Module 1

A journey from high level languages, through assembly, to the running process

https://github.com/hasherezade/malware_training_vol1

Basics of PE (Portable Executable)

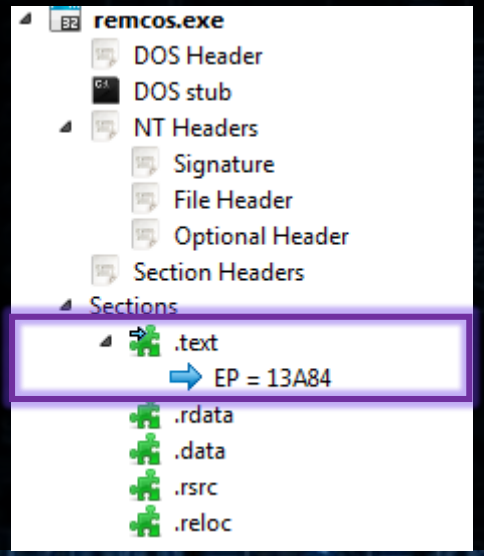
Basics of a PE file

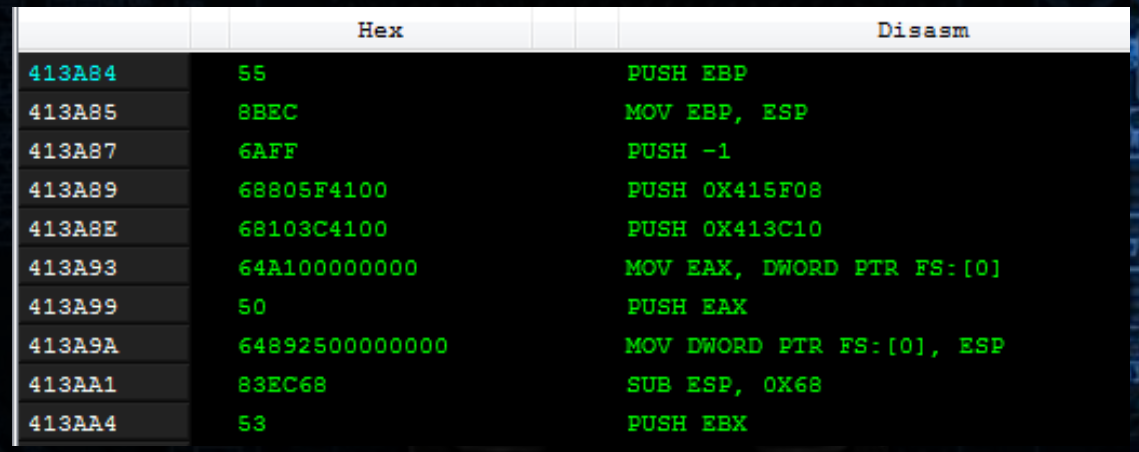
- PE (Portable Executable) is a native executable format on Windows
- PE files:
  - user mode: EXE, DLL
  - kernel more: driver (.sys), kernel image (ntoskrnl.exe)
  - UEFI (run in SMM – System Managemant Mode)
  - Also OBJ files have structures similar to PE

Basics of a PE file

- PE (Portable Executable) contains information:
  - What to execute: the compiled code
  - How to execute: headers with data neccessery for loading it

remcos.exe





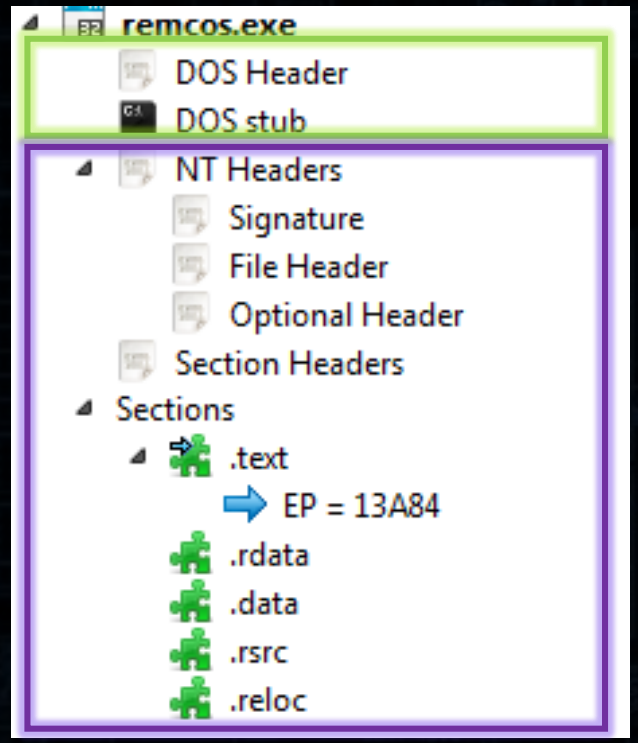
	Hex	Disasm
413A84	55	PUSH EBP
413A85	8BEC	MOV EBP, ESP
413A87	6AFF	PUSH -1
413A89	68805F4100	PUSH 0X415F08
413A8E	68103C4100	PUSH 0X413C10
413A93	64A100000000	MOV EAX, DWORD PTR FS:[0]
413A99	50	PUSH EAX
413A9A	64892500000000	MOV DWORD PTR FS:[0], ESP
413AA1	83EC68	SUB ESP, 0X68
413AA4	53	PUSH EBX

Basics of a PE file

- PE format is based on a Unix format COFF – that was used in VAX/VMS
- It was introduced as a part of specification Win32
- Throughout many years, the core of the format didn't change, only some new fields of some structures have been added
- Since introduction of 64 bit environment, PE needed to be adjusted to it: 64 bit PE was introduced
- Also, new variants have been introduced, like .NET PE – containing additional structures with intermediate code and metadata

Basics of a PE file

- PE file structure: the DOS part (legacy) and the Windows Part



https://gist.github.com/hashereware/54398aad2ef449c255d2bd2563bb62ca#file-pe_hdr-h

Basics of a PE file

- DOS Header: only e_magic, and e_lfnew must be filled:

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

https://gist.github.com/hashereware/54398aad2ef449c255d2bd2563bb62ca#file-pe_hdr-h

Basics of a PE file

- PE sections
  - PE is divided into sections with different permissions
  - Sections introduce a logical layout of the binary, that compilers/linkers can follow
  - Dividing PE on section improves security: the code is isolated from the data
  - HOWEVER:
    - if DEP is disabled, page without execution permission can still be executed
    - The section containing the Entry Point will always be treated as executable

Basics of a PE file

- PE sections are defined by sections header

Name	Raw Addr.	Raw size	Virtual Addr.	Virtual Size	Characteristics	Ptr to Reloc.	Num. of Reloc.	Num. of Linenum.
.text	1000	13000	1000	12D26	60000020	0	0	0
>	14000	^	13D26	^	r-x			
.rdata	14000	6000	14000	5490	40000040	0	0	0
>	1A000	^	19490	^	r--			
.data	1A000	1000	1A000	114C	C0000040	0	0	0
>	1B000	^	1B14C	^	rw-			
.rsrc	1B000	1000	1C000	B80	40000040	0	0	0
>	1C000	^	1CB80	^	r--			
.reloc	1C000	3000	1D000	268A	42000040	0	0	0
>	1F000	^	1F68A	^	r--			

Basics of a PE file

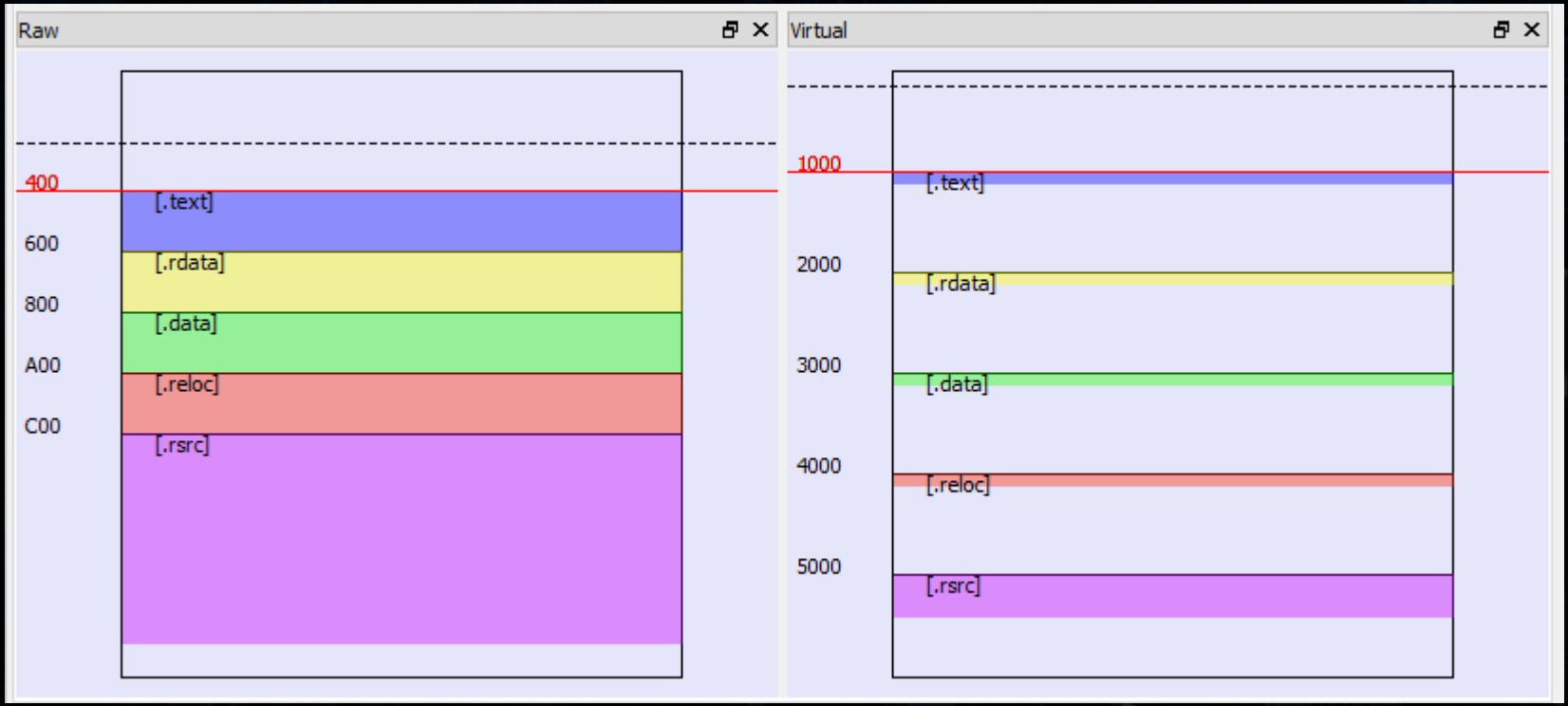
- PE sections
  - on the disk PE is stored in a raw format (the unit is defined by File Alignment)
  - In memory PE is mapped to its virtual format (the unit is defined by Section Alignment) – usually of the granularity of one page (0x1000)

Disasm: .rdata | General | DOS Hdr | Rich Hdr | File Hdr | Optional Hdr

Offset	Name	Value	Value
110	Entry Point	47A3	
114	Base of Code	1000	
118	Base of Data	1D000	
11C	Image Base	400000	
120	Section Alignment	1000	
124	File Alignment	200	

Basics of a PE file

- Raw (file on the disk), Virtual (mapped in the memory)



https://gist.github.com/hashereware/54398aad2ef449c255d2bd2563bb62ca#file-remap-h

Basics of a PE file

- The most information lies in data directories

	Data Directory	Address	Size
F8	Export Directory	294000	1E62
100	Import Directory	296000	3600
108	Resource Directory	29C000	4B134
110	Exception Directory	0	0
118	Security Directory	0	0
120	Base Relocation Table	2E8000	16ED0
128	Debug Directory	0	0
130	Architecture Specific Data	0	0
138	RVA of GlobalPtr	0	0
140	TLS Directory	29B000	18
148	Load Configuration Directory	0	0
150	Bound Import Directory in headers	0	0
158	Import Address Table	2968D4	7A8
160	Delay Load Import Descriptors	0	0
168	.NET header	0	0

Basics of a PE file

- Relocation Table

Disasm: .rdata | General | DOS Hdr | Rich Hdr | File Hdr | Optional Hdr | Section Hdrs | Imports | Resources | BaseReloc.

Offset	Page RVA	Block Size	Entries Count
24A00	1000	94	46
24A94	2000	74	36
24B08	3000	70	34
24B78	4000	11C	8A
24C94	5000	A0	4C
24D24	6000	74	36

Relocation Block [70 entries]

Offset	Value	Type	Offset from Page	Reloc RVA
24A08	300A	32 bit field	A	100A
24A0A	3043	32 bit field	43	1043
24A0C	3055	32 bit field	55	1055
24A0E	305B	32 bit field	5B	105B
24A10	30C7	32 bit field	C7	10C7
24A12	30F6	32 bit field	F6	10F6
24A14	3119	32 bit field	119	1119
24A16	3153	32 bit field	153	1153
24A18	318A	32 bit field	18A	118A
24A1A	31D6	32 bit field	1D6	11D6

https://gist.github.com/hashereware/54398aad2ef449c255d2bd2563bb62ca#file-relocate-h

Basics of a PE file

- Import Table

Disasm: .rdata | General | DOS Hdr | Rich Hdr | File Hdr | Optional Hdr | Section Hdrs | Imports | Resources | BaseRel

Offset	Name	Func. Count	Bound?	OriginalFirstThun	TimeDateStamp	Forwarder	NameRVA	FirstThunk
22474	KERNEL32.dll	93	FALSE	2309C	0	0	23328	1D000

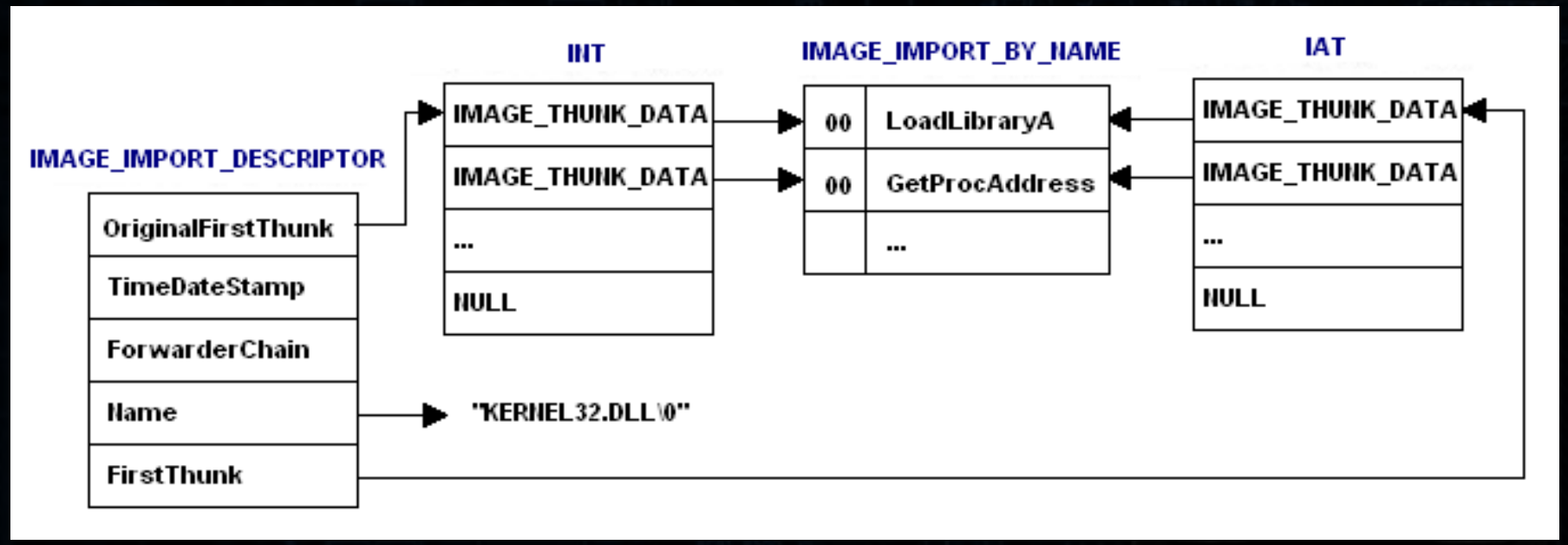
KERNEL32.dll [93 entries]

Call via	Name	Ordinal	Original Thunk	Thunk	Forwarder	Hint
1D000	CreateDirectoryA	-	23214	23214	-	C1
1D004	CloseHandle	-	23228	23228	-	8E
1D008	GetLastError	-	23236	23236	-	26A
1D00C	OpenProcess	-	23246	23246	-	408
1D010	VirtualFree	-	23254	23254	-	5AE
1D014	CreateToolhelp...	-	23262	23262	-	10A
1D018	Module32First	-	2327E	2327E	-	3DF
1D01C	Module32Next	-	2328E	2328E	-	3E1
1D020	CreateFileA	-	2329E	2329E	-	CE
1D024	GetFileSize	-	232AC	232AC	-	254
1D028	MapViewOfFile	-	232BA	232BA	-	3DB
1D02C	UnmapViewOfF...	-	232CA	232CA	-	593
1D030	CreateFileMapp...	-	232DC	232DC	-	CF

https://gist.github.com/hashereware/54398aad2ef449c255d2bd2563bb62ca#file-imports_load-h

Basics of a PE file

- Import Table



Basics of a PE file

- Before filling imports:

Disasm: .text | General | DOS Hdr | Rich Hdr | File Hdr | Optional Hdr | Section Hdrs | Imports | Resources

Offset	Name	Func. Count	Bound?	OriginalFirstThun	TimeDateStamp	Forwarder	NameRVA
2E00	KERNEL32.dll	17	FALSE	4134	0	0	42EC
2E14	USER32.dll	21	FALSE	417C	0	0	4462
2E28	GDI32.dll	1	FALSE	412C	0	0	4480
2E3C	ADVAPI32.dll	3	FALSE	411C	0	0	44CE

USER32.dll [21 entries]

Call via	Name	Ordinal	Original Thunk	Thunk	Forwarder	Hint
406C	CreateWindowExW	-	4394	4394	-	6E
4070	GetDesktopWindow	-	43A6	43A6	-	123
4074	CharUpperW	-	43BA	43BA	-	3C
4078	LoadIconW	-	43C8	43C8	-	1ED

Basics of a PE file

- After filling imports:

Disasm: .text | General | DOS Hdr | Rich Hdr | File Hdr | Optional Hdr | Section Hdrs | Imports | Resources

Offset	Name	Func. Count	Bound?	OriginalFirstThun	TimeDateStamp	Forwarder	NameRVA
2E00	KERNEL32.dll	17	FALSE	4134	0	0	42EC
2E14	USER32.dll	21	FALSE	417C	0	0	4462
2E28	GDI32.dll	1	FALSE	412C	0	0	4480
2E3C	ADVAPI32.dll	3	FALSE	411C	0	0	44CE

USER32.dll [21 entries]

Call via	Name	Ordinal	Original Thunk	Thunk	Forwarder	Hint
406C	CreateWindowExW	-	4394	768A2480	-	6E
4070	GetDesktopWindow	-	43A6	768AA100	-	123
4074	CharUpperW	-	43BA	768B53B0	-	3C
4078	LoadIconW	-	43C8	768B5890	-	1ED

Basics of a PE file

Disasm: .text | General | DOS Hdr | Rich Hdr | File Hdr | Optional Hdr | Section Hdrs | Imports | Resources

Offset	Name	Func. Count	Bound?	OriginalFirstThun	Tin
2E00	KERNEL32.dll	17	FALSE	4134	0
2E14	USER32.dll	21	FALSE	417C	0
2E28	GDI32.dll	1	FALSE	412C	0
2E3C	ADVAPI32.dll	3	FALSE	411C	0

USER32.dll [21 entries]

Call via	Name	Ordinal	Original Thunk	Thunk
406C	CreateWindowExW	-	4394	768A2480
4070	GetDesktopWindow	-	43A6	768AA100
4074	CharUpperW	-	43B4	768B53B0
4078	LoadIconW	-	[illegible]	768B5890

Disasm: .text | General | DOS Hdr | Rich Hdr | File Hdr | Optional Hdr | Section Hdrs | Exports

Offset	Name	Value	Meaning
9BCF0	Characteristics	0	
9BCF4	TimeDateStamp	7BD785F8	Saturday, 03.11.2035 17:01:44 UTC
9BCF8	MajorVersion	0	
9BCFA	MinorVersion	0	
9BCFC	Name	9F384	USER32.dll
9BD00	Base	5DE	
9BD04	NumberOfFunc...	4BF	
9BD08	NumberOfNames	3E8	
9BD0[illegible]	AddressOfFunc...	9C918	
9BD10	AddressOfNames	9DC14	

Exported Functions [1215 entries]

Offset	Ordinal	Function RVA	Name RVA	Name
9BF04	659	2C340	9FBC9	CreateWindowExA
9BF08	65A	32480	9FBD9	CreateWindowExW
9BF0C	65B	88D90	9FBE9	CreateWindowInBand
9BF10	65C	88DE0	9FBFC	CreateWindowInBandEx
9BF14	65D	88E30	9FC11	CreateWindowIndirect
9BF18	65E	907F0	9FC26	CreateWindowStationA

DLL Base + Function RVA
Example:
76B70000 + 32480 =
76BA2480

Basics of a PE file

- Export Table

Disasm: .text | General | DOS Hdr | Rich Hdr | File Hdr | Optional Hdr | Section Hdrs | Exports | Imports | Reso

Offset	Name	Value	Meaning
B47C4	Characteristics	0	
B47C8	TimeDateStamp	4CE78B54	sobota, 20.11.2010 08:48:20 UTC
B47CC	MajorVersion	0	
B47CE	MinorVersion	0	
B47D0	Name	B8502	KERNEL32.dll
B47D4	Base	1	
B47D8	NumberOfFunctions	54F	
B47DC	NumberOfNames	54F	
B47E0	AddressOfFunctions	B4FEC	
B47E4	AddressOfNames	B6528	
B47E8	AddressOfNameOrdinals	B7A64	

Exported Functions [1359 entries]

Offset	Ordinal	Function RVA	Name RVA	Name	Forwarder
B481C	D	3BFFB	B85E3	AddRefActCtx	
B4820	E	36399	B85F0	AddSIDToBoundaryDescriptor	
B4824	F	8BD00	B860B	AddSecureMemoryCacheCallback	
B4828	10	BEE06	B8628	AddVectoredContinueHandler	NTDLL.RtlAddVectoredContinueHandler
B482C	11	BEE2A	B8643	AddVectoredExceptionHandler	NTDLL.RtlAddVectoredExceptionHandler
B4830	12	5DF86	B865F	AdjustCalendarDate	
B4834	13	AC328	B8672	AllocConsole	
B4838	14	97CA8	B867F	AllocateUserPhysicalPages	

https://gist.github.com/hashereware/54398aad2ef449c255d2bd2563bb62ca#file-export_lookup-h

Exercise

- Compile the given code of a custom PE loader and get familiar with it