# Tssk.sys分析报告

该驱动主要用于防止各类已知rootkit驱动的攻击，有很多恢复系统函数的操作，具体过程不详细介绍，看idb即可。

目录

[Tssk.sys分析报告 1](#_Toc433295717)

[一、 基础库 2](#_Toc433295718)

## 基础库

1.1 由符号链接名获取设备名

BOOLEAN GetDeviceNameBySymbolicName(PUNICODE\_STRING SymbolicName,PUNICODE\_STRING DeviceName)

{

OBJECT\_ATTRIBUTES ObjectAttributes;

UNICODE\_STRING LinkTarget = {0};

HANDLE LinkHandle = NULL;

NTSTATUS Status;

ULONG ReturnedLength = 0;

BOOLEAN Ret = FALSE;

if(!SymbolicName || !DeviceName)

return FALSE;

InitializeObjectAttributes(&ObjectAttributes,SymbolicName,OBJ\_CASE\_INSENSITIVE | OBJ\_KERNEL\_HANDLE,NULL,NULL);

ZwOpenSymbolicLinkObject(&LinkHandle,GENERIC\_READ,&ObjectAttributes);

if(LinkHandle)

{

if(ZwQuerySymbolicLinkObject(LinkHandle, &LinkTarget, &ReturnedLength) == STATUS\_BUFFER\_TOO\_SMALL)

{

if(ReturnedLength)

{

ReturnedLength += 2;

LinkTarget.Buffer = (PWCH)ExAllocatePool(NonPagedPool,ReturnedLength);

if(LinkTarget.Buffer)

{

RtlZeroMemory(LinkTarget.Buffer,ReturnedLength);

LinkTarget.Length = 0;

LinkTarget.MaximumLength = ReturnedLength;

if(ZwQuerySymbolicLinkObject(LinkHandle, &LinkTarget, &ReturnedLength) == STATUS\_SUCCESS)

Ret = TRUE;

}

}

}

if(LinkHandle)

{

ZwClose(LinkHandle);

LinkHandle = NULL;

}

if(Ret)

\*DeviceName = LinkTarget;

}

if(LinkTarget.Buffer)

ExFreePool(LinkTarget.Buffer);

}

1.2 遍历所有硬盘分区

BOOLEAN GetHarddiskPartitioNumber(PDEVICE\_OBJECT DeviceObject, PULONG OutPartitionNum)

{

KEVENT Event;

IO\_STATUS\_BLOCK Ios;

NTSTATUS Status;

BOOLEAN Ret = FALSE;

if(!DeviceObject || !OutPartitionNum)

return FALSE;

KeInitializeEvent(&Event);

PVOID Buf = ExAllocatePool(NonPagedPool,0x2000);

RtlZeroMemory(&Buf,0x2000);

PIRP Irp = IoBuildDeviceIoControlRequest(0x70050,DeviceObject,NULL,0,Buf,0x2000,FALSE,&Event,&Ios);

if(Irp)

{

Status = IoCallDriver(DeviceObject,Irp);

if(Status == STATUS\_PENDING)

{

KeWaitForSingleObject(&Event,Suspended,KernelMode,FALSE,NULL);

Status = Ios.Status;

}

if(NT\_SUCCESS(Status))

{

\*OutPartitionNum = \*((ULONG\*)Buf+1);

Ret = TRUE;

}

}

if(Buf)

ExFreePool(Buf);

return Ret;

}

void EnumPartition()

{

if(KeGetCurrentIrql() != PASSIVE\_LEVEL)

return;

PCONFIGURATION\_INFORMATION config = IoGetConfigurationInformation();

for(int DiskIndex=0;DiskIndex < config->DiskCount;DiskIndex++)

{

WCHAR Disk[260];

UNICODE\_STRING UDisk;

PFILE\_OBJECT FileObj = NULL;

PDEVICE\_OBJECT DevObj = NULL;

vsnwprintfW(Disk,260,"\\Device\\Harddisk%d\\Partition0",DiskIndex);

RtlInitUnicodeString(&UDisk,Disk);

if(NT\_SUCCESS(IoGetDeviceObjectPointer(&UDisk,FILE\_READ\_ATTRIBUTES,&FileObj,&DevObj))

{

if(MmIsAddressValid(FileObj->DeviceObject) && MmIsAddressValid(FileObj->DeviceObject->DriverObject))

{

ULONG PartitionNumber;

if(GetHarddiskPartitionNumber(DevObj,&PartitionNumber))

{

for(int PartitionIndex=0;PartitionIndex < PartitionNumber;PartitionIndex++)

{

WCHAR Partition[260];

UNICODE\_STRING UPartition;

PFILE\_OBJECT FileObjx = NULL;

PDEVICE\_OBJECT DevObjx = NULL;

vsnwprintfW(Partition,260,"\\Device\\Harddisk%d\\Partition%d",DiskIndex,PartitionIndex);

RtlInitUnicodeString(&UPartition,Partition);

if(NT\_SUCCESS(IoGetDeviceObjectPointer(&UPartition,FILE\_READ\_ATTRIBUTES,&FileObjx,&DevObjx))

{

if(MmIsAddressValid(FileObjx->DeviceObject) && MmIsAddressValid(FileObjx->DeviceObject->DriverObject))

{

}

}

}

}

}

}

}

}

1.3 由进程对象获取文件对象

Xp之前：

由PsGetProcessSectionBaseAddress得到SectionBaseAddress在EPROCESS中的偏移，前4字节为SectionObject偏移

由PEPROCESS得到\_SECTION.ControlArea.FilePointer

Xp之后：

PsReferenceProcessFilePointer得到