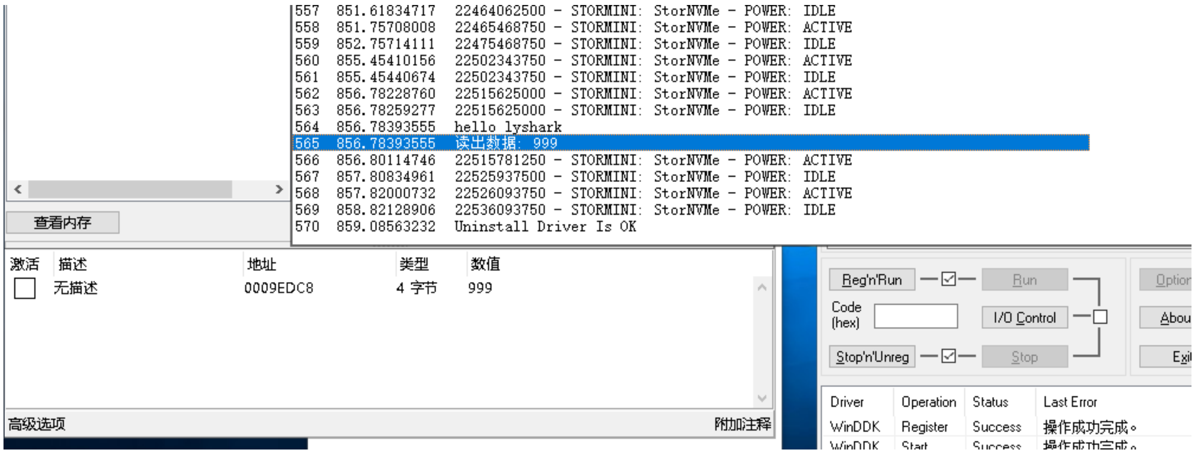
内核中读写内存的方式有很多，典型的读写方式有CR3读写，MDL读写，以及今天要给大家分享的内存拷贝实现读写，拷贝读写的核心是使用`MmCopyVirtualMemory`这个内核API函数实现，通过调用该函数即可很容易的实现内存的拷贝读写。

封装`KeReadProcessMemory()`内存读取。

```
#include <ntifs.h>
#include <windef.h>
#include <stdlib.h>

NTKERNELAPI NTSTATUS PsLookupProcessByProcessId(HANDLE ProcessId, PEPROCESS *Process);
NTKERNELAPI CHAR* PsGetProcessImageFileName(PEPROCESS Process);
NTSTATUS NTAPI MmCopyVirtualMemory(PEPROCESS SourceProcess, PVOID SourceAddress, PEPROCESS TargetProcess, PVOID TargetAddress, SIZE_T BufferSize, KPROCESSOR_MODE PreviousMode, PSIZE_T ReturnSize);

// 定义全局EProcess结构
PEPROCESS Global_Peprocess = NULL;

// 普通Ke内存读取
NTSTATUS KeReadProcessMemory(PVOID SourceAddress, PVOID TargetAddress, SIZE_T Size)
{
    __try
    {
        PEPROCESS TargetProcess = PsGetCurrentProcess();
        SIZE_T Result;
        if (NT_SUCCESS(MmCopyVirtualMemory(Global_Peprocess, SourceAddress, TargetProcess, TargetAddress, Size, KernelMode, &Result)))
            return STATUS_SUCCESS;
        else
            return STATUS_ACCESS_DENIED;
    }
    __except (EXCEPTION_EXECUTE_HANDLER)
    {
        return STATUS_ACCESS_DENIED;
    }
    return STATUS_ACCESS_DENIED;
}

VOID UnDriver(PDRIVER_OBJECT driver)
{
    DbgPrint("Uninstall Driver Is OK \n");
}

// By:lyshark.cnblogs.com
NTSTATUS DriverEntry(IN PDRIVER_OBJECT Driver, PUNICODE_STRING RegistryPath)
{
    DbgPrint("hello lyshark \n");

    // 根据PID打开进程
    DWORD PID = 6672;
    NTSTATUS nt = PsLookupProcessByProcessId((HANDLE)PID, &Global_Peprocess);

    DWORD ref_value = 0;

    // 将地址处读取4字节到ref_value中
    NTSTATUS read_nt = KeReadProcessMemory((PVOID)0x0009EDC8, &ref_value, 4);

    DbgPrint("读出数据: %d \n", ref_value);

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

读取效果如下:



```
557  851.61834717  22464062500 - STORMINI: StorNVMe - POWER: IDLE
558  851.75708008  22465468750 - STORMINI: StorNVMe - POWER: ACTIVE
559  852.75714111  22475468750 - STORMINI: StorNVMe - POWER: IDLE
560  855.45410156  22502343750 - STORMINI: StorNVMe - POWER: ACTIVE
561  855.45440674  22502343750 - STORMINI: StorNVMe - POWER: IDLE
562  856.78228760  22515625000 - STORMINI: StorNVMe - POWER: ACTIVE
563  856.78259277  22515625000 - STORMINI: StorNVMe - POWER: IDLE
564  856.78393555  hello lyshark
565  856.78393555  读出数据: 999
566  856.80114746  22515781250 - STORMINI: StorNVMe - POWER: ACTIVE
567  857.80834961  22525937500 - STORMINI: StorNVMe - POWER: IDLE
568  857.82000732  22526093750 - STORMINI: StorNVMe - POWER: ACTIVE
569  858.82128906  22536093750 - STORMINI: StorNVMe - POWER: IDLE
570  859.08563232  Uninstall Driver Is OK
```

激活	描述	地址	类型	数值
☐	无描述	0009EDC8	4 字节	999

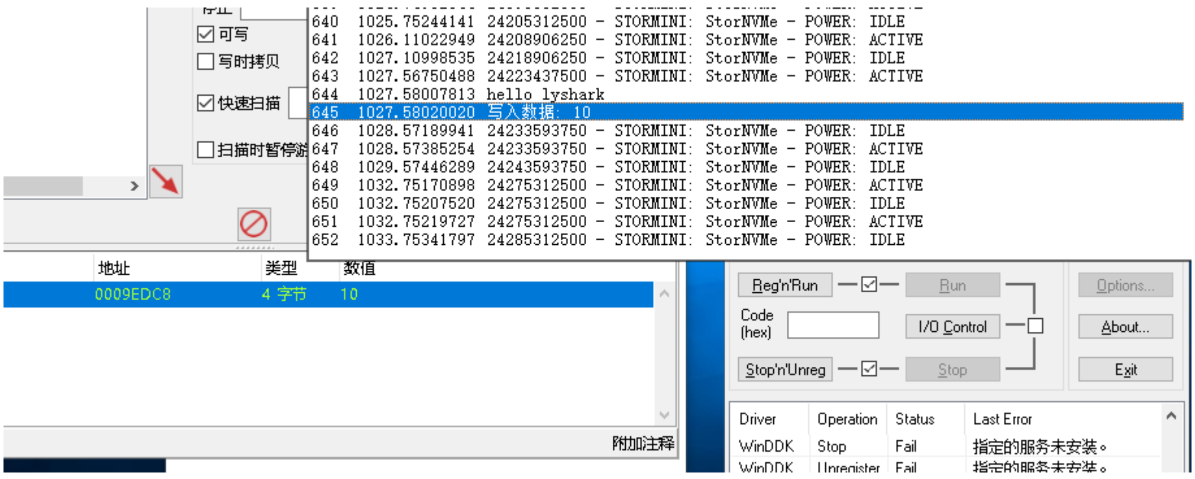
Driver	Operation	Status	Last Error
WinDDK	Register	Success	操作成功完成。
WinDDK	Start	Success	操作成功完成。

封装 `KeWriteProcessMemory()` 内存读取。

```
#include <ntifs.h>
#include <windef.h>
#include <stdlib.h>

NTKERNELAPI NTSTATUS PsLookupProcessByProcessId(HANDLE ProcessId, PEPROCESS
*Process);
NTKERNELAPI CHAR* PsGetProcessImageFileName(PEPROCESS Process);
NTSTATUS NTAPI MmCopyVirtualMemory(PEPROCESS SourceProcess, PVOID SourceAddress,
PEPROCESS TargetProcess, PVOID TargetAddress, SIZE_T BufferSize, KPROCESSOR_MODE
PreviousMode, PSIZE_T ReturnSize);

// 定义全局EProcess结构
PEPROCESS Global_Peprocess = NULL;


// 普通Ke内存写入
NTSTATUS KeWriteProcessMemory(PVOID SourceAddress, PVOID TargetAddress, SIZE_T
Size)
{
    PEPROCESS SourceProcess = PsGetCurrentProcess();
    PEPROCESS TargetProcess = Global_Peprocess;
    SIZE_T Result;

    if (NT_SUCCESS(MmCopyVirtualMemory(SourceProcess, SourceAddress,
TargetProcess, TargetAddress, Size, KernelMode, &Result)))
        return STATUS_SUCCESS;
```

```c
    else
        return STATUS_ACCESS_DENIED;
}

VOID UnDriver(PDRIVER_OBJECT driver)
{
    DbgPrint("Uninstall Driver Is OK \n");
}

NTSTATUS DriverEntry(IN PDRIVER_OBJECT Driver, PUNICODE_STRING RegistryPath)
{
    DbgPrint("hello lyshark \n");

    // 根据PID打开进程
    DWORD PID = 6672;
    NTSTATUS nt = PsLookupProcessByProcessId((HANDLE)PID, &Global_Peprocess);

    DWORD ref_value = 10;

    // 将地址处写出4字节
    NTSTATUS read_nt = KeWriteProcessMemory((PVOID)0x0009EDC8, &ref_value, 4);

    DbgPrint("写入数据: %d \n", ref_value);

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

写出内存效果:



```
640  1025.75244141  24205312500 - STORMINI: StorNVMe - POWER: IDLE
641  1026.11022949  24208906250 - STORMINI: StorNVMe - POWER: ACTIVE
642  1027.10998535  24218906250 - STORMINI: StorNVMe - POWER: IDLE
643  1027.56750488  24223437500 - STORMINI: StorNVMe - POWER: ACTIVE
644  1027.58007813  hello lyshark
645  1027.58020020  写入数据: 10
646  1028.57189941  24233593750 - STORMINI: StorNVMe - POWER: IDLE
647  1028.57385254  24233593750 - STORMINI: StorNVMe - POWER: ACTIVE
648  1029.57446289  24243593750 - STORMINI: StorNVMe - POWER: IDLE
649  1032.75170898  24275312500 - STORMINI: StorNVMe - POWER: ACTIVE
650  1032.75207520  24275312500 - STORMINI: StorNVMe - POWER: IDLE
651  1032.75219727  24275312500 - STORMINI: StorNVMe - POWER: ACTIVE
652  1033.75341797  24285312500 - STORMINI: StorNVMe - POWER: IDLE
```

☑ 可写
☐ 写时拷贝
☑ 快速扫描
☐ 扫描时暂停游

地址	类型	数值
0009EDC8	4 字节	10

附加注释

Reg'n'Run ☑ Run
Code (hex)
I/O Control ☐
Stop'n'Unreg ☑ Stop
Options...
About...
Exit

Driver	Operation	Status	Last Error
WinDDK	Stop	Fail	指定的服务未安装。
WinDDK	Unregister	Fail	指定的服务未安装。

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