

Malware Analysis

The Goals of Malware Analysis

Incident Response

- Case history
 - A medical clinic with 10 offices found malware on one of their workstations
 - Hired a consultant to clean & re-image that machine
- All done—case closed?

Incident Response

- After malware is found, you need to know
 - Did an attacker implant a rootkit or trojan on your systems?
 - Is the attacker really gone?
 - What did the attacker steal or add?
 - How did the attack get in
 - Root-cause analysis

Breach clean-up cost LinkedIn nearly \$1 million, another \$2-3 million in upgrades

Summary: LinkedIn executives reveal on quarterly earnings call just what the June theft of 6.5 million passwords cost the company in forensic work and on-going security updates.



By John Fontana for Identity Matters | August 3, 2012 -- 17:10 GMT (10:10 PDT)

 Follow @johnfontana

Comments

0



Vote

1



Like

4



Tweet

51



Share

more +

LinkedIn spent nearly \$1 million investigating and unraveling the theft of 6.5 million passwords in June and plans to spend up to \$3 million more updating security on its social networking site.

- Link Ch 1a

Malware Analysis

- Dissecting malware to understand
 - How it works
 - How to identify it
 - How to defeat or eliminate it
- A critical part of incident response



The Goals of Malware Analysis

- Information required to respond to a network intrusion
 - Exactly what happened
 - Ensure you've located all infected machines and files
 - How to measure and contain the damage
 - Find signatures for intrusion detection systems

Signatures





- Host-based signatures
 - Identify files or registry keys on a victim computer that indicate an infection
 - Focus on what the malware did to the system, not the malware itself
 - Different from antivirus signature
- Network signatures
 - Detect malware by analyzing network traffic
 - More effective when made using malware analysis


False Positives


CBS San Francisco Your Home Buy Tickets More FOLLOW US   LOGIN

City College Of San Francisco Computer Lab Security Breached

January 13, 2012 1:56 PM

Share this  1  3  0  2 View Comments

 [Share CBS Local with your friends. Add us to your Timeline. What's this?](#)




City College of San Francisco (CCSF)


SAN FRANCISCO (KCBS) – The personal banking data from thousands of City College of San Francisco students, faculty and staff may be at risk because of a virus that infiltrated one [computer](#) lab – perhaps years ago.

Incredibly, the breach was only discovered recently – over the Thanksgiving holiday weekend.


KCBS' Holly Quan Reports:



What's most disturbing isn't that the IP addresses identified as receiving transmissions belong to the Russian Mafia –

 [Click here to play audio](#)

Sponsored Links



\$28/Hr Data Entry Jobs At Home
\$28/hr Part-Time Job Openi...
[StunningLifeStyle.com/Finance](#)

Malware Analysis Techniques

Static v. Dynamic Analysis

- Static Analysis
 - Examines malware without running it
 - Tools: VirusTotal, strings, a disassembler like IDA Pro
- Dynamic Analysis
 - Run the malware and monitor its effect
 - Use a virtual machine and take snapshots
 - Tools: RegShot, Process Monitor, Process Hacker, CaptureBAT
 - RAM Analysis: Mandant Redline and Volatility

Basic Analysis

- Basic static analysis
 - View malware without looking at instructions
 - Tools: VirusTotal, strings
 - Quick and easy but fails for advanced malware and can miss important behavior
- Basic dynamic analysis
 - Easy but requires a safe test environment
 - Not effective on all malware

Advanced Analysis

- Advanced static analysis
 - Reverse-engineering with a disassembler
 - Complex, requires understanding of assembly code
- Advanced Dynamic Analysis
 - Run code in a debugger
 - Examines internal state of a running malicious executable

Types of Malware

Types of Malware

- Backdoor
 - Allows attacker to control the system
- Botnet
 - All infected computers receive instructions from the same Command-and-Control (C&C) server
- Downloader
 - Malicious code that exists only to download other malicious code
 - Used when attacker first gains access

Types of Malware

- Information-stealing malware
 - Sniffers, keyloggers, password hash grabbers
- Launcher
 - Malicious program used to launch other malicious programs
 - Often uses nontraditional techniques to ensure stealth or greater access to a system
- Rootkit
 - Malware that conceals the existence of other code
 - Usually paired with a backdoor

Types of Malware

- Scareware
 - Frightens user into buying something
 - Link Ch 1b

Fake FBI warning tricks man into surrendering himself for possession of child porn

29 Jul, 2013 | by Nishtha Kanal



3



0



3



Secure Your Application Today!



CHECKMARX

Learn more >

Here's a weird one. We've heard of viruses and malware bringing harm to computers but in a rare instance, a "ransomware" has brought a positive outcome. A man in the US turned himself in to the police after a pop-up caused by a ransomware informed him that child porn had been identified on his machine.

Jay Matthew Riley, a 21-year-old from Virginia was browsing the Internet, when a pop-up containing an "FBI warning" informed him that it had detected child pornography on his machine. The message went on to tell Riley to pay up a fine online or face the consequences.

Types of Malware

- Spam-sending malware
 - Attacker rents machine to spammers
- Worms or viruses
 - Malicious code that can copy itself and infect additional computers

Mass v. Targeted Malware

- Mass malware
 - Intended to infect as many machines as possible
 - Most common type
- Targeted malware
 - Tailored to a specific target
 - Very difficult to detect, prevent, and remove
 - Requires advanced analysis
 - Ex: Stuxnet

General Rules for Malware Analysis

General Rules for Malware Analysis

- Don't Get Caught in Details
 - You don't need to understand 100% of the code
 - Focus on key features
- Try Several Tools
 - If one tool fails, try another
 - Don't get stuck on a hard issue, move along
- Malware authors are constantly raising the bar

Ch 2: Basic Static Analysis

Techniques

- Antivirus scanning
- Hashes
- A file's strings, functions, and headers

Antivirus Scanning

Only a First Step

- Malware can easily change its signature and fool the antivirus
- VirusTotal is convenient, but using it may alert attackers that they've been caught
 - Link Ch 2a



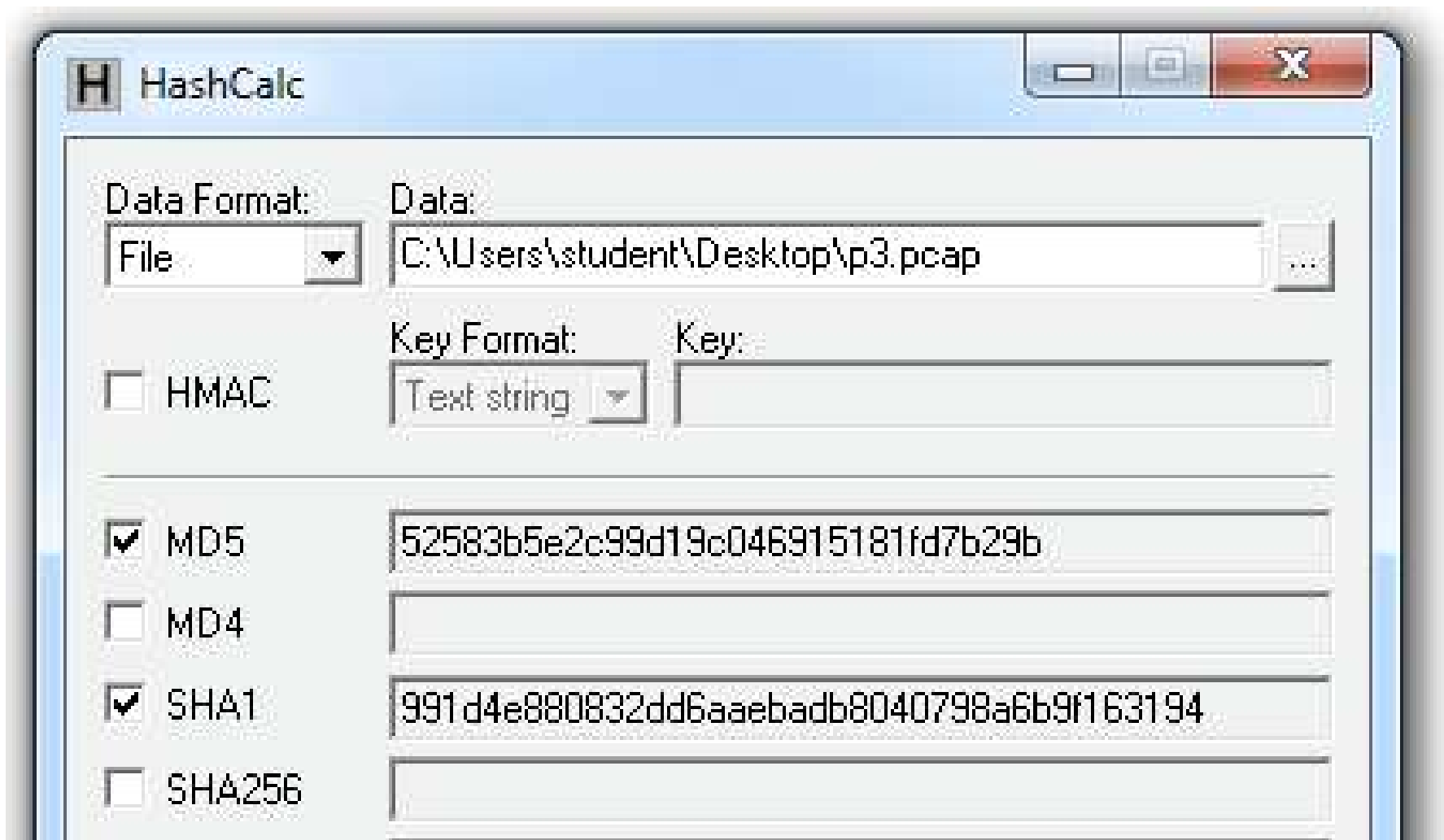
Hashing

A fingerprint for malware

Hashes

- MD5 or SHA-1
- Condenses a file of any size down to a fixed-length fingerprint
- Uniquely identifies a file well in practice
 - There are MD5 collisions but they are not common
 - Collision: two different files with the same hash

HashCalc



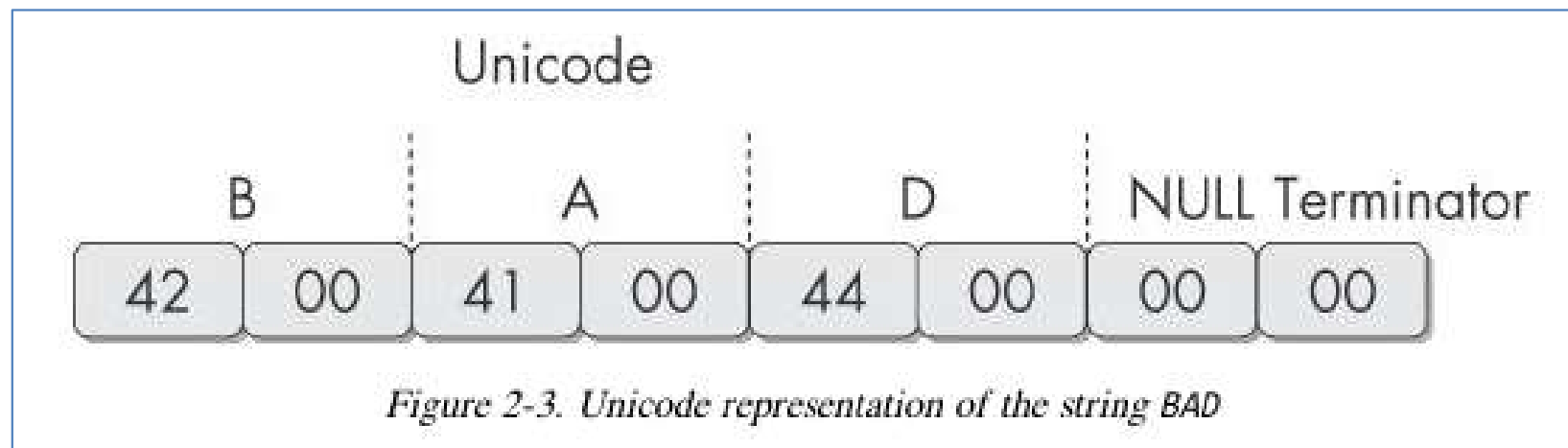
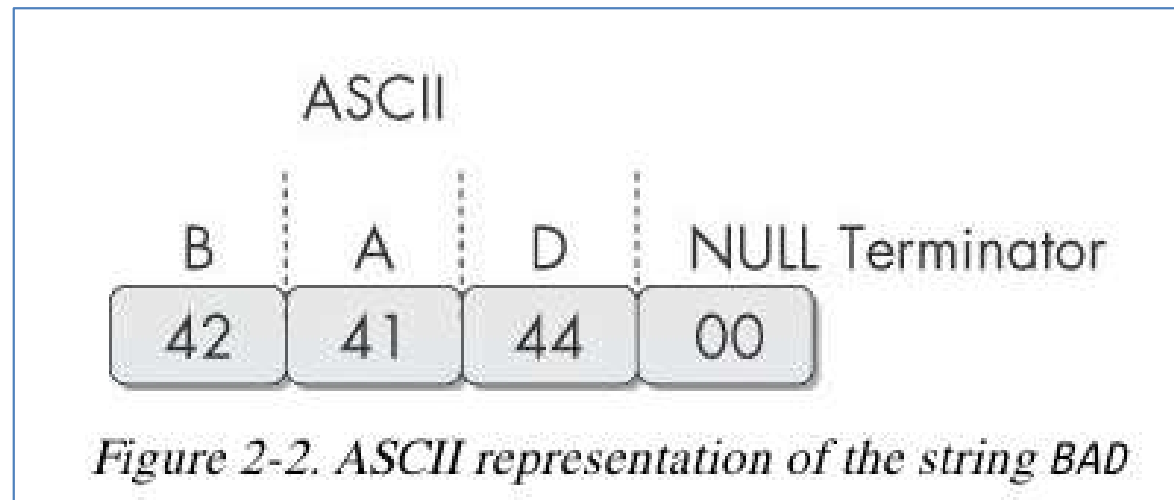
Hash Uses

- Label a malware file
- Share the hash with other analysts to identify malware
- Search the hash online to see if someone else has already identified the file

Finding Strings

Strings

- Any sequence of printable characters is a **string**
- Strings are terminated by a **null** (0x00)
- ASCII characters are 8 bits long
 - Now called ANSI
- Unicode characters are 16 bits long
 - Microsoft calls them "wide characters"



The strings Command

- Native in Linux, also available for Windows
- Finds all strings in a file 3 or more characters long

The strings Command

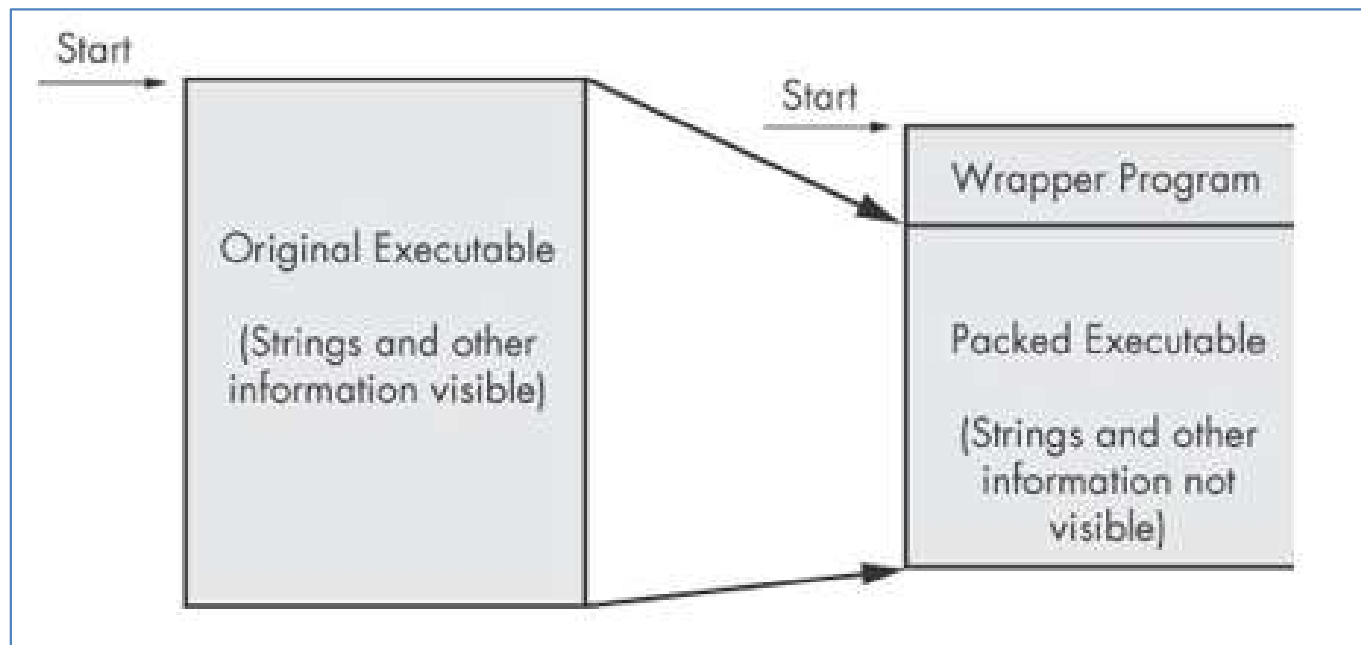
- Bold items can be ignored
- GetLayout and SetLayout are Windows functions
- GDI32.DLL is a Dynamic Link Library

```
C:>strings bp6.ex_  
VP3  
VW3  
t$@  
D$4  
99.124.22.1 4  
e-@  
GetLayout 1  
GDI32.DLL 3  
SetLayout 2  
M}C  
Mail system DLL is invalid.!Send Mail failed to  
send message. 5
```

Packed and Obfuscated Malware

Packing Files

- The code is compressed, like a Zip file
- This makes the strings and instructions unreadable
- All you'll see is the **wrapper** – small code that unpacks the file when it is run



Detecting Packers with PEiD

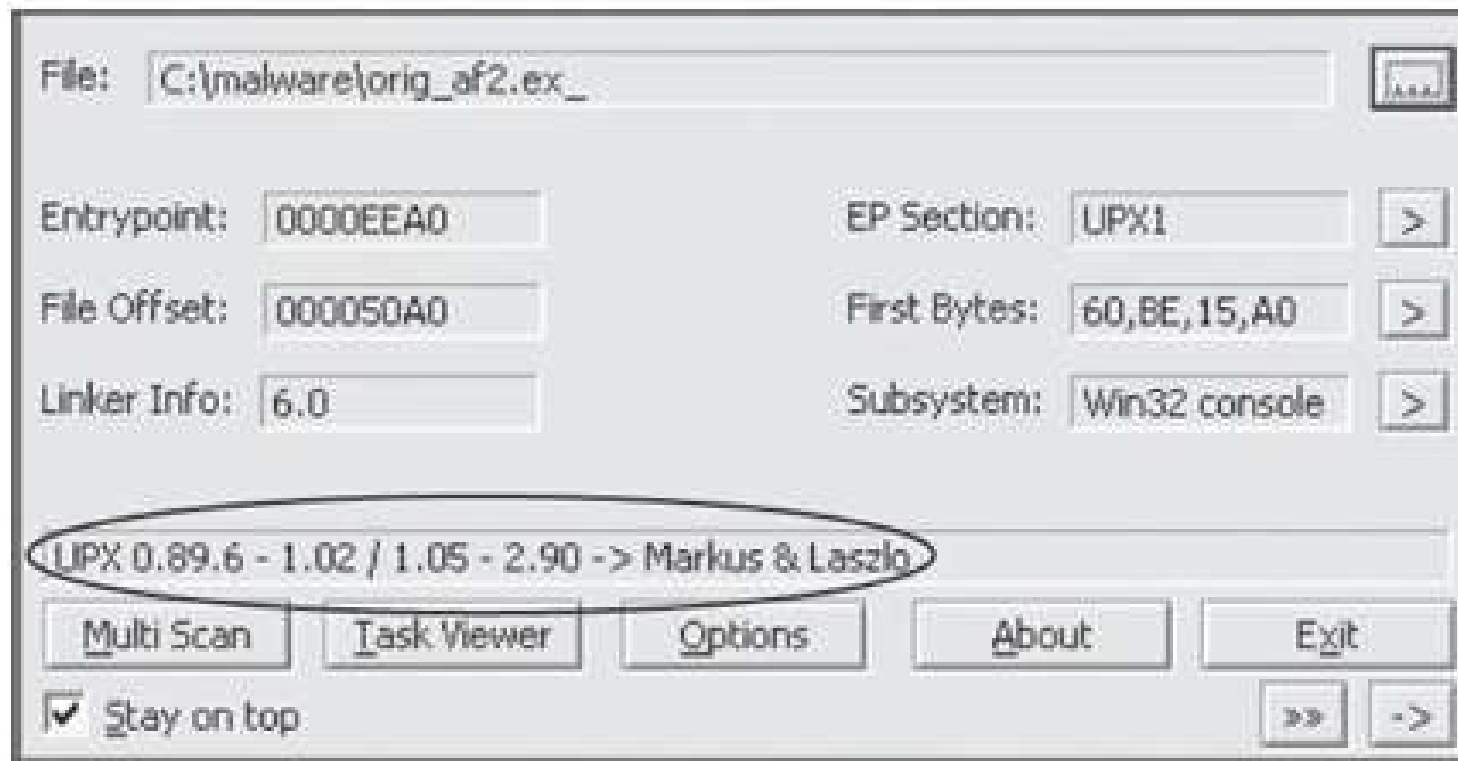


Figure 2-5. The PEiD program

Demo: UPX

```
root@kali: ~/126
File Edit View Search Terminal Help
root@kali:~/126# cat chatty.c
#include <stdio.h>
main()
{
char name[10];
printf("This program contains readable strings\n");
printf("Enter your name: ");
scanf("%s", name);
printf("Hello %s\n", name);
}

root@kali:~/126# gcc -static chatty.c -o chatty
root@kali:~/126# upx -o chatty-packed chatty
Ultimate Packer for eXecutables
Copyright (C) 1996 - 2011
UPX 3.08 Markus Oberhumer, Laszlo Molnar & John Reiser Dec 12th 2011

File size      Ratio      Format      Name
-----
592800 -> 272588 45.98% linux/elf386 chatty-packed

Packed 1 file.
root@kali:~/126# ls -l
total 852
-rwxr-xr-x 1 root root 592800 Aug 16 20:34 chatty
-rw-r--r-- 1 root root 174 Aug 16 20:27 chatty.c
-rwxr-xr-x 1 root root 272588 Aug 16 20:34 chatty-packed
root@kali:~/126#
```

Packing Obfuscates Strings

```
root@kali:~/126# strings chatty | wc
1962    4498    33817
root@kali:~/126# strings chatty-packed | wc
3950    4290    23623
root@kali:~/126#
```

NOTE

Many PEiD plug-ins will run the malware executable without warning! (See [Chapter 3](#) to learn how to set up a safe environment for running malware.) Also, like all programs, especially those used for malware analysis, PEiD can be subject to vulnerabilities. For example, PEiD version 0.92 contained a buffer overflow that allowed an attacker to execute arbitrary code. This would have allowed a clever malware writer to write a program to exploit the malware analyst's machine. Be sure to use the latest version of PEiD.

Portable Executable File Format

EXE Files

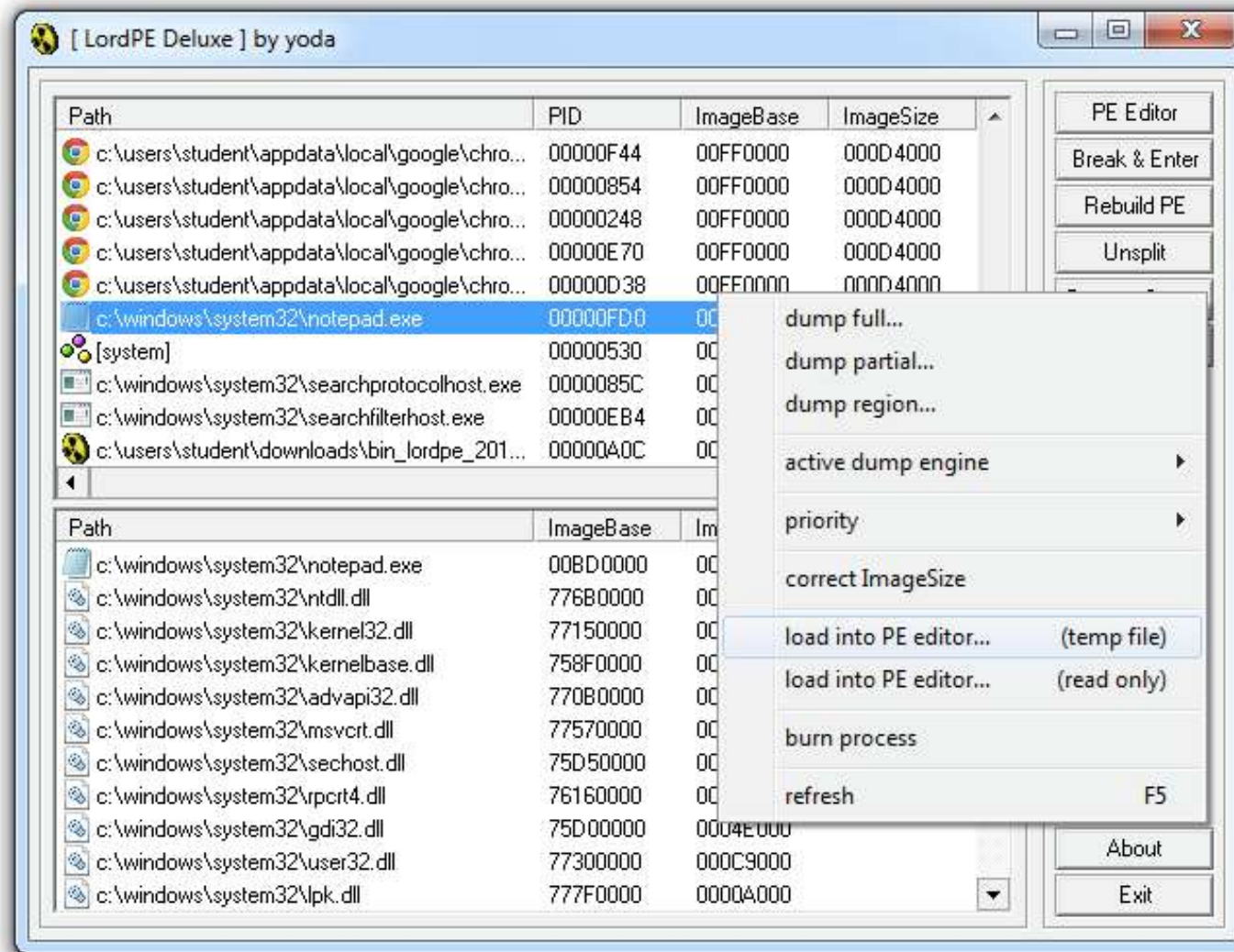
PE Files

- Used by Windows executable files, object code, and DLLs
- A data structure that contains the information necessary for Windows to load the file
- Almost every file executed on Windows is in PE format

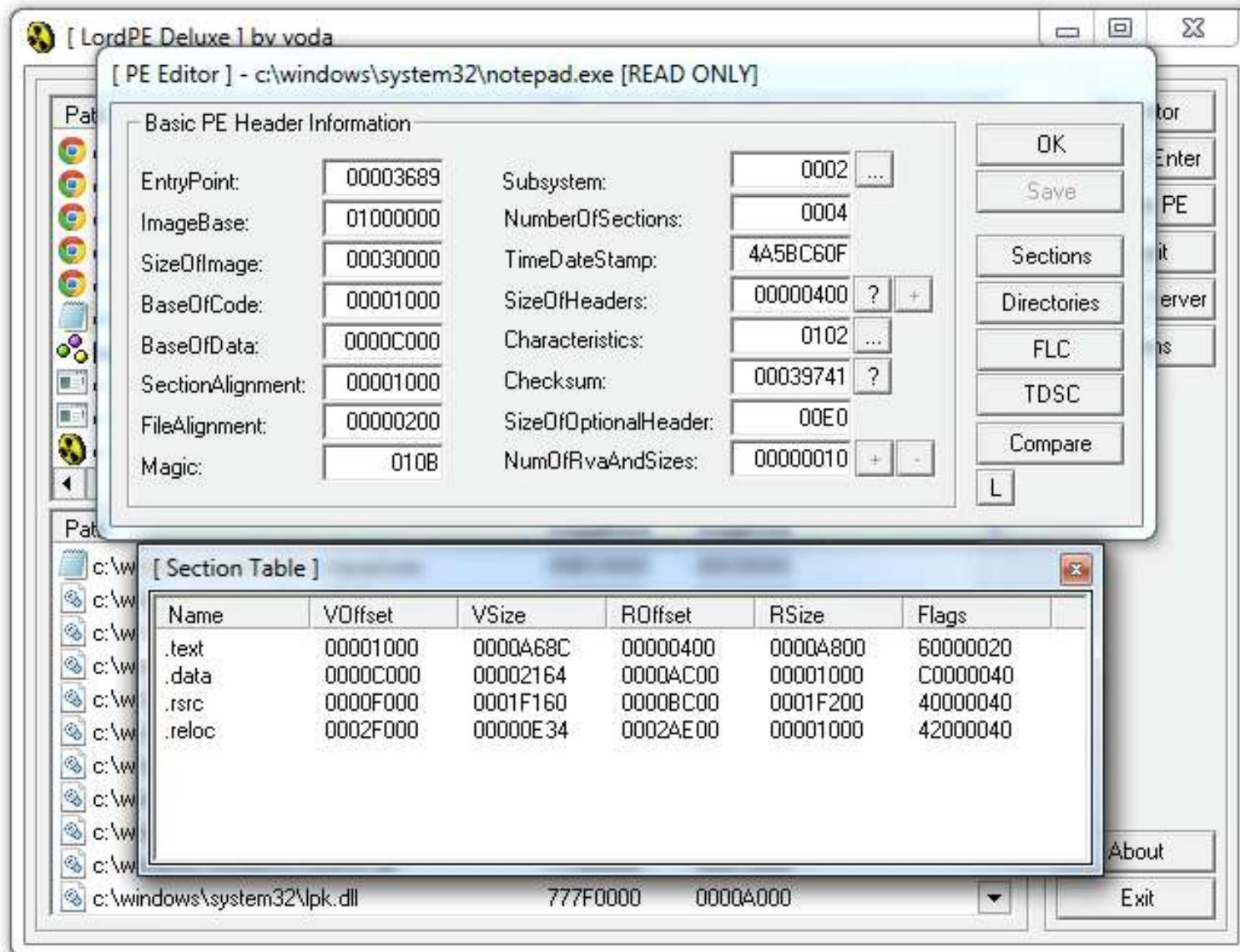
PE Header

- Information about the code
- Type of application
- Required library functions
- Space requirements

LordPE Demo



Main Sections



There are
a lot more
sections

- But the
main ones
are enough
for now
- Link Ch 2c

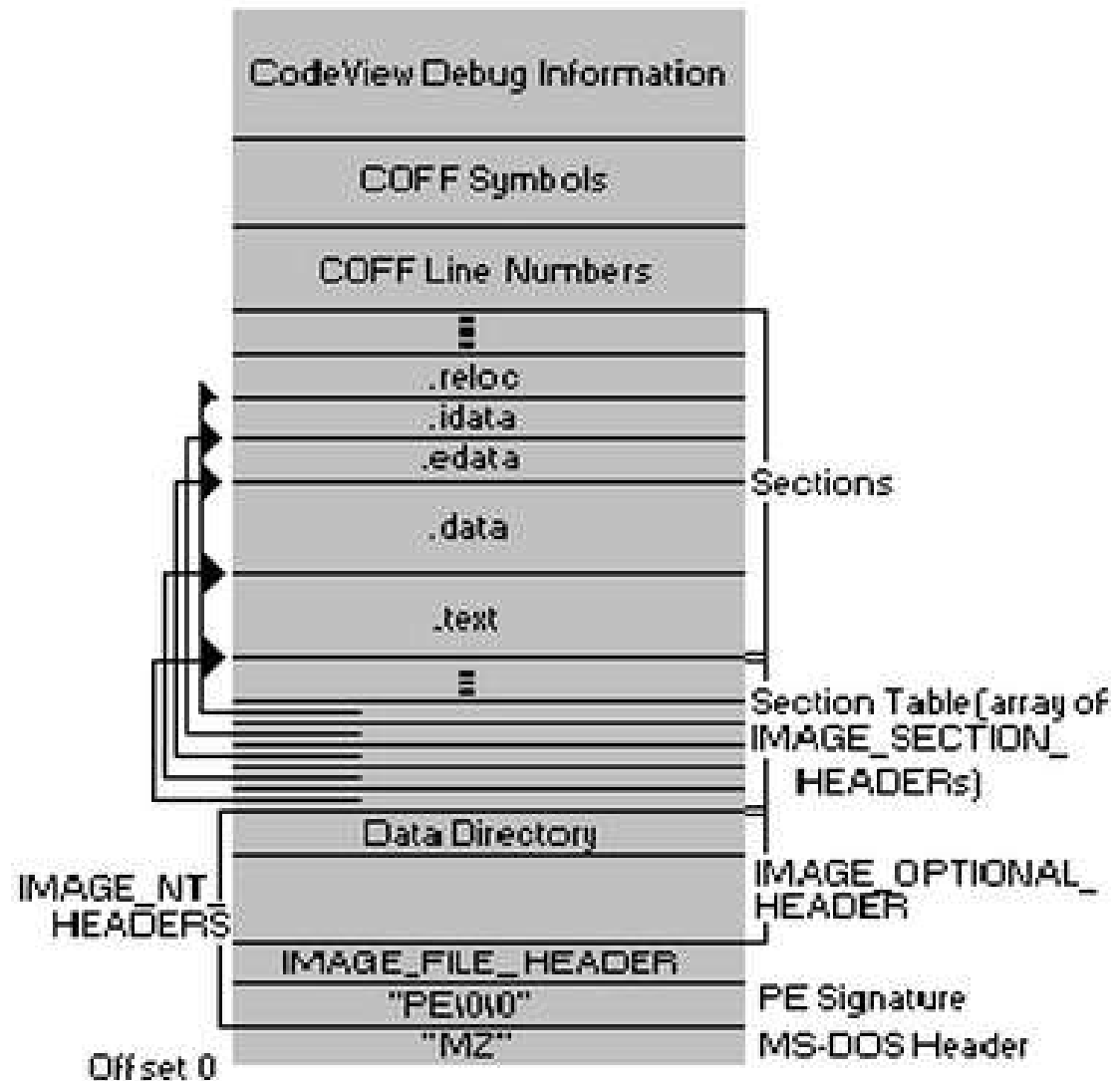


Figure 1. The PE file format

Linked Libraries and Functions

Imports

- Functions used by a program that are stored in a different program, such as library
- Connected to the main EXE by **Linking**
- Can be linked three ways
 - **Statically**
 - **At Runtime**
 - **Dynamically**

Static Linking

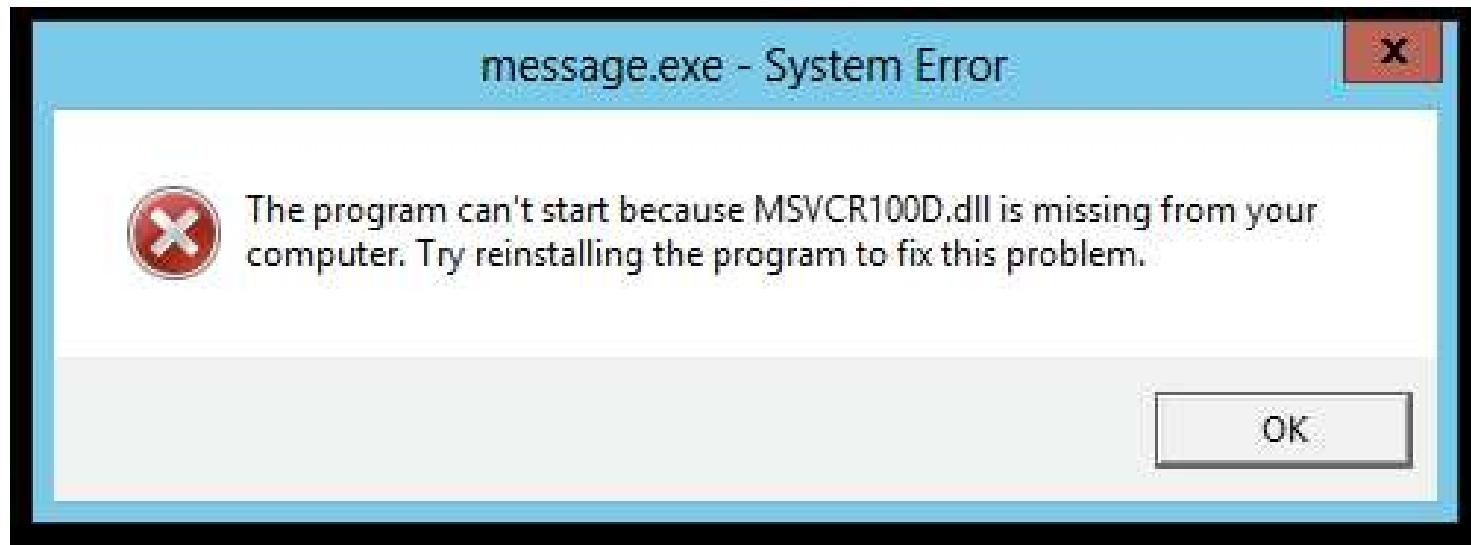
- Rarely used for Windows executables
- Common in Unix and Linux
- All code from the library is copied into the executable
- Makes executable large in size

Runtime Linking

- Unpopular in friendly programs
- Common in malware, especially packed or obfuscated malware
- Connect to libraries only when needed, not when the program starts
- Most commonly done with the **LoadLibrary** and **GetProcAddress** functions

Dynamic Linking

- Most common method
- Host OS searches for necessary libraries when the program is loaded



Clues in Libraries

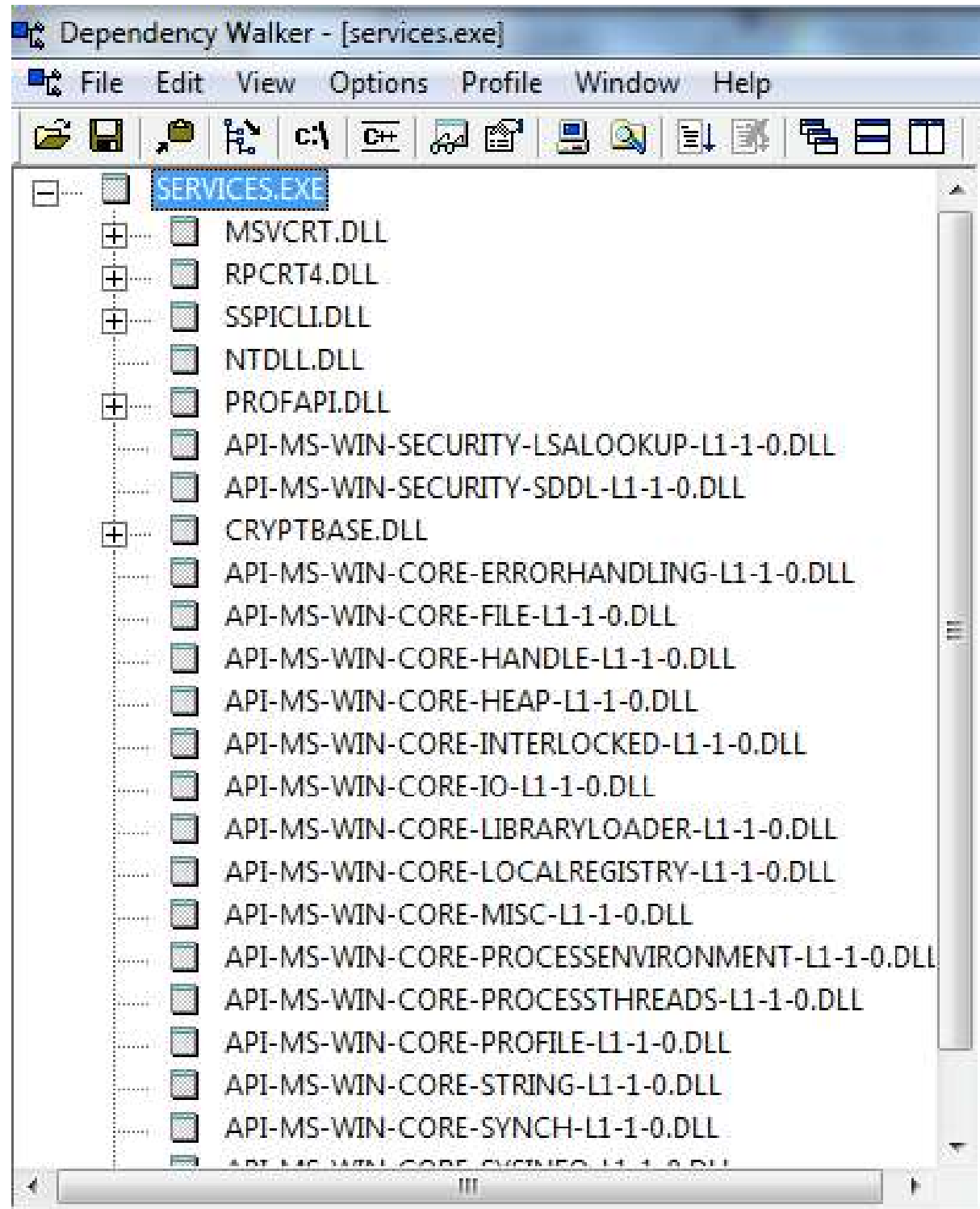
- The PE header lists every library and function that will be loaded
- Their names can reveal what the program does
- **URLDownloadToFile** indicates that the program downloads something

Dependency Walker

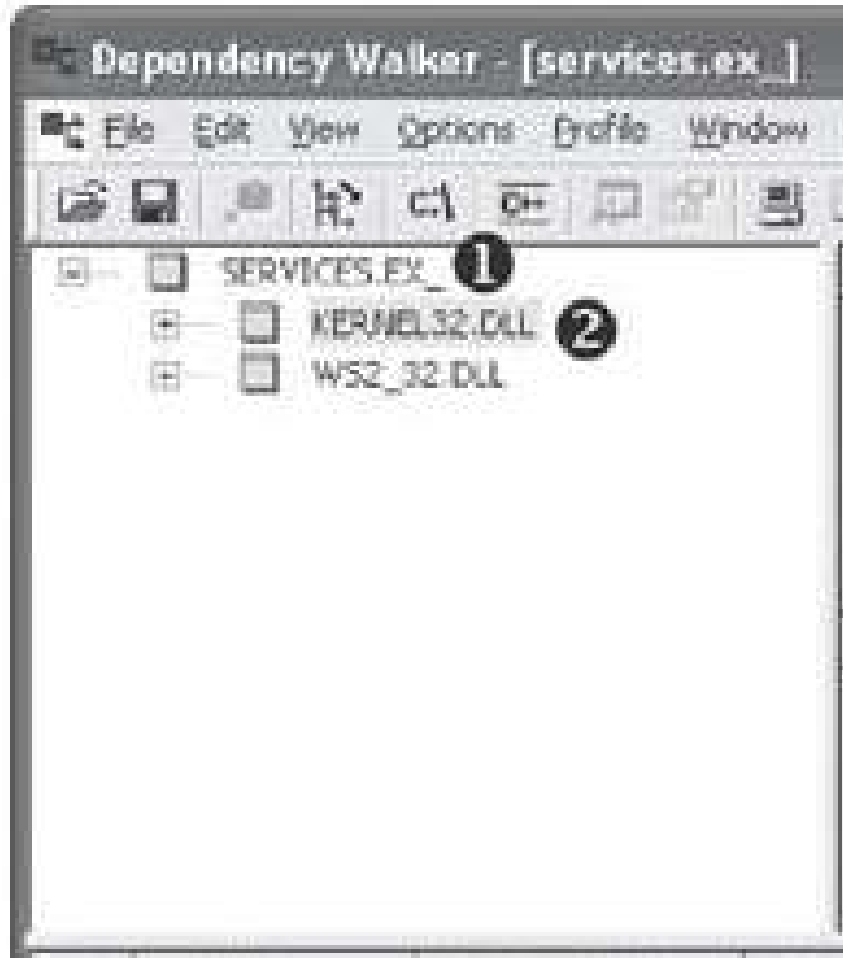
Shows Dynamically Linked Functions

- Normal programs have a lot of DLLs
- Malware often has very few DLLs

Services.exe



Services.ex_ (malware)



Imports & Exports in Dependency Walker

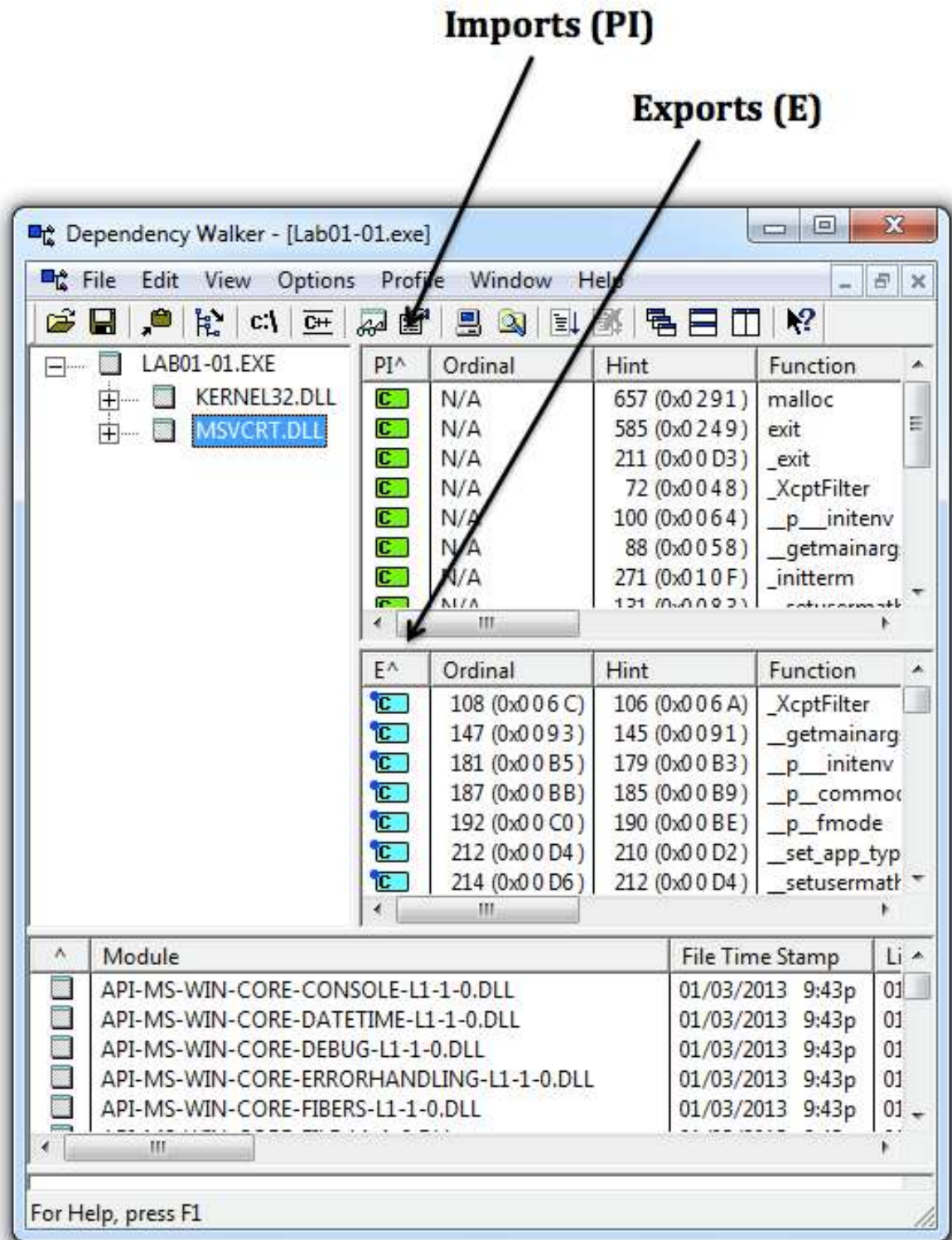


Table 2-1. Common DLLs

DLL	Description
<i>Kernel32.dll</i>	This is a very common DLL that contains core functionality, such as access and manipulation of memory, files, and hardware.
<i>Advapi32.dll</i>	This DLL provides access to advanced core Windows components such as the Service Manager and Registry.
<i>User32.dll</i>	This DLL contains all the user-interface components, such as buttons, scroll bars, and components for controlling and responding to user actions.
<i>Gdi32.dll</i>	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.

Exports

- DLLs **export** functions
- EXEs **import** functions
- Both exports and imports are listed in the PE header
- The book says exports are rare in EXEs, but I see a ton of exports in innocent EXEs

Example: Keylogger

- Imports User32.dll and uses the function **SetWindowsHookEx** which is a popular way keyloggers receive keyboard inputs
- It exports **LowLevelKeyboardProc** and **LowLevelMouseProc** to send the data elsewhere
- It uses **RegisterHotKey** to define a special keystroke like Ctrl+Shift+P to harvest the collected data

Ex: A Packed Program

- Very few functions
- All you see is the unpacker

Table 2-3. DLLs and Functions Imported from PackedProgram.exe

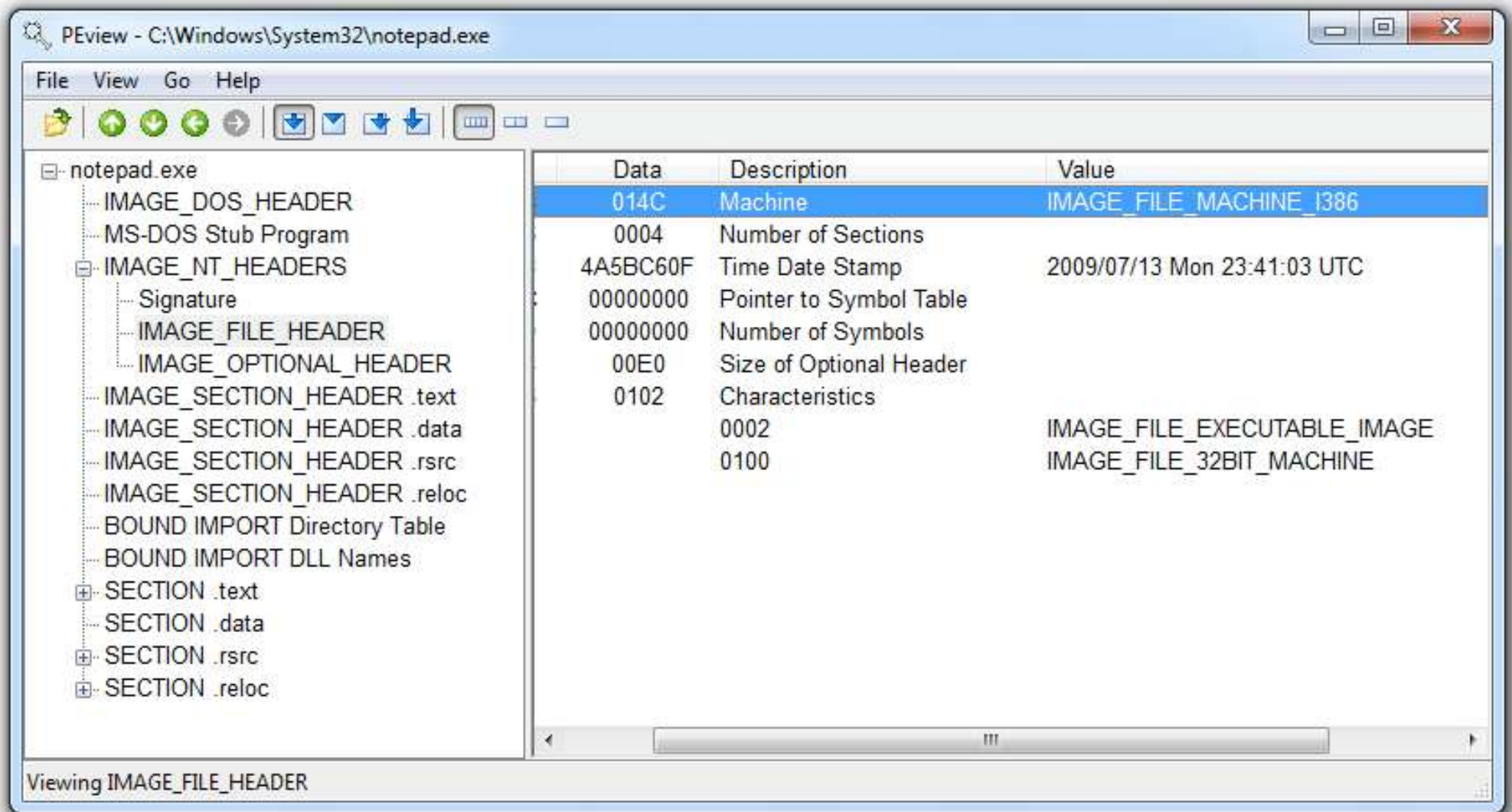
Kernel32.dll	User32.dll
GetModuleHandleA	MessageBoxA
LoadLibraryA	
GetProcAddress	
ExitProcess	
VirtualAlloc	
VirtualFree	

The PE File Headers and Sections

Important PE Sections

- **.text** -- instructions for the CPU to execute
- **.rdata** -- imports & exports
- **.data** – global data
- **.rsrc** – strings, icons, images, menus

PEView (Link Ch 2e)



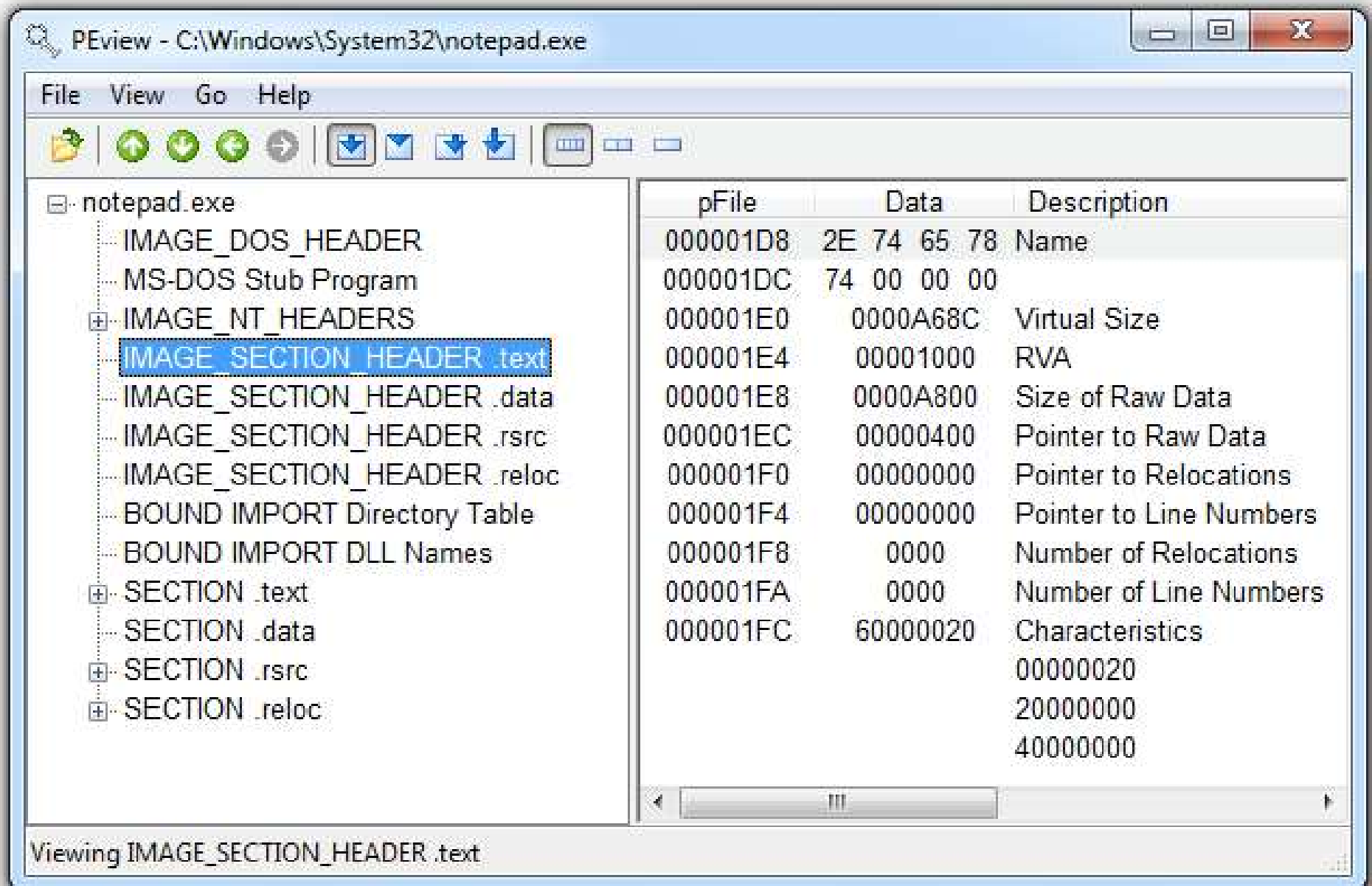
Time Date Stamp

- Shows when this executable was compiled
- Older programs are more likely to be known to antivirus software
- But sometimes the date is wrong
 - All Delphi programs show June 19, 1992
 - Date can also be faked

IMAGE_SECTION_HEADER

- Virtual Size – RAM
- Size of Raw Data – DISK
- For **.text** section, normally equal, or nearly equal
- Packed executables show Virtual Size much larger than Size of Raw Data for **.text** section

Not Packed



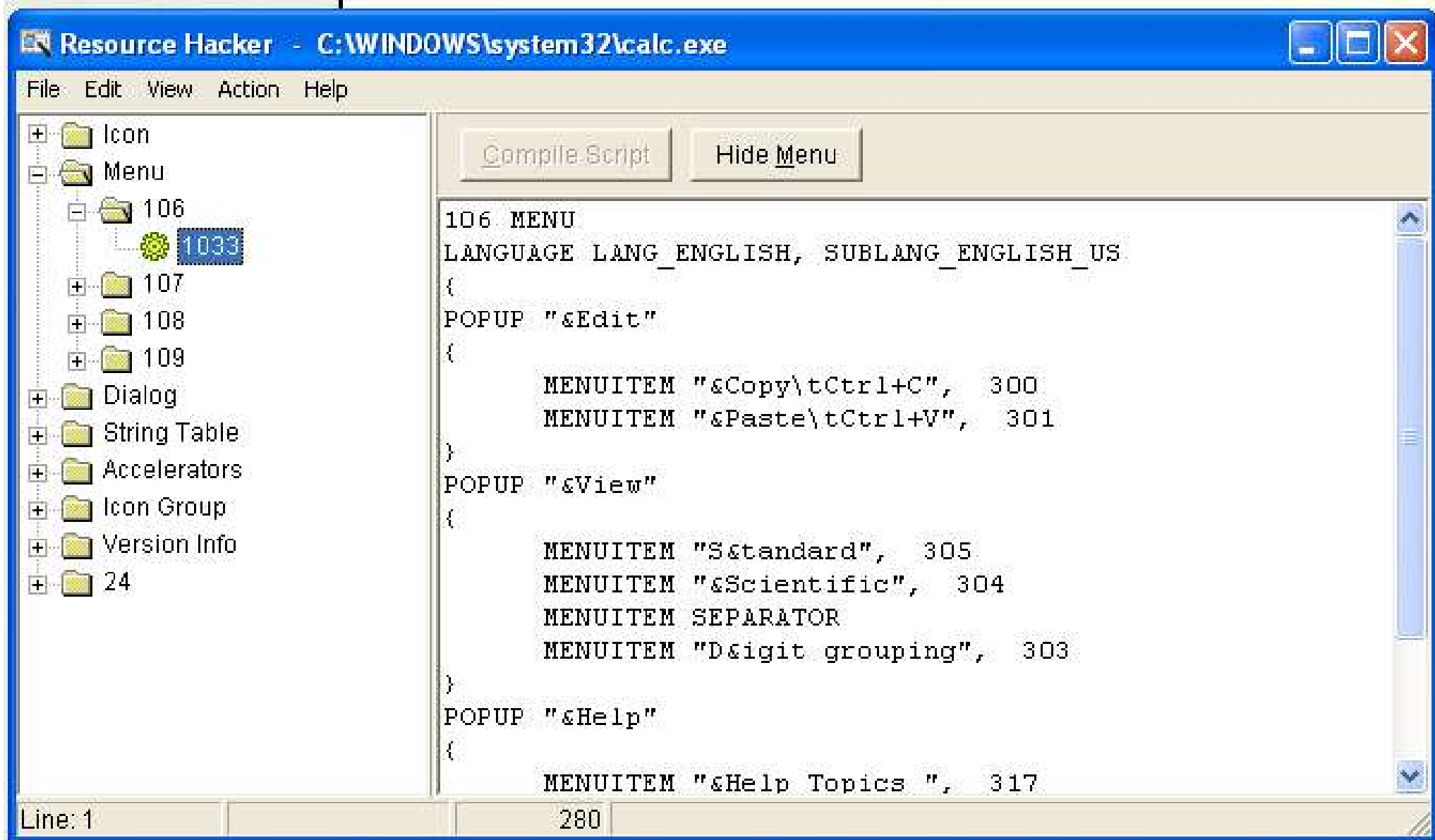
*Table 2-6. Section Information for
PackedProgram.exe*

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

- Lets you browse the **.rsrc** section
- Strings, icons, and menus
- Link Ch 2f

Resource Hacker in Windows XP



Resource Hacker in Windows 7

