Info 499

Practical Malware Analysis
Lecture 1 | Basic Static Techniques

Static Analysis

Determine functionality without running the program

1. Antivirus (AV) scan can determine maliciousness
    a. But sometimes you shouldn’t...
2. Use hash to identify malware
3. Look at strings, functions, and headers

Antivirus Scanning

Has this been seen before?

- Most AV uses
  - File Signatures
  - Heuristics

Can you fool AV?

- Change or obfuscate code
- New or rare samples

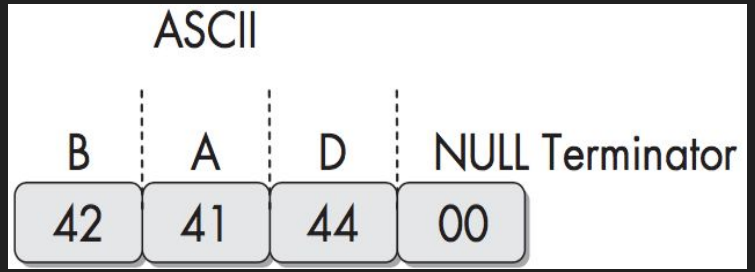
Fingerprinting via Hash

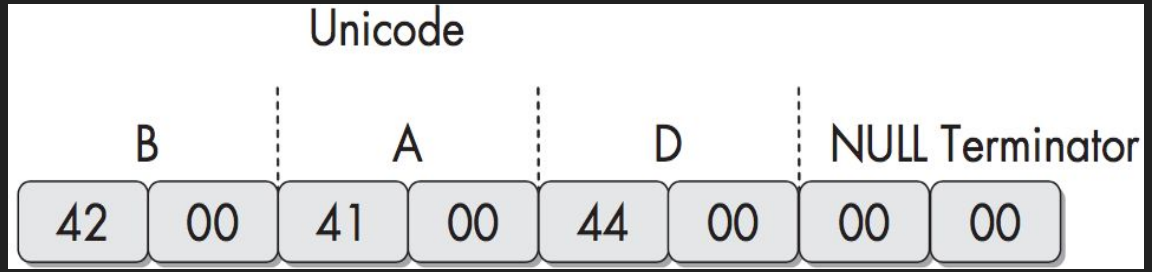
File => hash

- MD5 / SHA-1
  - 9e107d9d372bb6826bd81d3542a419d6 / 2fd4e1c67a2d28fced849ee1bb76e7391b93eb12
- Same input => same output
  - Fingerprint
  - Identify change
- Collision-resistant, quick, one-way
- Little change in input => completely different output

Finding Strings

- Hint towards functionality
- Stored as ASCII or Unicode with NULL terminator
  - ASCII uses 1 byte per character
  - Unicode uses 2 bytes per character





Strings

- Searches for 3+ characters + NULL
- Also outputs
  - Memory addresses
  - CPU instructions
  - Data, garbage

```
C:>strings bp6.ex_
VP3
VW3
t$@
D$4
99.124.22.1 ❹
e-@
GetLayout ❶
GDI32.DLL ❸
SetLayout ❷
M}C
Mail system DLL is invalid.!Send Mail failed to send message. ❺
```

Packing and Obfuscation

Hide program execution

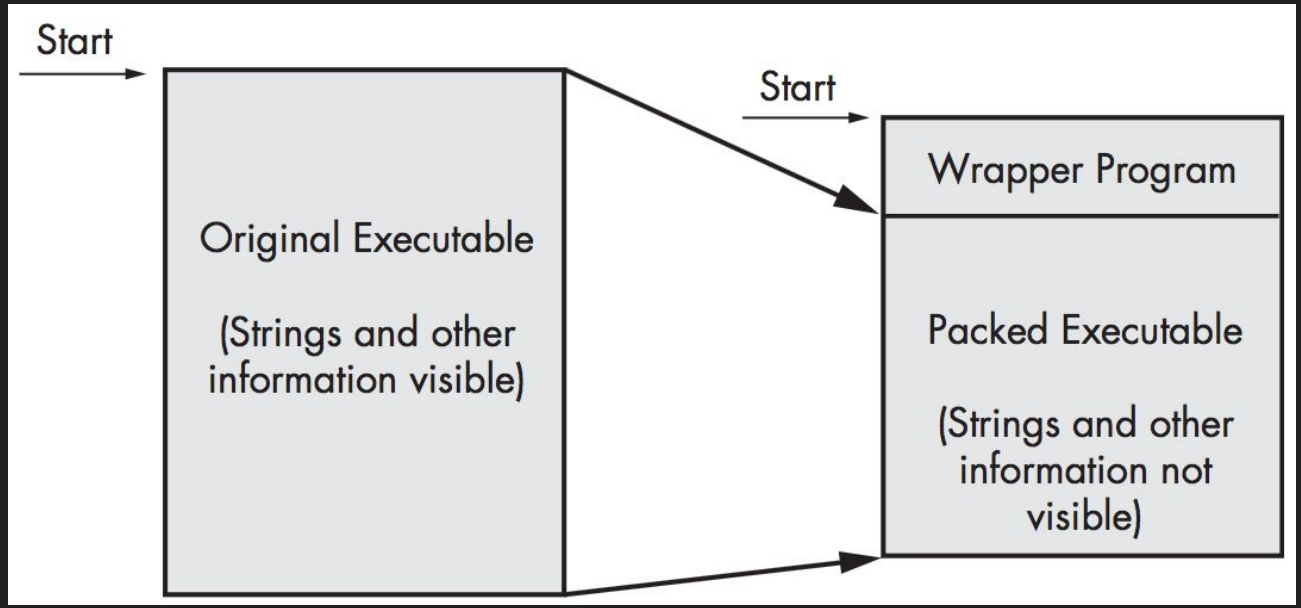
Packing compresses, cannot analyze

- Small strings output
- Indicates maliciousness

LoadLibrary and **GetProcAddress** functions allow access to more functions

Packed Files

- Wrapper decompresses and runs
- Analyze wrapper statically



Detecting Packers

Identify packers / compilers with **PEiD**

Download and use the packer

*Always run malware in a secure **sandbox** environment

*Know your tools' vulnerabilities

Portable Executable File Format

- Windows executables, object code, and dynamic-link libraries (DLLs)
- A data structure used by the Windows OS loader to manage the wrapped executable code
- Begins with a header including
  - Information about the code
  - Type of application
  - Required library functions
  - Space requirements

Linked Libraries and Functions

Borrowing code

- Imports - functions stored in another program
  - Functions shared by many programs
  - Code libraries
- Linking - connecting libraries to the main executable
  - Static or dynamic
  - Type affects information in PE header

Linking

- Static linking
  - Least common, used in UNIX/Linux
  - Code copied into executable
  - PE file header does not indicate linked code
- Runtime linking
  - Commonly used in malware
  - Connect to function only when needed
  - PE file header does not indicate linked code
  - Windows - **LoadLibrary** and **GetProcAddress** allow access to any function in any library on the system

Linking

- Dynamic Linking
  - Most common
  - OS finds libraries on program load
  - When the linked function is called, executes within the library
  - Every library and function used shows up in the PE file header
  - Use **Dependency Walker** to view linked functions

Possible to import functions by Ordinal

- Names of functions won't show up in the executable
- Match the ordinal number in the top pane with its corresponding function in the exports pane.

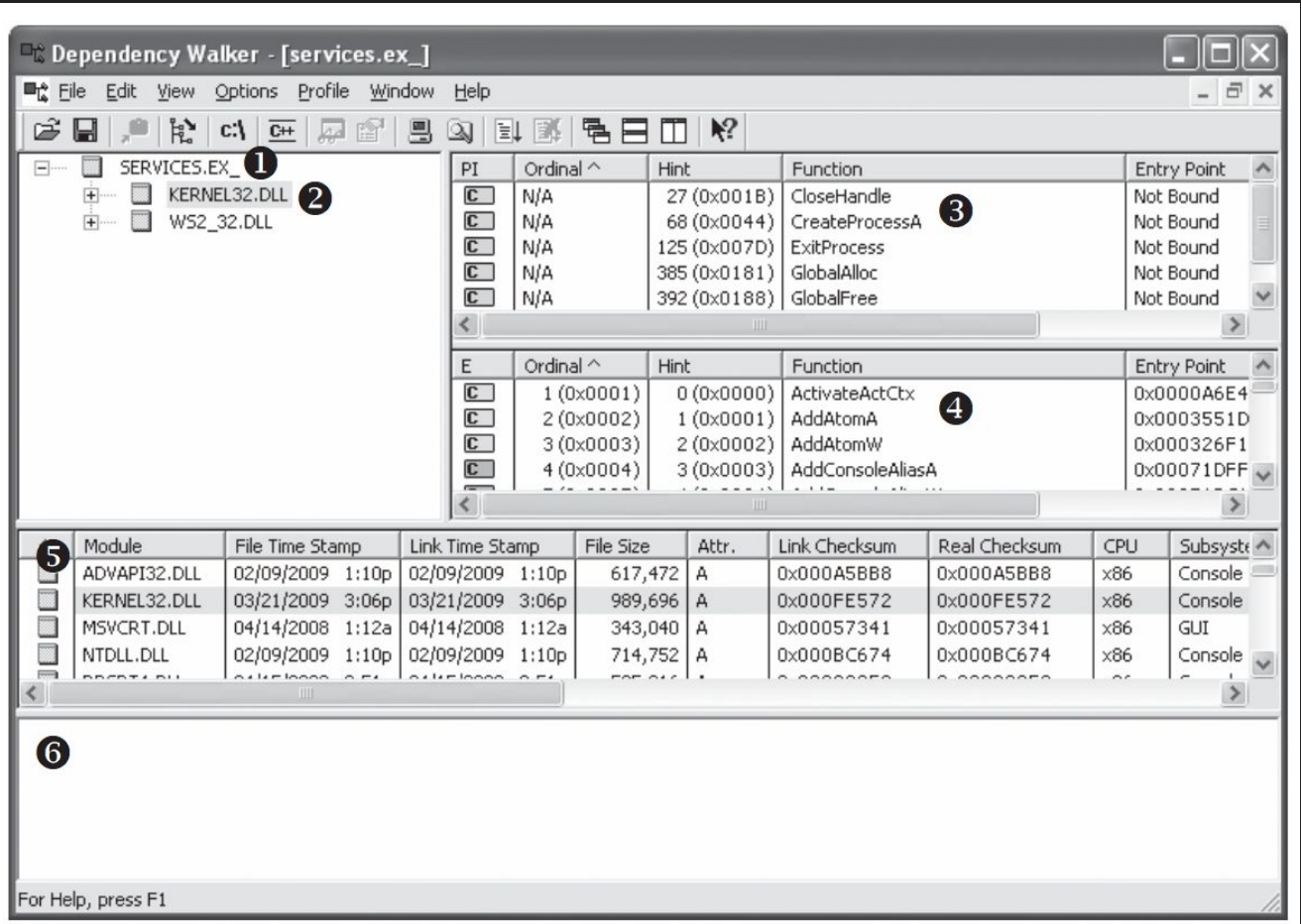


Figure 1-6: The Dependency Walker program

Common DLLs

- Kernel32
  - Core functionality: Access/change memory, files, hardware
- Advapi32
  - Advanced Windows access: Service manager, registry
- User32
  - UI components
- Gdi32
  - Display/change graphics

Common DLLs

- NtdlI
  - Interface to Windows kernel
  - Imported indirectly by Kernel32
  - Explicit import => access to uncommon functions
- WSock32 / Ws2_32
  - Networking
- Wininet
  - High-level networking functions

Notable Naming Conventions

- -Ex (**CreateWindowEx**)
  - Updated function, incompatible with old
  - Twice-updated will have -ExEx
- -A or -W (**CreateDirectoryW**)
  - Accepts a string as a parameter
  - Not in function documentation
  - ASCII or wide (ex. UTF-16)

Imported Functions

- Included in PE file header
- Windows API documented via Microsoft Developer Network (MSDN) library
- Give you a good idea of what the executable does

Exported Functions

- Listed in PE file
- DLL implements and exports functions
- Executables typically import, exports are rare
- Authors name exports following MS docs
  - But they don’t have to (obfuscation)
- Can also be viewed with Dependency Walker

Example: PotentialKeylogger.exe

- Is it packed?
- What interesting functionality can be derived from the functions?

Table 1-2: An Abridged List of DLLs and Functions Imported from *PotentialKeylogger.exe*

Kernel32.dll	User32.dll	User32.dll (continued)
CreateDirectoryW	BeginDeferWindowPos	**ShowWindow**
CreateFileW	CallNextHookEx	ToUnicodeEx
CreateThread	CreateDialogParamW	TrackPopupMenu
DeleteFileW	CreateWindowExW	TrackPopupMenuEx
ExitProcess	DefWindowProcW	TranslateMessage
FindClose	DialogBoxParamW	UnhookWindowsHookEx
FindFirstFileW	EndDialog	UnregisterClassW
FindNextFileW	GetMessageW	UnregisterHotKey
GetCommandLineW	GetSystemMetrics	
GetCurrentProcess	GetWindowLongW	**GDI32.dll**
GetCurrentThread	GetWindowRect	GetStockObject
GetFileSize	GetWindowTextW	SetBkMode
GetModuleHandleW	InvalidateRect	SetTextColor
GetProcessHeap	IsDlgButtonChecked	
GetShortPathNameW	IsWindowEnabled	**Shell32.dll**
HeapAlloc	LoadCursorW	CommandLineToArgvW
HeapFree	LoadIconW	SHChangeNotify
IsDebuggerPresent	LoadMenuW	SHGetFolderPathW
MapViewOfFile	MapVirtualKeyW	ShellExecuteExW
OpenProcess	MapWindowPoints	ShellExecuteW
ReadFile	MessageBoxW	
SetFilePointer	**RegisterClassExW**	**Advapi32.dll**
WriteFile	**RegisterHotKey**	RegCloseKey
	SendMessageA	RegDeleteValueW
	SetClipboardData	RegOpenCurrentUser
	SetDlgItemTextW	RegOpenKeyExW
	SetWindowTextW	RegQueryValueExW
	SetWindowsHookExW	RegSetValueExW

- Open and modify processes and files
  - OpenProcess
  - GetCurrentProcess
  - GetProcessHeap
  - Read/Create/WriteFile
- Search directories
  - FindFirst/FindNextFile
- GUI manipulation
  - RegisterClassEx
  - SetWindowText
  - ShowWindow
- Hook events ಠ_ಠ
  - SetWindowsHookEx
- Notified on hotkey press
  - RegisterHotKey

Table 1-2: An Abridged List of DLLs and Functions Imported from *PotentialKeylogger.exe*

Kernel32.dll	User32.dll	User32.dll (continued)
CreateDirectoryW	BeginDeferWindowPos	**ShowWindow**
CreateFileW	CallNextHookEx	ToUnicodeEx
CreateThread	CreateDialogParamW	TrackPopupMenu
DeleteFileW	CreateWindowExW	TrackPopupMenuEx
ExitProcess	DefWindowProcW	TranslateMessage
FindClose	DialogBoxParamW	UnhookWindowsHookEx
FindFirstFileW	EndDialog	UnregisterClassW
FindNextFileW	GetMessageW	UnregisterHotKey
GetCommandLineW	GetSystemMetrics	
GetCurrentProcess	GetWindowLongW	**GDI32.dll**
GetCurrentThread	GetWindowRect	GetStockObject
GetFileSize	GetWindowTextW	SetBkMode
GetModuleHandleW	InvalidateRect	SetTextColor
GetProcessHeap	IsDlgButtonChecked	
GetShortPathNameW	IsWindowEnabled	**Shell32.dll**
HeapAlloc	LoadCursorW	CommandLineToArgvW
HeapFree	LoadIconW	SHChangeNotify
IsDebuggerPresent	LoadMenuW	SHGetFolderPathW
MapViewOfFile	MapVirtualKeyW	ShellExecuteExW
OpenProcess	MapWindowPoints	ShellExecuteW
ReadFile	MessageBoxW	
SetFilePointer	**RegisterClassExW**	**Advapi32.dll**
WriteFile	**RegisterHotKey**	RegCloseKey
	SendMessageA	RegDeleteValueW
	SetClipboardData	RegOpenCurrentUser
	SetDlgItemTextW	RegOpenKeyExW
	SetWindowTextW	RegQueryValueExW
	SetWindowsHookExW	RegSetValueExW

- More GUI-related imports
  - GDI32.dll
- Launch other programs
  - Shell32.dll
- Use the registry
  - Advapi32.dll

Searching for strings that look like registry keys turns up:

Software\Microsoft\Windows\CurrentVersion\Run

which controls Windows startup programs ಠ_ಠ

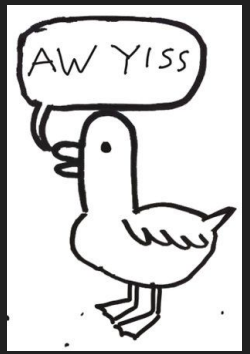
Table 1-2: An Abridged List of DLLs and Functions Imported from *PotentialKeylogger.exe*

Kernel32.dll	User32.dll	User32.dll (continued)
CreateDirectoryW	BeginDeferWindowPos	**ShowWindow**
CreateFileW	CallNextHookEx	ToUnicodeEx
CreateThread	CreateDialogParamW	TrackPopupMenu
DeleteFileW	CreateWindowExW	TrackPopupMenuEx
ExitProcess	DefWindowProcW	TranslateMessage
FindClose	DialogBoxParamW	UnhookWindowsHookEx
FindFirstFileW	EndDialog	UnregisterClassW
FindNextFileW	GetMessageW	UnregisterHotKey
GetCommandLineW	GetSystemMetrics	
GetCurrentProcess	GetWindowLongW	**GDI32.dll**
GetCurrentThread	GetWindowRect	GetStockObject
GetFileSize	GetWindowTextW	SetBkMode
GetModuleHandleW	InvalidateRect	SetTextColor
GetProcessHeap	IsDlgButtonChecked	
GetShortPathNameW	IsWindowEnabled	**Shell32.dll**
HeapAlloc	LoadCursorW	CommandLineToArgvW
HeapFree	LoadIconW	SHChangeNotify
IsDebuggerPresent	LoadMenuW	SHGetFolderPathW
MapViewOfFile	MapVirtualKeyW	ShellExecuteExW
OpenProcess	MapWindowPoints	ShellExecuteW
ReadFile	MessageBoxW	
SetFilePointer	**RegisterClassExW**	**Advapi32.dll**
WriteFile	**RegisterHotKey**	RegCloseKey
	SendMessageA	RegDeleteValueW
	SetClipboardData	RegOpenCurrentUser
	SetDlgItemTextW	RegOpenKeyExW
	SetWindowTextW	RegQueryValueExW
	SetWindowsHookExW	RegSetValueExW

PotentialKeylogger.exe Exports

- LowLevelKeyboardProc and LowLevelMouseProc
  - Used with SetWindowsHookEx (MSDN) to connect an event to a function
  - Function called on low-level keyboard events

- Author named export after MS function



PotentialKeylogger.exe Hypothesis

- Probably a keylogger
  - Uses SetWindowsHookEx to log keystrokes
  - Has a GUI for the author
  - Author uses hotkey to access GUI
  - Uses a registry key to run the keylogger on startup

Example: PackedProgram.exe

Is it packed?

- Short import list
- No readable strings
- Requires dynamic

Table 1-3: DLLs and Functions Imported from *PackedProgram.exe*

Kernel32.dll	User32.dll
GetModuleHandleA	MessageBoxA
LoadLibraryA	
GetProcAddress	
ExitProcess	
VirtualAlloc	
VirtualFree	

PE File Headers and Sections

Header => metadata about file

Sections => pieces of the file

- .text
  - CPU instructions
  - Only section with code
- .rdata
  - Import/export information
  - Globally accessible read-only data
  - Sometimes split into .idata and .edata

PE File Sections

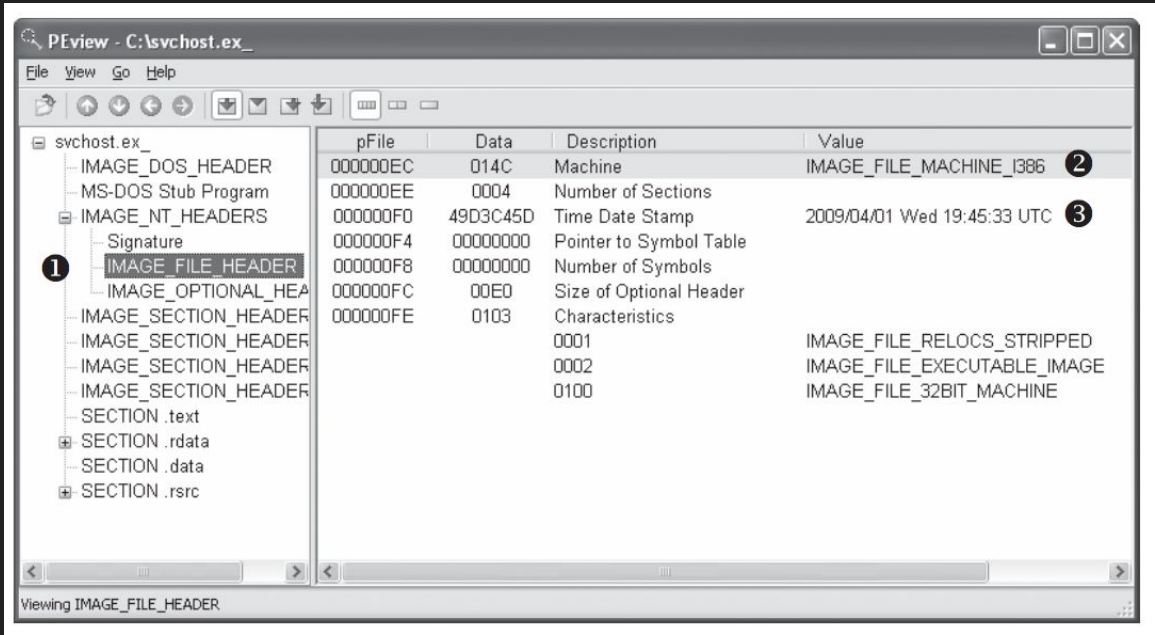
- .data
  - Global data
- .pdata (x64 only)
  - Exception-handling info
- .rsrc
  - Resources (icons, images, menus, strings)
- .reloc
  - Info regarding relocation of library files

PE File Sections

- Section names may vary per compiler
- Windows doesn’t care
- Section names can be obfuscated

PEview

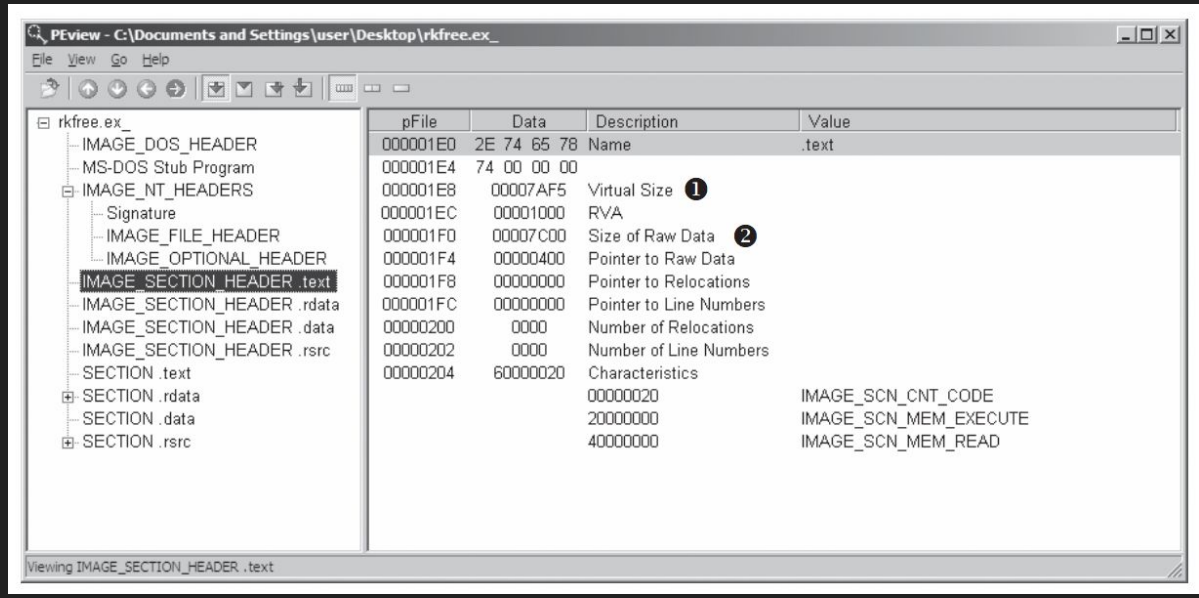
1. Main parts
2. Basic info
3. Time Date Stamp



IMAGE_OPTIONAL_HEADER (not shown)

PEview

1. Virtual Size
2. Size of Raw Data



PEview

Section Information for PotentialKeylogger.exe

Section	Virtual size	Size of raw data
.text	7AF5	7C00
.data	17A0	0200
.rdata	1AF5	1C00
.rsrc	72B8	7400

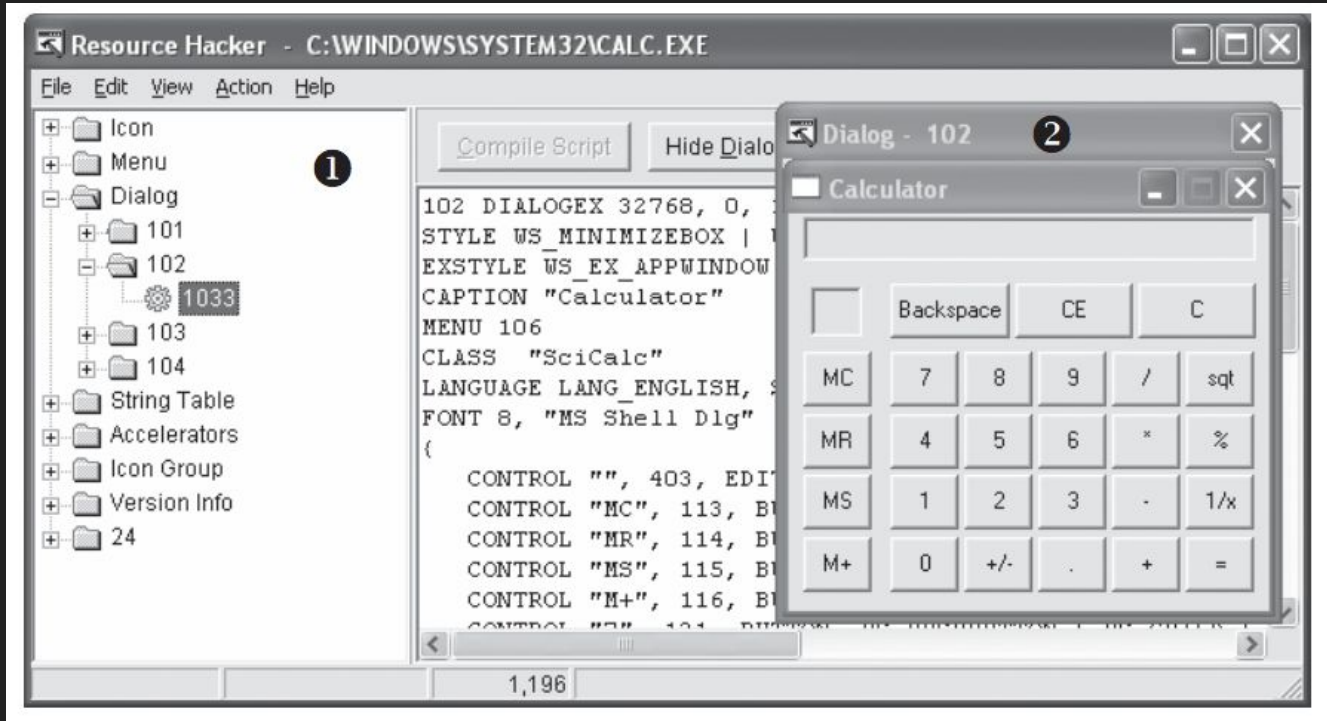
Section Information for PackedProgram.exe

Name	Virtual size	Size of raw data
.text	A000	0000
.data	3000	0000
.rdata	4000	0000
.rsrc	19000	3400
Dijfpds	20000	0000
.sdfuok	34000	3313F
Kijijl	1000	0200

Resource Hacker

Only section accessible thus far is .rsrc

- Icon
- Menu
- Dialog
- String Table
- Version Info



Other PE File Tools

PEBrowse Professional

- Similar to PEview, better view of .rsrc

PE Explorer

- GUI for navigating / editing, including a resource editor
- $$$

Summary

Field	Information revealed
Imports	Functions from other libraries that are used by the malware
Exports	Functions in the malware that are meant to be called by other programs or libraries
Time Date Stamp	Time when the program was compiled
Sections	Names of sections in the file and their sizes on disk and in memory
Subsystem	Indicates whether the program is a command-line or GUI application
Resources	Strings, icons, menus, and other information included in the file

Static analysis is useful, but only the first step...

Labs

- Given little or no information about the samples
- Generically named
- Malicious file provided
- Short questions, short answers
- Detailed analysis
- Answers in Appendix C

Resources

Use VirusTotal to scan or search for information on files, URLs, domains, IPs, hashes, etc.

Use a public sandbox such as Hybrid Analysis or Malwr to perform static (and some dynamic) analysis for you

Set up a private sandbox such as cuckoo for private analysis

Strings for Windows

Dependency Walker (Windows)

Resources

PEview (Windows)

Resource Hacker (Windows)

PEBrowse Professional (Windows)

PE Explorer (Windows)

PEiD (Windows)