ОНЛАЙН-ОБРАЗОВАНИЕ

Меня хорошо слышно && видно?

Напишите в чат, если есть проблемы!

Ставьте + если все хорошо

OTUS

Внутреннее устройство **Windows**

Объекты ядра

Пакулов Артур

1

Объекты ядра (ОЯ)

Что такое объект ядра?

Объект ядра представляет из себя блок памяти, выделенный ядром операционной системы и доступный только ему

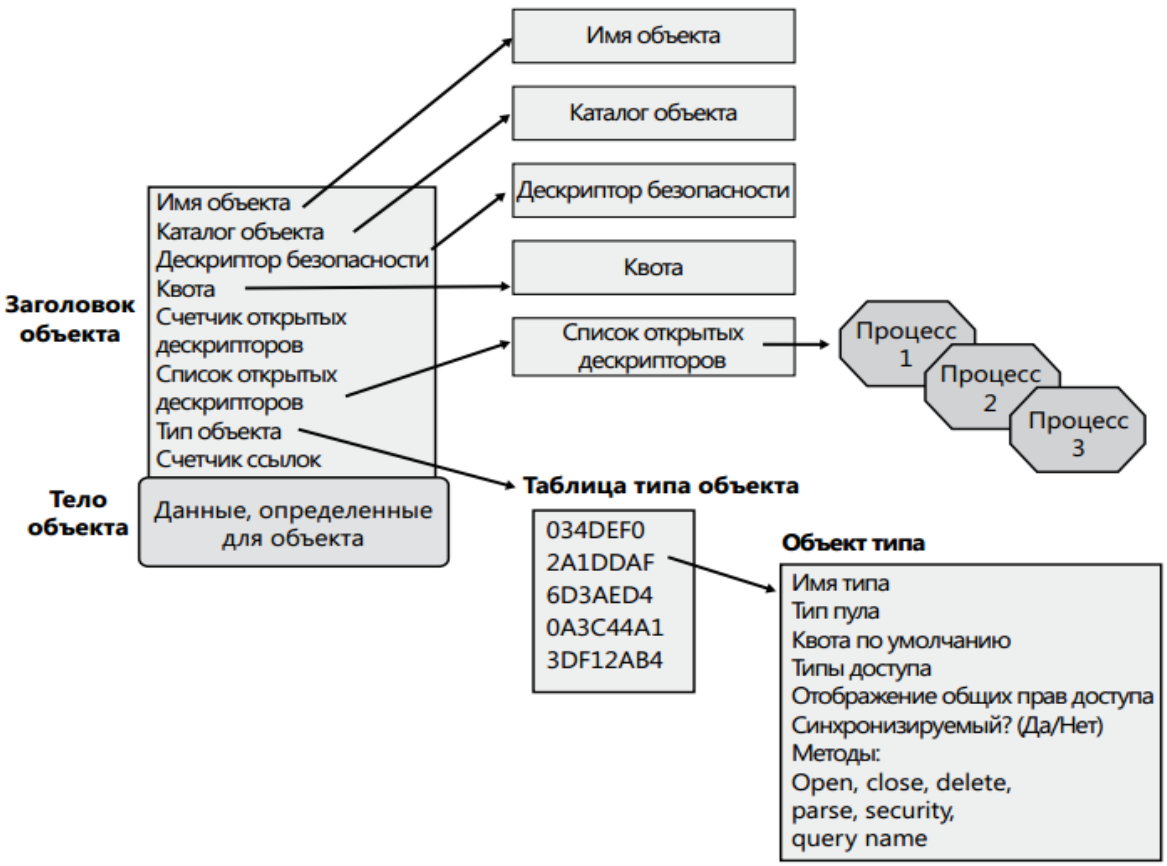
Что является объектом ядра

Пример:

Kernel object	Объект ядра	Kernel object	Объект ядра
Access token	Маркер доступа	Module	Подгружаемый модуль (DLL)
Change notification	Уведомление об изменениях на диске	Mutex	Мьютекс
I/O completion ports	Порт завершения ввода-вывода	Pipe	Канал
Event	Событие	Process	Процесс
File	Файл	Semaphore	Семафор
File mapping	Проекция файла	Socket	Сокет
Heap	Куча	Thread	Поток
Job	Задание	Timer	Ожидаемый таймер
Mailslot	Почтовый слот		

Структура объекта ядра

Общая часть + специфичная каждому объекты



Структура объекта “процесс” - _EPROCESS

Часть структуры, описывающей ОЯ «процесс»

```
      +0x030 IopmOffset       : 0x20ac
      +0x032 Iopl             : 0 ''
      +0x033 Unused           : 0 ''
      +0x034 ActiveProcessors : 0
      +0x038 KernelTime       : 0x29
      +0x03c UserTime         : 5
      +0x040 ReadyListHead    : _LIST_ENTRY [ 0x81f24c10 - 0x81f24c10 ]
      +0x048 SwapListEntry    : _SINGLE_LIST_ENTRY
      +0x04c VdmTrapcHandler  : (null)
      +0x050 ThreadListHead   : _LIST_ENTRY [ 0x8221cda0 - 0x81f60d28 ]
      +0x058 ProcessLock      : 0
      +0x05c Affinity         : 1
      +0x060 AutoAlignment    : 0y0
      +0x060 DisableBoost     : 0y0
      +0x060 DisableQuantum   : 0y0
      +0x060 ReservedFlags    : 0y00000000000000000000000000000000 (0)
      +0x060 ProcessFlags     : 0n0
      +0x064 BasePriority     : 8 ''
      +0x065 QuantumReset     : 6 ''
      +0x066 State            : 0 ''
      +0x067 ThreadSeed       : 0 ''
      +0x068 PowerState       : 0 ''
      +0x069 IdealNode        : 0 ''
      +0x06a Visited          : 0 ''
      +0x06b Flags            : _KEXECUTE_OPTIONS
      +0x06b ExecuteOptions   : 0 ''
      +0x06c StackCount       : 6
      +0x070 ProcessListEntry : _LIST_ENTRY [ 0x0 - 0x0 ]
   +0x078 ProcessLock      : _EX_PUSH_LOCK
      +0x000 Locked           : 0y0
```

Структура процесса _KPROCESS

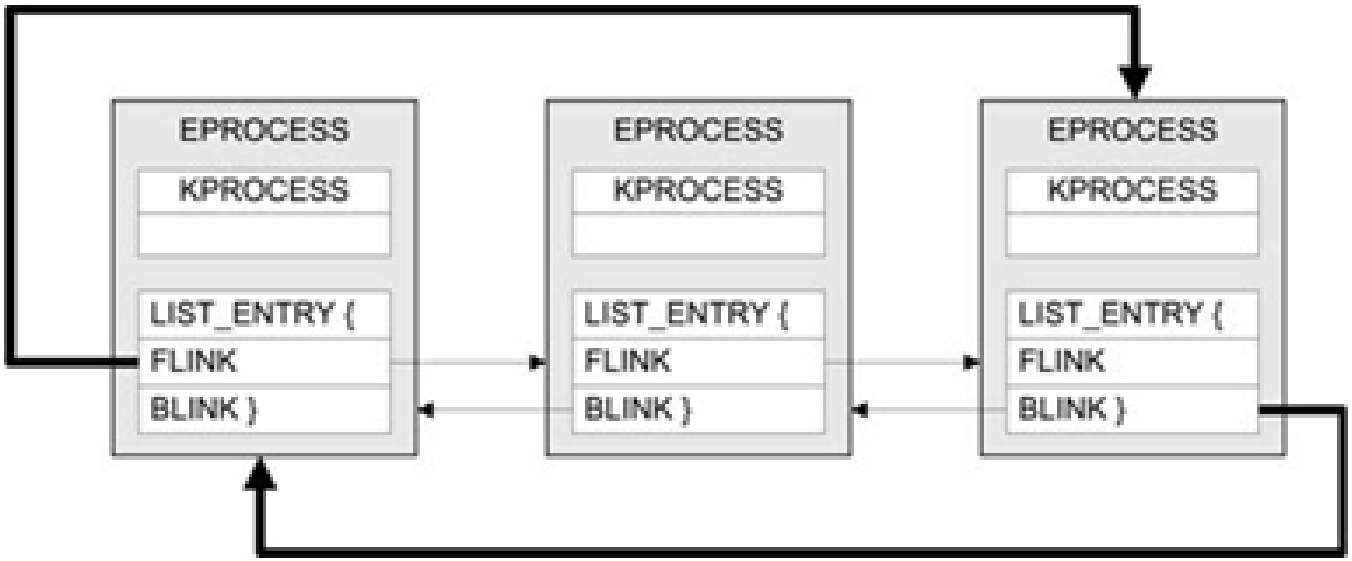
_KPROCESS – важная подструктура структуры _EPROCESS

```
nt!_KPROCESS
   +0x000 Header           : _DISPATCHER_HEADER
   +0x010 ProfileListHead  : _LIST_ENTRY
   +0x018 DirectoryTableBase : [2] Uint4B
   +0x020 LdtDescriptor    : _KGDTENTRY
   +0x028 Int21Descriptor  : _KIDTENTRY
   +0x030 IopmOffset       : Uint2B
   +0x032 Iopl             : UChar
   +0x033 Unused           : UChar
   +0x034 ActiveProcessors : Uint4B
```

DirectoryTableBase – указатель на каталог страниц

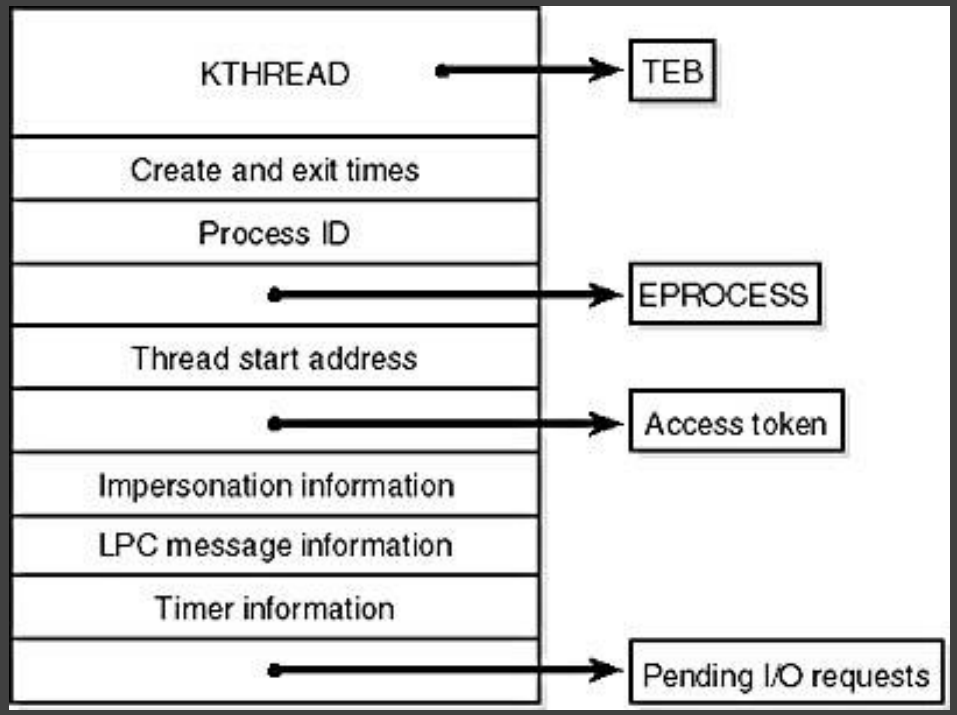
Расположение объектов “процесс” в памяти

Двусвязный кольцевой список



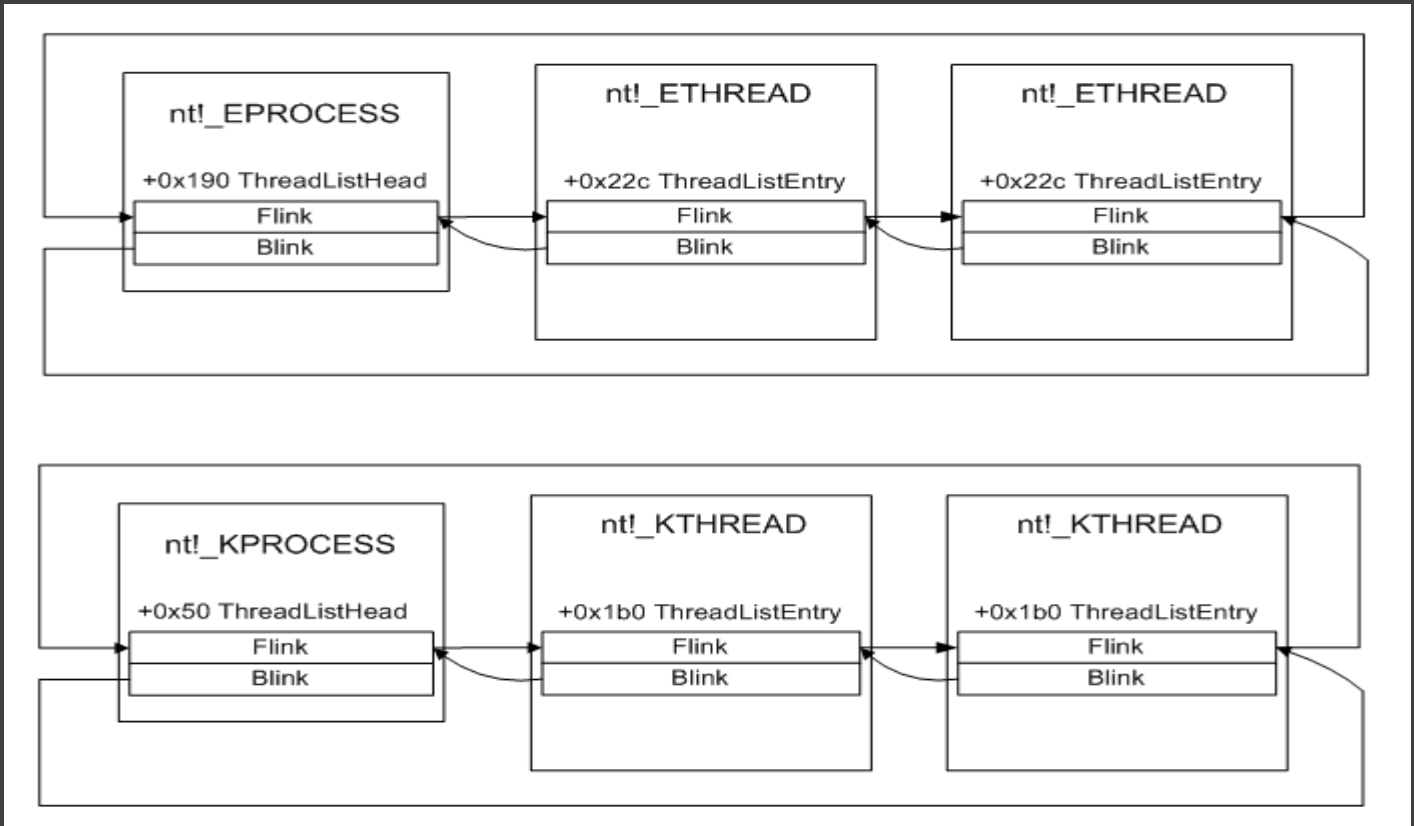
Структура потока _ETHREAD

Fs:0x124 – указатель на “текущий поток”



Расположение потоков в памяти

Список из _ETHREAD замыкается структурой _EPROCESS



Пример постоянной части структуры ОЯ

OTUS

Счётчик пользователей	Атрибуты безопасности
• Учёт использования объекта	• Контроль использования объекта

Примечания

- ✓ Если счётчик объекта ядра становится **== 0,** то ОЯ уничтожается системой
- ✓ Процессы управляют ОЯ с помощью **API** функций
- ✓ Фукции, пораждающие объект ядра начинаются с **«Create»**
- ✓ Процесс должен освобождать объект с помощью функции **CloseHandle** или её подобной

Структура **SECURITY_ATTRIBUTES**

lpSecuntyDescriptor - дескриптор защиты

```
typedef struct _SECURITY_ATTRIBUTES {
  DWORD  nLength;
  LPVOID lpSecurityDescriptor;
  BOOL   bInheritHandle;
} SECURITY_ATTRIBUTES, *PSECURITY_ATTRIBUTES, *LPSECURITY_ATTRIBUTES;
```

Структура **SECURITY_ATTRIBUTES**

lpSecuntyDescriptor - дескриптор защиты

```
typedef struct _SECURITY_ATTRIBUTES {
  DWORD  nLength;
  LPVOID lpSecurityDescriptor;
  BOOL   bInheritHandle;
} SECURITY_ATTRIBUTES, *PSECURITY_ATTRIBUTES, *LPSECURITY_ATTRIBUTES;
```

Если lpSecuntyDescriptor == NULL, то только создатель объекта и любой член группы администраторов получают к нему полный доступ

Таблица дескрипторов объектов ядра

OTUS

При инициализации процесса система создаёт в нём таблицу описателей

- Каждый ОЯ в системе имеет запись в этой таблице
- Хендл (дескриптор, описатель) – индекс ОЯ в этой таблице
- Первый индекс имеет значение 4, второй 8 и тд
- У каждого процесса своя таблица, которая заполняется по мере создания ОЯ
- В EPROCESS есть поле ObjectTable – указывающий на таблицу

Расшаривание ОЯ между процессами

OTUS

- Передача по «наследству»
- Именнованные объекты
- Копирование элемента таблицы в свою таблицу дескрипторов

DuplicateHandle

Дескриптор скопированного ОЯ будет другим

```
BOOL DuplicateHandle(
  HANDLE   hSourceProcessHandle,
  HANDLE   hSourceHandle,
  HANDLE   hTargetProcessHandle,
  LPHANDLE lpTargetHandle,
  DWORD    dwDesiredAccess,
  BOOL     bInheritHandle,
  DWORD    dwOptions
);
```

Это часть пользовательской Win32 подсистемы

Хэндлы всех работающих процессов находятся в сервере подсистемы Win32 (csrss.exe)

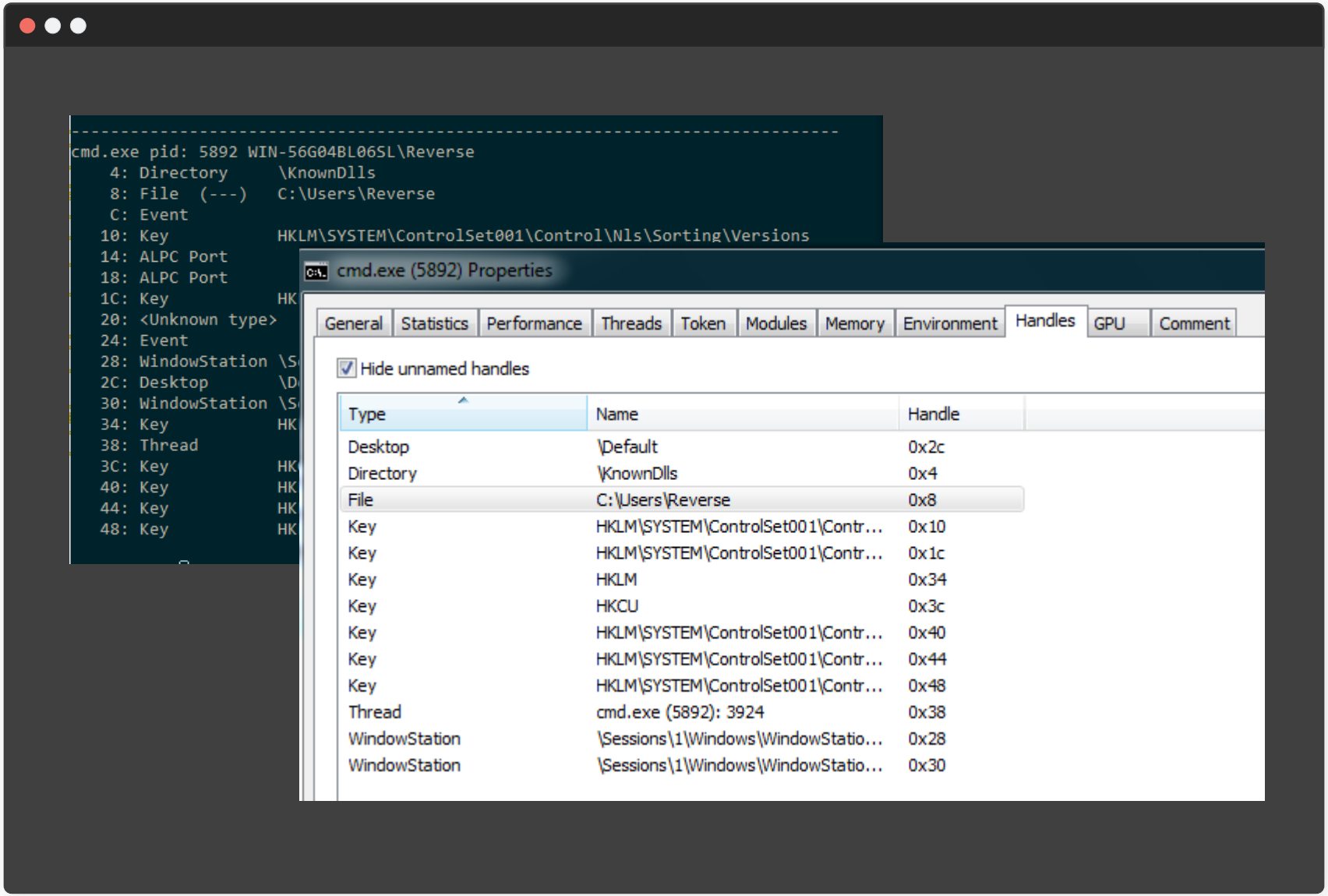
Просмотр хендлов

Утилита «handle» от sysinternals

```
------------------------------------------------------------------------------
cmd.exe pid: 5892 WIN-56G04BL06SL\Reverse
    4: Directory      \KnownDlls
    8: File  (---)    C:\Users\Reverse
    C: Event
   10: Key            HKLM\SYSTEM\ControlSet001\Control\Nls\Sorting\Versions
   14: ALPC Port
   18: ALPC Port
   1C: Key            HKLM\SYSTEM\ControlSet001\Control\Session Manager
   20: <Unknown type>
   24: Event
   28: WindowStation \Sessions\1\Windows\WindowStations\WinSta0
   2C: Desktop       \Default
   30: WindowStation \Sessions\1\Windows\WindowStations\WinSta0
   34: Key            HKLM
   38: Thread
   3C: Key            HKCU
   40: Key            HKLM\SYSTEM\ControlSet001\Control\Nls\Locale
   44: Key            HKLM\SYSTEM\ControlSet001\Control\Nls\Locale\Alternate Sorts
   48: Key            HKLM\SYSTEM\ControlSet001\Control\Nls\Language Groups
```

handle –p cmd.exe -a

Process Hacker



2

Графические объекты

Адрес массива Gdi объектов

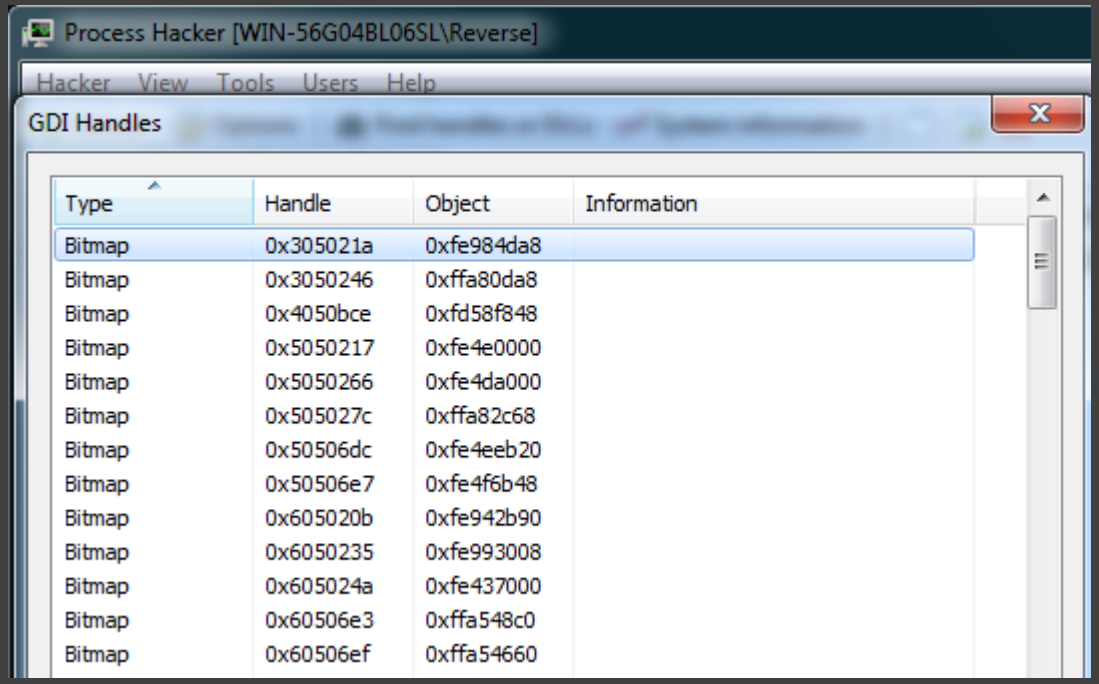
OTUS

- ✓ Таблица с GDI объектами хранится в пользовательской части процесса, в поле GdiSharedHandleTable находится указатель на неё
- ✓ Сами графические объекты хранятся в памяти ядра

```
kd> dt _PEB
ntdll!_PEB
   +0x000 InheritedAddressSpace : UChar
   +0x001 ReadImageFileExecOptions : UChar
    ..............................................
   +0x08c MaximumNumberOfHeaps : Uint4B
   +0x090 ProcessHeaps     : Ptr32 Ptr32 Void
   +0x094 GdiSharedHandleTable : Ptr32 Void
   +0x098 ProcessStarterHelper : Ptr32 Void
   +0x09c GdiDCAttributeList : Uint4B
   +0x0a0 LoaderLock       : Ptr32 _RTL_CRITICAL_SECTION
   +0x0a4 OSMajorVersion   : Uint4B
   +0x0a8 OSMinorVersion   : Uint4B
   +0x0ac OSBuildNumber    : Uint2B
   +0x0ae OSCSDVersion     : Uint2B
   +0x0b0 OSPlatformId     : Uint4B
   +0x0b4 ImageSubsystem   : Uint4B
   +0x0b8 ImageSubsystemMajorVersion : Uint4B
   +0x0bc ImageSubsystemMinorVersion : Uint4B
   +0x0c0 ActiveProcessAffinityMask : Uint4B
```

Просмотр графических объектов

Process Hacker



OTUS

Вопросы???

Пакулов Артур

A.Pakulov.Otus@Gmail.com

Спасибо
за внимание!