetKeServiceDescriptorTable();
    // DbgPrint("SSDT基地址 = %p \n", ssdt_address);

    // 根据序号得到函数地址
    ULONGLONG address = GetSSDTFunctionAddress(lyshark);
    ULONG offset = GetOffsetAddress(address);

    RtlInitAnsiString(&ansi, SSDT[lyshark]);
    RtlAnsiStringToUnicodeString(&unicode, &ansi, TRUE);
    PULONGLONG source_address = MmGetSystemRoutineAddress(&unicode);
    DbgPrint("[LyShark] 序号 => [%d] | 当前地址 => %p | 起源地址 => %p | 相对地址
=> %p | SSDT => %s \n", lyshark, address, source_address, offset,
SSDT[lyshark]);
}

DriverObject->DriverUnload = UnDriver;
return STATUS_SUCCESS;
}

```

我们运行这段程序，即可得到整个系统中所有的SSDT表地址信息；

```

Debug Print
hello lyshark.com
[LyShark] 序号 => [0] | 当前地址 => FFFFFFFF80449F12340 | 起源地址 => 0000000000000000 | 相对地址 => 00000000FCED...
[LyShark] 序号 => [1] | 当前地址 => FFFFFFFF80449F1C3D0 | 起源地址 => 0000000000000000 | 相对地址 => 00000000FCF7...
[LyShark] 序号 => [2] | 当前地址 => FFFFFFFF8044A4DE450 | 起源地址 => 0000000000000000 | 相对地址 => 0000000002B9...
[LyShark] 序号 => [3] | 当前地址 => FFFFFFFF8044A6992F0 | 起源地址 => 0000000000000000 | 相对地址 => 000000000474...
[LyShark] 序号 => [4] | 当前地址 => FFFFFFFF8044A3F3B50 | 起源地址 => FFFFFFFF8044A3F3B50 | 相对地址 => 0000000001CE...
[LyShark] 序号 => [5] | 当前地址 => FFFFFFFF80449FC4E10 | 起源地址 => 0000000000000000 | 相对地址 => 00000000FDA0...
[LyShark] 序号 => [6] | 当前地址 => FFFFFFFF8044A3E5220 | 起源地址 => FFFFFFFF8044A3E5220 | 相对地址 => 0000000001C0...
[LyShark] 序号 => [7] | 当前地址 => FFFFFFFF8044A3E8770 | 起源地址 => FFFFFFFF8044A3E8770 | 相对地址 => 0000000001C3...
[LyShark] 序号 => [8] | 当前地址 => FFFFFFFF8044A445EF0 | 起源地址 => FFFFFFFF8044A445EF0 | 相对地址 => 000000000221...
[LyShark] 序号 => [9] | 当前地址 => FFFFFFFF8044A4AEDA0 | 起源地址 => 0000000000000000 | 相对地址 => 00000000028A...
[LyShark] 序号 => [10] | 当前地址 => FFFFFFFF8044A4B0D10 | 起源地址 => 0000000000000000 | 相对地址 => 00000000028...
[LyShark] 序号 => [11] | 当前地址 => FFFFFFFF8044A3CE4F0 | 起源地址 => 0000000000000000 | 相对地址 => 0000000001A...
[LyShark] 序号 => [12] | 当前地址 => FFFFFFFF8044A407DD0 | 起源地址 => 0000000000000000 | 相对地址 => 0000000001E...
[LyShark] 序号 => [13] | 当前地址 => FFFFFFFF8044A3E7640 | 起源地址 => FFFFFFFF8044A3E7640 | 相对地址 => 0000000001C...
[LyShark] 序号 => [14] | 当前地址 => FFFFFFFF8044A4B06D0 | 起源地址 => FFFFFFFF8044A4B06D0 | 相对地址 => 00000000028...
[LyShark] 序号 => [15] | 当前地址 => FFFFFFFF8044A3F1670 | 起源地址 => FFFFFFFF8044A3F1670 | 相对地址 => 0000000001C...
[LyShark] 序号 => [16] | 当前地址 => FFFFFFFF8044A446FF0 | 起源地址 => 0000000000000000 | 相对地址 => 00000000022...
[LyShark] 序号 => [17] | 当前地址 => FFFFFFFF8044A3E45B0 | 起源地址 => FFFFFFFF8044A3E45B0 | 相对地址 => 0000000001B...
[LyShark] 序号 => [18] | 当前地址 => FFFFFFFF8044A4BBCC0 | 起源地址 => 0000000000000000 | 相对地址 => 00000000029...
[LyShark] 序号 => [19] | 当前地址 => FFFFFFFF8044A423620 | 起源地址 => 0000000000000000 | 相对地址 => 0000000001F...

```

在WinDBG中可看到完整的输出内容，当然有些函数没有被导出，起源地址是拿不到的。

Command									
10958750000 - STORINI: StorNVMe - POWER: ACTIVE									
hello lyshark.com									
[LyShark]	序号 => [0]	当前地址 => FFFF80449F12340	起源地址 => 0000000000000000	相对地址 => 00000000FCED7300	SSDT => NtAccessCheck				
[LyShark]	序号 => [1]	当前地址 => FFFF80449F1C3D0	起源地址 => 0000000000000000	相对地址 => 00000000FCF77C00	SSDT => NtWorkerFactoryWorkerReady				
[LyShark]	序号 => [2]	当前地址 => FFFF8044A4DE450	起源地址 => 0000000000000000	相对地址 => 0000000002B98400	SSDT => NtAcceptConnectPort				
[LyShark]	序号 => [3]	当前地址 => FFFF8044A932F0	起源地址 => 0000000000000000	相对地址 => 0000000004746E00	SSDT => NtMapUserPhysicalPagesScatter				
[LyShark]	序号 => [4]	当前地址 => FFFF8044A3F3B50	起源地址 => FFFF8044A43F3B50	相对地址 => 0000000001CEF400	SSDT => NtWaitForSingleObject				
[LyShark]	序号 => [5]	当前地址 => FFFF80449FC4E10	起源地址 => 0000000000000000	相对地址 => 00000000FDA02000	SSDT => NtCallbackReturn				
[LyShark]	序号 => [6]	当前地址 => FFFF8044A3E5220	起源地址 => FFFF8044A3E5220	相对地址 => 0000000001C06100	SSDT => NtReadFile				
[LyShark]	序号 => [7]	当前地址 => FFFF8044A3E8770	起源地址 => FFFF8044A3E8770	相对地址 => 0000000001C3B600	SSDT => NtDeviceIoControlFile				
[LyShark]	序号 => [8]	当前地址 => FFFF8044A445EF0	起源地址 => FFFF8044A445EF0	相对地址 => 0000000002212E00	SSDT => NtWriteFile				
[LyShark]	序号 => [9]	当前地址 => FFFF8044A4AEDA0	起源地址 => 0000000000000000	相对地址 => 00000000028A1900	SSDT => NtRemoveIoCompletion				
[LyShark]	序号 => [10]	当前地址 => FFFF8044A4B0D10	起源地址 => 0000000000000000	相对地址 => 00000000028C1000	SSDT => NtReleaseSemaphore				
[LyShark]	序号 => [11]	当前地址 => FFFF8044A3CE4F0	起源地址 => 0000000000000000	相对地址 => 0000000001A98E00	SSDT => NtReplyWaitReceivePort				
[LyShark]	序号 => [12]	当前地址 => FFFF8044A4A07DD0	起源地址 => 0000000000000000	相对地址 => 0000000001E31C00	SSDT => NtReplyPort				
[LyShark]	序号 => [13]	当前地址 => FFFF8044A3E7640	起源地址 => FFFF8044A3E7640	相对地址 => 0000000001C2A300	SSDT => NtSetInformationThread				
[LyShark]	序号 => [14]	当前地址 => FFFF8044A4B06D0	起源地址 => FFFF8044A4B06D0	相对地址 => 00000000028BAC00	SSDT => NtSetEvent				
[LyShark]	序号 => [15]	当前地址 => FFFF8044A3F1670	起源地址 => FFFF8044A3F1670	相对地址 => 0000000001CCA600	SSDT => NtClose				
[LyShark]	序号 => [16]	当前地址 => FFFF8044A4A46FF0	起源地址 => 0000000000000000	相对地址 => 0000000002223E00	SSDT => NtQueryObject				
[LyShark]	序号 => [17]	当前地址 => FFFF8044A3E45B0	起源地址 => FFFF8044A3E45B0	相对地址 => 0000000001BF9A00	SSDT => NtQueryInformationFile				
[LyShark]	序号 => [18]	当前地址 => FFFF8044A4BEC00	起源地址 => 0000000000000000	相对地址 => 0000000002970B00	SSDT => NtOpenKey				
[LyShark]	序号 => [19]	当前地址 => FFFF8044A4A23620	起源地址 => 0000000000000000	相对地址 => 0000000001FEA100	SSDT => NtEnumerateValueKey				
[LyShark]	序号 => [20]	当前地址 => FFFF8044A3CD510	起源地址 => FFFF8044A3CD510	相对地址 => 0000000001A89000	SSDT => NtFindAtom				
[LyShark]	序号 => [21]	当前地址 => FFFF8044A4A061A0	起源地址 => 0000000000000000	相对地址 => 0000000001E15900	SSDT => NtQueryDefaultLocale				
[LyShark]	序号 => [22]	当前地址 => FFFF8044A3F6200	起源地址 => 0000000000000000	相对地址 => 0000000001D15F00	SSDT => NtQueryKey				
[LyShark]	序号 => [23]	当前地址 => FFFF8044A3F40F0	起源地址 => 0000000000000000	相对地址 => 0000000001CF4E00	SSDT => NtQueryValueKey				
[LyShark]	序号 => [24]	当前地址 => FFFF8044A450AD0	起源地址 => FFFF8044A450AD0	相对地址 => 00000000022BEC00	SSDT => NtAllocateVirtualMemory				
[LyShark]	序号 => [25]	当前地址 => FFFF8044A4A11AC0	起源地址 => FFFF8044A4A11AC0	相对地址 => 0000000001F55B00	SSDT => NtQueryInformationProcess				
[LyShark]	序号 => [26]	当前地址 => FFFF8044A4A21410	起源地址 => 0000000000000000	相对地址 => 0000000001FC8000	SSDT => NtWaitForMultipleObjects32				
[LyShark]	序号 => [27]	当前地址 => FFFF8044A4AD450	起源地址 => 0000000000000000	相对地址 => 0000000002888400	SSDT => NtWriteFileGather				
[LyShark]	序号 => [28]	当前地址 => FFFF8044A4E0B0	起源地址 => FFFF8044A4E0B0	相对地址 => 0000000002294A00	SSDT => NtSetInformationProcess				
[LyShark]	序号 => [29]	当前地址 => FFFF8044A4B1F50	起源地址 => 0000000000000000	相对地址 => 00000000028D3E00	SSDT => NtCreateKey				
[LyShark]	序号 => [30]	当前地址 => FFFF8044A45B3D0	起源地址 => FFFF8044A45B3D0	相对地址 => 0000000002367C00	SSDT => NtFreeVirtualMemory				
[LyShark]	序号 => [31]	当前地址 => FFFF8044A46863A0	起源地址 => 0000000000000000	相对地址 => 0000000004617900	SSDT => NtImpersonateClientOfPort				
[LyShark]	序号 => [32]	当前地址 => FFFF8044A3FA450	起源地址 => 0000000000000000	相对地址 => 0000000001D58400	SSDT => NtReleaseMutant				
[LyShark]	序号 => [33]	当前地址 => FFFF8044A3F7AB0	起源地址 => FFFF8044A3F7AB0	相对地址 => 0000000001D2EA00	SSDT => NtQueryInformationToken				
[LyShark]	序号 => [34]	当前地址 => FFFF8044A4C2D40	起源地址 => FFFF8044A4C2D40	相对地址 => 00000000029E1300	SSDT => NtRequestWaitReplyPort				
[LyShark]	序号 => [35]	当前地址 => FFFF8044A4A59200	起源地址 => 0000000000000000	相对地址 => 0000000002345F00	SSDT => NtQueryVirtualMemory				
[LyShark]	序号 => [36]	当前地址 => FFFF8044A3CCA10	起源地址 => FFFF8044A3CCA10	相对地址 => 0000000001A7E000	SSDT => NtOpenThreadToken				
[LyShark]	序号 => [37]	当前地址 => FFFF8044A4175C0	起源地址 => FFFF8044A4175C0	相对地址 => 0000000001F29B00	SSDT => NtQueryInformationThread				
[LyShark]	序号 => [38]	当前地址 => FFFF8044A4A1D4F0	起源地址 => FFFF8044A4A1D4F0	相对地址 => 0000000001F88E00	SSDT => NtOpenProcess				
[LyShark]	序号 => [39]	当前地址 => FFFF8044A9E28EC0	起源地址 => FFFF8044A9E28EC0	相对地址 => 00000000FC042B00	SSDT => NtSetInformationFile				
[LyShark]	序号 => [40]	当前地址 => FFFF8044A3D7A50	起源地址 => FFFF8044A3D7A50	相对地址 => 0000000001B2E400	SSDT => NtMapViewOfSection				
[LyShark]	序号 => [41]	当前地址 => FFFF8044A4EDB0	起源地址 => 0000000000000000	相对地址 => 0000000002C67700	SSDT => NtAccessCheckAndAuditAlarm				
[LyShark]	序号 => [42]	当前地址 => FFFF8044A3C73D0	起源地址 => 0000000000000000	相对地址 => 0000000001A27C00	SSDT => NtUnmapViewOfSection				
[LyShark]	序号 => [43]	当前地址 => FFFF8044A3CE510	起源地址 => 0000000000000000	相对地址 => 0000000001A93000	SSDT => NtReplyWaitReceivePortEx				
[LyShark]	序号 => [44]	当前地址 => FFFF8044A439D00	起源地址 => 0000000000000000	相对地址 => 0000000002150F00	SSDT => NtTerminateProcess				
[LyShark]	序号 => [45]	当前地址 => FFFF8044A70E580	起源地址 => 0000000000000000	相对地址 => 0000000004E93700	SSDT => NtSetEventBoostPriority				
[LyShark]	序号 => [46]	当前地址 => FFFF8044A4ADE20	起源地址 => 0000000000000000	相对地址 => 00000000028F1F00	SSDT => NtReadFileScatter				
[LyShark]	序号 => [47]	当前地址 => FFFF8044A3CCA30	起源地址 => FFFF8044A3CCA30	相对地址 => 0000000001A7E200	SSDT => NtOpenThreadTokenEx				

本书作者：王瑞 (LyShark)

作者邮箱：me@lyshark.com

作者博客：<https://lyshark.cnblogs.com>

团队首页：www.lyshark.com