

# RuxCon 2004

# Reverse Engineering for Malware Analysis

by: Peter Taylor

email: nevar<insert at symbol>feline<no space>menace<full stop>com

RuxCon 2004

# Introduction

- What is "Malware"
  - Software with malicious intent
  - Software with sinister motives

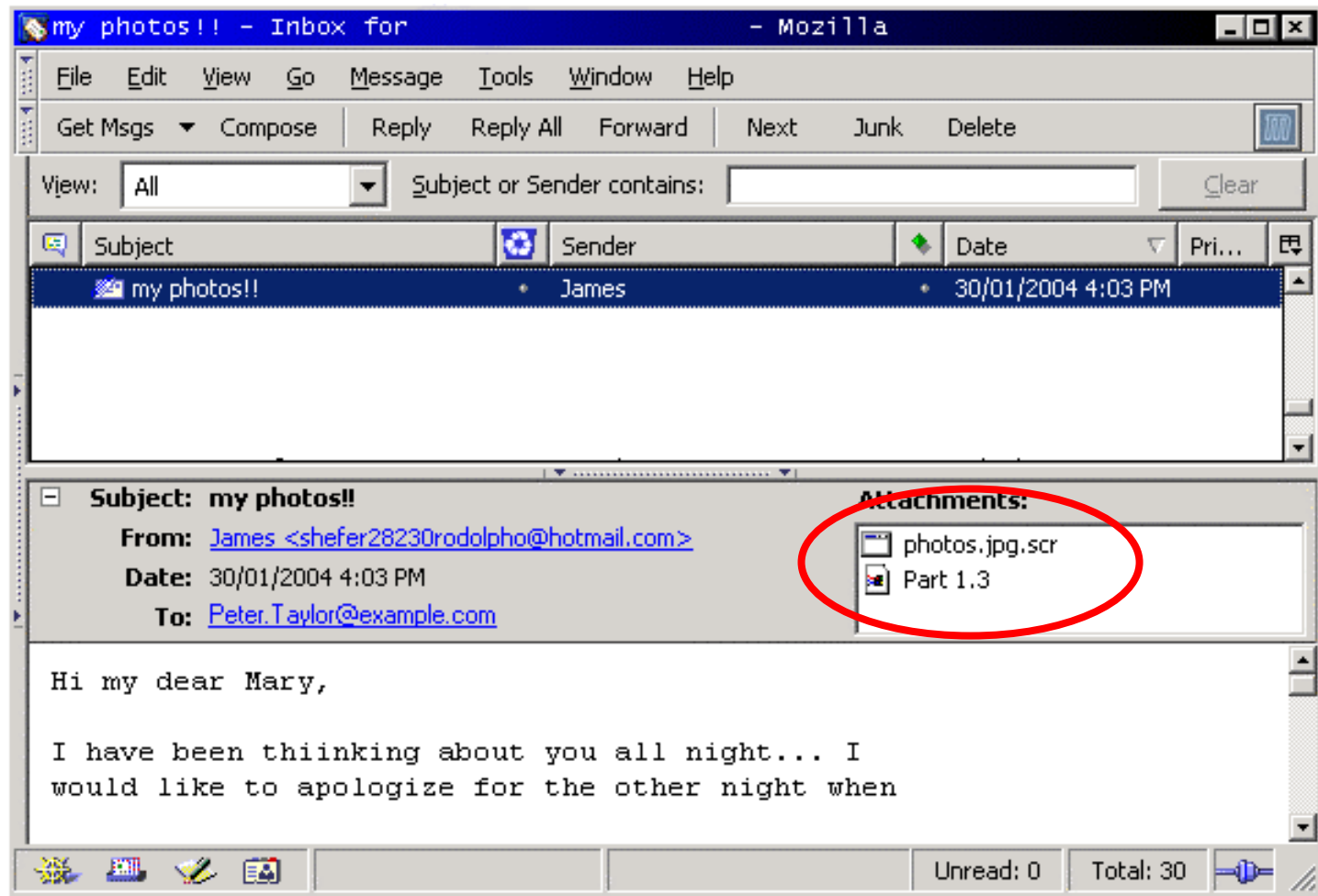
# Introduction

- What is "Malware"
  - Software with malicious intent
  - Software with sinister motives
- Is there a need for specialized R.E. when dealing with malware

# R.E. and Malware

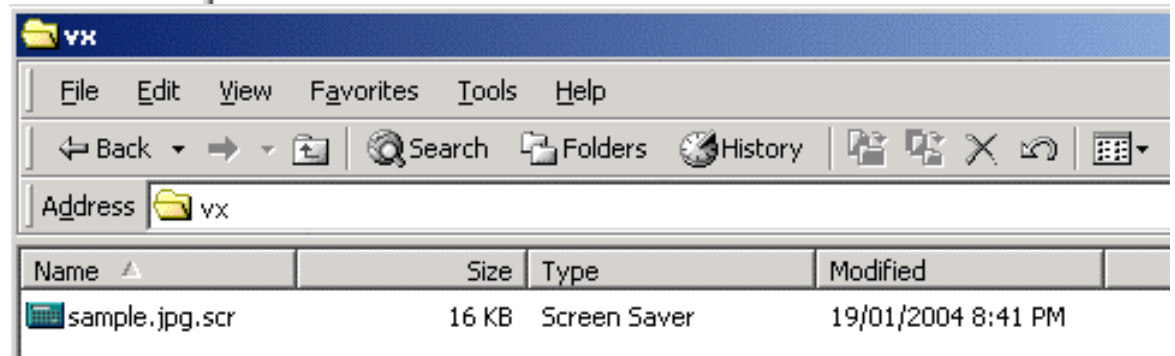
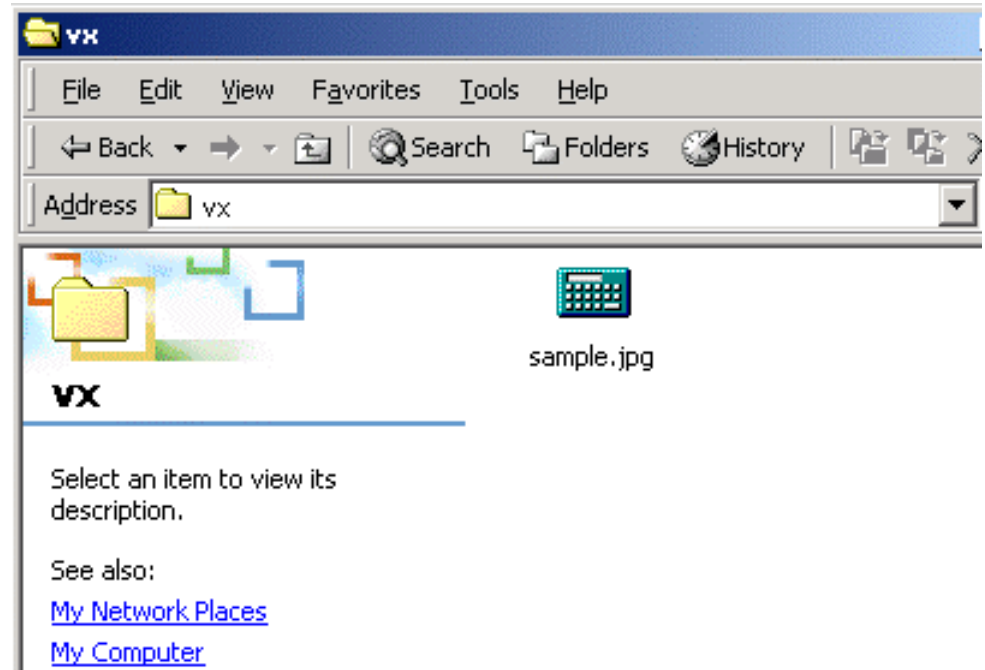
- Time is of the essence
- No complex algorithms or data-structures to reverse engineer
- A small subset of system functions work together to exhibit malicious behaviour
- Lack of imagination and release of malware source leads to rampant function re-use
- Code to intricately manipulate executable file headers has few reason to appear in notepad.exe

# Contracting Malware - Did you get my email ?



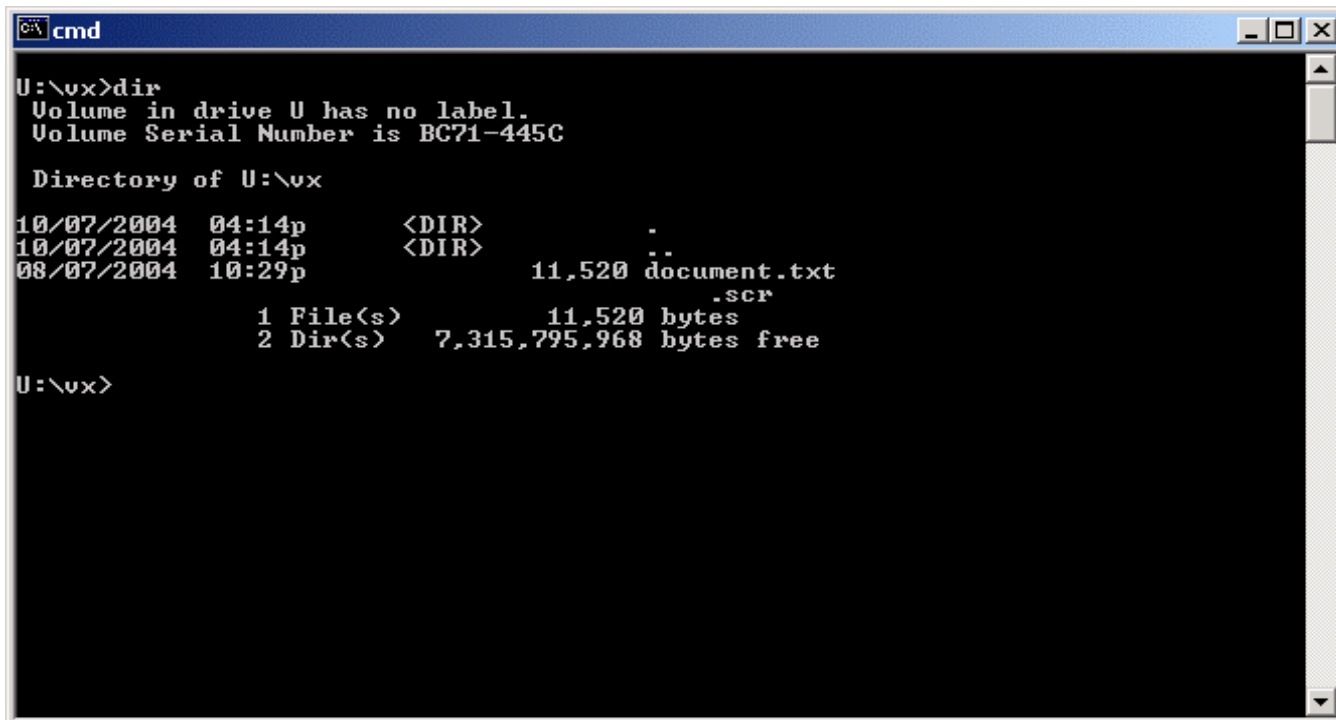
# Explorer Trickery

• Too many smarts spoil the Puter. Sometimes software tries to be too clever and malware authors like to exploit this fact



# Cursory Analysis

'dir' yields much to the keen observer



```
cmd
U:\vx>dir
Volume in drive U has no label.
Volume Serial Number is BC71-445C

Directory of U:\vx

10/07/2004  04:14p    <DIR>          .
10/07/2004  04:14p    <DIR>          ..
08/07/2004  10:29p                11,520 document.txt
                                .scr
          1 File(s)              11,520 bytes
          2 Dir(s)  7,315,795,968 bytes free

U:\vx>
```

# Mary had a little *beagle*...?

```
cmd
U:\>gettext beagle.ex
GetText v1.2 by Rudnai -- Unicode characters are supported
0x00004D: '!This program cannot be run in DOS mode.'
0x0001C0: 'beagle'
0x0001E8: '.rdata'
0x000CDB: '8-updt'
0x002E34: 'et_addr'
0x002E6A: 'wsock32.dll'
0x002E7A: 'Initialize'
0x002E8A: 'eateStreamOnHGGlobal'
0x002E9E: 'ole32.dll'
0x002ED6: 'shlwapi.dll'
0x002EE6: 'ternetCloseHandle'
0x002EFC: 'ternetGetConnectedState'
0x002F26: 'InternetOpenUrlA'
0x002F38: 'wininet.dll'
0x002F46: 'RegCloseKey'
0x002F54: 'RegCreateKeyA'
0x002F64: 'RegQueryValueExA'
0x002F78: 'RegSetValueExA'
0x002F88: 'advapi32.dll'
0x003036: '@hotmail.com'
0x003043: '@msn.com'
0x00304C: '@microsoft'
0x00348D: 'http://www.example.com/1.php'
0x0034AC: 'Date: %s'
0x0034BE: 'Subject: Hi'
0x0034CB: 'From: %s'
0x0034D5: 'Message-ID: <%s>'
0x0034E9: 'MIME-Version: 1.0'
0x0034FC: 'Content-Type: multipart/mixed;'
0x00354C: 'Content-Type: text/plain; charset="us-ascii"'
0x00357A: 'Content-Transfer-Encoding: 7bit'
0x0035AC: 'Content-Type: application/x-msdownload; name="
0x0035EB: 'Content-Transfer-Encoding: base64'
0x00360E: 'Content-Disposition: attachment; filename="I%RAND%.exe"'
0x003687: 'Test. yep.'
0x00369F: 'if exist %1 goto l'
0x0036E1: 'calc.exe'
0x003706: 'SOFTWARE\Windows98'
0x003706: 'SOFTWARE\Microsoft\Windows\CurrentU
0x003734: 'd3dupdate.exe'
0x003742: '\beagle.exe'
0x00377B: 'HELO %s'
0x00378C: 'MAIL FROM:<%s>'
0x00379D: 'RCPT TO:<%s>'
0x0037BC: 'ddd', ' dd MMM yyyy '
0x0037D0: 'HH:mm:ss '
0x0037ED: 'agle_beagle'
0x0037F9: '\bsupld'
0x003B69: 'dDDDDDDDDDDDDDE'
0x003B7A: 'fffffffffffff'
U:\>_

02E34: 'et_addr'
02E6A: 'wsock32.dll'
02E7A: 'Initialize'
02E8A: 'eateStreamOnHGGlobal'
02E9E: 'ole32.dll'
02ED6: 'shlwapi.dll'
02EE6: 'ternetCloseHandle'
02EFC: 'ternetGetConnectedState'
02F26: 'InternetOpenUrlA'
02F38: 'wininet.dll'
02F46: 'RegCloseKey'
02F54: 'RegCreateKeyA'
02F64: 'RegQueryValueExA'
02F78: 'RegSetValueExA'
02F88: 'advapi32.dll'
03036: '@hotmail.com'
03043: '@msn.com'
0304C: '@microsoft'
0348D: 'http://www.example.com/1.php'
034AC: 'Date: %s'
034BE: 'Subject: Hi'
034CB: 'From: %s'
034D5: 'Message-ID: <%s>'
034E9: 'MIME-Version: 1.0'
034FC: 'Content-Type: multipart/mixed;'
0354C: 'Content-Type: text/plain; charset="us
369F: 'if exist %1 goto l'
36E1: 'calc.exe'
3706: 'SOFTWARE\Windows98'
3706: 'SOFTWARE\Microsoft\Windows\CurrentU
3734: 'd3dupdate.exe'
3742: '\beagle.exe'
377B: 'HELO %s'
378C: 'MAIL FROM:<%s>'
379D: 'RCPT TO:<%s>'
37BC: 'ddd', ' dd MMM yyyy '
37D0: 'HH:mm:ss '
37ED: 'agle_beagle'
```

# Section Dissection

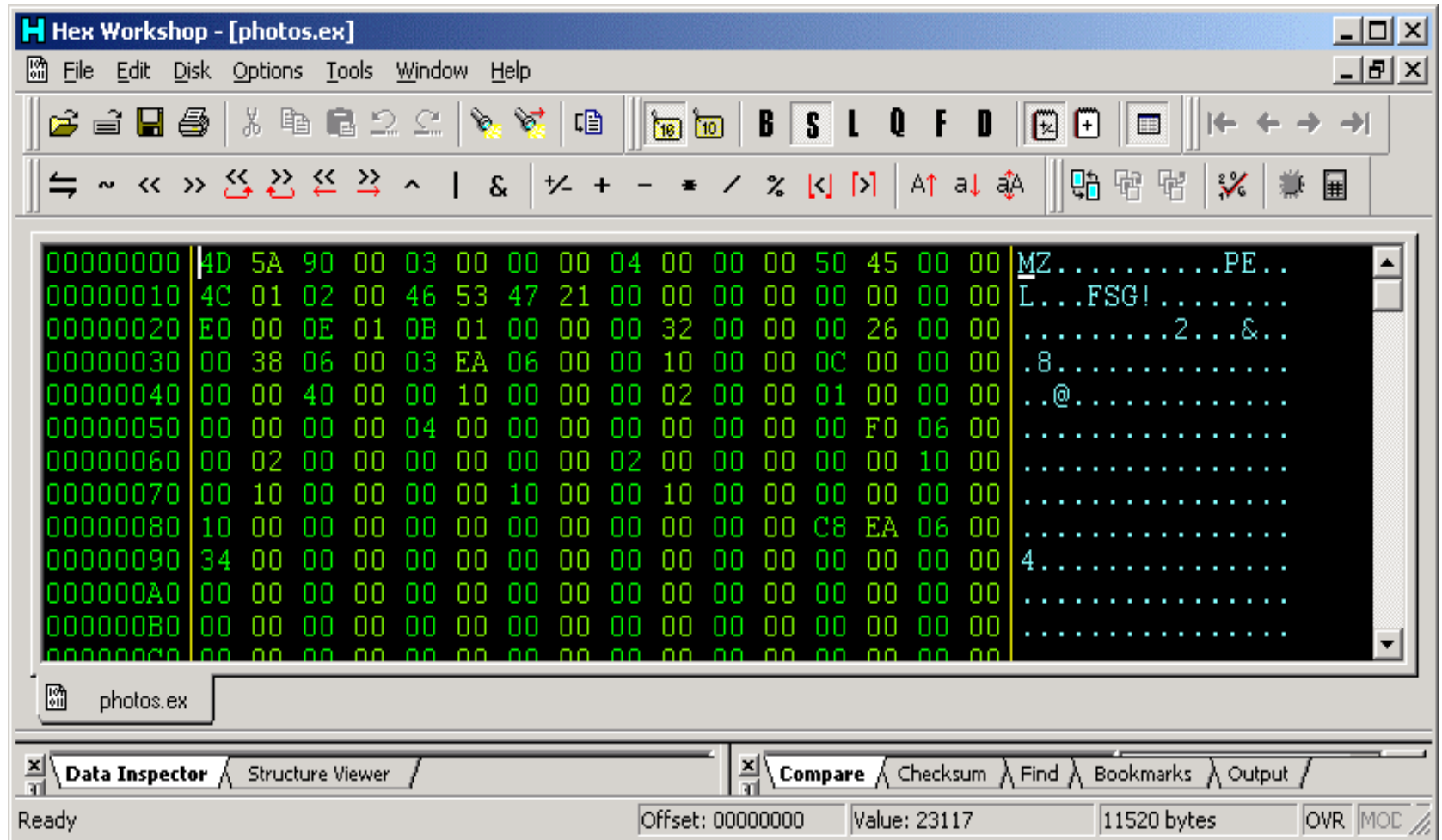
```
cmd
U:\vx>pe photos.ex
PE v1.4 (c) Sophos Plc 1999-2003
Last modified 20030113 by US

lfanew : c
Image base : 400000          Image size : 6f000
Entrypoint RVA : 6ea03
Sec Name      Virtual Physical  Virtual Physical  Flags      CRC32
              Address Address   Size      Size
  0      t      1000      0      6b000      0      rw-      0
  1      6c000   200      3000     2afc      rw-     4b222f67
bytes after last sec      2cfc      4
Entrypoint in section 1
Entrypoint in file at 2c03

be a4 01 40 00 ad 93 ad 97 ad 56 96 b2 80 a4 b6
80 ff 13 73 f9 33 c9 ff 13 73 16 33 c0 ff 13 73
1f b6 80 41 b0 10 ff 13 12 c0 73 fa 75 3c aa eb
e0 ff 53 08 02 f6 83 d9 01 75 0e ff 53 04 eb 26
ac d1 e8 74 2f 13 c9 eb 1a 91 48 c1 e0 08 ac ff
53 04 3d 00 7d 00 00 73 0a 80 fc 05 73 06 83 f8
7f 77 02 41 41 95 8b c5 b6 00 56 8b f7 2b f0 f3
a4 5e eb 9d 8b d6 5e ad 48 74 0a 79 02 ad 50 56

U:\vx>_
```

# All this HEX is Byting me!



# Identifying some common file formats

```
00000000 55 45 73 44 42 41 6F 41 41 41 41 41 41 43 57 6C UESDBAoAAAAACWl
00000010 4D 7A 43 4B 6C 49 41 76 41 44 34 41 41 41 41 2B MzCKlIAVAD4AAAA+
00000020 41 41 41 4B 41 41 41 41 63 32 46 74 63 47 78 6C AAKAAAAC2FtcGxl
00000030 4C 6D 56 34 5A 55 31 61 6B 41 41 44 0D 0A 41 41 LmV4ZU1akAAD..AA
00000040 41 41 42 41 41 41 41 50 2F 2F 41 41 43 34 41 41 AABAAAAP//AAC4AA
00000050 41 41 41 41 41 41 41 45 41 41 41 41 41 41 41 41 AAAAAAAEAAAAAAA
```

# Identifying some common file formats

```
00000000 50 4B 03 04 14 00 00 00 08 00 25 A5 33 30 8A 94 PK.....%.30..
00000010 80 2F 34 1E 00 00 00 3E 00 00 0A 00 00 00 73 61 ./4....>.....sa
00000020 6D 70 6C 65 2E 65 78 65 ED 7B 0B 78 94 D5 B5 E8 mple.exe.{.x....
00000030 9E C9 4C 32 24 13 32 62 A0 BC AC 03 12 3D 15 32 ..L2$.2b.....=.2
00000040 06 02 16 03 D1 09 61 22 54 02 43 26 24 BC 04 26 .....a"T.C&$..&
00000050 99 3F CC 0C F3 EA CC FF 27 C4 4A 9D 34 86 42 A7 .?.....'.J.4.B.
```

# Identifying some common file formats

```
00000000 50 4B 03 04 0A 00 00 00 00 00 25 A5 33 30 8A 94 PK.....%.30..
00000010 80 2F 00 3E 00 00 00 3E 00 00 0A 00 00 00 73 61 ./.>...>.....sa
00000020 6D 70 6C 65 2E 65 78 65 4D 5A 90 00 03 00 00 00 mple.exeMZ.....
00000030 04 00 00 00 FF FF 00 00 B8 00 00 00 00 00 00 00 .....
00000040 40 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 @.....
00000050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
```

# Identifying some common file formats

```
00000000 50 4B 03 04 14 00 01 00 00 00 25 A5 33 30 8A 94 PK.....%.30..
00000010 80 2F 0C 3E 00 00 00 3E 00 00 0A 00 00 00 73 61 ./.>...>.....sa
00000020 6D 70 6C 65 2E 65 78 65 86 72 C6 BD 88 96 EC AA mple.exe.r.....
00000030 D3 82 C3 18 85 91 4C C2 A1 0D 61 7C DF 20 2B E9 .....L...a|. +.
00000040 1D 13 1F B4 B1 84 AE 6E F9 49 41 48 E1 07 19 63 .....n.IAH...c
00000050 A8 8C 43 25 82 53 0F 90 FA 07 50 86 C0 6B 79 A4 ..C%.S....P..ky.
```

# Identifying some common file formats

```
00000000 52 61 72 21 1A 07 00 CF 90 73 00 00 0D 00 00 00 Rar!.....s.....
00000010 00 00 00 00 36 2D 74 20 80 2A 00 00 3E 00 00 00 ....6-t .*..>...
00000020 3E 00 00 02 8A 94 80 2F 25 A5 33 30 14 30 0A 00 >...../%.30.0..
00000030 20 00 00 00 73 61 6D 70 6C 65 2E 65 78 65 4D 5A ...sample.exeMZ
00000040 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 B8 00 .....
00000050 00 00 00 00 00 00 40 00 00 00 00 00 00 00 00 00 .....@.....
```

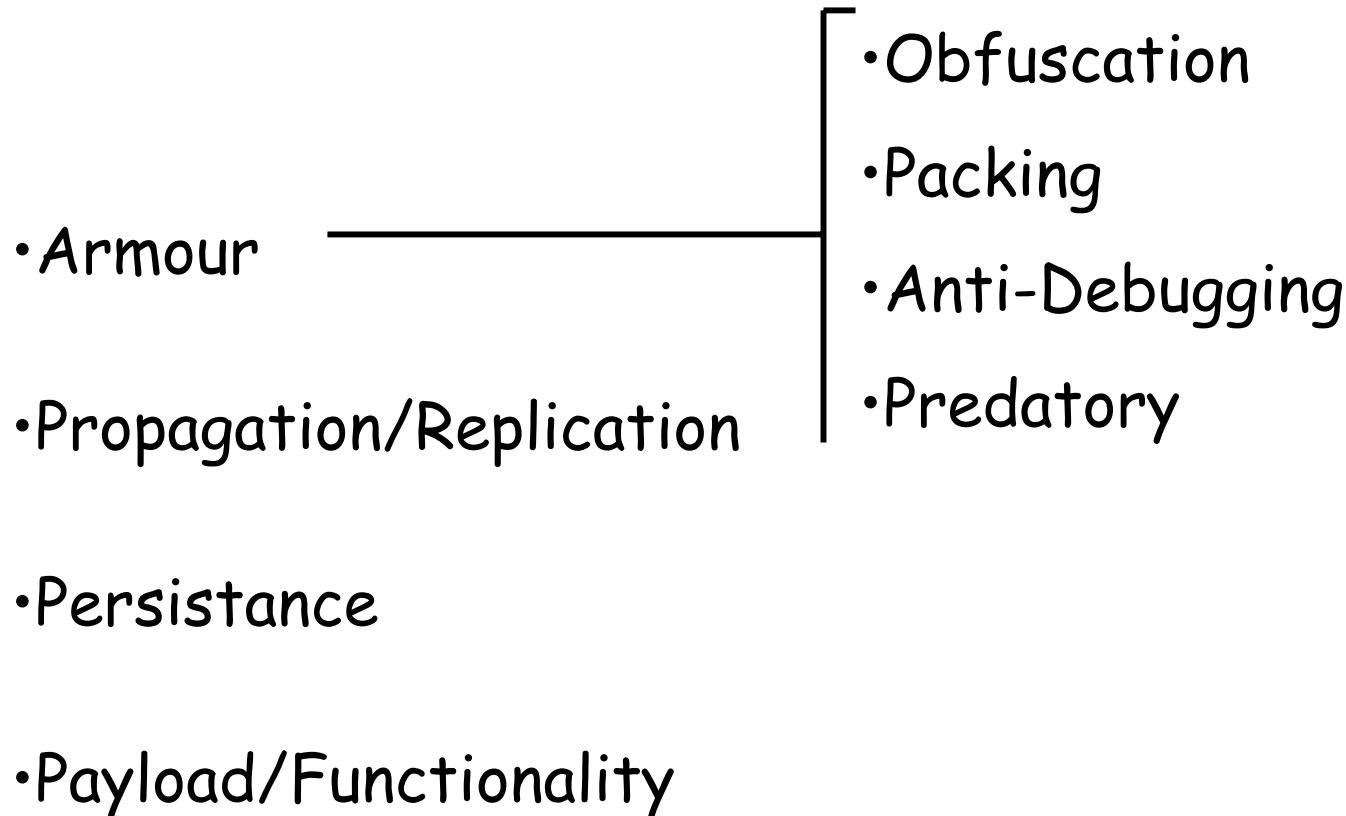
# Identifying some common file formats

```
00000000 52 61 72 21 1A 07 00 CE 99 73 80 00 0D 00 00 00 Rar!.....s.....
00000010 00 00 00 00 18 D5 17 4F 00 FB 97 C1 70 DC 6E C8 .....O....p.n.
00000020 04 97 55 27 00 58 A1 6D 5C B0 AA 6F 46 1A 95 E8 ..U'.X.m\..oF...
00000030 72 83 91 0F 10 1D D0 77 CF 4B D9 96 B8 28 BB 09 r.....w.K...(..
00000040 C3 6B 06 4D 77 3E 67 ED F4 40 F8 5C 20 98 55 D2 .k.Mw>g..@.\ .U.
00000050 69 AE 6F 46 B6 9D 1C 3C 0E 91 69 EF 9E BA 64 EE i.oF...<..i...d.
```

# Identifying some common file formats

```
00000000 D0 CF 11 E0 A1 B1 1A E1 00 00 00 00 00 00 00 00 .....
00000010 00 00 00 00 00 00 00 00 3E 00 03 00 FE FF 09 00 .....>.....
00000020 06 00 00 00 00 00 00 00 00 00 00 00 01 00 00 00 .....
00000030 21 00 00 00 00 00 00 00 00 10 00 00 23 00 00 00 !.....#...
00000040 01 00 00 00 FE FF FF FF 00 00 00 00 20 00 00 00 .....
00000050 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF .....
```

# Physiology of malware



# Physiology of malware

- Armour
- Propagation/Replication
  - By copy
  - By Email
  - By Exploit
  - By Network
  - By Host infection
- Persistence
- Payload/Functionality

# Physiology of malware

- Armour

- Propagation/Replication

- Persistence

- Payload/Functionality

- Autostart folders

- Registry Run keys

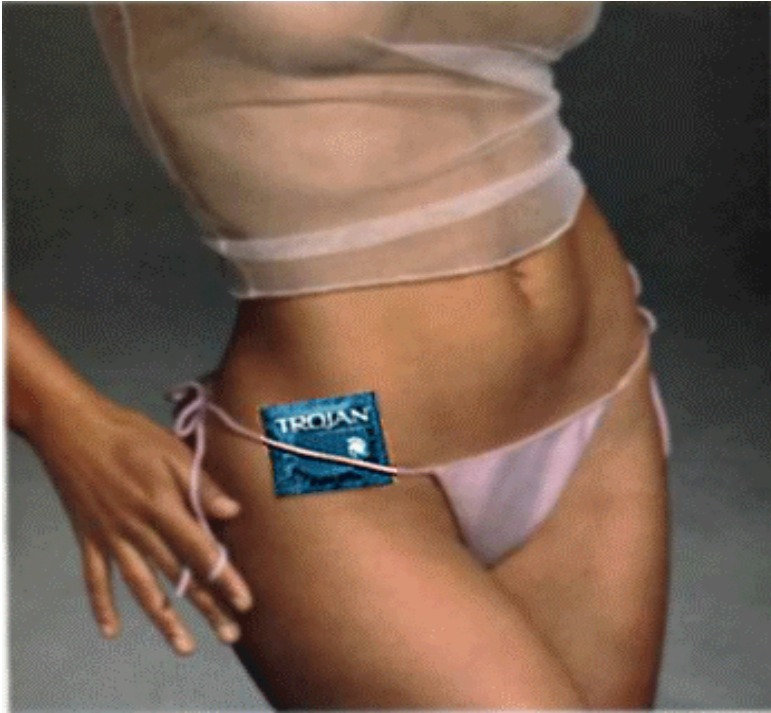
- Browser Help Objects

- Creating Service

# Physiology of malware

- Armour
  - Propagation/Replication
  - Persistence
  - Payload/Functionality
- 
- Harmless
  - Spybot
  - Anon. Proxy
  - Data theft or destruction

# Trojans



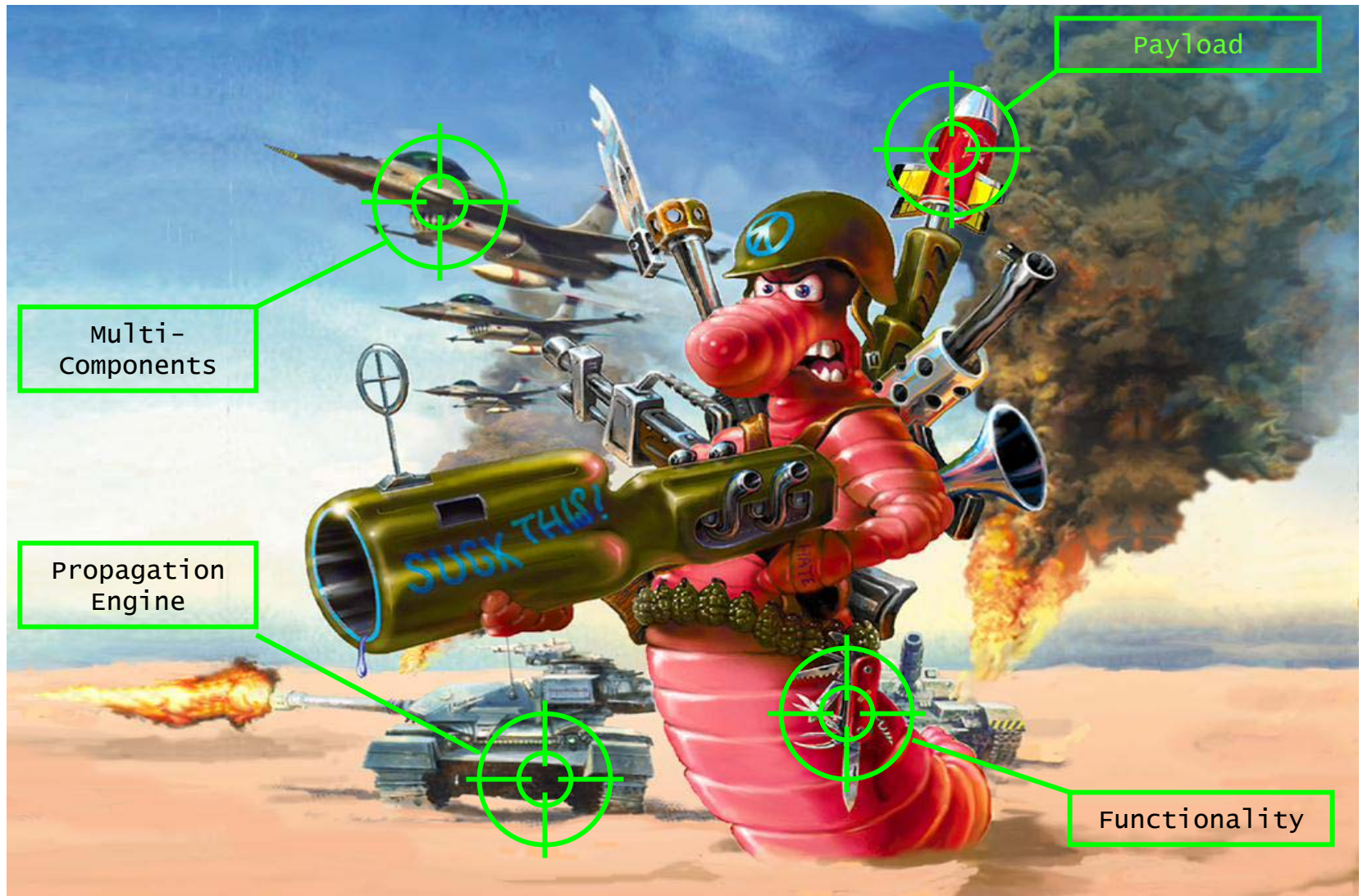
RuxCon 2004

# Trojans



- Appear to be something other than claimed
- Dont replicate
- Size depends on nature of payload
- Typically written in HLL

# Worms



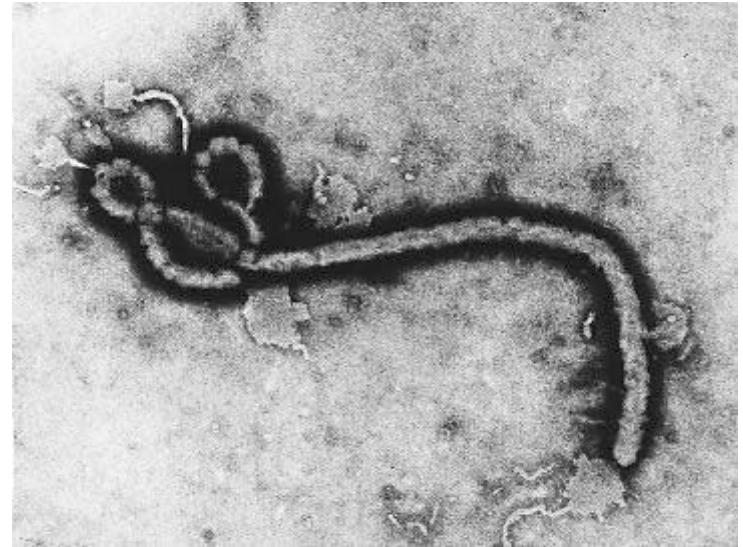
# Viruses (Virii)

- Replicate by parasitic means (infect files/MBR)

- Intimate knowledge of system data-structures

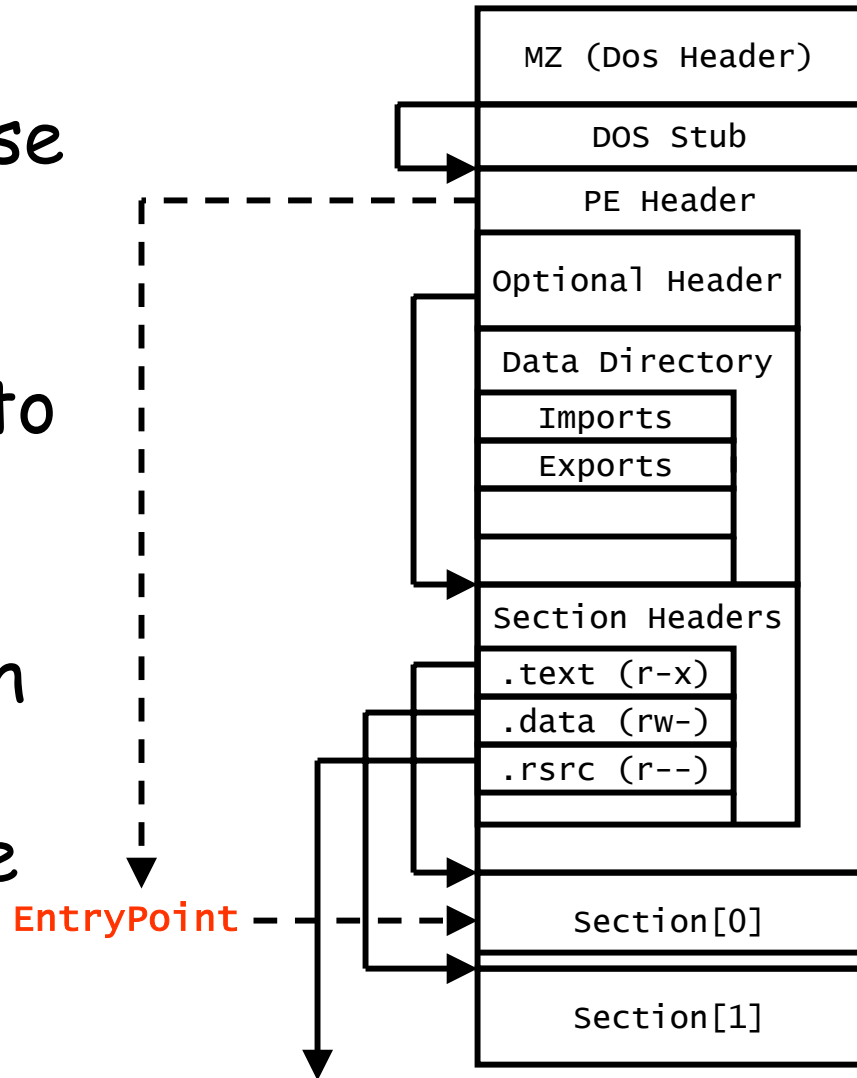
- Generally written in *ASM* and are thus small in size

- Majority of authors tend to be skilled individuals and only produce never-released proof-of-concept. Ofcourse there are always going to be bad apples...



# Basics of Microsoft PE file

•Unpackers and Viral code make use of intimate knowledge of the PE spec. in order to accomplish their goal. An Analyst being familiar with PE can rapidly identify such code and determine its nature



# Toolz of the Trade

- Datarescues Interactive DisAssembler
- SoftIce (debugger)
- Netmon (packet sniffer)
- Tcpmon (sysinternals tcp monitor)
- Regmon (sysinternals registry monitor)
- Filemon (sysinternals file monitor)
- HexWorkshop
- LordPE and/or other PE toolz

# Conducting the Analysis

- Lab Environment (inmates running the Assylum)
- Methodical Madness (chase what glitters brightest)
- Keen observation
- Magic numbers arent magic
- Once bitten, twice shy
- Trail of crumbs always leads to a Cookie Monster

# Static (Dead-Listing) Analysis

- Life begins at the EntryPoint
- Functionality flows from Imports
  - Services rendered via Exports
- One mans trash is anothers treasure  
(where have those strings gone?)

# Behavioural Analysis

- Involves executing the sample on an isolated network and observing its interaction
- When performed in parallel with static analysis can be used to verify coded behaviour and gain any created data not easily divulged through dead-listing
- Under duress of a debugger, the sample may be stopped at specific locations to examine particular system and sample state(s)



# Other wonders to be found at EP

- Position Independent Code {Virii, Libs}
- Unpacking code {UPX, FSG, ASPack, etc}
- Decryptors {simple XOR, complex}
- Obfuscation {Morphine}

# Have I seen you before ?

The screenshot shows the IDA Pro interface with two windows. The top window, 'IDA View-B', displays a list of memory addresses and their corresponding hex values. The bottom window, 'IDA View-A', shows the assembly code for a function named 'start'. The assembly code includes instructions like 'pusha', 'mov esi, offset packed\_data', 'lea edi, [esi-14015h]', 'push edi', 'or ebp, 0FFFFFFFh', and 'jmp short loc\_4190F2'. Below this, there are several 'db' instructions for constants. Further down, there are labels 'loc\_4190E8', 'loc\_4190F2', 'loc\_4190F9', and 'loc\_419100' with their respective assembly instructions. A red dashed box in the hex view highlights a sequence of bytes: 8A 06, 46 07, 88 07, 47 07, 01 DB, 75 07, 8B 1E, 83 FF, 11 DB, 72 ED, B8 01, 00 00, 00, 01 DB, 75 07, 8B 1E, 83 EE. These bytes correspond to the assembly instructions in the code view.

```
public start
start:
pusha
mov esi, offset packed_data
lea edi, [esi-14015h]
push edi
or ebp, 0FFFFFFFh
jmp short loc_4190F2

;
db 90h
db 90h
db 90h
db 90h
db 90h
db 90h
db 90h
;
loc_4190E8:                                ; CODE XREF: UPX1:004190E8
mov al, [esi]
inc esi
mov [edi], al
inc edi
add ebx, ebx
jnz short loc_4190F9

loc_4190F2:                                ; CODE XREF: UPX1:004190F2
mov ebx, [esi]
sub esi, -4
adc ebx, ebx

loc_4190F9:                                ; CODE XREF: UPX1:004190F9
jb short loc_4190E8
mov eax, 1

loc_419100:                                ; CODE XREF: UPX1:00419100
add ebx, ebx
jnz short loc_41910B
mov ebx, [esi]
sub esi, -4
```

Hex View: 8A 06, 46 07, 88 07, 47 07, 01 DB, 75 07, 8B 1E, 83 FF, 11 DB, 72 ED, B8 01, 00 00, 00, 01 DB, 75 07, 8B 1E, 83 EE

# Hiding startup by SEH magic

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

- IDA View-A:** Shows assembly code for the `start` function. It includes instructions like `mov eax, offset loc_4027DD`, `add byte ptr [eax], 28h`, `inc eax`, `add dword ptr [eax], 1234567h`, and `mov eax, offset seh`. It also shows a `patch:` section with `mov [eax], ecx` and `start endp`.
- IDA View-B:** Shows the `seh proc near` function. It includes instructions like `mov eax, 0F04087B1h`, `lea ecx, [eax+10001082h]`, `mov [ecx+1], eax`, `mov edx, [esp+arg_0]`, `mov byte ptr [edx], 0E9h`, `add edx, 5`, `sub ecx, edx`, `mov [edx-4], ecx`, `xor eax, eax`, and `retn`.
- Structures:** A window showing the definition of the `_EXCEPTION_RECORD` structure. The structure is defined as follows:

```
0000 ; Ins/Del : create/delete structure
0000 ; D/A/* : create structure member (d
0000 ; N : rename structure or struct
0000 ; U : delete structure member
0000 ;
0000 _EXCEPTION_RECORD struc ; (sizeof=0x50
0000 ExceptionCode dd ?
0004 ExceptionFlags dd ?
0008 ExceptionRecord dd ?
000C ExceptionAddress dd ?
0010 NumberParameters dd ?
0014 ExceptionInformation dd 15 dup(?)
0050 _EXCEPTION_RECORD ends
```
- Bottom Panel:** Shows the status bar with "AU: idle", "Down", "Disk: 6GB", "00001C0C", and "0040280C".





# Imports, sign-posts to functionality

- Examination of the Imports is a good starting point when fishing for malicious code blocks
- In order to satisfy Persistence, most malware will use a handfull of well defined APIs such as CopyFileA, MoveFileA, CreateFileA, RegCreateKey

# Import Following - CopyFileA

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

- IDA View-A:** Shows assembly code for the `CopyFileA` function. The code includes instructions like `mov esi, 400h`, `lea eax, [ebp+filename_self]`, `push eax`, `call ds:GetModuleFileNameA`, `call ds:GetWindowsDirectoryA`, `call _strlen`, `call unknown_libname_2`, and `call ds:CopyFileA`. It also shows local variables like `loc_4020C9` and `loc_4020F9`.
- Imports:** A window listing imported functions from the kernel library (KERN). The list includes functions such as `GetLastError`, `CreateMutexA`, `GetTickCount`, `CopyFileA` (highlighted), `GetWindowsDirectoryA`, `GetProcAddress`, `VirtualAlloc`, `HeapAlloc`, `GetOEMCP`, `GetACP`, `GetCPInfo`, `GetStringTypeW`, `GetStringTypeA`, `MultiByteToWideChar`, `WriteFile`, `RtlUnwind`, `HeapFree`, `VirtualFree`, `HeapCreate`, `HeapDestroy`, `GetFileType`, `GetStdHandle`, `LCMapStringW`, `LCMapStringA`, `HeapReAlloc`, `GetModuleHandleA`, `GetStartupInfoA`, `GetCommandLineA`, `GetVersion`, `ExitProcess`, `TerminateProcess`, `GetCurrentProcess`, `UnhandledExceptionFilter`, `FreeEnvironmentStringsA`, `FreeEnvironmentStringsW`, and `WideCharToMultiByte`.
- Code Window:** Shows a snippet of assembly code with comments: `; CODE XREF: drop`, `phkResult`, `"SOFTWARE\\Micr`, `hKey`, and `; char *`.
- Status Bar:** Displays system information: `AU: idle`, `Down`, `Disk: 6GB`, `000020F3`, `004020F3: drop_hook+82`.

# Imports, sign-posts to functionality

- If the malware includes networking capability its Imports will show this
- Malware of the Viral variety often calls upon `MapViewOfFileA` during infection
- Malware which spreads by email may contain its own SMTP engine (which can be found by following socket APIs)
- Trojans and backdoor bots often run listen servers awaiting commands

# Import Following - listen

The screenshot displays the IDA Pro interface for a function named `listen_4660`. The assembly code is shown in the main window, with the `Imports` window open on the right. The assembly code includes instructions for pushing registers, calling `socket`, `bind`, and `listen`, and finally `closesocket`. The `Imports` window lists various system functions, with `listen` highlighted.

```
listen_4660 proc near
name=sockaddr ptr -14h
ThreadId=dword ptr -4
push ebp
mov ebp, esp
sub esp, 14h
push esi
xor esi, esi
push edi
push esi
push 1
push 2
call ds:socket
mov edi, eax
cmp edi, 0FFFFFFFFh
jnz short loc_401E8E

loc_401E86:
pop edi
xor eax, eax
pop esi
leave
ret 4

loc_401E8E:
push 4660
mov [ebp+name.sa_family], 2
call ds:htons
word ptr [ebp+name.sa_data], ax
lea eax, [ebp+name]
push 10h
push eax
push edi
mov dword ptr [ebp+name.sa_data+2], esi
call ds:bind
cmp eax, 0FFFFFFFFh
jz short loc_401EC6
push 5
push edi
call ds:listen
cmp eax, 0FFFFFFFFh
jnz short loc_401ECF

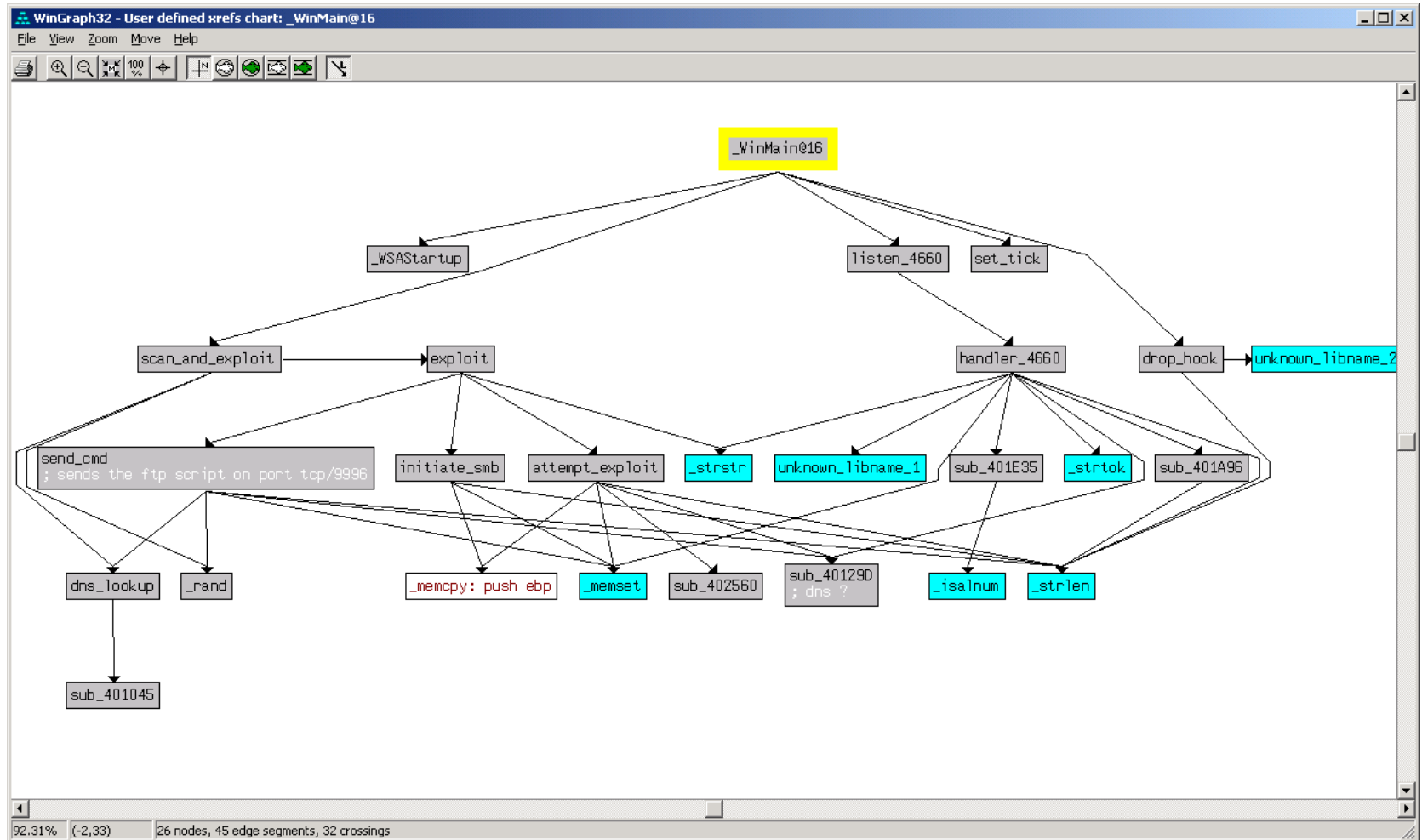
loc_401EC6:
push edi
call ds:closesocket
call ds:closesocket
jmp short loc_401E86
```

The `Imports` window shows the following list of imported functions:

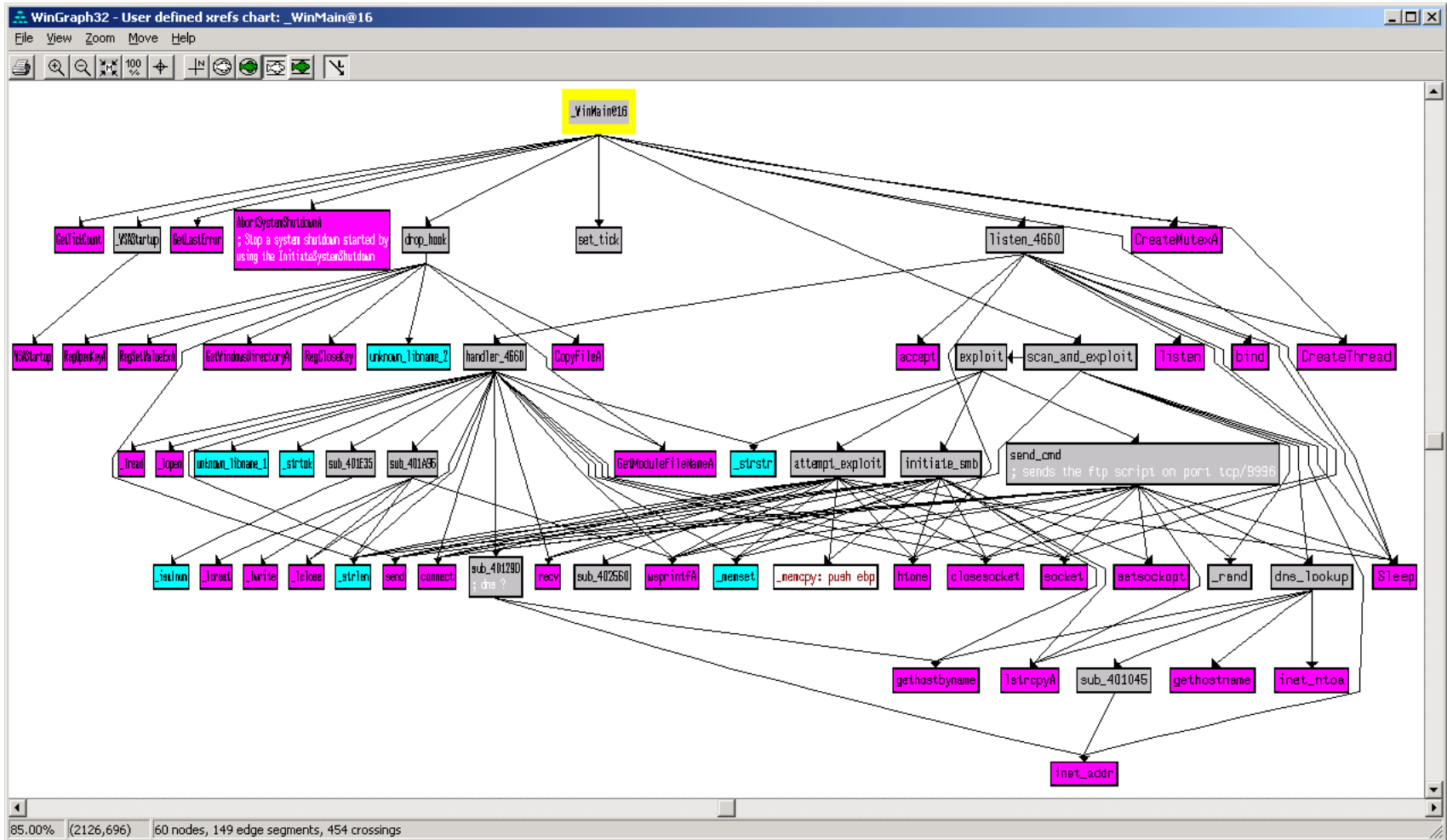
Address	Ordinal	Name	Library
004050C4		FreeEnvironmentStringsW	KERNEL32
004050C8		WideCharToMultiByte	KERNEL32
004050CC		GetEnvironmentStrings	KERNEL32
004050D0		GetEnvironmentStringsW	KERNEL32
004050D4		SetHandleCount	KERNEL32
004050DC		wsprintfA	USER32
004050E4	1	accept	WS2_32
004050E8	16	recv	WS2_32
004050EC	9	htons	WS2_32
004050F0	23	socket	WS2_32
004050F4	21	setsockopt	WS2_32
004050F8	4	connect	WS2_32
004050FC	13	listen	WS2_32
00405100	3	closesocket	WS2_32
00405104	57	gethostname	WS2_32
00405108	12	inet_ntoa	WS2_32
0040510C	11	inet_addr	WS2_32
00405110	52	gethostbyname	WS2_32
00405114	115	WSAStartup	WS2_32
00405118	2	bind	WS2_32
0040511C	19	send	WS2_32



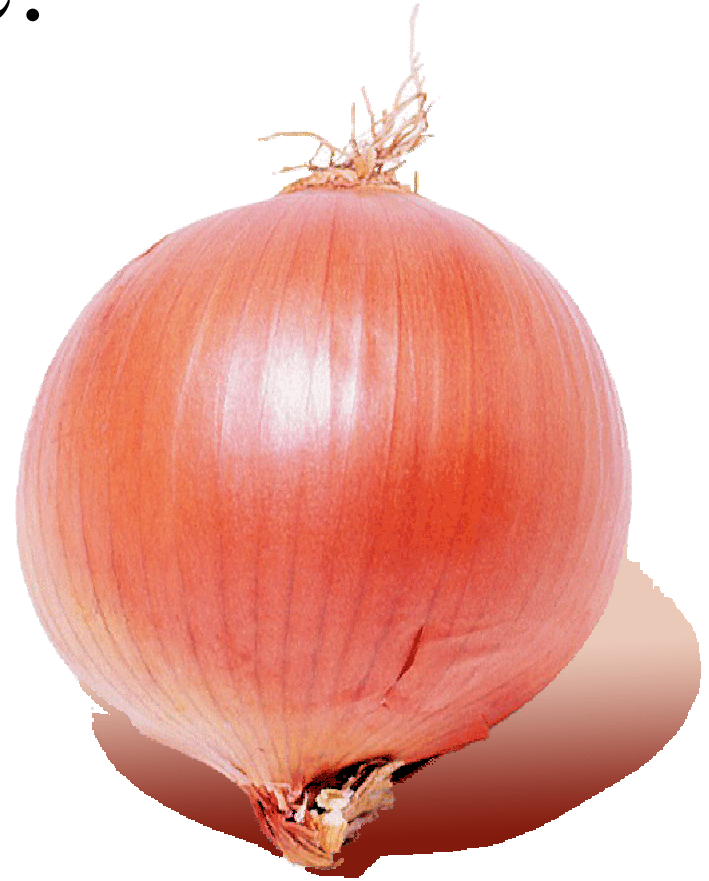
# Structure of most malware is simple



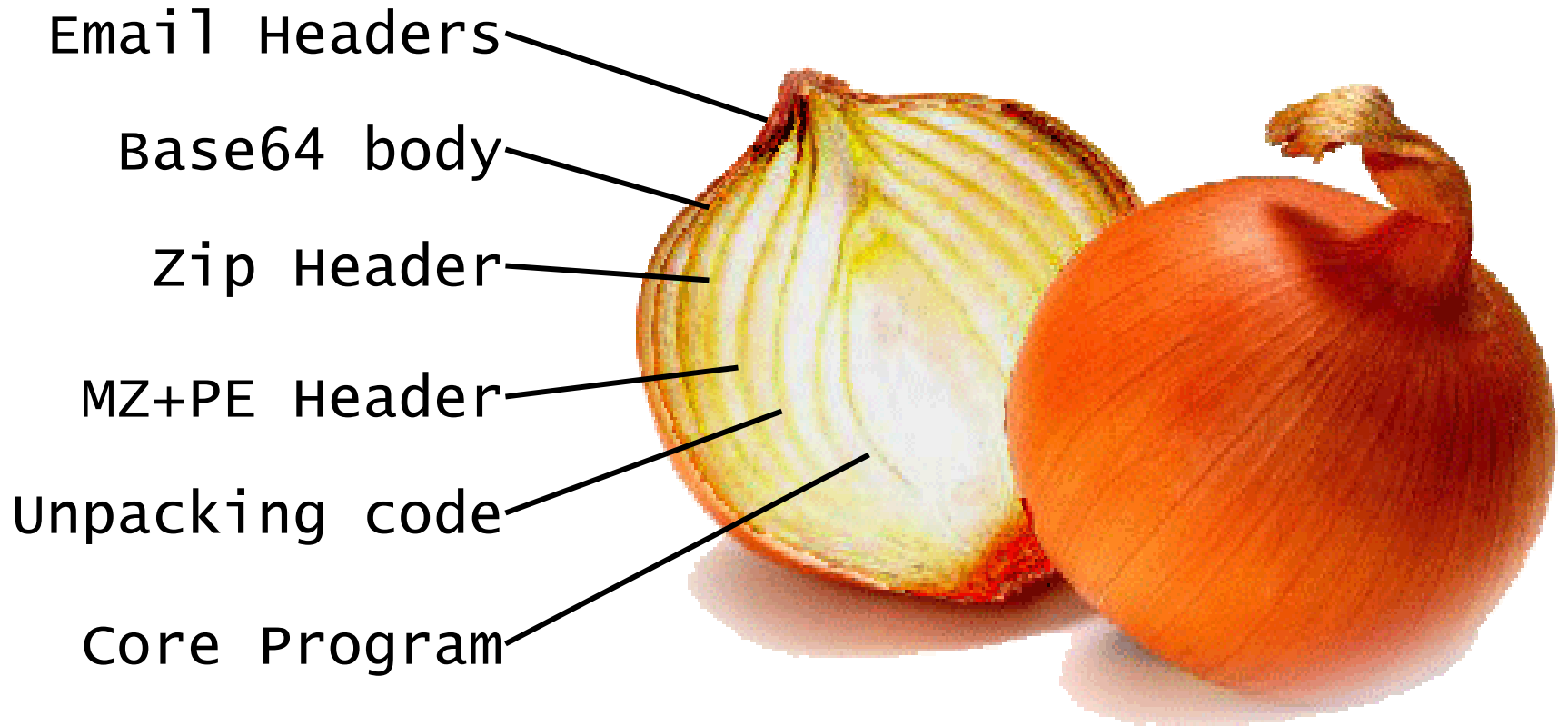
# Module behaviour can be inferred



Analysis can bring a tear to your  
eye!



# Analysis can bring a tear to your eye!



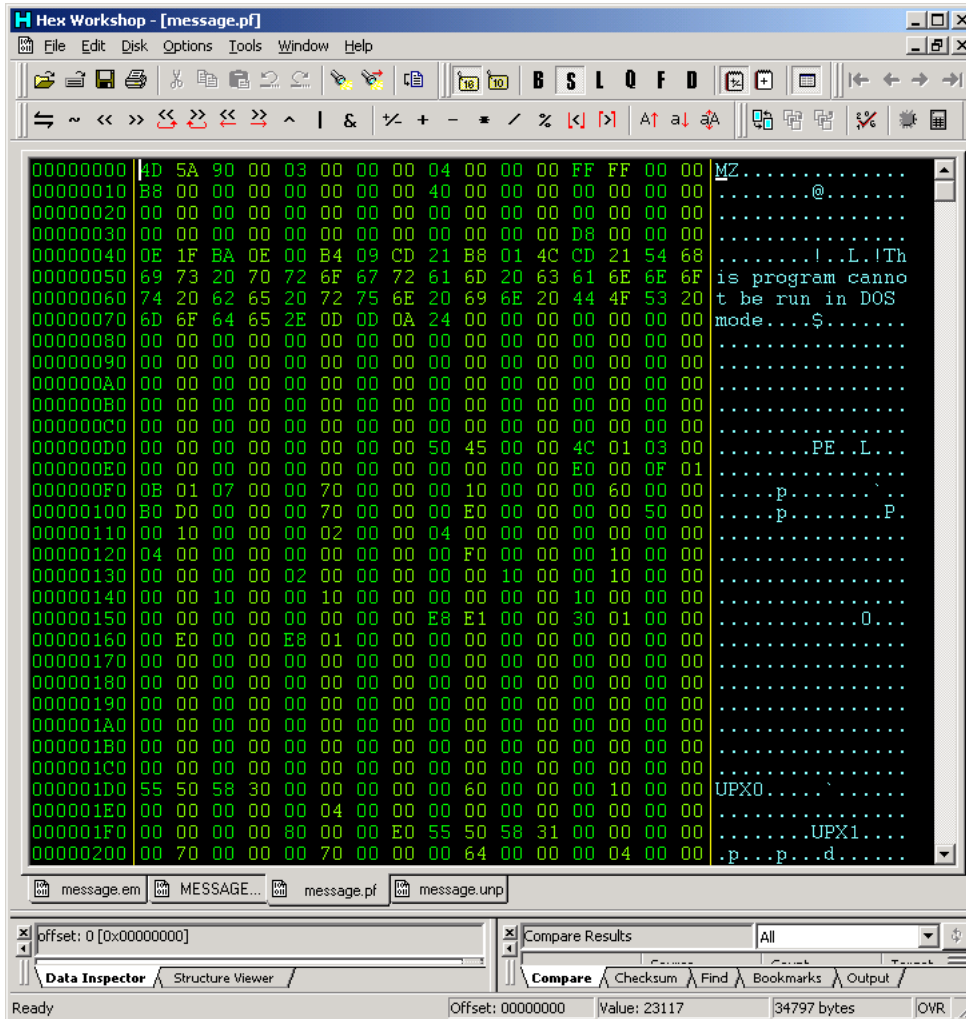
# Brief Example

```
Hex Workshop - [message.em]
File Edit Disk Options Tools Window Help
B S L O F D
~ << >> <> <> ^ | & % / < > A1 a1 aA
000008A0 6C 74 69 2D 70 61 72 74 20 6D 65 73 73 61 67 65 lti-part message
000008B0 20 69 6E 20 4D 49 4D 45 20 66 6F 72 6D 61 74 2E in MIME format.
000008C0 0D 0A 0D 0A 2D 2D 2D 2D 2D 2D 3D 5F 4E 65 78 74 ....-----_Next
000008D0 50 61 72 74 5F 30 30 30 5F 30 30 30 33 5F 38 35 Part_000_0003_85
000008E0 46 36 45 32 30 38 2E 45 41 46 32 43 34 34 45 0D F6E208.EAF2C44E.
000008F0 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 74 .Content-Type: t
00000900 65 78 74 2F 70 6C 61 69 6E 3B 0D 0A 09 63 68 61 ext/plain;...cha
00000910 72 73 65 74 3D 22 57 69 6E 64 6F 77 73 2D 31 32 rset="windows-12
00000920 35 32 22 0D 0A 43 6F 6E 74 65 6E 74 2D 54 72 61 52"..Content-Tra
00000930 6E 73 66 65 72 2D 45 6E 63 6F 64 69 6E 67 3A 20 nsfer-Encoding:
00000940 37 62 69 74 0D 0A 0D 0A 48 65 72 65 20 69 73 20 7bit...Here is
00000950 74 68 65 20 64 6F 63 75 6D 65 6E 74 2E 0D 0A 0D the document...
00000960 0A 0D 0A 2D 2D 2D 2D 2D 2D 3D 5F 4E 65 78 74 50 ....-----_NextP
00000970 61 72 74 5F 30 30 30 5F 30 30 30 33 5F 38 35 46 art_000_0003_85F
00000980 36 45 32 30 38 2E 45 41 46 32 43 34 34 45 0D 0A 6E208.EAF2C44E..
00000990 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 61 70 Content-Type: ap
000009A0 70 6C 69 63 61 74 69 6F 6E 2F 6F 63 74 65 74 2D plication/octet-
000009B0 73 74 72 65 61 6D 3B 0D 0A 09 6E 61 6D 65 3D 22 stream;...name="
000009C0 6D 65 73 73 61 67 65 2E 7A 69 70 22 0D 0A 43 6F message.zip"..Co
000009D0 6E 74 65 6E 74 2D 54 72 61 6E 73 66 65 72 2D 45 ntent-Transfer-E
000009E0 6E 63 6F 64 69 6E 67 3A 20 62 61 73 65 36 34 0D ncoding: base64.
000009F0 0A 43 6F 6E 74 65 6E 74 2D 44 69 73 70 6F 73 69 .Content-Disposi
00000A00 74 69 6F 6E 3A 20 61 74 74 61 63 68 6D 65 6E 74 tion: attachment
00000A10 3B 0D 0A 09 66 69 6C 65 6E 61 6D 65 3D 22 6D 65 ;...filename="me
00000A20 73 73 61 67 65 2E 7A 69 70 22 0D 0A 0D 0A 55 45 ssage.zip"...UE
00000A30 73 44 42 41 6F 41 41 41 41 41 41 43 4F 57 57 54 sDBAcAAAAAACOWWT
00000A40 41 78 73 73 54 4C 37 59 63 41 41 4F 32 48 41 41 AxssTL7YcAAO2HAA
00000A50 41 36 41 41 41 41 62 57 56 7A 63 32 46 6E 5A 53 A6AAAAAbWzce2FnZS
00000A60 35 34 62 48 4D 67 49 43 41 67 49 43 41 67 49 43 54bHMgICAgICAgIC
00000A70 41 67 49 43 41 67 49 43 41 67 0D 0A 49 43 41 67 AgICAgICAg..ICAg
00000A80 49 43 41 67 49 43 41 67 49 43 41 67 49 43 41 67 ICAGICAgICAgICAg
00000A90 49 43 41 67 49 43 41 67 49 43 41 67 49 43 41 67 ICAGICAgICAgICAg
00000AA0 4C 6E 42 70 5A 6B 31 61 6B 41 41 44 41 41 41 LnBpZk1akAADAAAA
```

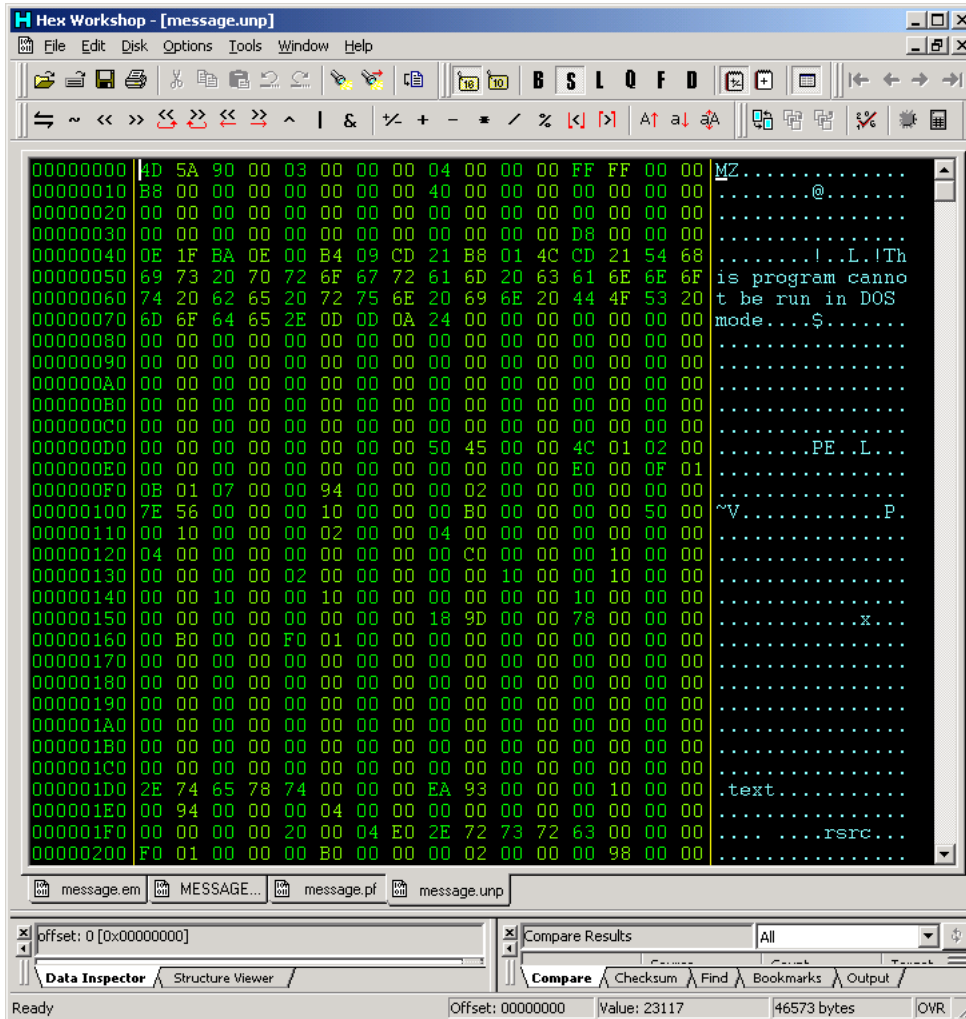
# Brief Example

```
00000000 50 4B 03 04 0A 00 00 00 00 23 96 59 30 31 B2 EK.....#.Y01.
00000010 C4 CB ED 87 00 00 ED 87 00 00 3A 00 00 00 6D 65 .....me
00000020 73 73 61 67 65 2E 78 6C 73 20 20 20 20 20 20 ssage.xls
00000030 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
00000040 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
00000050 20 20 20 20 2E 70 69 66 4D 5A 90 00 03 00 00 00 .pifMZ.....
00000060 04 00 00 00 FF FF 00 00 B8 00 00 00 00 00 00 .....
00000070 40 00 00 00 00 00 00 00 00 00 00 00 00 00 00 @.....
00000080 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000090 00 00 00 00 D8 00 00 00 0E 1F BA 0E 00 B4 09 CD .....
000000A0 21 B8 01 4C CD 21 54 68 69 73 20 70 72 6F 67 72 !..L.!This progr
000000B0 61 6D 20 63 61 6E 6E 6F 74 20 62 65 20 72 75 6E am cannot be run
000000C0 20 69 6E 20 44 4F 53 20 6D 6F 64 65 2E 0D 0D 0A in DOS mode....
000000D0 24 00 00 00 00 00 00 00 00 00 00 00 00 00 00 $.
000000E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000000F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000100 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000110 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000120 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000130 50 45 00 00 4C 01 03 00 00 00 00 00 00 00 00 PE..L.....
00000140 00 00 00 00 E0 00 0F 01 0B 01 07 00 00 70 00 00 .....p.
00000150 00 10 00 00 00 60 00 00 B0 D0 00 00 00 70 00 00 .....p.
00000160 00 E0 00 00 00 00 50 00 00 10 00 00 00 02 00 00 .....P.
00000170 04 00 00 00 00 00 00 04 00 00 00 00 00 00 00 .....
00000180 00 F0 00 00 00 10 00 00 00 00 00 00 02 00 00 .....
00000190 00 00 10 00 00 10 00 00 00 00 10 00 00 10 00 .....
000001A0 00 00 00 00 10 00 00 00 00 00 00 00 00 00 00 .....
000001B0 E8 E1 00 00 30 01 00 00 E0 00 00 E8 01 00 00 .....0.
000001C0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000001D0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000001E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000001F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000200 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
```

# Brief Example



# Brief Example



# In Conclusion

- Malware analysis is focused on identifying malicious modules and documenting behaviour in a timely fashion at the expense of detailed source reconstruction.
- Although the fundamental techniques remain the same, their application is somewhat reactionary toward the sample at hand.
- It is only through laborious analysis of many samples that one begins to notice patterns and trends which can be used to refine the analysis process.

Questions ?