



دانشکده مهندسی کامپیوتر

گروه مهندسی نرم افزار

عنوان پروژه:

بررسی ساختار فایل های PE

پیش نویسی اول پروژه کلاسی شماره ۳ درس کامپایلر پیشرفته

دانشجویان:

مرتضی ذاکری

استاد:

دکتر سعید پارسا

پاییز ۱۳۹۵

فهرست مطالب

۱.....	مقدمه	۱
۱.....	ساختار کلی	۲
۴.....	DOS HEADER	۱-۲
۴.....	PE FILE HEADER	۲-۲
۴.....	OPTIONAL HEADER	۳-۲
۴.....	DATA DIRECTORY	۱-۳-۲
۵.....	SECTION TABLE	۴-۲
۵.....	SECTION ها	۵-۲
۵.....	برخی از SECTION های رایج	۱-۵-۲
۷.....	مطالعه موردی فایل PE یک برنامه ساده	۳
۷.....	معرفی برنامه	۱-۳
۷.....	کد برنامه و فایل اجرایی	۲-۳
۹.....	برنامه نویسی نمایش ساختار فایل PE	۳-۳
۱۰.....	بررسی فایل اجرایی برنامه اعداد اول	۴-۳
۱۲.....	تفاوت نسخه های ۳۲ بیتی و ۶۴ بیتی فایل های PE	۵-۳
۱۳.....	نتیجه گیری	۶-۳
۱۳.....	منابع و ماخذ	۴

فهرست شکل ها

- شکل ۱-۲ ساختار کلی فایل PE ۲
- شکل ۲-۲ ساختار فایل PE با جزئیات بیش تر ۳
- شکل ۳-۲ نمایش HEX از یک فایل PE ۶
- شکل ۱-۳ کد برنامه تعیین عدد اول ۸
- شکل ۲-۳ ساختار فایل PE در ابزار NIKPEVIEWER ۱۰
- شکل ۳-۳ بخش DOS HEADER ۱۱
- شکل ۴-۳ نمایش دودویی بخش DOS HEADER ۱۱
- شکل ۵-۳ پیدا کردن ثابت های رشته ای برنامه در فایل دودویی ۱۲

۱ مقدمه

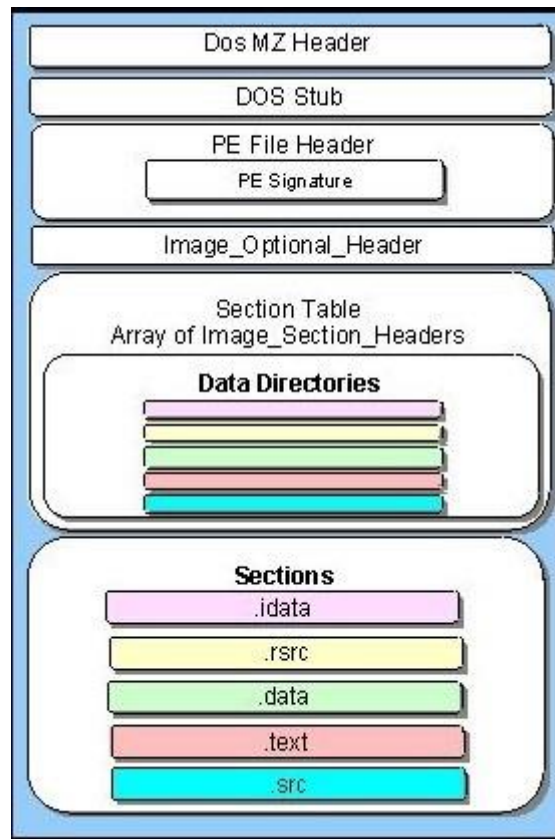
فایل PE یا Portable Executable قالب فایلی است که در سیستم ها عامل ویندوز به کار می رود (در هر دو نسخه ۳۲ و ۶۴ بیتی). PE قالب استاندارد شده ای برای فایل هایی با پسوند های رایج زیر است:

.exe, .dll, .obj, .acm, .ax, .cpl, .drv, .efi, ...

در واقع PE داده ساختاری است شامل اطلاعات لازم برای ماژول بارکننده برنامه در سیستم عامل که نهایتا فایل را برای اجرا شدن در حافظه قرار می دهد. قبل از ابداع این قالب، قالبی به نام COFF وجود داشت که در سیستم های Windows NT استفاده می شد.

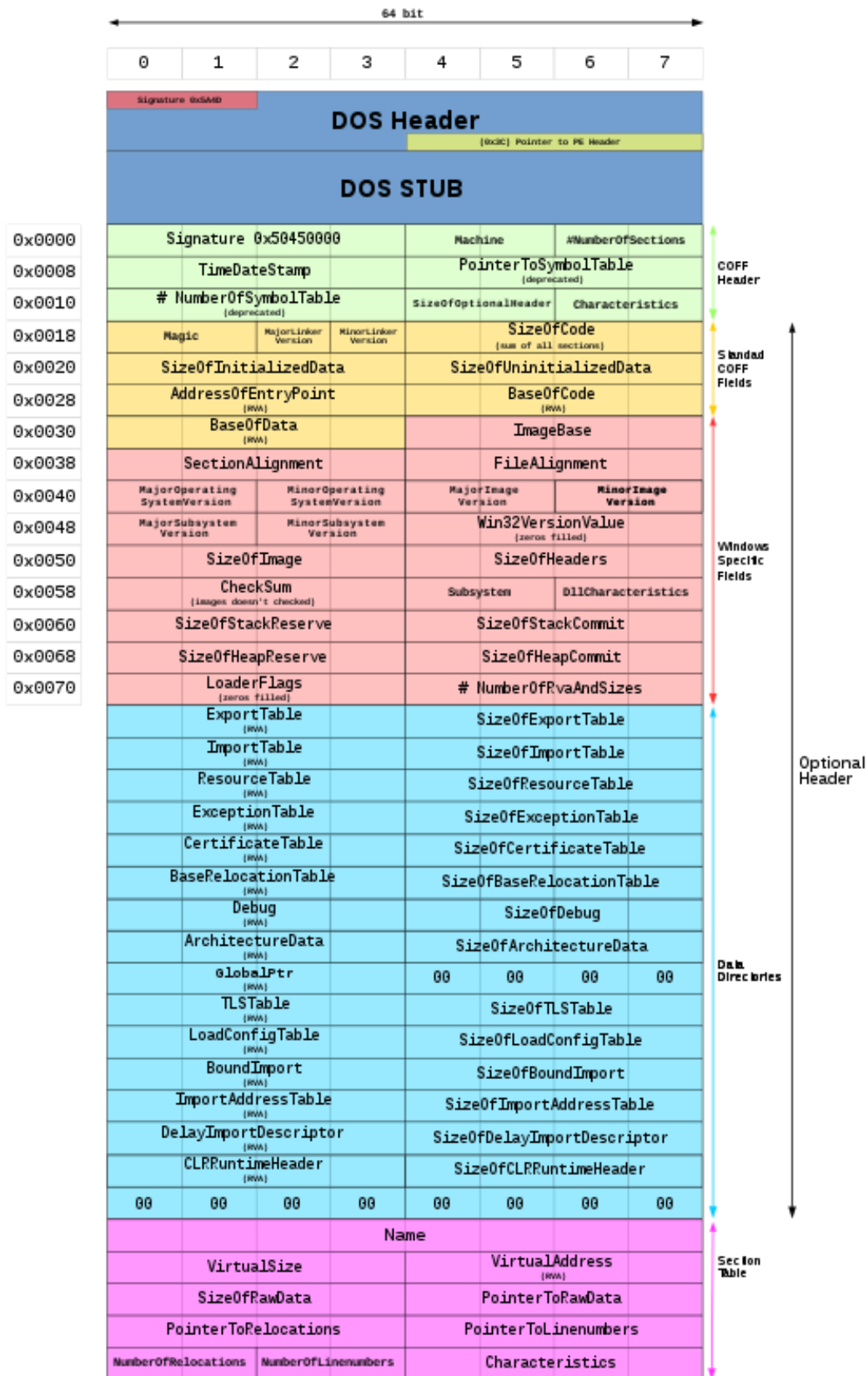
۲ ساختار کلی

اساسا فایل PE از دو بخش تقسیم شده است که هر یک دارای زیر بخش های مختلفی می باشد. بخش اول Header و بخش دیگر Section نام دارد. شکل ۱-۲ بخش های اصلی و زیر بخش های درون هر یک را به صورت گرافیکی نشان می دهد.



شکل ۲-۱ ساختار کلی فایل PE

در شکل ۲-۲ جزئیات بیش تری را می توان مشاهده کرد.



شکل ۲-۲ ساختار فایل PE با جزئیات بیشتر

۱-۲ DOS Header

DOS Header یا DOS MZ Header ۶۴ بایت اول هر فایل PE را تشکیل می دهد. این قسمت توسط بارکننده DOS بررسی می شود و در صورتی که معتبر باشد برنامه DOS STUB اجرا می شود. اما هنگامی که فایل PE حاوی برنامه Win32 باشد، در بخش DOS Header پرش به ابتدای PE File Header انجام شده و DOS STUB نیز حاوی کد چاپ پیام زیر می شود.

"This program cannot run in DOS mode"

بنابراین هنگامی که فایل PE در محیط DOS اجرا شود، با ورود بخش حاوی برنامه کوچک DOS STUB پیام فوق به کاربر نمایش داده می شود.

۲-۲ PE File Header

شامل اطلاعاتی مانند نوع پردازنده، تعداد کل Section های فایل PE جاری، مهر زمان (TimeStamp) و غیره است. آغاز این قسمت با فیلد موسوم به PE Signature است که محتوی کد اسکی 45 00 یا همان عبارت PE 00 می باشد.

۳-۲ Optional Header

علی رقم وجود واژه اختیاری در نام این قسمت، اطلاعات مهمی در آن قرار می گیرند. چند قلم از این اطلاعات عبارتند از: MajorLinkerVersion و MinorLinkerVersion شماره نسخه برنامه Linker که این فایل را ایجاد کرده است. SizeOfCode اندازه همه Section های حاوی کد برنامه. AddressOfEntryPoint آدرسی که بار کننده اجرا را از آنجا آغاز می کند. این فیلد حاوی آدرس مجازی نسبی یه همان RVA می باشد که به محلی در بخش text اشاره می کند و ...

۱-۳-۲ Data Directory

یک زیر بخش مهم در Optional Header فیلدی تحت عنوان Data Directory است. این بخش در واقع آرایه ای از ساختارهای IMAGE_DATA_DIRECTORY است. هر ساختار آدرس

مجازی نسبی، یک ساختار داده مهم در فایل PE همچون جدول آدرس واردات فایل (import address table) را ارائه می دهد.

۴-۲ Section Table

بین بخش PE Header و Section ها این قسمت قرار گرفته است. Section Table در واقع شبیه یک دفترچه تلفن است که اطلاعاتی راجع به هر Section موجود در تصویر حافظه^۱ برنامه را در بردارد. Section ها بر اساس آدرس شروع مجازی خود در حافظه مرتب می شوند نه عناوینشان.

۵-۲ Section ها

تعداد و نام Section ها در فایل های PE می تواند مختلف باشد. اما برای هر Section اطلاعات یکسانی باید نگهداری گردد. فایل EXE در مورد Section هایش اطلاعات نظیر آدرس فیزیکی، آدرس مجازی، اندازه اطلاعات و نوع دسترسی های موجود از قبیل خواندن، نوشتن و اجرا را نگهداری می کند.

۱-۵-۲ برخی از Section های رایج

در این قسمت تعدادی از Section هایی را که در فایل های .exe و .obj وجود دارند معرفی می کنیم. معمولاً عنوان هر Section با یک نطقه آغاز می شود ولی الزامی برای این کار وجود ندارد، یعنی می تواند بدون نقطه هم باشد.

.text این Section جایی است که همه کدهای برنامه که توسط کامپایلر یا اسمبلر تولید شده است نوشته می شود.

.rsrc شامل تمام مازول های به کار رفته در برنامه است.

^۱ Image

idata. شامل داده ها و تابع هایی است که از ماژول های DLL دیگر در این فایل درج شده است.

و ...

آن چه گفته شده تو ضیح بسیار مختصری از اجزای موجود در ساختار فایل PE بود و وارد بسیاری از جزئیات نشدیم. شکل ۲-۳ نمایش یک فایل PE به صورت کد HEX می باشد که بخش های گفته شده در آن مشخص شده اند. همان طور که پیداست این پایین ترین سطح نمایشی از یک برنامه و در واقع نمایش در سطح ماشین است (صفر و یک). در بخش سوم این نوشتار یک فایل PE را در عمل بررسی می کنیم.

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f		
00000000h:	4b	5a	50	00	02	00	00	00	04	00	0f	00	ff	ff	00	00	: MZP.....yy..	
00000010h:	b8	00	00	00	00	00	00	00	40	00	1a	00	00	00	00	00	:8.....	DOS HEADER
00000020h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	:	
00000030h:	00	00	00	00	00	00	00	00	00	00	00	00	00	01	00	00	:	
00000040h:	ba	10	00	0e	1f	84	09	cd	21	8b	01	4c	cd	21	90	90	: *...'.i!..Li![]	
00000050h:	54	68	69	73	20	70	72	6f	67	72	61	6d	20	6d	75	73	: This program mus	DOS STUB
00000060h:	74	20	62	65	20	72	75	6e	20	75	6e	64	65	72	20	57	: t be run under W	
00000070h:	69	6e	33	32	0d	0a	24	37	00	00	00	00	00	00	00	00	: in32..\$7.....	
00000080h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	:	
00000090h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	:	
000000a0h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	:	
000000b0h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	:	
000000c0h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	:	
000000d0h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	:	
000000e0h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	:	
000000f0h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	:	
00000100h:	50	45	00	00	4c	01	08	00	19	5e	42	2a	00	00	00	00	: PE..L....^B*....	PE HEADER
00000110h:	00	00	00	00	00	00	8e	81	08	01	02	19	00	a0	02	00	:ä.2[].....	
00000120h:	00	de	00	00	00	00	00	00	b4	ad	02	00	00	10	00	00	: .P.....'-.....	Signature
00000130h:	00	b0	02	00	00	00	40	00	10	00	00	00	00	02	00	00	: *.....8.....	
00000140h:	01	00	00	00	00	00	00	00	04	00	00	00	00	00	00	00	:	FileHeader
00000150h:	00	d0	03	00	00	04	00	00	00	00	00	00	02	00	00	00	: .D.....	
00000160h:	00	00	10	00	00	40	00	00	00	00	10	00	00	10	00	00	:8.....	
00000170h:	00	00	00	00	10	00	00	00	00	00	00	00	00	00	00	00	:	OptionalHeader
00000180h:	00	d0	02	00	1e	18	00	00	00	40	03	00	00	8e	00	00	: .D.....8...2..	
00000190h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	:	
000001a0h:	00	10	03	00	04	2b	00	00	00	00	00	00	00	00	00	00	:+.....	DATA DIRECTORY
000001b0h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	:	
000001c0h:	00	00	03	00	18	00	00	00	00	00	00	00	00	00	00	00	:	
000001d0h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	:	
000001e0h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	:	
000001f0h:	00	00	00	00	00	00	00	00	43	4f	44	45	00	00	00	00	:CODE.....	
00000200h:	88	9e	02	00	00	10	00	00	00	a0	02	00	00	04	00	00	: "ä.....	SECTION TABLE
00000210h:	00	00	00	00	00	00	00	00	00	00	00	00	20	00	00	60	:	
00000220h:	44	41	54	41	00	00	00	00	d4	06	00	00	00	80	02	00	: DATA....ô....*	

شکل ۲-۳ نمایش از یک فایل PE از HEX

۳ مطالعه موردی فایل PE یک برنامه ساده

۱-۳ معرفی برنامه

در این قسمت یک برنامه بسیار کوچک به زبان C++ در Win32 می نویسیم و سپس آن را کامپایل می کنیم تا فایل اجرایی در قالب EXE تولید شود. در ادامه ساختار فایل EXE تولید شده را که از قالب فایل های PE پیروی می کند بررسی می کنیم تا موارد شرح داده شده در بخش ۲ این نوشتار را به صورت عملی مشاهده نماییم.

دلیل آن که خود به نوشتن یک برنامه مبادرت ورزیدیم این است که از ساختار و عملکرد بیرونی کد اطلاع داشته باشیم و بتوانیم نتایج مشاهده شده را با آن چه بایستی وجود داشته باشد تطابق دهیم. به عبارت دیگر می خواهیم ساختار زمان اجرای یک برنامه کاربردی کوچک را که قرار است توسط بارکننده های ویندوزی بارگذاری شود با آگاهی و تسلط بر نحوه عملکرد و محتوای کد منبع و منابع مورد استفاده آن، مشاهده و بررسی نماییم. مینیمال بودن برنامه این امکان را به ما می دهد تا با پیچیدگی های ناشی از بزرگ بودن برنامه مواجه نشویم و بر روی هدف اصلی که شناخت ساختار فایل PE است تمرکز داشته باشیم.

برنامه ای که در این جا نوشته ایم در واقع یک عدد را از کنسول ورودی می خواند و تعیین می کند که اول است یا خیر و در نهایت پیام مناسب صادر می کند. برنامه شامل دستورات شرطی، حلقه تکرار، فراخوانی تابع، ثابت های رشته ای و دیگر جنبه های برنامه نویسی است که به نحوی که بتواند خیلی خوب ما را در یک روند آموزشی مناسب هدایت کند.

۲-۳ کد برنامه و فایل اجرایی

کد برنامه به صورت شکل ۱-۳ است. برنامه در محیط Microsoft Visual Studio 2015 نوشته و کامپایل شده است. دو نسخه اجرایی 64 و 32 بیتی از برنامه ایجاد کرده ایم تا بتوانیم تفاوت های ساختار فایل های PE در نسخه های مختلف را بررسی کنیم.

```
#include <iostream>
using namespace std;

void checkPrime(int n)
{
    int flag = 1;
    for (int i = 2; i*i < n && flag; i++)
    {
        if (!(n%i))
        {
            flag = 0;
        }
    }

    if (!flag)
    {
        cout << "Is Not Prime!\n";
    }
    else
    {
        cout << "Is Prime!\n";
    }
}

int main()
{
    int n;
    cout << "Enter Number: ";
    cin >> n;
    checkPrime(n);
    system("pause");
    return 0;
}
```

شکل ۳-۱ کد برنامه تعیین عدد اول

۳-۳ برنامه نویسی نمایش ساختار فایل PE

فایل PE یک محتوی یک کد دودویی است بنابراین از نوع متنی نیست و نمی توان با ویراستار های متنی آن را گشود و ساختار آن را مشاهده کرد. اما می توان آن را با یک ویرایشگر فایل های HEX یا همان باینری گشود. برنامه های بسیاری هستند که برای نمایش ساختار فایل PE استفاده می شوند و علاوه بر نمایش مقادیر دودویی اجزا را به صورت طبقه بندی شده نشان داده و اطلاعات را از پیکره فایل PE بیرون می کشند. این برنامه ها عمدتاً در امکاناتی نظیر جست و جو و واسط کاربر تفاوت هایی دارند. ما فهرستی از برنامه ها را تست کردیم تا گزینه بهتر را انتخاب کنیم؛ از جمله:

PE Explorer, PEBrowse64, NikPEViewer, x64/32dbg, ...

کلیه اجزای موجود در فایل های PE توسط ساختارهای برنامه نویسی زبان C/C++ تعریف می شوند. برای نمونه بخش DOS Header به صورت زیر در زبان C تعریف می شود، این ساختار در داخل فایل سراینده WINNT.H وجود دارد.

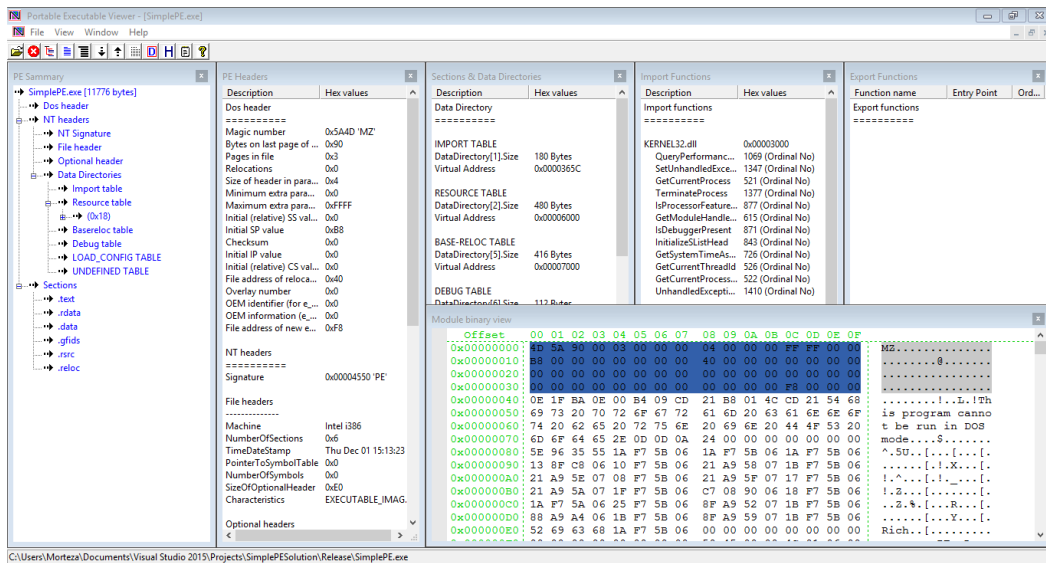
```
typedef struct _IMAGE_DOS_HEADER { // DOS .EXE header
    USHORT e_magic;           // Magic number
    USHORT e_cblp;           // Bytes on last page of file
    USHORT e_cp;             // Pages in file
    USHORT e_crlc;           // Relocations
    USHORT e_cparhdr;        // Size of header in paragraphs
    USHORT e_minalloc;       // Minimum extra paragraphs needed
    USHORT e_maxalloc;       // Maximum extra paragraphs needed
    USHORT e_ss;             // Initial (relative) SS value
    USHORT e_sp;             // Initial SP value
    USHORT e_csum;           // Checksum
    USHORT e_ip;             // Initial IP value
    USHORT e_cs;             // Initial (relative) CS value
    USHORT e_lfarlc;         // File address of relocation table
    USHORT e_ovno;           // Overlay number
    USHORT e_res[4];         // Reserved words
    USHORT e_oemid;          // OEM identifier (for e_oeminfo)
    USHORT e_oeminfo;        // OEM information; e_oemid specific
    USHORT e_res2[10];       // Reserved words
    LONG e_lfanew;           // File address of new exe header
} IMAGE_DOS_HEADER, *PIMAGE_DOS_HEADER;
```

بدیهی است که API ویندوز توابعی را برای کار با این داده ساختارها تدارک دیده است. برنامه های نمایش ساختار PE که در بالا به برخی از آن ها اشاره شد، از چنین توابعی استفاده می

کنند. بنابراین نوشتن یک برنامه که بتواند نمایش صحیحی از فایل PE ارائه دهد کار ساده ای است.

۴-۳ بررسی فایل اجرایی برنامه اعداد اول

فایل با عنوان SimplePE_x86.exe را توسط NikPEViewer باز می کنیم (شکل زیر). همان طور که پیداست، پنجره های زیر هر کدام بخش مجزایی از ساختار را نشان می دهند.



شکل ۳-۲ ساختار فایل PE در ابزار NikPEViewer

بخش DOS MZ Header خالی است و تنها یک دستور پرش به F8 که ابتدای PE File Header است دارد. اطلاعات شکل ۳-۳ از شکل ۴-۳ که معادل دودویی آن است بیرون کشیده شده است.

Dos header	
=====	
Magic number	0x5A4D 'MZ'
Bytes on last page of file	0x90
Pages in file	0x3
Relocations	0x0
Size of header in paragraphs	0x4
Minimum extra paragraphs needed	0x0
Maximum extra paragraphs needed	0xFFFF
Initial (relative) SS value	0x0
Initial SP value	0xB8
Checksum	0x0
Initial IP value	0x0
Initial (relative) CS value	0x0
File address of relocation table	0x40
Overlay number	0x0
OEM identifier (for e_oemid)	0x0
OEM information (e_oemid specific)	0x0
File address of new exe header	0xF8

شکل ۳-۳ بخش DOS Header

Offset	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
0x00000000	4D	5A	90	00	03	00	00	00	04	00	00	00	FF	FF	00	00	MZ.....
0x00000010	B8	00	00	00	00	00	00	00	40	00	00	00	00	00	00	00@.
0x00000020	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x00000030	00	00	00	00	00	00	00	00	00	00	00	00	FB	00	00	00

شکل ۳-۴ نمایش دودویی بخش DOS Header

به همین ترتیب می توان سایر قسمت های بخش دوم این نوشتار را مشاهده کرد. به عنوان مثال کد برنامه در text. و ثابت های رشته ای در rdata. تعریف شده اند. هنگامی که برنامه اجرا می شود، پیام "Enter Number:" روی کنسول به نمایش در می آید. می خواهیم محل این پیام را در فایل باینری ببینیم. در منوی سمت چپ روی rdata. دوبار کلیک می کنیم تا محدوده مربوط به این Section در کد باینری یافت شود. حال این محدوده را مشاهده می کنیم.

چنان چه مشاهده می شود کلیه ثابت های رشته ای برنامه در قالب کد اسکی و به صورت متوالی ذخیره شده اند. ادرس محل ذخیره موجودیت غیر فعال (کد برنامه) نیز در این قسمت وجود دارد.

Offset	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
0x000015D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x000015E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x000015F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x00001600	DE	3C	00	00	76	3D	00	00	8C	3D	00	00	A2	3D	00	00	<...v=...=...=.
0x00001610	BC	3D	00	00	D2	3D	00	00	E6	3D	00	00	40	3D	00	00	.=...=...=..@=..
0x00001620	2C	3D	00	00	18	3D	00	00	FA	3C	00	00	5C	3D	00	00	,=...=...<..\=. .
0x00001630	00	00	00	00	F8	37	00	00	D6	39	00	00	96	39	00	007...9...9..
0x00001640	54	39	00	00	14	39	00	00	D6	38	00	00	90	38	00	00	T9...9...8...8..
0x00001650	56	38	00	00	34	38	00	00	00	00	00	00	42	3A	00	00	V8..48.....B:..
0x00001660	38	3A	00	00	08	3E	00	00	26	3A	00	00	00	00	00	00	8:...>..&:.....
0x00001670	B8	3B	00	00	00	00	00	00	A2	3B	00	00	00	00	00	00	.;.....;.....
0x00001680	9A	3A	00	00	00	00	00	00	D8	3B	00	00	F4	3B	00	00;.....;..
0x00001690	10	3C	00	00	1E	3C	00	00	2E	3C	00	00	6E	3A	00	00	.<...<...<..n:..
0x000016A0	78	3A	00	00	6A	3B	00	00	74	3B	00	00	60	3B	00	00	x:..z;..t;..`;..
0x000016B0	52	3B	00	00	8A	3A	00	00	2E	3B	00	00	26	3B	00	00	R;.....;..&;..
0x000016C0	18	3B	00	00	0C	3B	00	00	EA	3A	00	00	C8	3A	00	00	.;.....;.....
0x000016D0	AE	3A	00	00	44	3B	00	00	00	00	00	00	C8	3B	00	00D;.....;..
0x000016E0	36	3B	00	00	00	00	00	00	E4	1E	40	00	00	00	00	00	6;.....@.....
0x000016F0	7F	14	40	00	00	00	00	00	00	00	00	00	D3	13	40	00	l.@.....@.
0x00001700	77	14	40	00	00	00	00	00	00	00	00	00	00	00	00	00	w.@.....
0x00001710	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x00001720	18	40	40	00	68	40	40	00	49	73	20	4E	6F	74	20	50	.@@.h@@.Is Not P
0x00001730	72	69	6D	65	21	0A	00	00	49	73	20	50	72	69	6D	65	prime!...Is Prime
0x00001740	21	0A	00	00	45	6E	74	65	72	20	4E	75	6D	62	65	72	!...Enter Number
0x00001750	3A	20	00	00	70	61	75	73	65	00	00	00	00	00	00	00	: ..pause.....
0x00001760	00	00	00	00	DB	0C	40	58	00	00	00	00	02	00	00	00@X.....
0x00001770	75	00	00	00	40	32	00	00	40	18	00	00	00	00	00	00	u...@2...@.....
0x00001780	DB	0C	40	58	00	00	00	00	0C	00	00	00	14	00	00	00	..@X.....
0x00001790	B8	32	00	00	B8	18	00	00	00	00	00	00	DB	0C	40	58	.2.....@X
0x000017A0	00	00	00	00	0D	00	00	00	78	02	00	00	CC	32	00	00x...2..
0x000017B0	CC	18	00	00	00	00	00	00	DB	0C	40	58	00	00	00	00@X.....
0x000017C0	0E	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x000017D0	5C	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	\.....
0x000017E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x000017F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

شکل ۳-۵ پیدا کردن ثابت های رشته ای برنامه در فایل دودویی

۵-۳ تفاوت نسخه های ۳۲ بیتی و ۶۴ بیتی فایل های PE

قالب فایل های اجرایی در ویندوزهای ۶۴ بیتی (x64) نیز مانند ویندوز ۳۲ بیتی (x86) همان قالب فایل PE است، اما تفاوت هایی دارد. در این جا کشف این تفاوت ها یک نسخه ۶۴ بیتی از برنامه بررسی اعداد اول که در قسمت قبل دیدیم را ایجاد می کنیم. سپس آن را در محیط ابزار NikPEViewer باز می کنیم.

اولین تفاوت در حجم نسخه های ۳۲ بیتی و ۶۴ بیتی است. نسخه ۶۴ بیتی ما حدوداً ۱ کیلوبایت از نسخه ۳۲ بیتی، بزرگتر است (۱۳ کیلوبایت در مقابل ۱۲ کیلوبایت).

آفست شروع PE Header File متفاوت است (0x00000100). آدرس های مجازی ۶۴ بیتی هستند. به طور کلی اندازه Section ها و مقدار RVA ها نیز متفاوت است.

۶-۳ نتیجه گیری

از ابزار NikPEViewer تفاوت های بیش تری را نمی توان تشخیص داد. با استفاده از ابزار IDA Pro مشاهده می شود از آن جایی که برنامه از نوع Win32 می باشد لذا در نسخه ۶۴ بیتی نیز همچنان از ثبات های ۳۲ بیتی استفاده شده است و از این لحاظ مزیتی ندارد. می توان این نتیجه را گرفت که با وجود تفاوت های جزئی ساختار کلی همان ساختار PE است و در واقع برای به حداکثر رسانی قابلیت حمل این فایل ها تفاوت اساسی در آن ها ایجاد نشده است.

۴ منابع و ماخذ