

Работа 4. Управление памятью

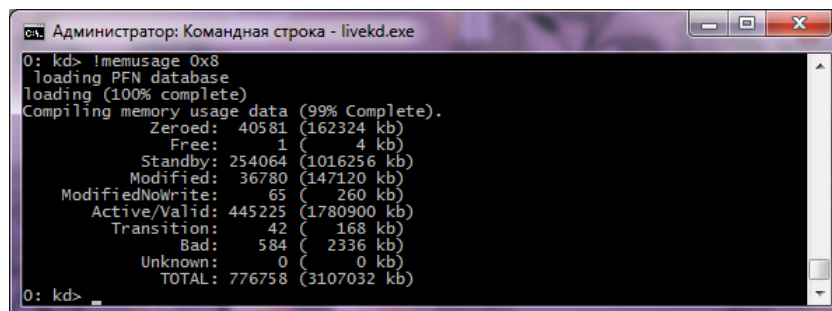
Цель работы: исследовать механизмы управления виртуальной памятью Win32.

Задание 4.1. Исследовать процесс трансляции виртуальных адресов в 32-разрядной операционной системе Windows.

Указания к выполнению.

1. Запустите командную строку от имени администратора, перейдите в каталог **c:\Tools\LiveKD** и запустите утилиту *LiveKd.exe*. В процессе запуска программы Вам могут быть заданы некоторые вопросы, касающиеся настроек запуска, отвечайте на них утвердительно.

2. Выполните команду *!memusage 0x8*, которая отображает размеры списков страниц виртуальной памяти, находящихся в различных состояниях. Ознакомьтесь с результатами выполнения и запишите их в отчет.



```
Администратор: Командная строка - livekd.exe
0: kd> !memusage 0x8
Loading PFN database
Loading (100% complete)
Compiling memory usage data (99% Complete).
Zeroed: 40581 (162324 kb)
Free: 1 (4 kb)
Standby: 254064 (1016256 kb)
Modified: 36780 (147120 kb)
ModifiedNoWrite: 65 (260 kb)
Active/Valid: 445225 (1780900 kb)
Transition: 42 (168 kb)
Bad: 584 (2336 kb)
Unknown: 0 (0 kb)
TOTAL: 776758 (3107032 kb)
0: kd>
```

3. Выполните команду *!vm*, которая выводит базовые сведения об управлении памятью, доступные через соответствующие счетчики производительности. В том числе команда *!vm* выводит список процессов с их идентификаторами и объемом занимаемой памяти. Ознакомьтесь с результатами выполнения и запишите их в отчет. Выберите один из процессов для исследования, например, в дальнейшем мы будем изучать процесс *chrome.exe* с идентификатором 1918.

```

0: kd> !vm

*** Virtual Memory Usage ***
Physical Memory:      777342 ( 3109368 Kb)
Page File: \??\C:\pagefile.sys
Current: 7409084 Kb Free Space: 5635512 Kb
Minimum: 3109368 Kb Maximum: 9328104 Kb
Available Pages: 294318 ( 1177272 Kb)
ResAvail Pages: 635754 ( 2543016 Kb)
Locked IO Pages: 0 ( 0 Kb)
Free System PTEs: 106812 ( 427248 Kb)
Modified Pages: 15770 ( 63080 Kb)
Modified PF Pages: 15675 ( 62700 Kb)
NonPagedPool Usage: 30117 ( 120468 Kb)
NonPagedPool Max: 523068 ( 2092272 Kb)
PagedPool 0 Usage: 47600 ( 190400 Kb)
PagedPool 1 Usage: 7073 ( 28292 Kb)
PagedPool 2 Usage: 4245 ( 16980 Kb)
PagedPool 3 Usage: 4184 ( 16736 Kb)
PagedPool 4 Usage: 4225 ( 16900 Kb)
PagedPool Usage: 67327 ( 269308 Kb)
PagedPool Maximum: 523264 ( 2093056 Kb)

***** 6 pool allocations have failed *****

Session Commit: 23067 ( 92268 Kb)
Shared Commit: 117984 ( 471936 Kb)
Special Pool: 0 ( 0 Kb)
Shared Process: 4728 ( 18912 Kb)
PagedPool Commit: 67377 ( 269508 Kb)
Driver Commit: 7113 ( 28452 Kb)
Committed pages: 957888 ( 3831552 Kb)
Commit limit: 2629174 ( 10516696 Kb)

***** 1 commit requests have failed *****

Total Private: 686267 ( 2745068 Kb)
19ec avp.exe 98513 ( 394052 Kb)
275c WINWORD.EXE 71430 ( 285720 Kb)
2080 AcroRd32.exe 54842 ( 219368 Kb)
1918 chrome.exe 45403 ( 181612 Kb)
046c svchost.exe 41844 ( 167376 Kb)

```

4. Чтобы просмотреть информацию о процессах, существует команда `!process 0 0`, которая выводит информацию о всех процессах. Ознакомьтесь с результатами выполнения, запишите их в отчет, выберите процесс для дальнейшего изучения его виртуального адресного пространства.

5. Выполните команду `!process <идентификатор процесса> 0`, в этом случае Вам отобразится краткая информация о выбранном процессе, например, адрес каталога страниц процесса **DirBase**, который понадобится для перехода в контекст процесса, также можно просмотреть адрес **PEB** процесса, имя файла процесса **Image**, адрес таблицы дескрипторов **ObjectTable**, размер таблицы дескрипторов **HandleCount**, адрес **VadRoot** корня дерева регионов виртуального адресного пространства процесса и т.д. Далее приведен пример вывода краткой информации о процессе *chrome.exe* с идентификатором *1918*. Повторите действия для выбранного процесса, ознакомьтесь с результатами и запишите их в отчет.

```
0: kd> !process 1918 1
Searching for Process with Cid == 1918
Cid handle table at 8d401098 with 2040 entries in use

PROCESS b0290d40 SessionId: 1 Cid: 1918 Peb: 7ffd4000 ParentCid: 162c
DirBase: bd59f1c0 ObjectTable: e6fc9b78 HandleCount: 1368.
Image: chrome.exe
VadRoot b030ff78 Vads 576 Clone 0 Private 39837. Modified 654781. Locked 116
72.
DeviceMap b0606c18
Token c2e6c030
ElapsedTime 1 Day 00:12:41.738
UserTime 00:00:10.810
KernelTime 00:00:28.158
QuotaPoolUsage[PagedPool] 483652
QuotaPoolUsage[NonPagedPool] 39072
Working Set Sizes (now,min,max) (38748, 50, 345) (154992KB, 200KB, 1380KB)
PeakWorkingSetSize 45484
VirtualSize 446 Mb
PeakVirtualSize 482 Mb
PageFaultCount 1183161
MemoryPriority BACKGROUND
BasePriority 8
CommitCharge 45403

0: kd>
```

6. Далее проанализируем регионы виртуального адресного пространства выбранного процесса. Для этого необходимо вызвать команду `!vad <VadRoot>`. В результате выполнения этой команды мы получим краткую информацию о всех регионах виртуального адресного пространства выбранного процесса, в первом столбце выводится а именно, виртуальный адрес описания региона (в скобках уровень иерархии в дереве), стартовый виртуальный адрес начала региона, адрес конца региона, статус региона и параметры защиты страниц. Список регионов может быть слишком большой, поэтому после запуска команды рекомендуется сразу приостановить вывод информации путем нажатия комбинации `Ctrl+Break`. Далее приведен пример вывода для процесса `chrome.exe` с идентификатором `1918`, у которого значение `VadRoot` было равно `b030aa78`. Обратим свое внимание на регион виртуальных адресов с адресом описания VAD `86825008`, в который спроецирован файл `\Program Files\Google\Chrome\chrome.exe`. Для получения подробной информации о выбранном регионе виртуального адресного пространства используем команду `!vad <VAD> 1`. Повторите действия для выбранного процесса, ознакомьтесь с результатами и запишите их в отчет. Далее мы попробуем определить, какие страницы физической памяти поставлены в соответствие этому региону, и попробуем прочитать из них информацию.

```

Администратор: Командная строка - livekd.exe

Total VADs: 8, average level: 2, maximum depth: 3
0: kd> !vad b030ff78
VAD      level      start      end      commit
8b996d78 ( 7)      10       1f       0 Mapped    READWRITE    Page
file-backed section
8babc9d8 ( 8)      20       20       1 Private    READWRITE    Page
86a9cea0 ( 6)      30       33       0 Mapped    READONLY     Page
file-backed section
8648b258 ( 7)      40       41       0 Mapped    READONLY     Page
file-backed section
88cec630 ( 5)      50       50       1 Private    READWRITE    \Win
b036f0f8 ( 8)      60       c6       0 Mapped    READONLY     \Win
dows\System32\locale.nls
862fe1a8 ( 7)      d0       1cf      256 Private    READWRITE    Page
86a4eb78 ( 8)     1d0      1d0      1 Private    READWRITE    Page
864aead0 ( 6)     1e0      1e0      0 Mapped    READONLY     Page
file-backed section
8648b9e8 ( 8)     1f0      1f0      0 Mapped    READWRITE    Page
file-backed section
86a50b50 ( 7)     200      200      0 Mapped    READWRITE    Page
file-backed section
897f1008 ( 8)     210      211      0 Mapped    READONLY     Page
file-backed section
86825008 ( 4)     220      367     11 Mapped    EXECUTE_WRITECOPY \Pro
gram Files\Google\Chrome\Application\chrome.exe
88c38ea8 ( 7)     370      375      0 Mapped    READONLY     Page
file-backed section
8645fe38 ( 6)     380      380      0 Mapped    READWRITE    Page
file-backed section
888d67d8 ( 8)     390      390      0 Mapped    READONLY     \Win

```

```

Администратор: Командная строка - livekd.exe

0: kd> !vad 86825008 1
VAD @ 86825008
  Start VPN      220  End VPN      367  Control Area  88b89890
  FirstProtoPte e0f67038 LastPte ffffffff Commit Charge  b (11.)
  Secured.Flink  0      Blink      0      Banked/Extend 0
  File Offset    0
  ImageMap ViewShare EXECUTE_WRITECOPY

ControlArea @ 88b89890
  Segment e0f67008 Flink 00000000 Blink 88abd58c
  Section Ref 8 Pfn Ref 86 Mapped Views 8
  User Ref 10 WaitForDel 0 Flush Count 0
  File Object 86892ab0 ModWriteCount 0 System Views ffff
  WritableRefs 80000015
  Flags (40a0) Image File Accessed

  \Program Files\Google\Chrome\Application\chrome.exe

Segment @ e0f67008
  ControlArea 88b89890 BasedAddress 00220000
  Total Ptes 148
  Segment Size 148000 Committed 0
  Image Commit 9 Image Info e0f67a78
  ProtoPtes e0f67038
  Flags (820000) ProtectionMask

Reload command: .reload chrome.exe=00220000,148000
0: kd>

```

7. Чтобы получить доступ в виртуальное адресное пространство выбранного процесса необходимо, чтобы регистр CR3 указывал на каталог таблицы страниц этого процесса. Для решения этой проблемы мы воспользуемся командой перезагрузки контекста процессора в режиме отладчика *.context*. Вызовем эту команду без параметров, чтобы снова проверить текущее значение базового адреса таблицы страниц, а затем вызовем команду *.context <DirBase >*. В нашем случае это *.context bd59f1c0*, затем снова проверим контекст процессора *.context*. Повторите действия для выбранного процесса, ознакомьтесь с результатами и запишите их в отчет.

```
Администратор: Командная строка - livekd.exe
0: kd> .context
User-mode page directory base is bd59f1c0
0: kd> .context bcad1a40
0: kd> .context
User-mode page directory base is bcad1a40
0: kd> .context bd59f1c0
0: kd> .context
User-mode page directory base is bd59f1c0
0: kd>
```

8. После смены контекста можно смело выполнять доступ в виртуальную и физическую память процесса. Для начала необходимо преобразовать виртуальный адрес ячейки в физический адрес. Для этого необходимо обратиться в каталог страниц с помощью команды *!pte <виртуальный адрес>*, в нашем случае это *!pte <виртуальный адрес начала региона>*. Теперь мы имеем физический адрес стартовой страницы процесса, в которую операционной системой было выполнено проецирование файла *\Program Files\Google\Chrome\chrome.exe*. Для рассматриваемого примера это адрес *pfn a0f3b*. Повторите действия для выбранного процесса, ознакомьтесь с результатами и запишите их в отчет.

```
Администратор: Командная строка - livekd.exe
0: kd> !pte 220*0x1000
                                VA 00220000
PDE at C0600008                PTE at C0001100
contains 0000000030E46867      contains 80000000A0F3B005
pfn 30e46        ---DA---UWV    pfn a0f3b        -----UR-V
0: kd>
```

9. Теперь прочитаем ячейки оперативной памяти по адресу *pfn*, т.к. по этому адресу был спроецирован исполнимый файл, то в первых байтах должна располагаться специальная сигнатура **MZ**. Чтение физической памяти можно выполнить с помощью команды *!dc*. Повторите действия для выбранного процесса, ознакомьтесь с результатами и запишите их в отчет. Удостоверимся, что в начале страницы действительно размещена сигнатура **MZ**, что подтверждает правильность выполненного нами преобразования виртуального адреса первой страницы процесса в физический адрес.

```
Администратор: Командная строка - livekd.exe
0: kd> !dc a0f3b*0x1000
#a0f3b000 00905a4d 00000003 00000004 0000ffff MZ.....
#a0f3b010 000000b8 00000000 00000040 00000000 .....@.....
#a0f3b020 00000000 00000000 00000000 00000000 .....
#a0f3b030 00000000 00000000 00000000 00000118 .....
#a0f3b040 0eba1f0e cd09b400 4c01b821 685421cd .....!.!.!Th
#a0f3b050 70207369 72676f72 63206d61 6f6e6e61 is program canno
#a0f3b060 65622074 6e757220 206e6920 20534f44 t be run in DOS
#a0f3b070 65646f6d 0a0d0d2e 00000024 00000000 mode....$.
0: kd>
```

10. Подготовьте итоговый отчет с развернутыми выводами по заданию.

Задание 4.2. Исследовать виртуальное адресное пространство процесса.

Указания к выполнению.

1. Создайте консольное приложение с меню (каждая выполняемая функция и/или операция должна быть доступна по отдельному пункту меню), которое выполняет:

- получение информации о вычислительной системе (функция Win32 API – **GetSystemInfo**);
- определение статуса виртуальной памяти (функция Win32 API – **GlobalMemoryStatus**);
- определение состояния конкретного участка памяти по заданному с клавиатуры адресу (функция Win32 API – **VirtualQuery**);
- раздельное резервирование региона и передачу ему физической памяти в автоматическом режиме и в режиме ввода адреса начала региона (функция Win32 API – **VirtualAlloc**, **VirtualFree**);
- одновременное резервирование региона и передача ему физической памяти в автоматическом режиме и в режиме ввода адреса начала региона (функция Win32 API – **VirtualAlloc**, **VirtualFree**);
- запись данных в ячейки памяти по заданным с клавиатуры адресам;
- установку защиты доступа для заданного (с клавиатуры) региона памяти и ее проверку (функция Win32 API – **VirtualProtect**).

2. Запустите приложение и проверьте его работоспособность на нескольких наборах вводимых данных. Запротоколируйте результаты в отчет. Дайте свои комментарии в отчете относительно выполнения функций Win32 API.

3. Перезапустите приложение.

4. Запустите командную строку от имени администратора, перейдите в каталог **c:\Tools\LiveKD** и запустите утилиту *LiveKd.exe*.

5. Определите в *LiveKd* с помощью команды *!process* идентификатор процесса приложения, параметры виртуальной памяти, в том числе адрес **VadRoot** корня дерева регионов виртуального адресного пространства. Запротоколируйте результаты в отчет.

6. В дальнейшем после выполнения каждого пункта меню делайте обновление снимка для *LiveKd* (нажимайте *Ctrl+Break* и затем 'y') и выполняйте анализ виртуального адресного пространства процесса с помощью команды *!vad*. В случае выполнения в приложении записи в выделенные ячейки памяти данных выполните преобразование виртуальных адресов ячеек в физические (команда *!pte*) и затем вывод содержимого ячеек физической памяти на экран (команда *!dc*). Протоколируйте результаты в отчет с комментариями изменений и раскрытием их взаимосвязи с выполненными инструкциями приложения.

7. Подготовьте итоговый отчет с развернутыми выводами по заданию.

Задание 4.3. Использование проецируемых файлов для обмена данными между процессами.

Указания к выполнению.

1. Создайте два консольных приложения с меню (каждая выполняемая функция и/или операция должна быть доступна по отдельному пункту меню), которые выполняют:

- приложение-писатель создает проецируемый файл (функции Win32 API – **CreateFile, CreateFileMapping**), проецирует фрагмент файла в память (функции Win32 API – **MapViewOfFile, UnmapViewOfFile**), осуществляет ввод данных с клавиатуры и их запись в спроецированный файл;
- приложение-читатель открывает проецируемый файл (функция Win32 API – **OpenFileMapping**), проецирует фрагмент файла в память (функции Win32 API – **MapViewOfFile, UnmapViewOfFile**), считывает содержимое из спроецированного файла и отображает на экран.

2. Запустите приложения и проверьте обмен данных между процессами, удостоверьтесь в надлежащем выполнении задания. Запротоколируйте результаты в отчет. Дайте свои комментарии в отчете относительно выполнения функций Win32 API.

3. Перезапустите разработанные приложения.

4. Запустите командную строку от имени администратора, перейдите в каталог **c:\Tools\LiveKD** и запустите утилиту *LiveKd.exe*.

5. Определите в *LiveKd* с помощью команды *!process* идентификаторы процессов приложений. Запротоколируйте результаты в отчет.

6. В дальнейшем после выполнения каждого пункта меню делайте обновление снимка для *LiveKd* (нажимайте *Ctrl+Break* и затем *'y'*), получайте информацию с помощью команды *!handle* об объектах процессов и выполняйте анализ виртуальных адресных пространств процессов с помощью команды *!vad*. В случае выполнения в приложении записи в выделенные ячейки памяти данных выполните преобразование виртуальных адресов ячеек в физические (команда *!pte*) и затем вывод содержимого ячеек физической памяти на экран (команда *!dc*). Протоколируйте результаты в отчет с комментариями

изменений и раскрытием их взаимосвязи с выполненными инструкциями приложения.

7. Подготовьте итоговый отчет с развернутыми выводами по заданию.