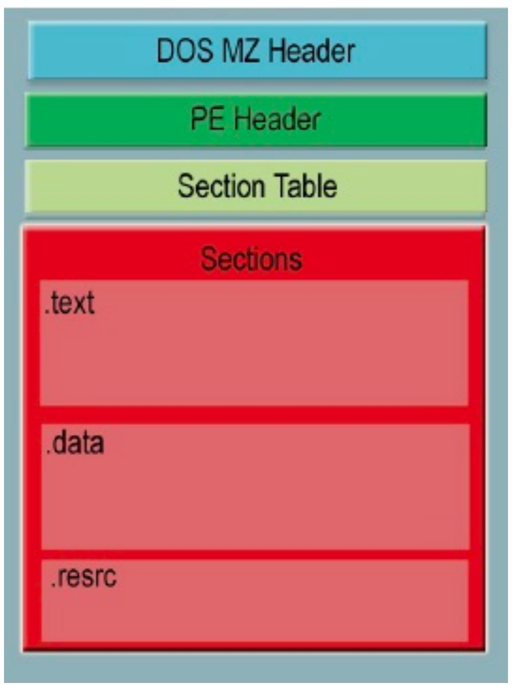
Class 4

CSC 471/583 Advanced Topics in Computer Security
Modern Malware Analysis
PE Format

Si Chen (schen@wcupa.edu)

Portable Executable (PE) file

- A Portable Executable (**PE**) **file** is the standard binary **file** format for an **Executable (.exe) or DLL** under Windows NT, Windows 95, and Win32.
- Derived from COFF (Common Object File Format) in UNIX platform, and it is not really "portable".

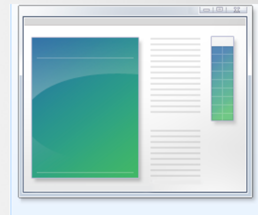


Now here is the kicker. Even though this specification is spelled out by Microsoft, compilers/linkers chose to ignore some parts of it.

To make things even worse, the Microsoft loader doesn't enforce a good portion of this specification and instead makes assumptions if things start getting weird.

So even though the spec outlined here says a particular field is supposed to hold a certain value, the compiler/linker or **even a malicious actor could put whatever they want in there and the program will likely still run**...

Portable Executable (PE) file

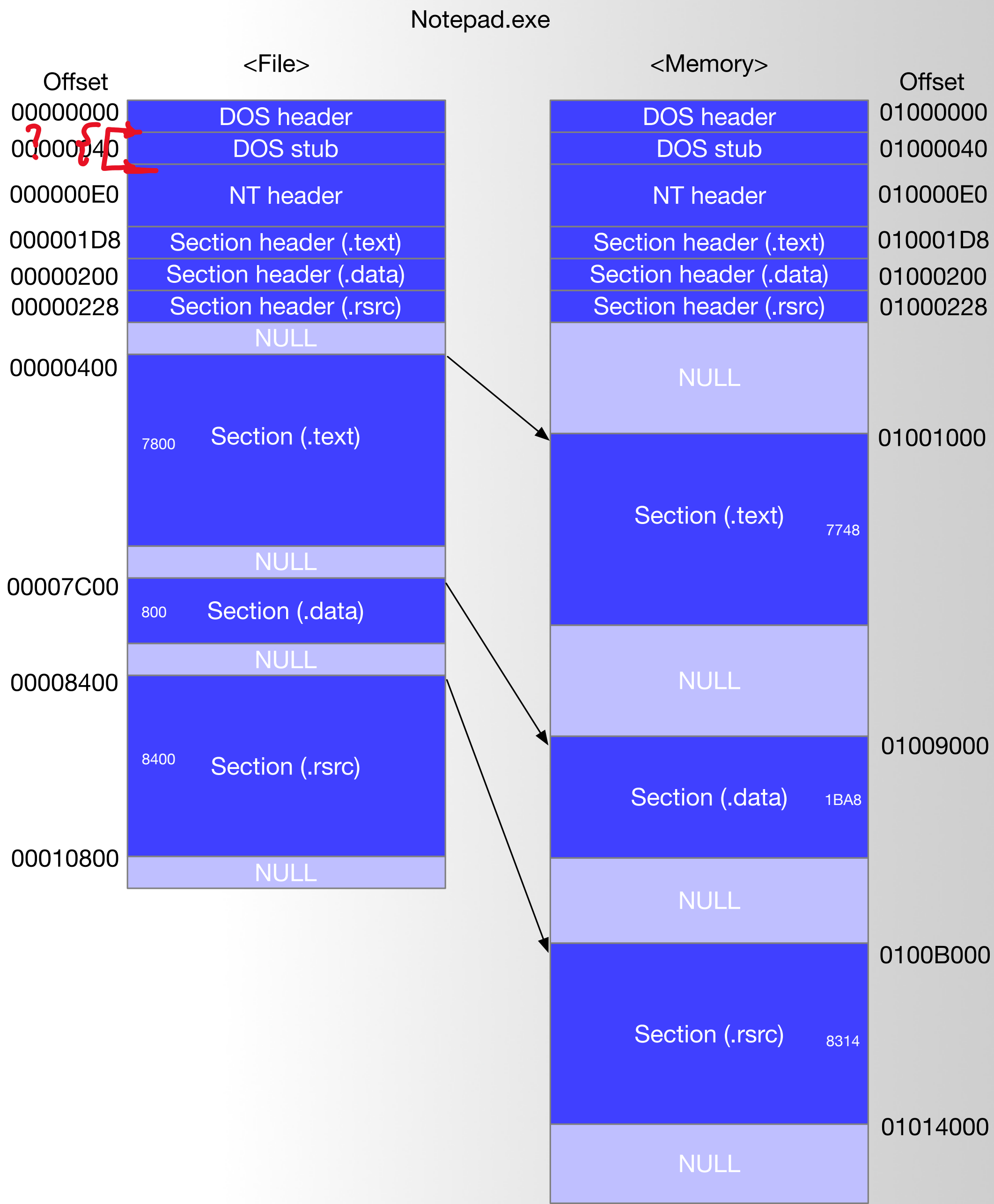


- PE formatted files include:
  - .exe, .scr (executable)
  - .dll, .ocx, .cpl, drv (library)
  - .sys, .vxd (driver files)
  - .obj (objective file)
- All PE formatted files can be executed, except obj file.
  - .exe, .scr can be directly executed inside Shell (explorer.exe)
  - others can be executed by other program/service
- **PE refers to 32 bit** executable file, or **PE32**. **64 bit** executable file is named as **PE+ or PE32+.** (Note that it is not PE64).

PE Example – Notepad.exe

```
00000000  4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00  MZÉ.........  ..
00000010  B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00  ╕.......@.......
00000020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000030  00 00 00 00 00 00 00 00 00 00 00 00 E8 00 00 00  ............Φ...
00000040  0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68  ..║..┤.═!╕.L═!Th
00000050  69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F  is.program.canno
00000060  74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20  t.be.run.in.DOS.
00000070  6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00  mode....$.......
00000080  A5 6D 16 9B E1 0C 78 C8 E1 0C 78 C8 E1 0C 78 C8  Ñm.¢ß.x╚ß.x╚ß.x╚
00000090  1B 2F 38 C8 E0 0C 78 C8 E1 0C 78 C8 E0 0C 78 C8  ./8╚α.x╚ß.x╚α.x╚
000000A0  1B 2F 61 C8 F2 0C 78 C8 E1 0C 79 C8 23 0C 78 C8  ./a╚≥.x╚ß.y╚#.x╚
000000B0  76 2F 3D C8 E0 0C 78 C8 3B 2F 64 C8 F2 0C 78 C8  v/=╚α.x╚;/d╚≥.x╚
000000C0  1B 2F 45 C8 E0 0C 78 C8 52 69 63 68 E1 0C 78 C8  ./E╚α.x╚Richß.x╚
000000D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000000E0  00 00 00 00 00 00 00 00 50 45 00 00 4C 01 03 00  ........PE..L...
000000F0  0D 84 7D 3B 00 00 00 00 00 00 00 00 E0 00 0F 01  .ä};........α...
00000100  0B 01 07 00 00 6E 00 00 00 A6 00 00 00 00 00 00  .....n...ª......
00000110  E0 6A 00 00 00 10 00 00 00 80 00 00 00 00 00 01  αj.......Ç......
00000120  00 10 00 00 00 02 00 00 05 00 01 00 05 00 01 00  ................
00000130  04 00 00 00 00 00 00 00 00 30 01 00 00 04 00 00  .........0......
00000140  55 D8 01 00 02 00 00 80 00 00 04 00 00 10 01 00  U╪.....Ç........
00000150  00 00 10 00 00 10 00 00 00 00 00 00 10 00 00 00  ................
00000160  00 00 00 00 00 00 00 00 20 6D 00 00 C8 00 00 00  .........m..╚...
00000170  00 A0 00 00 48 89 00 00 00 00 00 00 00 00 00 00  .á..Hë..........
00000180  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000190  40 13 00 00 1C 00 00 00 00 00 00 00 00 00 00 00  @...............
```

Load PE file (Notepad.exe) into Memory



DOS Header

```
struct DOS_Header
 {
// short is 2 bytes, long is 4 bytes
     char signature[2] = { 'M', 'Z' };
     short lastsize;
     short nblocks;
     short nreloc;
     short hdrsize;
     short minalloc;
     short maxalloc;
     void *ss; // 2 byte value
     void *sp; // 2 byte value
     short checksum;
     void *ip; // 2 byte value
     void *cs; // 2 byte value
     short relocpos;
     short noverlay;
     short reserved1[4];
     short oem_id;
     short oem_info;
     short reserved2[10];
     long  e_lfanew; // Offset to the 'PE\0\0' signature relative to the beginning of the file
 }
```

The first 2 letters are **always** the letters "**MZ**", the initials of Mark Zbikowski, who created the first linker for DOS. To some people, the first few bytes in a file that determine the type of file are called the "**magic number**,"

DOS Header

```
long  e_lfanew;
```

long → 32 bit → ? Byte

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	4D	5A	90	00	03	00	00	00	04	00	00	00	FF	FF	00	00	MZ..........ÿÿ..
00000010	B8	00	00	00	00	00	00	00	40	00	00	00	00	00	00	00	¸.......@.......
00000020	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000030	00	00	00	00	00	00	00	00	00	00	00	00	E0	00	00	00	............à...

E0 00 00 00

value for e_lfanew → ?

DOS stub

```
00000040  0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68  ..º..´.Í!¸.LÍ!Th
00000050  69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F  is program canno
00000060  74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20  t be run in DOS
00000070  6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00  mode....$.......
00000080  EC 85 5B A1 A8 E4 35 F2 A8 E4 35 F2 A8 E4 35 F2  ì…[¡¨ä5ò¨ä5ò¨ä5ò
00000090  6B EB 3A F2 A9 E4 35 F2 6B EB 55 F2 A9 E4 35 F2  kë:ò©ä5òkëUò©ä5ò
000000A0  6B EB 68 F2 BB E4 35 F2 A8 E4 34 F2 63 E4 35 F2  këhò»ä5ò¨ä4òcä5ò
000000B0  6B EB 6B F2 A9 E4 35 F2 6B EB 6A F2 BF E4 35 F2  këkò©ä5òkëjò¿ä5ò
000000C0  6B EB 6F F2 A9 E4 35 F2 52 69 63 68 A8 E4 35 F2  këoò©ä5òRich¨ä5ò
000000D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

https://virtualconsoles.com/online-emulators/dos/

```
C:\>notepad.exe
This program cannot be run in DOS mode.
```

NT Header

IMAGE_NT_HEADERS32 structure

12/04/2018 • 2 minutes to read

Represents the PE header format.

Syntax

C++ Copy

```
typedef struct _IMAGE_NT_HEADERS {
  DWORD                   Signature;
  IMAGE_FILE_HEADER       FileHeader;
  IMAGE_OPTIONAL_HEADER32 OptionalHeader;
} IMAGE_NT_HEADERS32, *PIMAGE_NT_HEADERS32;
```

Members

`Signature`

A 4-byte signature identifying the file as a PE image. The bytes are "PE\0\0".

`FileHeader`

An IMAGE_FILE_HEADER structure that specifies the file header.

`OptionalHeader`

An IMAGE_OPTIONAL_HEADER structure that specifies the optional file header.

https://docs.microsoft.com/en-us/windows/desktop/api/winnt/

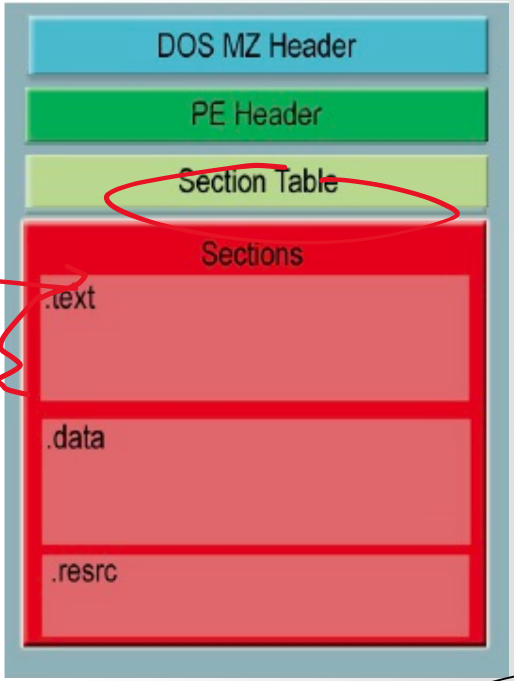
NT Header

Dos

NOTEPAD.EXE

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
000000E0	50	45	00	00	4C	01	03	00	A3	C3	B0	4A	00	00	00	00	PE..L...£Ã°J....
000000F0	00	00	00	00	E0	00	0F	01	0B	01	07	0A	00	78	00	00	....à........x..
00000100	00	A6	00	00	00	00	00	00	9D	73	00	00	00	10	00	00	.¦.......s......
00000110	00	90	00	00	00	00	00	01	00	10	00	00	00	02	00	00	....
00000120	05	00	01	00	05	00	01	00	04	00	00	00	00	00	00	00	....
00000130	00	40	01	00	00	04	00	00	33	30	01	00	02	00	00	80	.@......30.....€
00000140	00	00	04	00	00	10	01	00	00	00	10	00	00	10	00	00	....
00000150	00	00	00	00	10	00	00	00	00	00	00	00	00	00	00	00	....
00000160	04	76	00	00	C8	00	00	00	00	B0	00	00	58	89	00	00	.v..È....°..X‰..
00000170	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000180	00	00	00	00	00	00	00	00	50	13	00	00	1C	00	00	00	........P.......
00000190	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
000001A0	00	00	00	00	00	00	00	00	A8	18	00	00	40	00	00	00	........¨...@...
000001B0	00	00	00	00	00	00	00	00	00	10	00	00	48	03	00	00	............H...
000001C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
000001D0	00	00	00	00	00	00	00	00	2E	74	65	78	74	00	00	00	.... .text...

Section Header



Name	Privilege
.code	Executable, read
.data	Non-Executable, read/write
.resource	Non-Executable, read

Section Header

IMAGE_SECTION_HEADER structure

12/04/2018 • 4 minutes to read

Represents the image section header format.

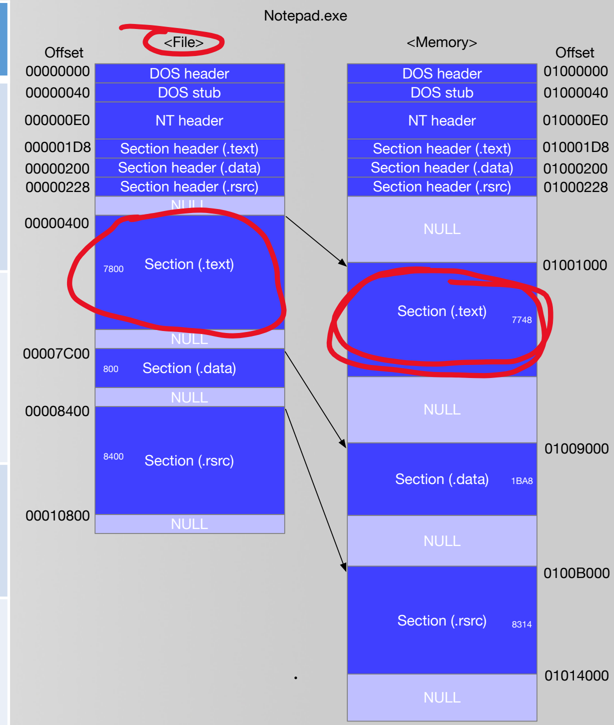
Syntax

C++ Copy

```
typedef struct _IMAGE_SECTION_HEADER {
  BYTE  Name[IMAGE_SIZEOF_SHORT_NAME];
  union {
    DWORD PhysicalAddress;
    DWORD VirtualSize;
  } Misc;
  DWORD VirtualAddress;
  DWORD SizeOfRawData;
  DWORD PointerToRawData;
  DWORD PointerToRelocations;
  DWORD PointerToLinenumbers;
  WORD  NumberOfRelocations;
  WORD  NumberOfLinenumbers;
  DWORD Characteristics;
} IMAGE_SECTION_HEADER, *PIMAGE_SECTION_HEADER;
```

Section Header

Members	Meaning
VirtualSize	**The total size of the section** when loaded into memory, in bytes.
VirtualAddress	**The address of the first byte of the section** when loaded into memory (RVA)
SizeOfRaw Data	**The size of the section data on disk**, in bytes.
PointerToRawData	**The address of the first byte of the section on disk.**
Characteristics	The characteristics of the image.



https://docs.microsoft.com/en-us/windows/desktop/api/winnt/ns-winnt-_image_section_header

Section Header

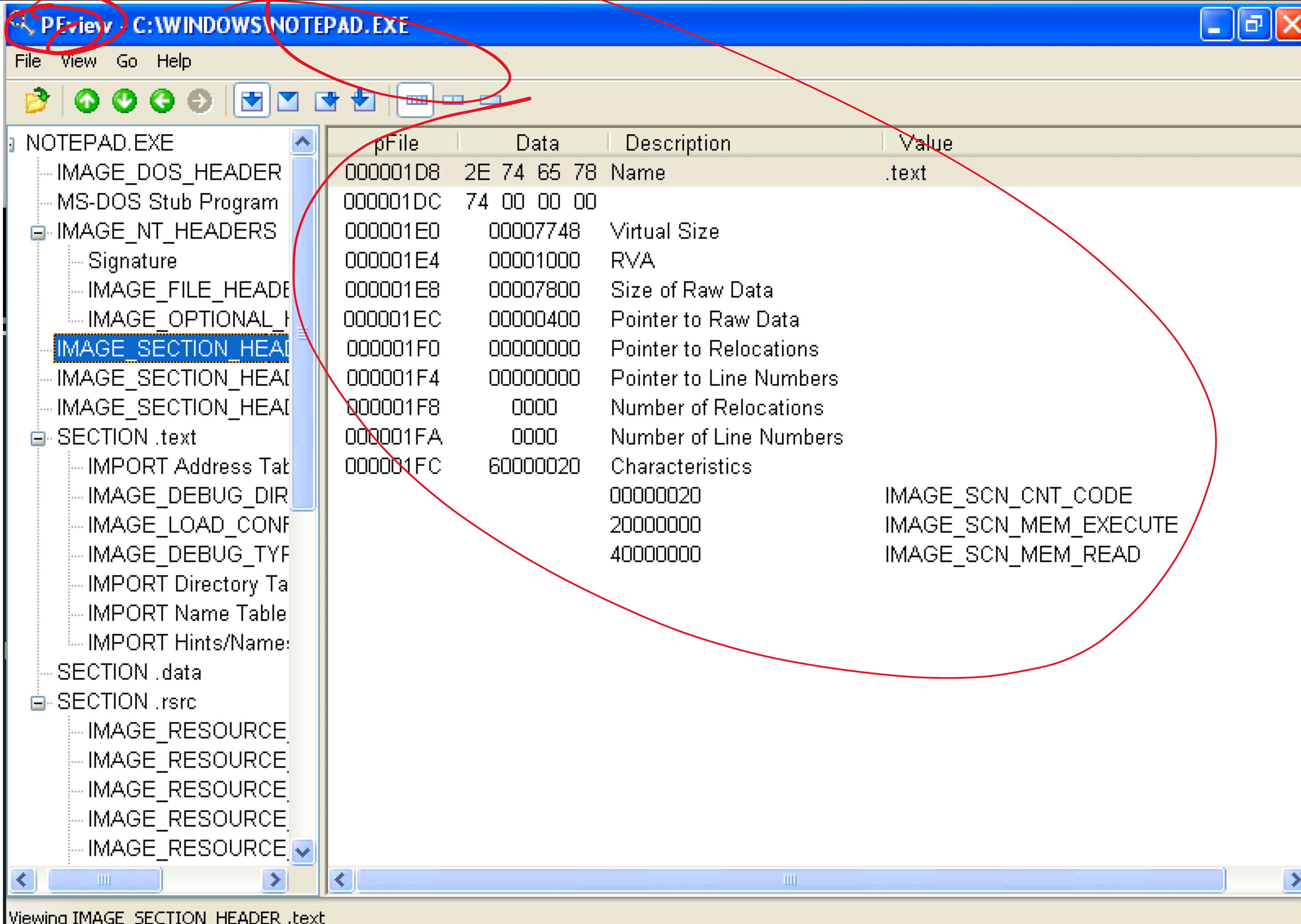
```
000001D0  00 00 00 00 00 00 00 00 2E 74 65 78 74 00 00 00  .........text...
000001E0  48 77 00 00 00 10 00 00 00 78 00 00 00 04 00 00  Hw.......x......
000001F0  00 00 00 00 00 00 00 00 00 00 00 00 20 00 00 60  ............ ..`
00000200  2E 64 61 74 61 00 00 00 A8 1B 00 00 00 90 00 00  .data...¨.......
00000210  00 08 00 00 00 7C 00 00 00 00 00 00 00 00 00 00  .....|..........
00000220  00 00 00 00 40 00 00 C0 2E 72 73 72 63 00 00 00  ....@..À.rsrc...
00000230  58 89 00 00 00 B0 00 00 00 8A 00 00 00 84 00 00  X‰...°...Š...„..
00000240  00 00 00 00 00 00 00 00 00 00 00 00 40 00 00 40  ............@..@
```

Inspecting PE Header Information in Linux

```python
import pefile
import sys

malware_file = sys.argv[1]
pe = pefile.PE(malware_file)
for section in pe.sections:
    print "Name: %s VirtualSize: %s VirtualAddr: %s  SizeofRawData: %s PointerToRawData: %s" %
                    (section.Name,  hex(section.Misc_VirtualSize), hex(section.VirtualAddress), section.SizeOfRawData, section.PointerToRawData)
```

```
root@localhost ~ python display_sections.py a99c01d5748b1bfd203fc1763e6612e8
Name: .text VirtualSize: 0x7378 VirtualAddr: 0x1000  SizeofRawData: 29696 PointerToRawData: 1024
Name: .rdata VirtualSize: 0x261c VirtualAddr: 0x9000  SizeofRawData: 10240 PointerToRawData: 30720
Name: .data VirtualSize: 0x2cac VirtualAddr: 0xc000  SizeofRawData: 3584 PointerToRawData: 40960
Name: .rsrc VirtualSize: 0x1b4 VirtualAddr: 0xf000  SizeofRawData: 512 PointerToRawData: 44544
```

Inspecting PE Header Information



Examining PE Section Table and Sections

- https://hub.docker.com/r/remnux/pescanner/

IAT (Import Address Table)

IAT (Import Address Table)

- Let’s review the concept of DLL (Dynamic Link Library) **again**…

Dynamic Linking

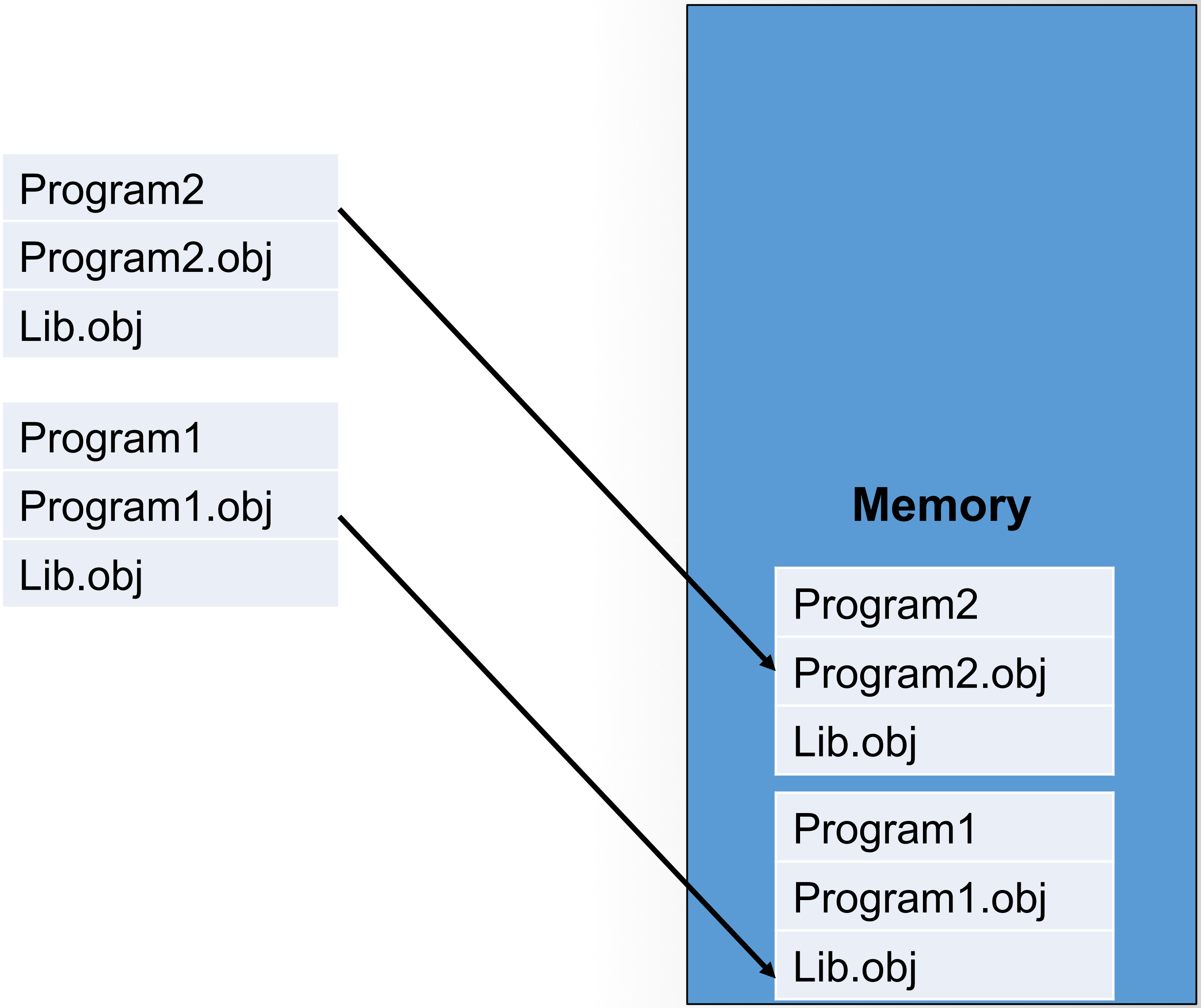
16-Bit DOS System

21. lab1.c (~) - VIM (ssh)

```c
#include <stdio.h>
#include <string.h>

void hacked()
{
>---/* change YOURNAME to your name :) */
>---puts("Hacked by YOURNAME!!!!");
}

void return_input(void)
{
>---/* Please set the array size equal to-
>--- the last two digits of your student ID
>--- e.g. 0861339 --> array size should set to 39 */
>---char array[39];--
>---gets(array);
>---printf("%s\n", array);
}

main()
{
>---return_input();
>---return 0;
}
```

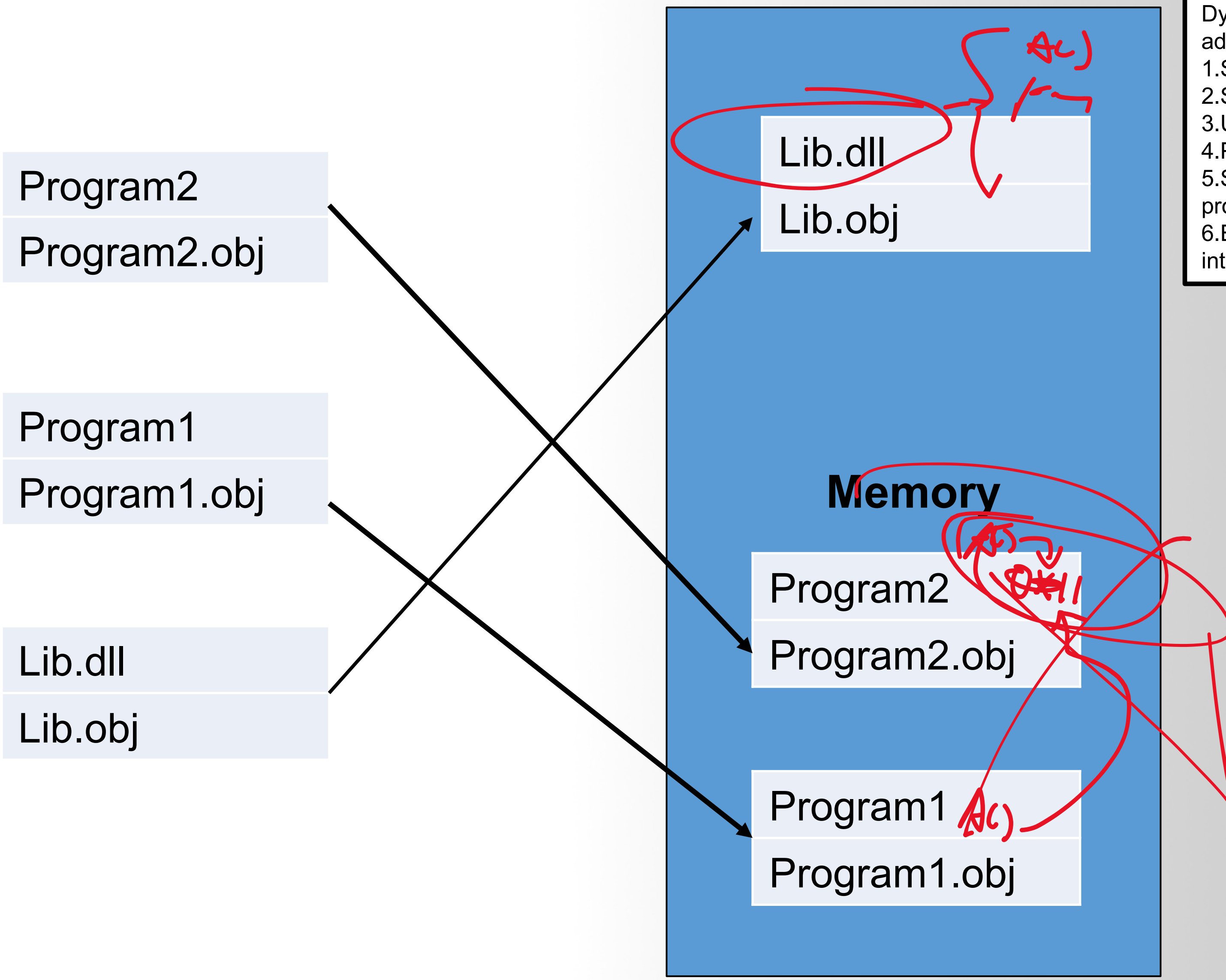
import Library → Put binary code of stdio library into the executable file

Static Linking



- Waste space
- Hard to maintain

Dynamic Linking

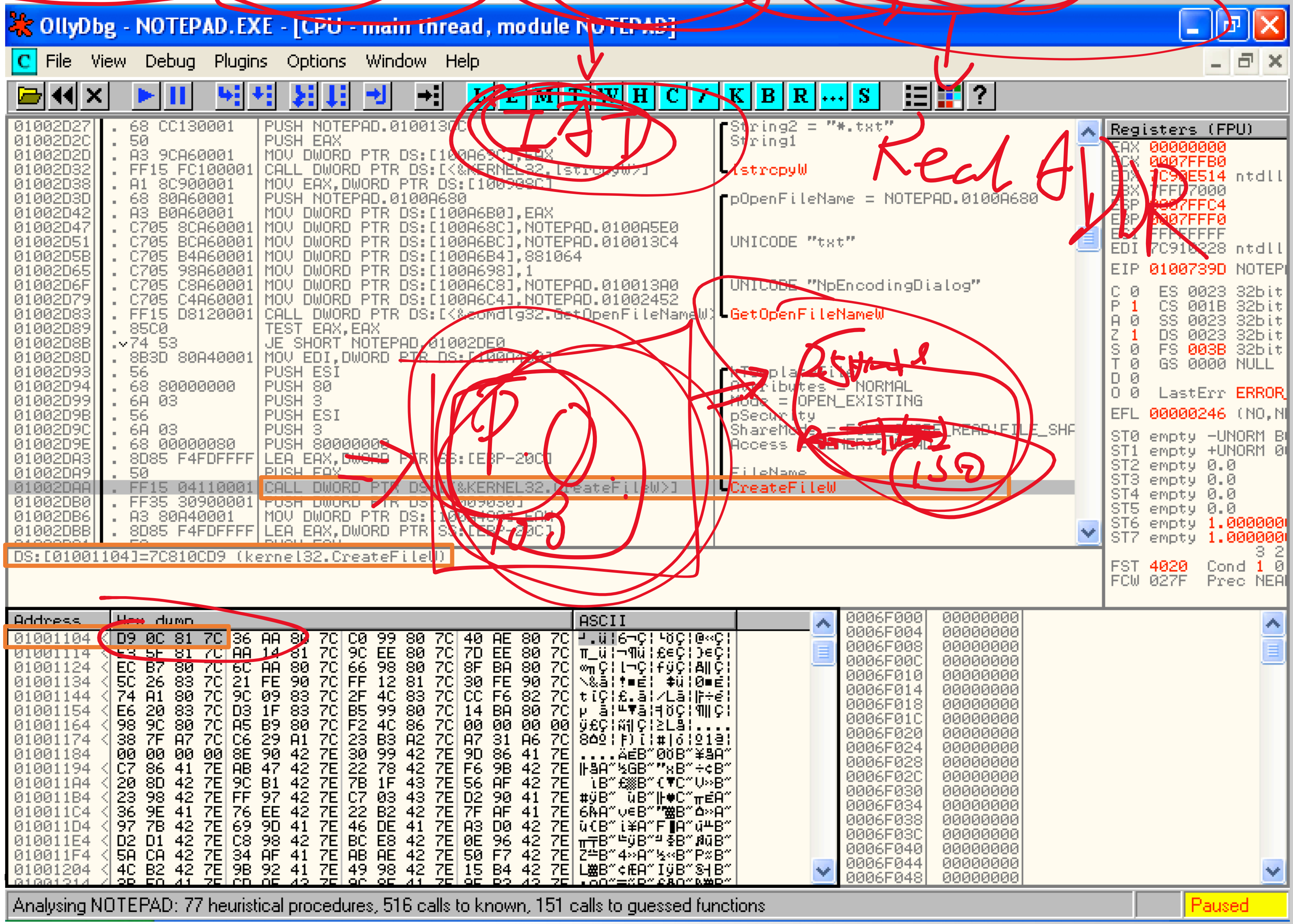


Dynamic linking has the following advantages:

1. Saves memory
2. Saves disk space.
3. Upgrades to the DLL are easier.
4. Provides after-market support.
5. Supports multi language programs.
6. Eases the creation of international versions

Implicit Linking and IAT (Import Address Table)

- Notepad.exe Call CreateFileW() → Call **0x01001104** → Call **0x7C810CD9**



Implicit Linking and IAT (Import Address Table)

- Notepad.exe Call CreateFileW() → Call **0x01001104** → Call **0x7C810CD9**

Call **0x01001104**

↓ Look up

IAT Table

Function Name	IAT Address	Real Address
…		
CreateFileW()	**0x01001104**	**0x7C810CD9**
…		

When the application was first compiled, it was designed so that all API calls will **NOT** use **direct hardcoded addresses** but rather work through a function pointer.

This was accomplished through the use of **an import address table**. This is a table of function pointers filled in by the windows loader as the dlls are loaded.

IAT (Import Address Table)

Why IAT?

IAT (Import Address Table)

- **Support different Windows Version (9X, 2K, XP, Vista, 7, 8, 10)**

Call CreateFileW() --> Call **0x01001104**

Look up

IAT Table

XP

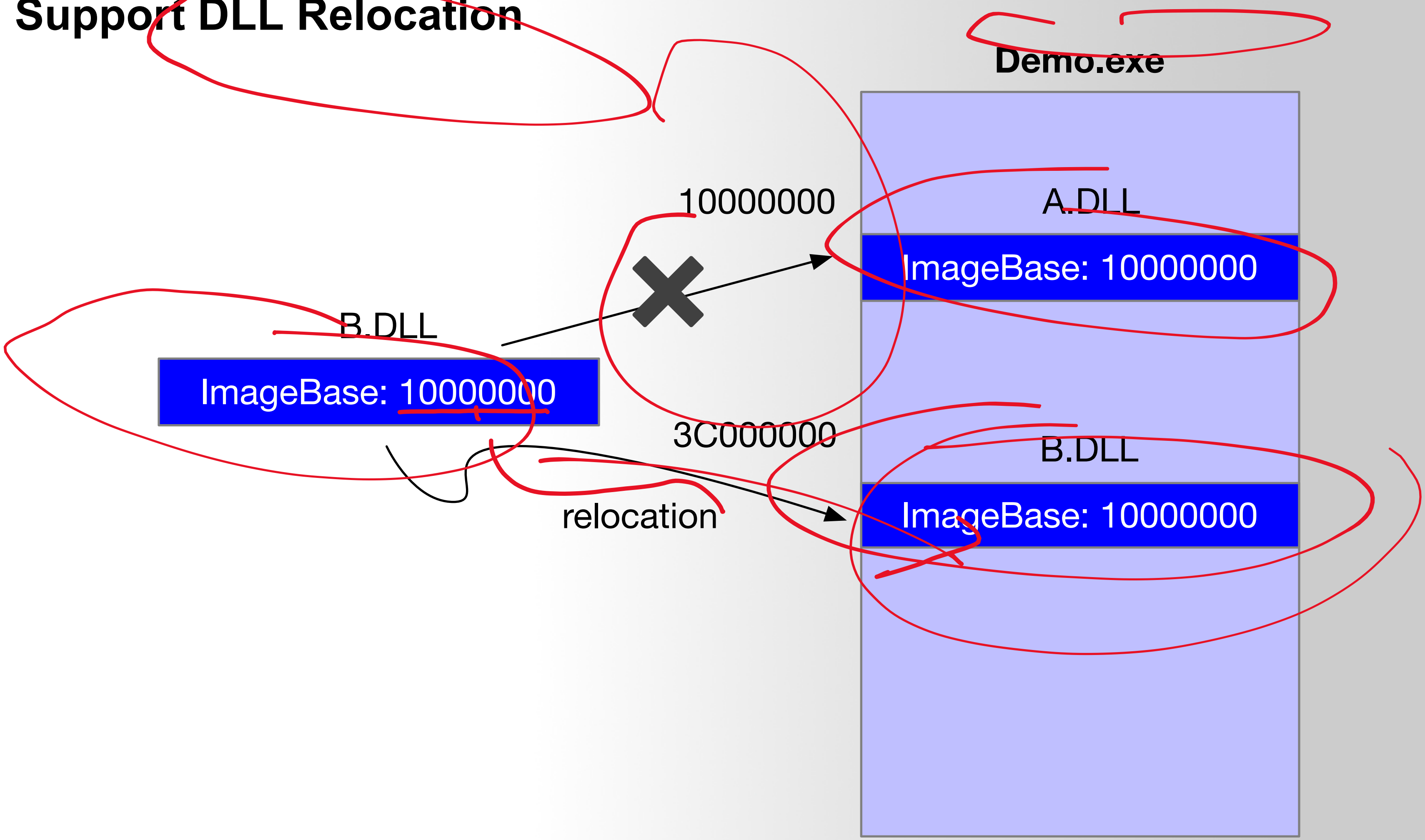
Function Name	IAT Address	Real Address
…		
CreateFileW()	**0x01001104**	**0x7C810CD9**
…		

Windows 7

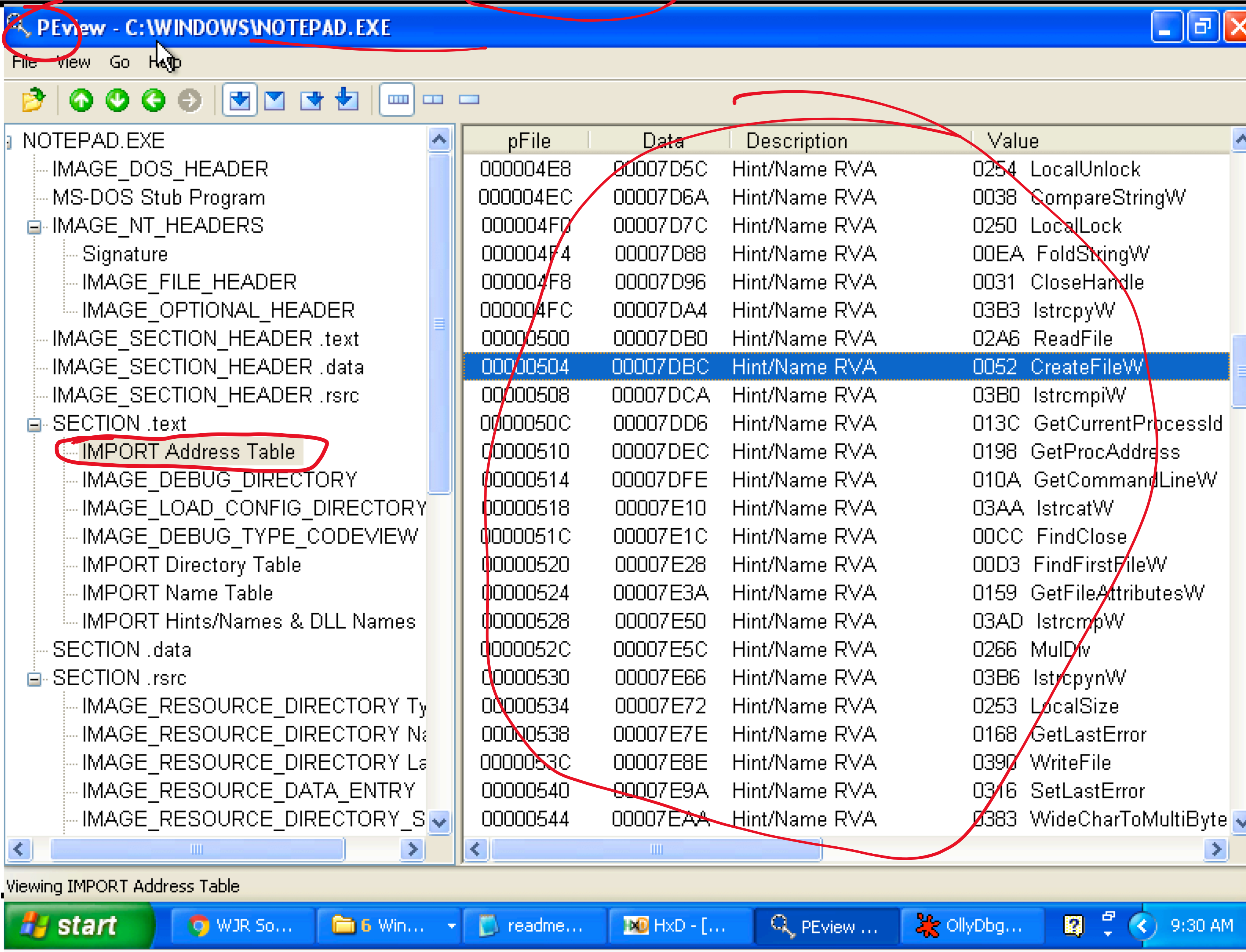
Function Name	IAT Address	Real Address
…		
CreateFileW()	**0x01001104**	**0x7C81FFFF**
…		

IAT (Import Address Table)

- **Support DLL Relocation**

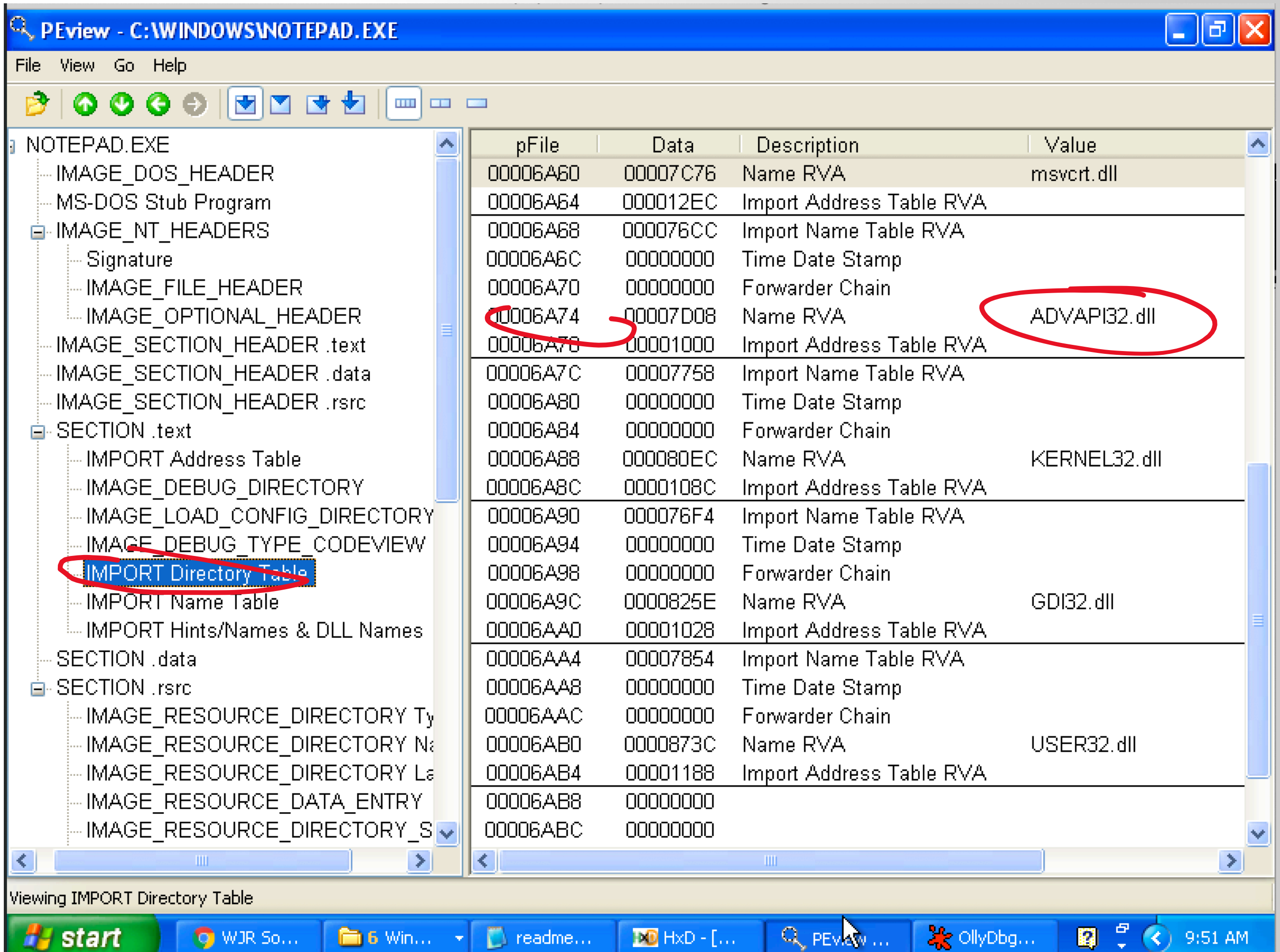


Look up IAT Table with PEview



Import Directory Table

- The Import Directory Table contains entries for every DLL which is loaded by the executable. Each entry contains, among other, Import Lookup Table (ILT) and **Import Address Table (IAT)**



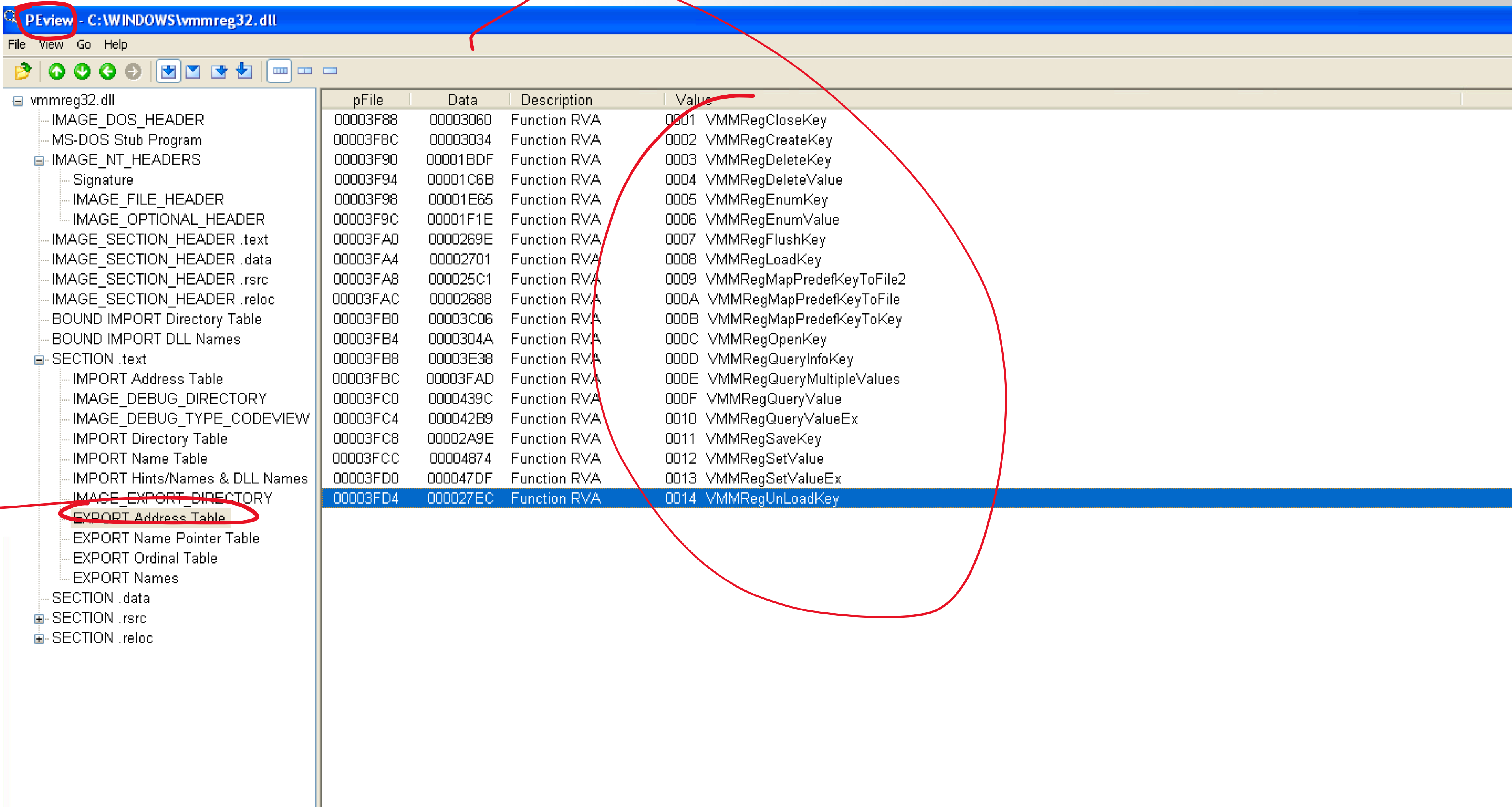
Inspecting file imports with pefile library

```
import pefile
import sys

malware_file = sys.argv[1]
pe = pefile.PE(malware_file)
if hasattr(pe, 'DIRECTORY_ENTRY_IMPORT'):
    for entry in pe.DIRECTORY_ENTRY_IMPORT:
        print("%s" % entry.dll)
        for imp in entry.imports:
            if imp.name != None:
                print ("\t %s \t %s" % (hex(imp.address), imp.name))
            else:
                print ("\tord(%s)" % (str(imp.ordinal)))
        print("\n")
```

```
root@localhost  python enum_imports.py myhack.dll
KERNEL32.dll
        0x1000c000      GetModuleFileNameW
        0x1000c004      OutputDebugStringW
        0x1000c008      CloseHandle
        0x1000c00c      CreateThread
        0x1000c010      WriteConsoleW
        0x1000c014      CreateFileW
        0x1000c018      UnhandledExceptionFilter
        0x1000c01c      SetUnhandledExceptionFilter
        0x1000c020      GetCurrentProcess
        0x1000c024      TerminateProcess
        0x1000c028      IsProcessorFeaturePresent
        0x1000c02c      QueryPerformanceCounter
        0x1000c030      GetCurrentProcessId
        0x1000c034      GetCurrentThreadId
        0x1000c038      GetSystemTimeAsFileTime
        0x1000c03c      InitializeSListHead
        0x1000c040      IsDebuggerPresent
        0x1000c044      GetStartupInfoW
        0x1000c048      GetModuleHandleW
        0x1000c04c      InterlockedFlushSList
        0x1000c050      RtlUnwind
        0x1000c054      GetLastError
        0x1000c058      SetLastError
        0x1000c05c      EnterCriticalSection
        0x1000c060      LeaveCriticalSection
        0x1000c064      DeleteCriticalSection
        0x1000c068      InitializeCriticalSectionAndSpinCount
        0x1000c06c      TlsAlloc
        0x1000c070      TlsGetValue
        0x1000c074      TlsSetValue
        0x1000c078      TlsFree
        0x1000c07c      FreeLibrary
        0x1000c080      GetProcAddress
        0x1000c084      LoadLibraryExW
        0x1000c088      RaiseException
        0x1000c08c      ExitProcess
        0x1000c090      GetModuleHandleExW
        0x1000c094      HeapAlloc
        0x1000c098      HeapFree
        0x1000c09c      FindClose
        0x1000c0a0      FindFirstFileExW
        0x1000c0a4      FindNextFileW
        0x1000c0a8      IsValidCodePage
        0x1000c0ac      GetACP
        0x1000c0b0      GetOEMCP
        0x1000c0b4      GetCPInfo
        0x1000c0b8      GetCommandLineA
        0x1000c0bc      GetCommandLineW
        0x1000c0c0      MultiByteToWideChar
        0x1000c0c4      WideCharToMultiByte
        0x1000c0c8      GetEnvironmentStringsW
        0x1000c0cc      FreeEnvironmentStringsW
        0x1000c0d0      GetStdHandle
        0x1000c0d4      GetFileType
        0x1000c0d8      LCMapStringW
        0x1000c0dc      GetProcessHeap
        0x1000c0e0      GetStringTypeW
        0x1000c0e4      HeapSize
        0x1000c0e8      HeapReAlloc
        0x1000c0ec      SetStdHandle
        0x1000c0f0      FlushFileBuffers
        0x1000c0f4      WriteFile
        0x1000c0f8      GetConsoleCP
        0x1000c0fc      GetConsoleMode
        0x1000c100      SetFilePointerEx
        0x1000c104      DecodePointer

urlmon.dll
        0x1000c10c      URLDownloadToFileW
```

Export Address Table (EAT)

- The EAT works in the same way as the IAT, apart from the library will be exporting the functions to the image executable, in which the program will import into the IAT.



pFile	Data	Description	Value	
00003F88	00003060	Function RVA	0001	VMMRegCloseKey
00003F8C	00003034	Function RVA	0002	VMMRegCreateKey
00003F90	00001BDF	Function RVA	0003	VMMRegDeleteKey
00003F94	00001C6B	Function RVA	0004	VMMRegDeleteValue
00003F98	00001E65	Function RVA	0005	VMMRegEnumKey
00003F9C	00001F1E	Function RVA	0006	VMMRegEnumValue
00003FA0	0000269E	Function RVA	0007	VMMRegFlushKey
00003FA4	00002701	Function RVA	0008	VMMRegLoadKey
00003FA8	000025C1	Function RVA	0009	VMMRegMapPredefKeyToFile2
00003FAC	00002688	Function RVA	000A	VMMRegMapPredefKeyToFile
00003FB0	00003C06	Function RVA	000B	VMMRegMapPredefKeyToKey
00003FB4	0000304A	Function RVA	000C	VMMRegOpenKey
00003FB8	00003E38	Function RVA	000D	VMMRegQueryInfoKey
00003FBC	00003FAD	Function RVA	000E	VMMRegQueryMultipleValues
00003FC0	0000439C	Function RVA	000F	VMMRegQueryValue
00003FC4	000042B9	Function RVA	0010	VMMRegQueryValueEx
00003FC8	00002A9E	Function RVA	0011	VMMRegSaveKey
00003FCC	00004874	Function RVA	0012	VMMRegSetValue
00003FD0	000047DF	Function RVA	0013	VMMRegSetValueEx
00003FD4	000027EC	Function RVA	0014	VMMRegUnLoadKey

Real-world Case Study

16d6b0e2c77da2776a88dd88c7cfc672

(Trojan.Win32.Dllhijack.a)

16d6b0e2c77da2776a88dd88c7cfc672

```
root@li254-249 ~ python enum_exports.py 16d6b0e2c77da2776a88dd88c7cfc672
0x100011e0      CreateDatabaseQueryObject       1
0x100011e0      DataImporterMain        2
0x100011e0      FlashboxMain    3
0x100010d0      KugouMain       4
```

KuGou

From Wikipedia, the free encyclopedia

KuGou (Chinese: 酷狗音乐) is a Chinese music streaming and download service established in 2004 and owned by Tencent Music.[1][2] It is the largest music streaming service in the world, with more than 450 million monthly active users.[2] KuGou is the largest online music service in China, with a market share of 28%.[1] It has more than 800 million users.[1] A merger between China Music Corporation and Tencent's QQ Music was announced on July 15, 2016.[1][3] The services are expected to continue being offered separately.[1] Together with Kuwo, another online music service also owned by Tencent Music and the third largest one in China,[1] KuGou holds a music award ceremony, the KU Music Asian Music Awards,[4] also known as Cool Music Asia Festival Award.[5]

KuGou

Developer(s)	Tencent Music
Initial release	2004; 15 years ago
Operating system	Android, iOS, Web, Windows
Type	Music streaming
Website	www.kugou.com

References [edit]

1. ^ a b c d e f Zen Soo (July 15, 2016). "Tencent to merge QQ Music service with China Music Corp to create streaming giant". *South China Morning Post*. Retrieved August 20, 2016.
2. ^ a b Jubb, Nathan (October 19, 2016). "The Future of Music Streaming Lies in China's Small Cities". *Sixth Tone*. Retrieved October 27, 2016.
3. ^ Millward, Steven (July 15, 2016). "In China, 'Spotify' is free". *Tech In Asia*. Retrieved August 20, 2016.
4. ^ Kim Dong-Joo (March 31, 2016). "Kang Ta & SHINee garner awards at 'KU MUSIC ASIAN MUSIC AWARDS'". *sg.style.yahoo.com*. Retrieved August 20, 2016.
5. ^ "FTISLAND Wins "Asia's Popular Band" Award at Cool Music Asia Festival Award". *Soompi*. April 23, 2015. Retrieved August 20, 2016.

External links [edit]

- Official website (in Chinese)

16d6b0e2c77da2776a88dd88c7cfc672

```
root@li254-249  ~  python enum_exports.py 16d6b0e2c77da2776a88dd88c7cfc672
0x100011e0      CreateDatabaseQueryObject       1
0x100011e0      DataImporterMain        2
0x100011e0      FlashboxMain    3
0x100010d0      KugouMain       4
```

```
Dump of assembler code for function kugou!FlashboxMain:
   0x100011e0 <+0>:     xor    eax,eax
   0x100011e2 <+2>:     ret
   0x100011e3 <+3>:     nop
   0x100011e4 <+4>:     nop
   0x100011e5 <+5>:     nop
   0x100011e6 <+6>:     nop
   0x100011e7 <+7>:     nop
   0x100011e8 <+8>:     nop
   0x100011e9 <+9>:     nop
   0x100011ea <+10>:    nop
   0x100011eb <+11>:    nop
   0x100011ec <+12>:    nop
   0x100011ed <+13>:    nop
   0x100011ee <+14>:    nop
   0x100011ef <+15>:    nop
```

16d6b0e2c77da2776a88dd88c7cfc672

Incident Response

Risk Assessment

Remote Access	Uses network protocols on unusual ports
Network Behavior	Contacts 2 domains and 2 hosts. View the network section for more details.

- https://www.hybrid-analysis.com/sample/037203d274cb66bad34559c0f426e9e1bf91a048155240581f4aa554be17925c?environmentId=100

40 > 12
52 > 12
64 > 12
76 > 12

0fd6e3fb1cd5ec397ff3cdbaac39d80c

```
root@li254-249  ~  python enum_exports.py 0fd6e3fb1cd5ec397ff3cdbaac39d80c
0x10002628      AheadLib_LpkPresent     36
0x10002634      AheadLib_ScriptApplyDigitSubstitution   37
0x10002640      AheadLib_ScriptApplyLogicalWidth        38
0x1000264c      AheadLib_ScriptBreak    39
0x10002658      AheadLib_ScriptCPtoX    40
0x10002664      AheadLib_ScriptCacheGetHeight   41
0x10002670      AheadLib_ScriptFreeCache        42
0x1000267c      AheadLib_ScriptGetCMap  43
0x10002688      AheadLib_ScriptGetFontProperties        44
0x10002694      AheadLib_ScriptGetGlyphABCWidth         45
0x1000271f      AheadLib_ScriptGetLogicalWidths         46
0x1000272b      AheadLib_ScriptGetProperties    47
0x10002737      AheadLib_ScriptIsComplex        48
0x10002743      AheadLib_ScriptItemize  49
0x10003091      AheadLib_ScriptJustify  50
0x1000309d      AheadLib_ScriptLayout   51
0x100030a9      AheadLib_ScriptPlace    52
0x100030b5      AheadLib_ScriptRecordDigitSubstitution          53
0x100030c1      AheadLib_ScriptShape    54
0x100030cd      AheadLib_ScriptStringAnalyse    55
0x100030d9      AheadLib_ScriptStringCPtoX      56
0x100030e5      AheadLib_ScriptStringFree       57
0x100030f1      AheadLib_ScriptStringGetLogicalWidths   58
0x100030fd      AheadLib_ScriptStringGetOrder   59
0x10003109      AheadLib_ScriptStringOut        60
0x10003115      AheadLib_ScriptStringValidate   61
0x10003121      AheadLib_ScriptStringXtoCP      62
0x1000312d      AheadLib_ScriptString_pLogAttr          63
0x10003139      AheadLib_ScriptString_pSize     64
0x10003145      AheadLib_ScriptString_pcOutChars        65
0x10003151      AheadLib_ScriptTextOut          66
0x1000315d      AheadLib_ScriptXtoCP    67
0x10003169      AheadLib_UspAllocCache          68
0x10003175      AheadLib_UspAllocTemp   69
0x10003181      AheadLib_UspFreeMem     70
0x100023cf      LpkDllInitialize        311
0x100023db      LpkDrawTextEx   411
0x1001d040      LpkEditControl          71
0x100023f3      LpkExtTextOut   611
0x100023ff      LpkGetCharacterPlacement        711
0x100025f3      LpkGetTextExtentExPoint         811
0x100023b7      LpkInitialize   111
0x10002604      LpkPSMTextOut   911
0x10002628      LpkPresent      1
0x100023c3      LpkTabbedTextOut        211
0x10002610      LpkUseGDIWidthCache     1011
0x100023cf      MemCode_LpkDllInitialize        72
0x100023db      MemCode_LpkDrawTextEx   73
0x100023e7      MemCode_LpkEditControl          74
0x100023f3      MemCode_LpkExtTextOut   75
0x100023ff      MemCode_LpkGetCharacterPlacement        76
0x100025f3      MemCode_LpkGetTextExtentExPoint         77
0x100023b7      MemCode_LpkInitialize   78
0x10002604      MemCode_LpkPSMTextOut   79
0x100023c3      MemCode_LpkTabbedTextOut        80
0x10002610      MemCode_LpkUseGDIWidthCache     81
0x1000261c      MemCode_ftsWordBreak    82
0x10002634      ScriptApplyDigitSubstitution    2
0x10002640      ScriptApplyLogicalWidth         3
```

```
gdb-peda$ disas ScriptBreak
Dump of assembler code for function drc!ScriptBreak:
   0x1000264c <+0>:     push   0x1001d804
   0x10002651 <+5>:     call   0x1000233d
   0x10002656 <+10>:    jmp    eax
End of assembler dump.
```

```
gdb-peda$ disas LpkPresent
Dump of assembler code for function drc!LpkPresent:
   0x10002628 <+0>:     push   0x1001d7c0
   0x1000262d <+5>:     call   0x1000233d
   0x10002632 <+10>:    jmp    eax
End of assembler dump.
```

6a764e4e6db461781d080034aab85aff

&

cc3c6c77e118a83ca0513c25c208832c

```
root@li254-249 ~ python enum_exports.py 6a764e4e6db461781d080034aab85aff
0x10004f00      AheadLib_ScriptApplyDigitSubstitution   36
0x10004f10      AheadLib_ScriptApplyLogicalWidth        37
0x10004f20      AheadLib_ScriptBreak    38
0x10004f30      AheadLib_ScriptCPtoX    39
0x10004f40      AheadLib_ScriptCacheGetHeight   40
0x10004f50      AheadLib_ScriptFreeCache        41
0x10004f60      AheadLib_ScriptGetCMap  42
0x10004f70      AheadLib_ScriptGetFontProperties        43
0x10004f80      AheadLib_ScriptGetGlyphABCWidth         44
0x10004f90      AheadLib_ScriptGetLogicalWidths         45
0x10004fa0      AheadLib_ScriptGetProperties    46
0x10004fb0      AheadLib_ScriptIsComplex        47
0x10004fc0      AheadLib_ScriptItemize          48
0x10004fd0      AheadLib_ScriptJustify          49
0x10004fe0      AheadLib_ScriptLayout   50
0x10004ff0      AheadLib_ScriptPlace    51
0x10005000      AheadLib_ScriptRecordDigitSubstitution          52
0x10005010      AheadLib_ScriptShape    53
0x10005020      AheadLib_ScriptStringAnalyse    54
0x10005030      AheadLib_ScriptStringCPtoX      55
0x10005040      AheadLib_ScriptStringFree       56
0x10005050      AheadLib_ScriptStringGetLogicalWidths   57
0x10005060      AheadLib_ScriptStringGetOrder   58
0x10005070      AheadLib_ScriptStringOut        59
0x10005080      AheadLib_ScriptStringValidate   60
0x10005090      AheadLib_ScriptStringXtoCP      61
0x100050a0      AheadLib_ScriptString_pLogAttr          62
0x100050b0      AheadLib_ScriptString_pSize     63
0x100050c0      AheadLib_ScriptString_pcOutChars        64
0x100050d0      AheadLib_ScriptTextOut          65
0x100050e0      AheadLib_ScriptXtoCP    66
0x100050f0      AheadLib_UspAllocCache          67
0x10005100      AheadLib_UspAllocTemp   68
0x10005110      AheadLib_UspFreeMem     69
0x10004ef0      AheadLib_mmLpkPresent   70
0x10004e50      LpkDllInitialize        311
0x10004e60      LpkDrawTextEx   411
0x1000e92c      LpkEditControl          71
0x10004e80      LpkExtTextOut   611
0x10004e90      LpkGetCharacterPlacement        711
0x10004ea0      LpkGetTextExtentExPoint         811
0x10004e30      LpkInitialize   111
0x10004ec0      LpkPSMTextOut   911
0x10004ef0      LpkPresent      1
0x10004e40      LpkTabbedTextOut        211
0x10004ed0      LpkUseGDIWidthCache     1011
0x10004e50      MemCode_LpkDllInitialize        72
0x10004e60      MemCode_LpkDrawTextEx   73
0x10004e70      MemCode_LpkEditControl          74
0x10004e80      MemCode_LpkExtTextOut   75
0x10004e90      MemCode_LpkGetCharacterPlacement        76
0x10004ea0      MemCode_LpkGetTextExtentExPoint         77
0x10004e30      MemCode_LpkInitialize   78
0x10004ec0      MemCode_LpkPSMTextOut   79
0x10004e40      MemCode_LpkTabbedTextOut        80
0x10004ed0      MemCode_LpkUseGDIWidthCache     81
0x10004ee0      MemCode_ftsWordBreak    82
0x10004f00      ScriptApplyDigitSubstitution    2
0x10004f10      ScriptApplyLogicalWidth         3
```

```
root@li254-249 ~ python enum_exports.py cc3c6c77e118a83ca0513c25c208832c
0x10001100      LpkPresent      1
0x10001120      ScriptApplyDigitSubstitution    2
0x10001140      ScriptApplyLogicalWidth         3
0x10001160      ScriptBreak     4
0x10001180      ScriptCPtoX     5
0x100011a0      ScriptCacheGetHeight    6
0x100011c0      ScriptFreeCache         7
0x100011e0      ScriptGetCMap   8
0x10001200      ScriptGetFontProperties         9
0x10001220      ScriptGetGlyphABCWidth          10
0x10001240      ScriptGetLogicalWidths          11
0x10001260      ScriptGetProperties     12
0x10001280      ScriptIsComplex         13
0x100012a0      ScriptItemize   14
0x100012c0      ScriptJustify   15
0x100012e0      ScriptLayout    16
0x10001300      ScriptPlace     17
0x10001320      ScriptRecordDigitSubstitution   18
0x10001340      ScriptShape     19
0x10001360      ScriptStringAnalyse     20
0x10001380      ScriptStringCPtoX       21
0x100013a0      ScriptStringFree        22
0x100013c0      ScriptStringGetLogicalWidths    23
0x100013e0      ScriptStringGetOrder    24
0x10001400      ScriptStringOut         25
0x10001420      ScriptStringValidate    26
0x10001440      ScriptStringXtoCP       27
0x10001460      ScriptString_pLogAttr   28
0x10001480      ScriptString_pSize      29
0x100014a0      ScriptString_pcOutChars         30
0x100014c0      ScriptTextOut   31
0x100014e0      ScriptXtoCP     32
0x10001890      ServiceMain     36
0x10001500      UspAllocCache   33
0x10001520      UspAllocTemp    34
0x10001540      UspFreeMem      35
```

e0bed0b33e7b6183f654f0944b607618

e0bed0b33e7b6183f654f0944b607618

```
 root@li254-249  ~  python enum_exports.py e0bed0b33e7b6183f654f0944b607618
0x100165f0       LsaApCallPackage        1
0x10016610       LsaApCallPackagePassthrough     2
0x10016600       LsaApCallPackageUntrusted       3
0x100165e0       LsaApInitializePackage          4
0x10016620       LsaApLogonTerminated    5
0x10016470       LsaApLogonUserEx2       6
0x10016560       SpInitialize    7
0x100165d0       SpInstanceInit          8
0x10016570       SpLsaModeInitialize     9
0x100165c0       SpUserModeInitialize    10
```

db8199eeb2d75e789df72cd8852a9fbb

(Rootkit.Win32.blackken.b)

db8199eeb2d75e789df72cd8852a9fbb

```
root@li254-249  ~  python enum_exports.py db8199eeb2d75e789df72cd8852a9fbb
0x10006707       ?Start@@YGKPAX@Z        1
0x10006707       MakeCache       2
```

Is this claim correct?

If two export functions share the same address, it's a malware.

1c1131112db91382b9d8b46115045097

1c1131112db91382b9d8b46115045097

```
root@li254-249  ~  python enum_exports.py 1c1131112db91382b9d8b46115045097
0x100014a0        AfxGetHttpRaquestMgr    3
0x100014b0        AfxGetHttpRequastMgr    4
0x10001490        InitInstance    2
0x10001490        MessageLoop     1
```

Lab2

- Create your own anti-malware system based on heuristic analysis.
- Check course website

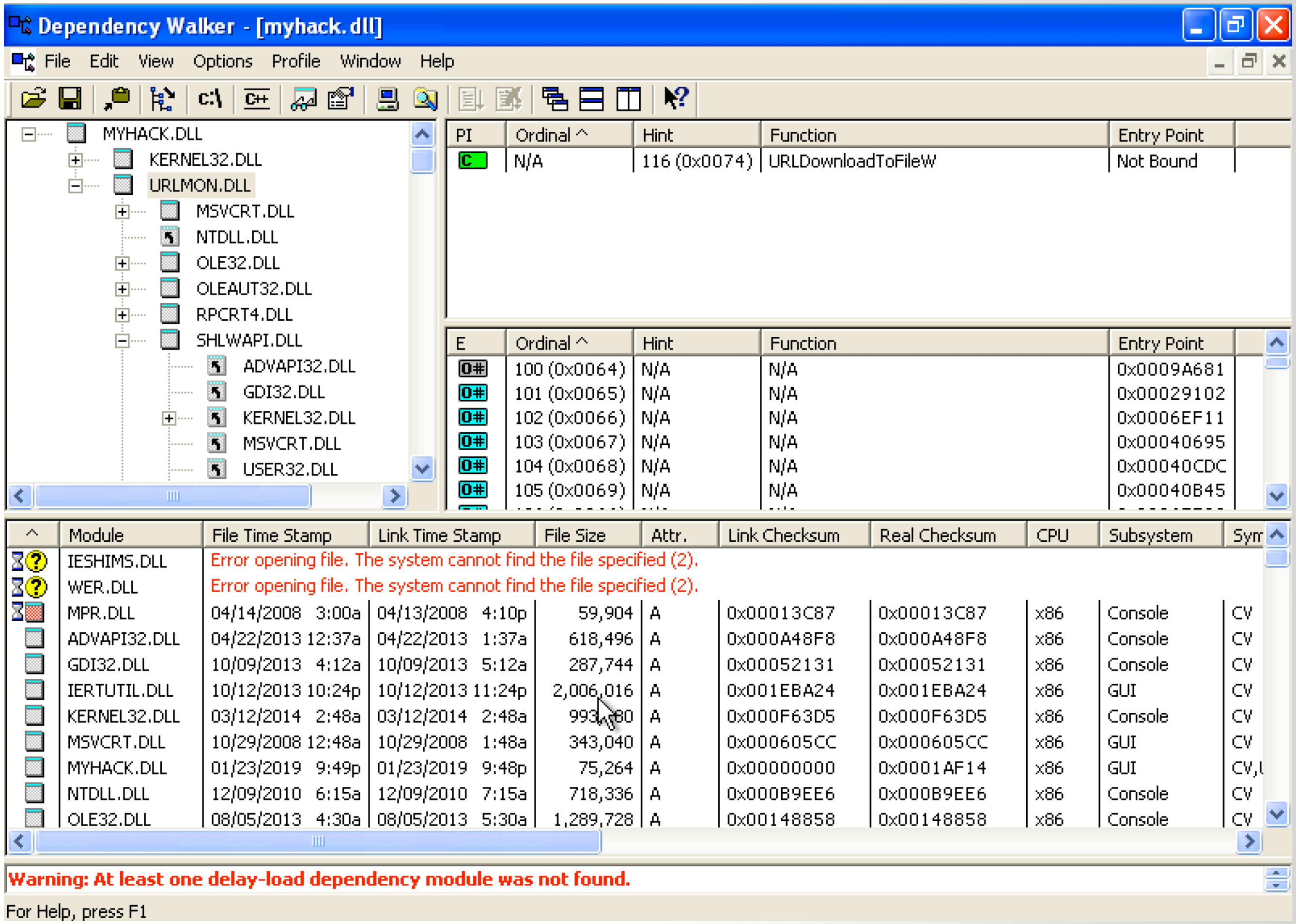
EAT (Export Address Table)

- Similar to IAT, EAT data is stored in IMAGE_EXPORT_DIRECTORY
- EAT contains an RVA that points to an array of pointers to **(RVAs of) the functions in the module**.

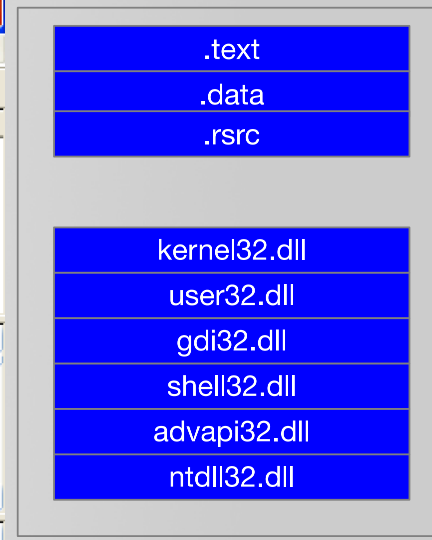
Inspecting file export with pefile library

```python
import pefile
import sys

malware_file = sys.argv[1]
pe = pefile.PE(malware_file)
if hasattr(pe, 'DIRECTORY_ENTRY_EXPORT'):
    for exp in pe.DIRECTORY_ENTRY_EXPORT.symbols:
        print "%s"  % exp.name
```

Exploring Dynamically Linked Functions with Dependency Walker



Notepad.exe Process



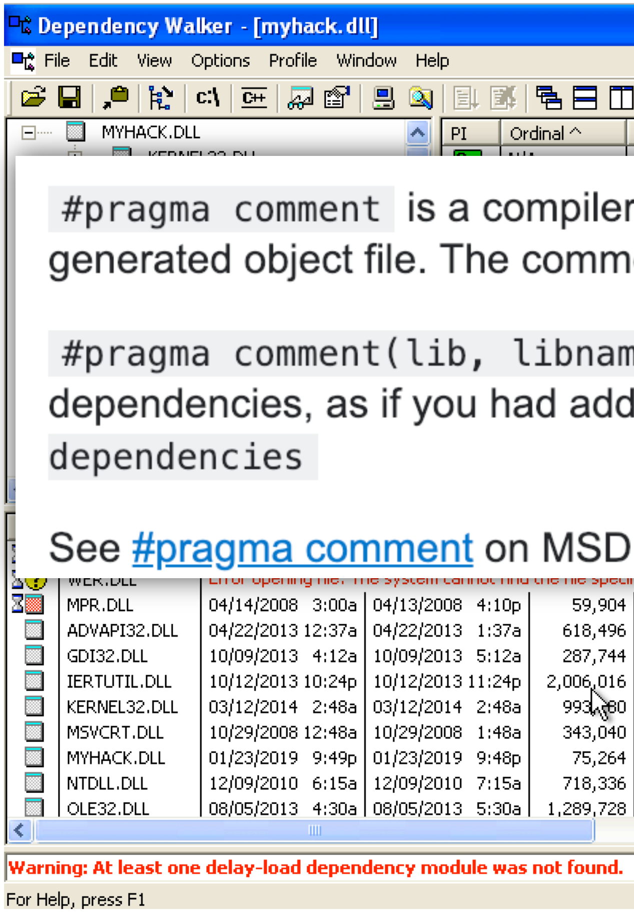
Common DLLs

- .text
- .data
- .rsrc

- kernel32.dll
- user32.dll
- gdi32.dll
- shell32.dll
- advapi32.dll
- ntdll32.dll

DLL	Description
Kernel32.dll	This is a very common DLL that contains core functionality, such as access and manipulation of memory, files, and hardware.
Advapi32.dll	This DLL provides access to advanced core Windows components such as the Service Manager and Registry.
User32.dll	This DLL contains all the user-interface components, such as buttons, scroll bars, and components for controlling and responding to user actions.
Gdi32.dll	This DLL contains functions for displaying and manipulating graphics.
Ntdll.dll	This DLL is the interface to the Windows kernel. Executables generally do not import this file directly, although it is always imported indirectly by *Kernel32.dll*. If an executable imports this file, it means that the author intended to use functionality not normally available to Windows programs. Some tasks, such as hiding functionality or manipulating processes, will use this interface.
WSock32.dll and *Ws2_32.dll*	These are networking DLLs. A program that accesses either of these most likely connects to a network or performs network-related tasks.
Wininet.dll	This DLL contains higher-level networking functions that implement protocols such as FTP, HTTP, and NTP.

Exploring Dynamically Linked Functions with Dependency Walker



myhack.cpp 〉 No Selection

```
#include "windows.h"
#include "tchar.h"

#pragma comment(lib, "urlmon.lib")

#define DEF_URL         (L"http://www.naver.com/index.html")
```

`#pragma comment` is a compiler directive which indicates Visual C++ to leave a comment in the generated object file. The comment can then be read by the linker when it processes object files.

`#pragma comment(lib, libname)` tells the linker to add the 'libname' library to the list of library dependencies, as if you had added it in the project properties at `Linker->Input->Additional dependencies`

See [#pragma comment](#) on MSDN

```
    URLDownloadToFile(NULL, DEF_URL, szPath, 0, NULL);

    return 0;
}

BOOL WINAPI DllMain(HINSTANCE hinstDLL, DWORD fdwReason, LPVOID lpvReserved)
{
    HANDLE hThread = NULL;

    g_hMod = (HMODULE)hinstDLL;

    switch( fdwReason )
    {
    case DLL_PROCESS_ATTACH :
        OutputDebugString(L"<myhack.dll> Injection!!! -- CSC 497/583 -- Dr. Chen");
        hThread = CreateThread(NULL, 0, ThreadProc, NULL, 0, NULL);
        CloseHandle(hThread);
        break;
    }

    return TRUE;
}
```

Q & A