ОНЛАЙН-ОБРАЗОВАНИЕ

Меня хорошо слышно && видно?

Напишите в чат, если есть проблемы!

Ставьте + если все хорошо

OTUS

Программирование **Native** приложений

Таблица экспорта

Пакулов Артур

1

Таблица экспорта

Таблица экспорта

Описывается структурой _IMAGE_EXPORT_DIRECTORY

```
typedef struct _IMAGE_EXPORT_DIRECTORY {
    DWORD   Characteristics;
    DWORD   TimeDateStamp;
    WORD    MajorVersion;
    WORD    MinorVersion;
    DWORD   Name;
    DWORD   Base;
    DWORD   NumberOfFunctions;
    DWORD   NumberOfNames;
/*+0x1C*/   DWORD   AddressOfFunctions;     // RVA from base of image
    DWORD   AddressOfNames;         // RVA from base of image
    DWORD   AddressOfNameOrdinals;  // RVA from base of image
} IMAGE_EXPORT_DIRECTORY, *PIMAGE_EXPORT_DIRECTORY;
```

Алгоритм получения адреса функции по таблице экспорта I

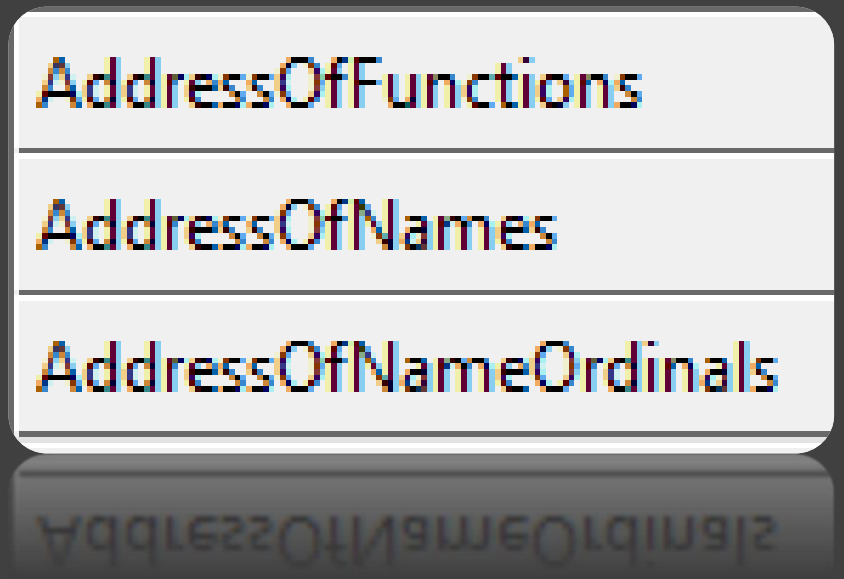
Получить указатель на _IMAGE_DOS_HEADER

По полю _IMAGE_DOS_HEADER.e_lfanew (0x3C) получить смещение до _IMAGE_NT_HEADERS

По смещению 0x78 от _IMAGE_NT_HEADERS читается смещение до таблицы экспорта

Алгоритм получения адреса функции по таблице экспорта II

Читаются три указателя на массивы



Алгоритм получения адреса функции по таблице экспорта III

Из массива имён находится нужная функция по имени (или хешу) и запоминается её индекс N

Из массива ординалов читается N-ый элемент - M

Из массива адресов функций читается M-ый элемент - address

Таблица экспорта

Итог:

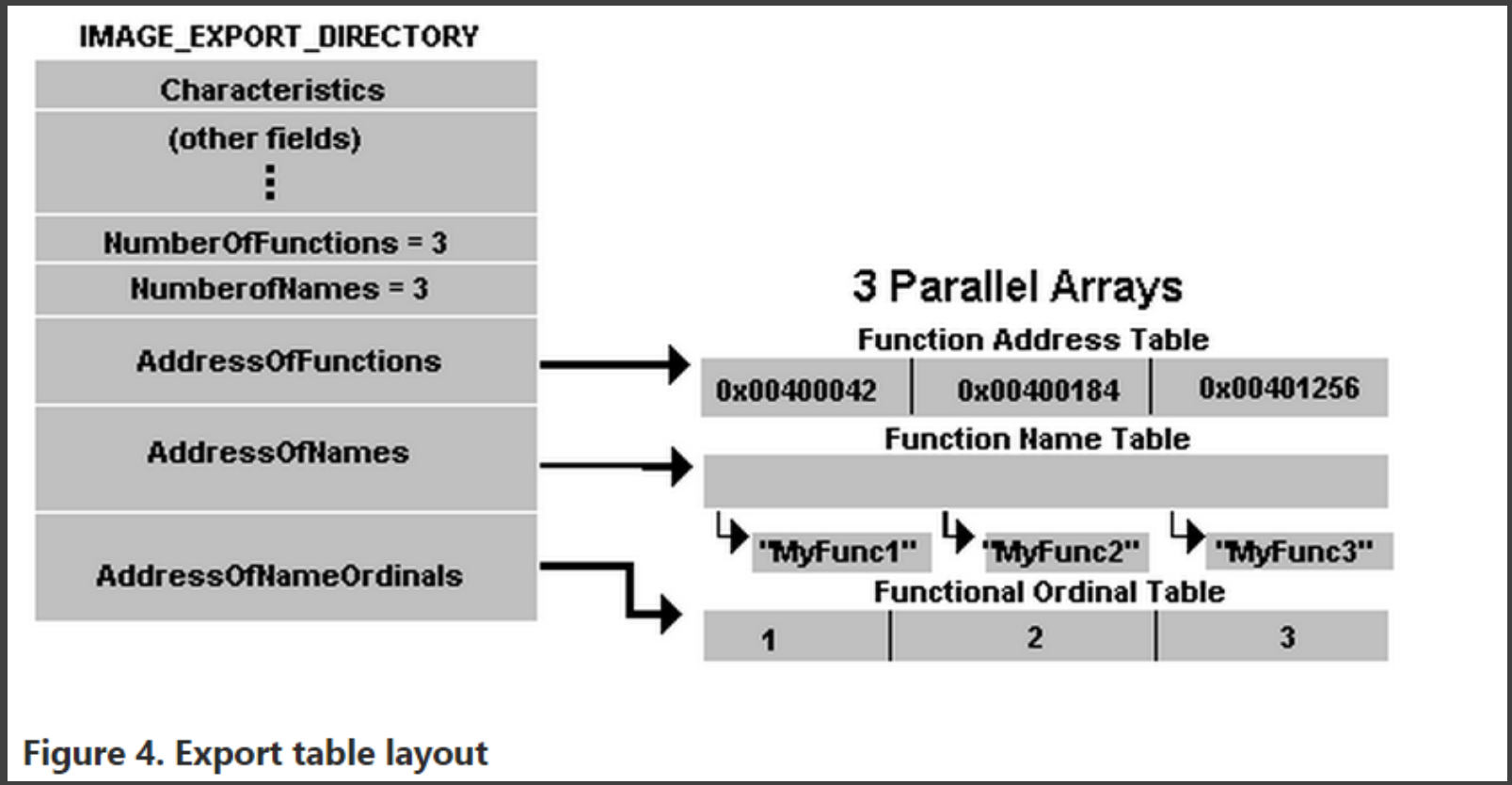


Figure 4. Export table layout

2

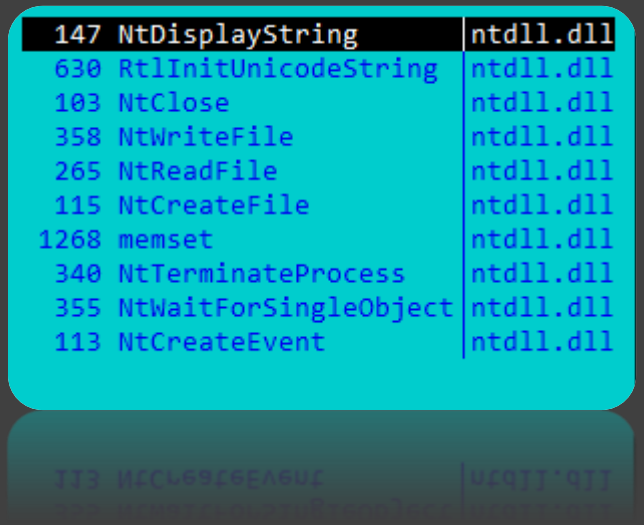
Программирование **Native** приложений

Windows Driver Kit Version 7.1.0 скачать
Native Development Kit скачать

Это программы, способные запускаться на раннем этапе загрузки Windows (до окна входа в систему)
Пример программы: chkdsk

Count of sections	2	Machine	Intel386
Symbol table	00000000[00000000]		Sat Apr 27 05:20:16 2019
Size of optional header	00E0	Magic optional header	010B
Linker version	9.00	OS version	6.01
Image version	6.01	Subsystem version	5.01
Entry point	00001560	Size of code	00000800
Size of init data	00000200	Size of uninit data	00000000
Size of image	00003000	Size of header	00000400
Base of code	00001000	Base of data	00002000
Image base	01000000	Subsystem	Native
Section alignment	00001000	File alignment	00000200
Stack	00040000/00002000	Heap	00100000/00001000
Checksum	00008E1F	Number of dirs	16

Native приложения используют только функции, экспортируемые из библиотеки ntdll.dll, которые частично документированы



147	NtDisplayString	ntdll.dll
630	RtlInitUnicodeString	ntdll.dll
103	NtClose	ntdll.dll
358	NtWriteFile	ntdll.dll
265	NtReadFile	ntdll.dll
115	NtCreateFile	ntdll.dll
1268	memset	ntdll.dll
340	NtTerminateProcess	ntdll.dll
355	NtWaitForSingleObject	ntdll.dll
113	NtCreateEvent	ntdll.dll

Недокументированные функции можно подсмотреть тут

Undocumented functions of NTDLL

NtDisplayString

```
NTSYSAPI
NTSTATUS
NTAPI

NtDisplayString(

  IN PUNICODE_STRING      String );
```

Function **NtDisplayString** display specified string in *text-mode* (typically: blue screen).
Warning: Trying to display string without previously switch to text-mode results as system hang.

- String Pointer to **UNICODE_STRING** contains string to display. Some basic control characters are implemented (like *CR*, *LF*).

Documented by:
Tomasz Nowak

Requirements:

Library: **ntdll.lib**
Privilege: **SE_TCB_PRIVILEGE**

See also:

Задача

Реализовать **Native** программу, которая сдампит **MBR**

Вспомогательные функции

Описываем функции задержки и вывода на экран

```
void Sleep(DWORD mSec)
{
    LARGE_INTEGER interval;
    interval.QuadPart = -1 * (int)(mSec * 10000);
    NtDelayExecution(FALSE, &interval);
}

void WriteLn(LPWSTR Message)
{
    UNICODE_STRING string;
    RtlInitUnicodeString(&string, Message);
    NtDisplayString(&string);
}
```

Задержка на 1 секунду (1000 мс)

```
LARGE_INTEGER interval;

interval.QuadPart = -1 * (int)(mSec * 10000);

NtDelayExecution(FALSE, &interval);
```

- Alertable If set, execution can break in a result of **NtAlertThread** call.
- DelayInterval Delay in 100-ns units. Negative value means delay relative to current.

```
10^3 * 100*10^-9 * 10^4 = 10^3 * 10^-7 * 10^4 =

= 10^0 = 1 sec
```

NtReadFile

OTUS

Reads data from an open file

```
NTSTATUS NtReadFile(
  _In_     HANDLE           FileHandle,
  _In_opt_ HANDLE           Event,
  _In_opt_ PIO_APC_ROUTINE  ApcRoutine,
  _In_opt_ PVOID            ApcContext,
  _Out_    PIO_STATUS_BLOCK IoStatusBlock,
  _Out_    PVOID            Buffer,
  _In_     ULONG            Length,
  _In_opt_ PLARGE_INTEGER   ByteOffset,
  _In_opt_ PULONG           Key
);
```

Event [in, optional]

Optionally, a handle to an event object to set to the signaled state after the read operation completes. Device and intermediate drivers should set this parameter to **NULL**.

Чтение с клавиатуры

OTUS

NtReadFile

```
RtlInitUnicodeString(&keyboard, L"\\Device\\KeyboardClass0");
InitializeObjectAttributes(&ObjectAttributes, &keyboard, OBJ_CASE_INSENSITIVE, NULL, NULL);

NtCreateFile(&hKeyBoard,
             SYNCHRONIZE | GENERIC_READ | FILE_READ_ATTRIBUTES,
             &ObjectAttributes,
             &Iosb,
             NULL,
             FILE_ATTRIBUTE_NORMAL,
             0,
             FILE_OPEN,FILE_DIRECTORY_FILE,
             NULL, 0);

InitializeObjectAttributes(&ObjectAttributes, NULL, 0, NULL, NULL);
NtCreateEvent(&hEvent, EVENT_ALL_ACCESS, &ObjectAttributes, 1, 0);

if (DumpMbr())
    WriteLn(L"Dump Created!\n");
else
    WriteLn(L"Dump Error!\n");
while (TRUE)
{
    NtReadFile(hKeyBoard, hEvent, NULL, NULL, &Iosb, &kbData, sizeof(KEYBOARD_INPUT_DATA), &ByteOffset, NULL);
    NtWaitForSingleObject(hEvent, TRUE, NULL);

    if (kbData.MakeCode == 0x01)    //ESC
    {
            break;
    }

}
```

Точка входа - NtProcessStartup

NtProcessStartup

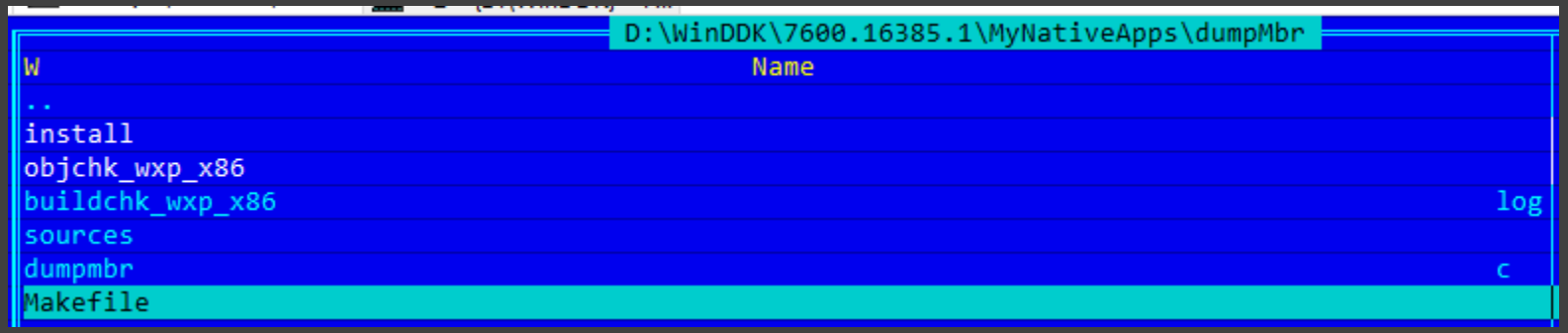
```
void NtProcessStartup(void* StartupArgument)
{
    HANDLE hKeyBoard, hEvent;
    UNICODE_STRING skull, keyboard;
    OBJECT_ATTRIBUTES ObjectAttributes;
    IO_STATUS_BLOCK Iosb;
    LARGE_INTEGER ByteOffset;
    KEYBOARD_INPUT_DATA kbData;

    RtlInitUnicodeString(&keyboard, L"\\Device\\KeyboardClass0");
    InitializeObjectAttributes(&ObjectAttributes, &keyboard, OBJ_CASE_INSENSITIVE, NULL, NULL);

    NtCreateFile(&hKeyBoard,
                 SYNCHRONIZE | GENERIC_READ | FILE_READ_ATTRIBUTES,
                 &ObjectAttributes,
                 &Iosb,
                 NULL,
                 FILE_ATTRIBUTE_NORMAL,
                 0,
                 FILE_OPEN,FILE_DIRECTORY_FILE,
                 NULL, 0);
```

Сборка

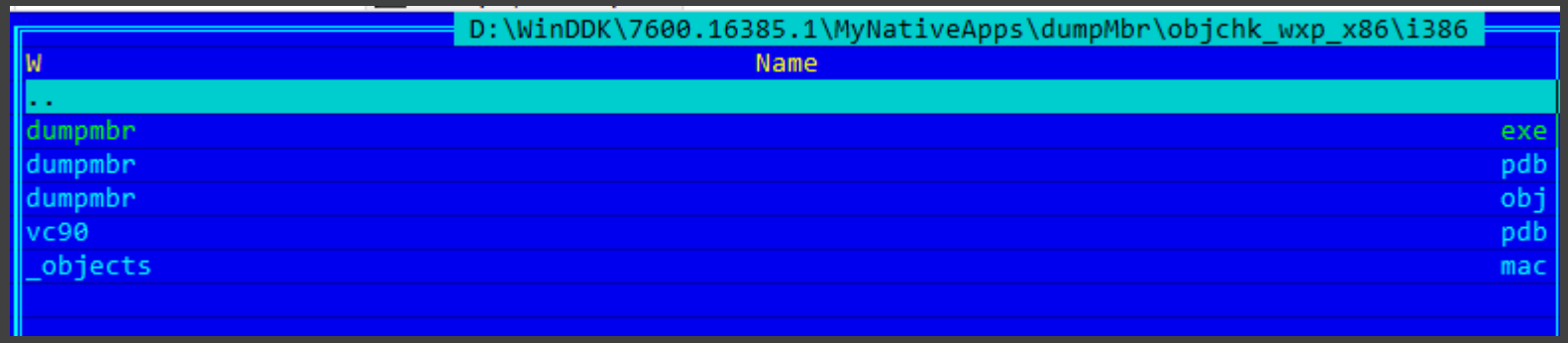
1. Установить WDK (например, на диск D)
2. Создать каталог MyNativeApps в D:\WinDDK\7600.16385.1\
3. Скопировать в него каталог с исходниками dumpMbr
4. Запустить x86 Checked Build Environment
5. Cd MyNativeApps\dumpMbr
6. Build

Сборка

Структура каталога

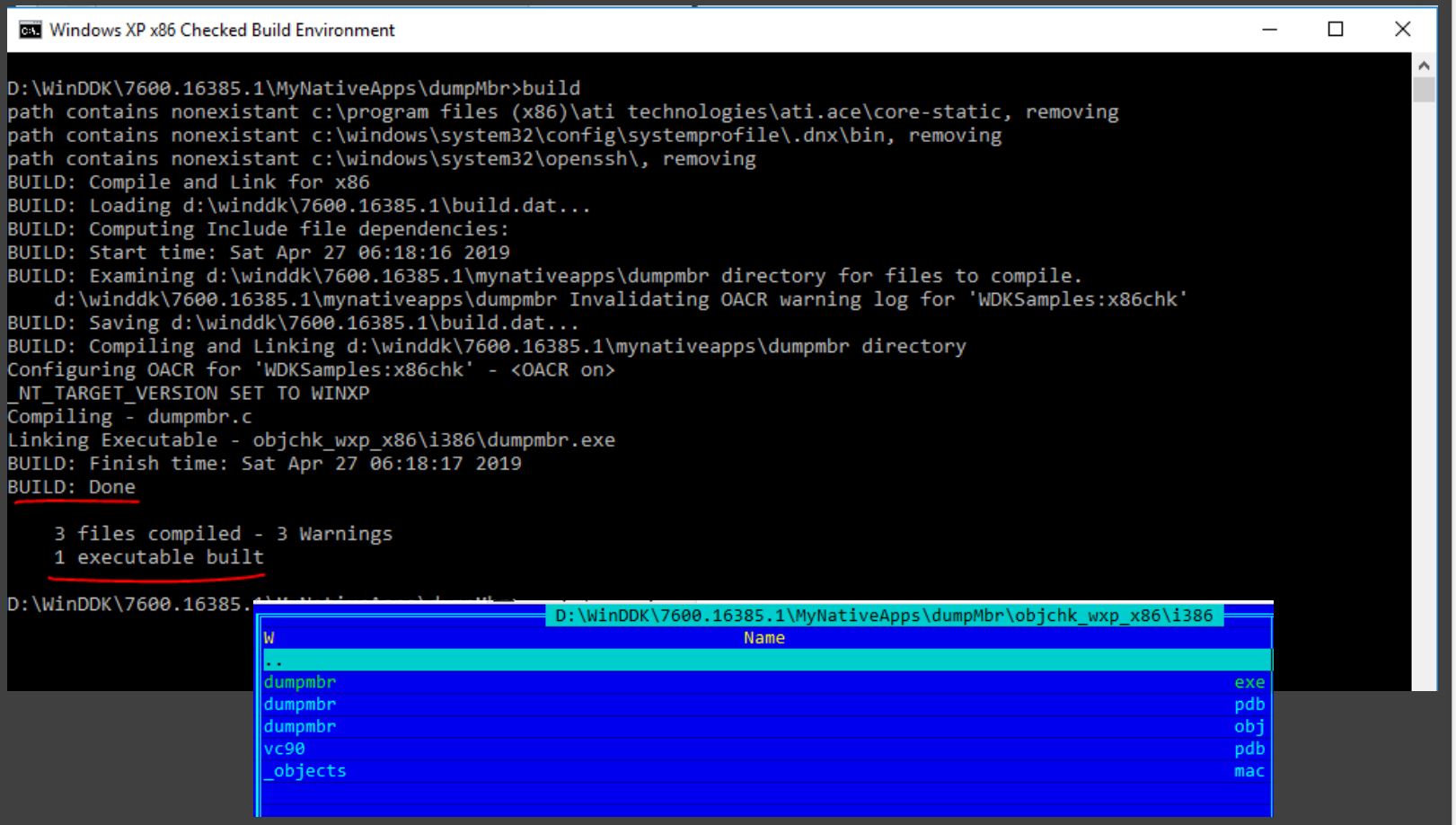


В случае успешной сборки:



Сборка

В случае успешной сборки:



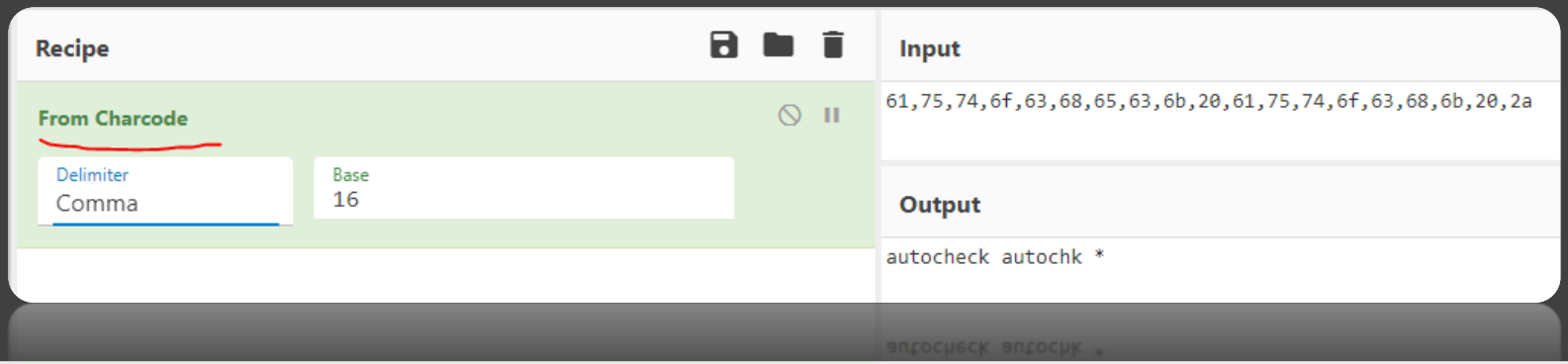
Установка

Для запуска Nativeприложения нужно его скопировать в system32 и прописать в реестре

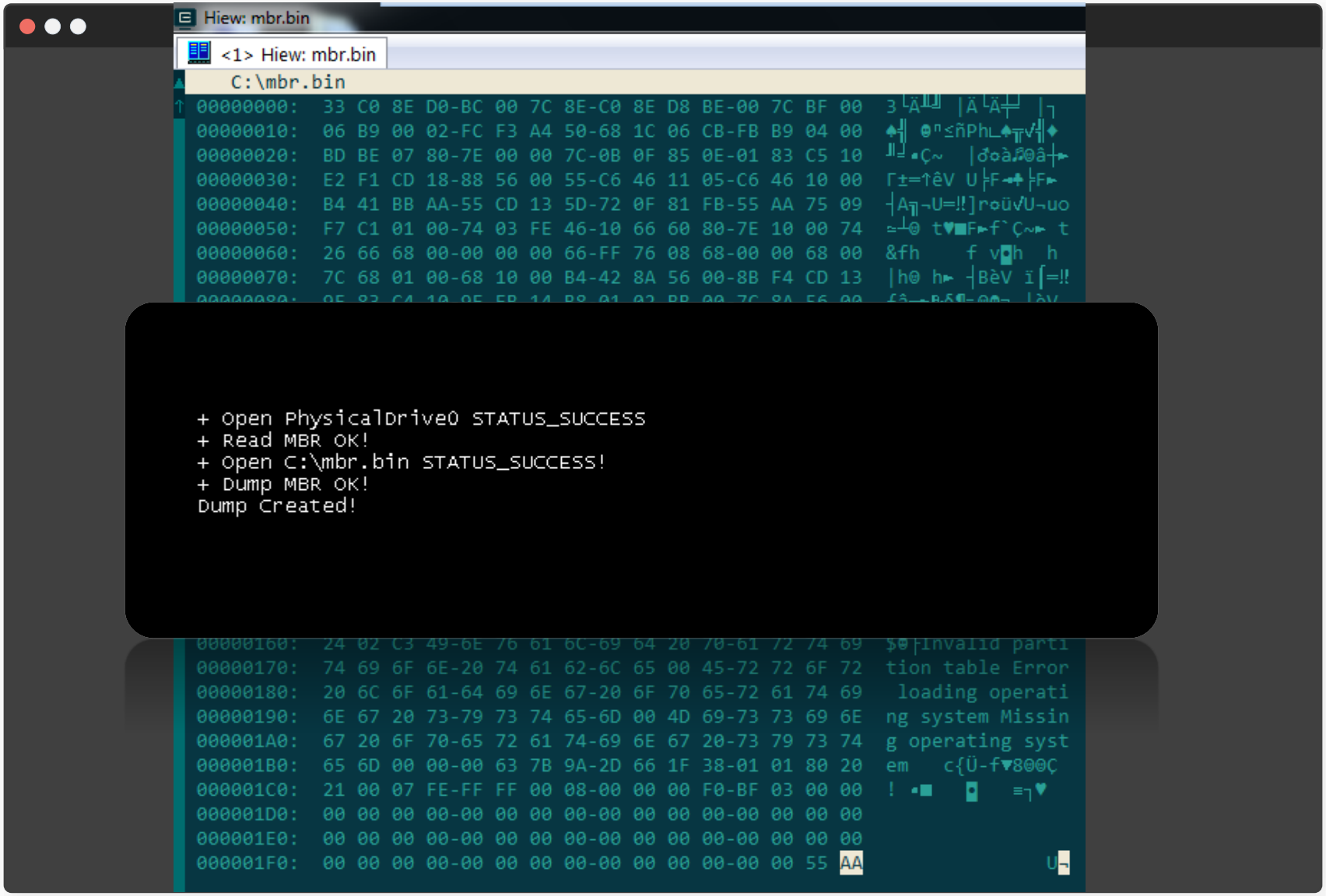
HKLM\System\CurrentControlSet\Control\Session Manager\BootExecute

После строки «Autocheck Autochk *»

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager]
"BootExecute"=hex(7):61,75,74,6f,63,68,65,63,6b,20,61,75,74,6f,63,68,6b,20,2a,\
  00,64,75,6d,70,6d,62,72,00,00
```



После перезагрузки



ДЗ

Собрать **DumpMBR** из исходников

OTUS

Вопросы???

Пакулов Артур

A.Pakulov.Otus@Gmail.com

Спасибо
за внимание!