

PE32 Password

Kyungpook National University
Taekwan Yang (CSE GSWC, KERT)

KERT

KNU computer Emergency Response Team

Contents

1. Outline

2. Implementation

3. Conclusion

Outline

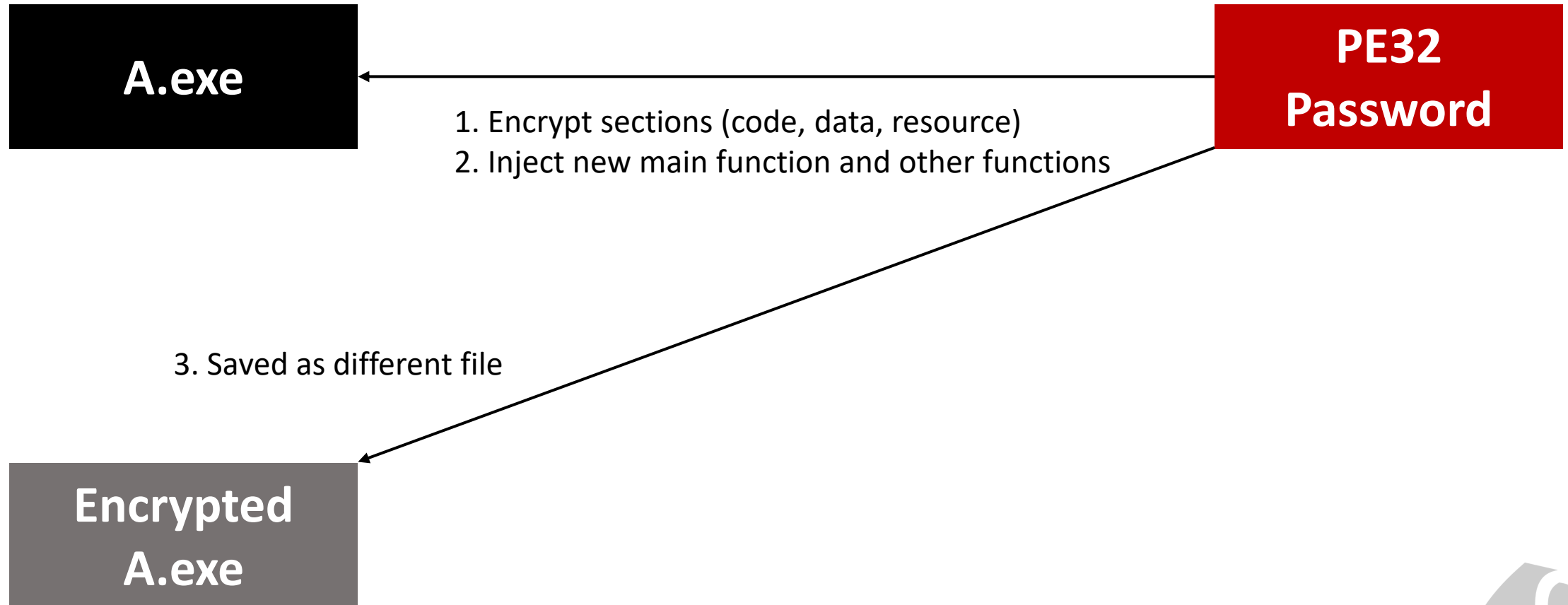
Outline

Project	PE32 Password
Description	Encrypting x32 executable files (.exe) to lock with the password
Programming Language	C++, x86 Assembly
Operating System	Windows 10 Pro x64
Goals	<p>1. Deeper understanding of Windows Internals (format of executable files and how operating system executes files) and SEED128 (symmetric encryption algorithm provided by Korea Internet & Security Agency)</p> <p>2. Developing a sample project based on my library (for modifying windows executable files)</p>

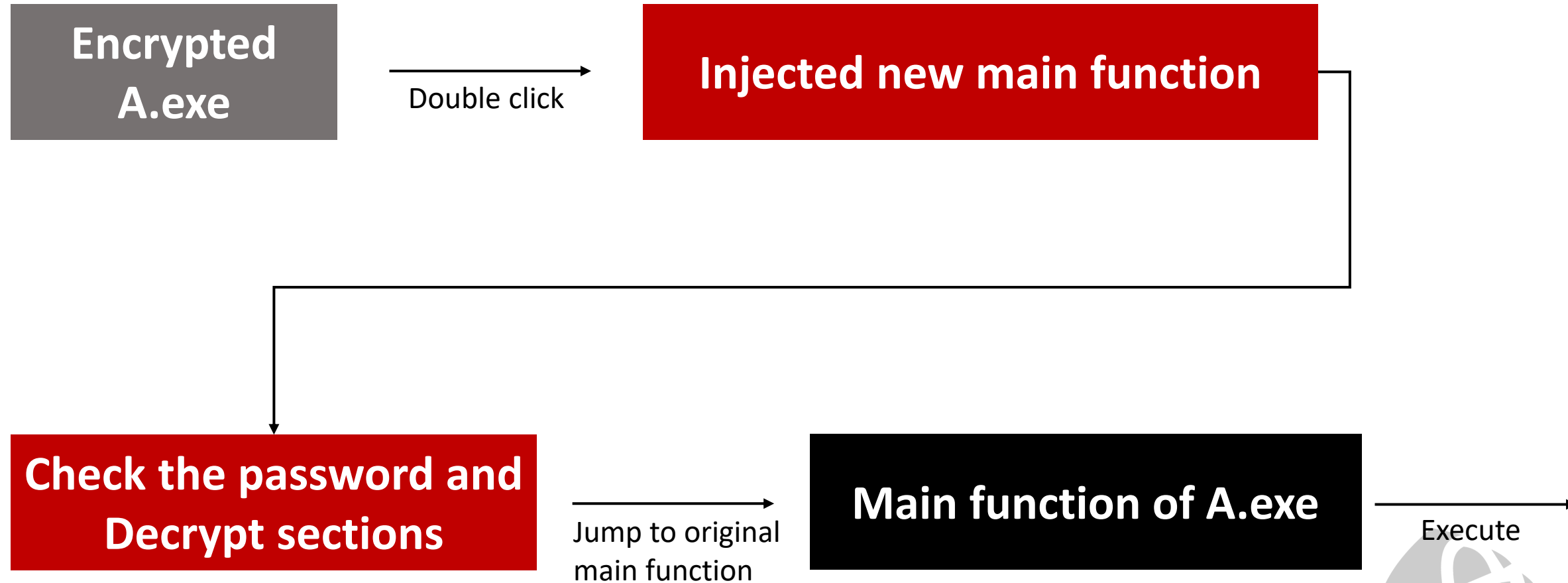
Outline – Process of Normal A



Outline - Encryption Process

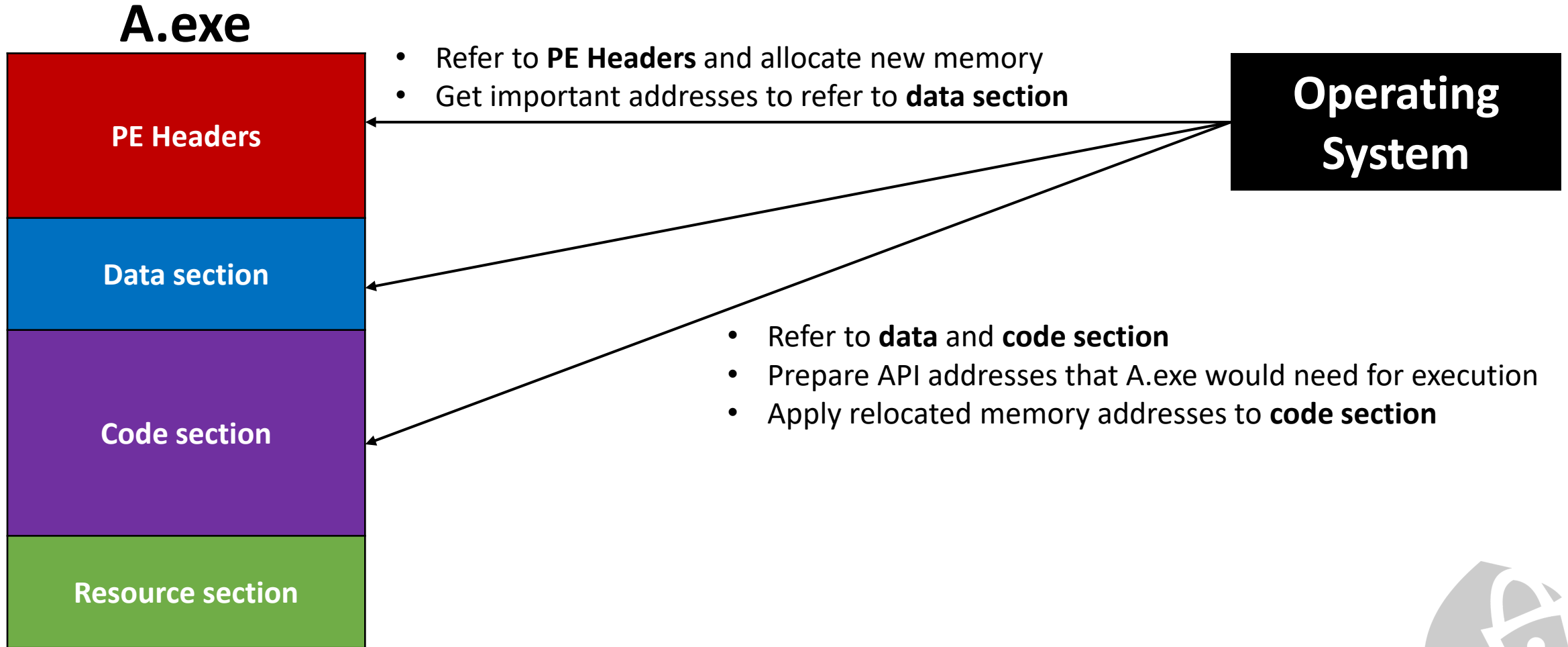


Outline - Process of Encrypted A

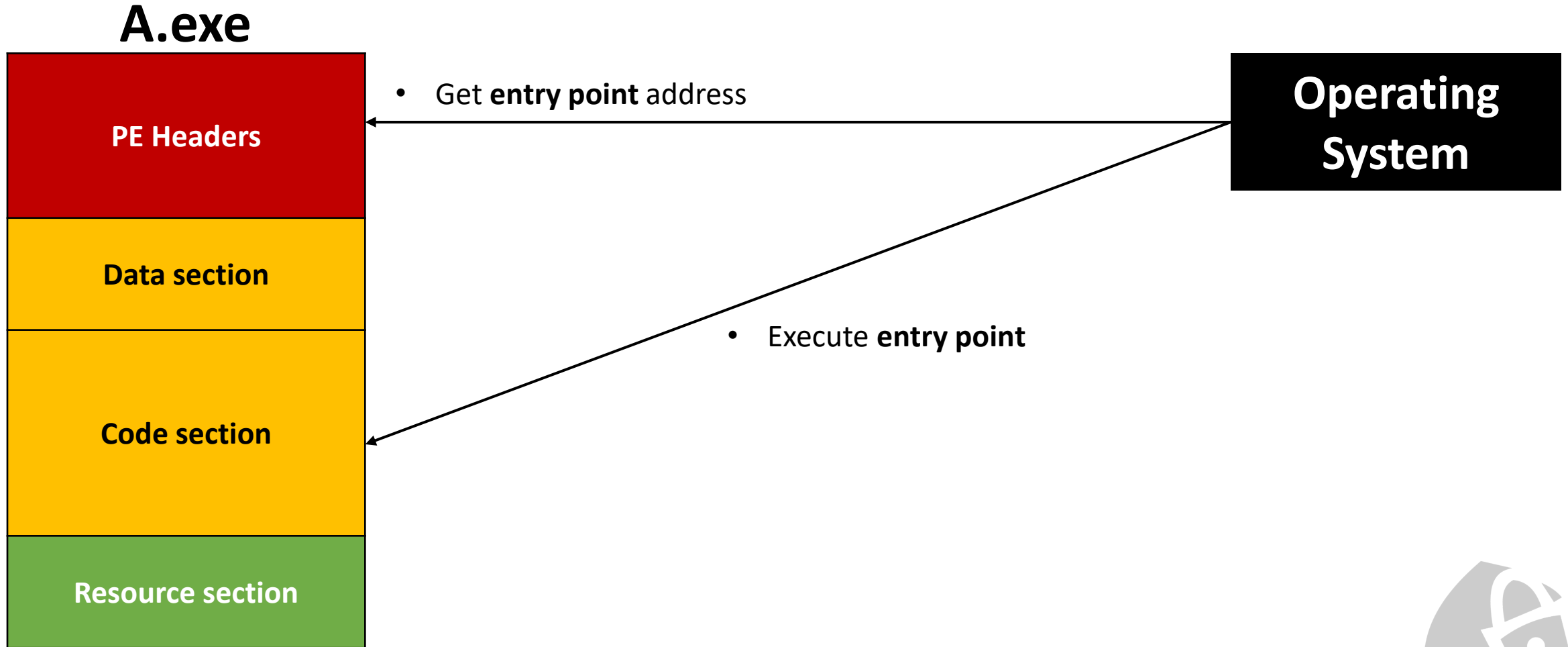


Implementation

Implementation - Normal Case

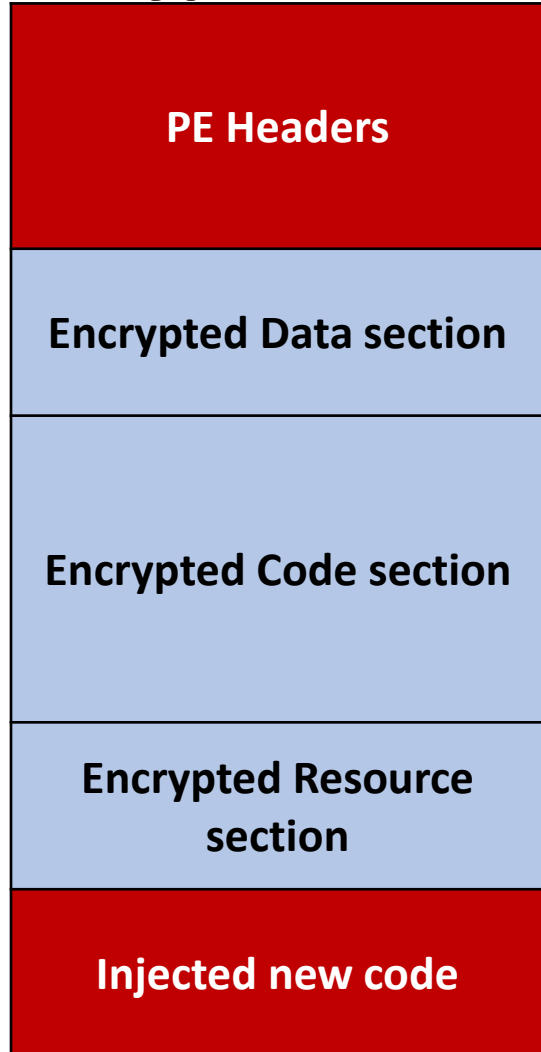


Implementation - Normal Case



Implementation – Error Case

Encrypted A.exe



- Refer to **PE Headers** and allocate new memory
- Get important addresses to refer to **data section**

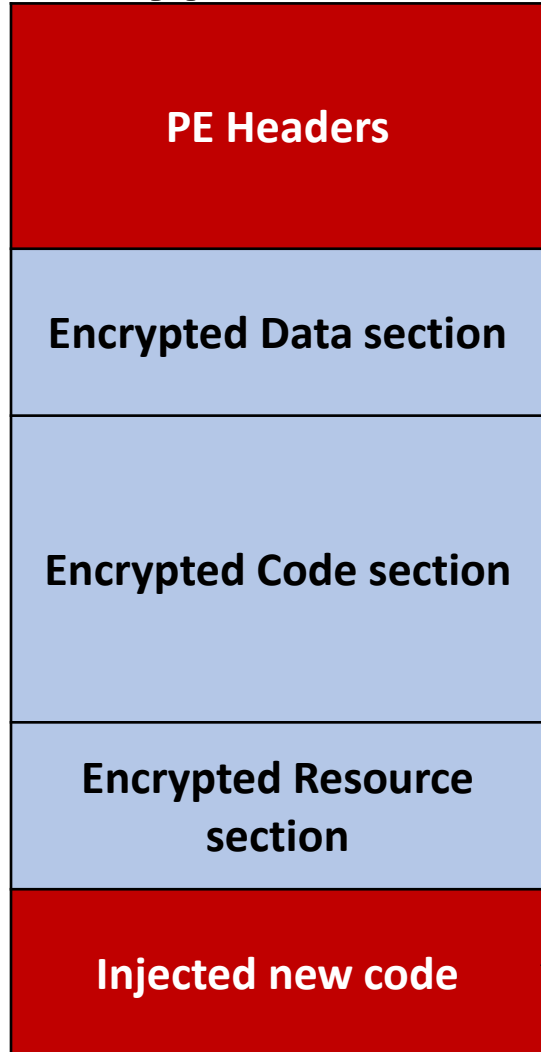
Operating System

- Refer to **data** and **code** section
- **Prepare API addresses that A.exe would need for execution**
- **Apply relocated memory addresses to code section**



Implementation – Encrypted Case

Encrypted A.exe



- Refer to **PE Headers** and allocate new memory
- Get **entry point** address

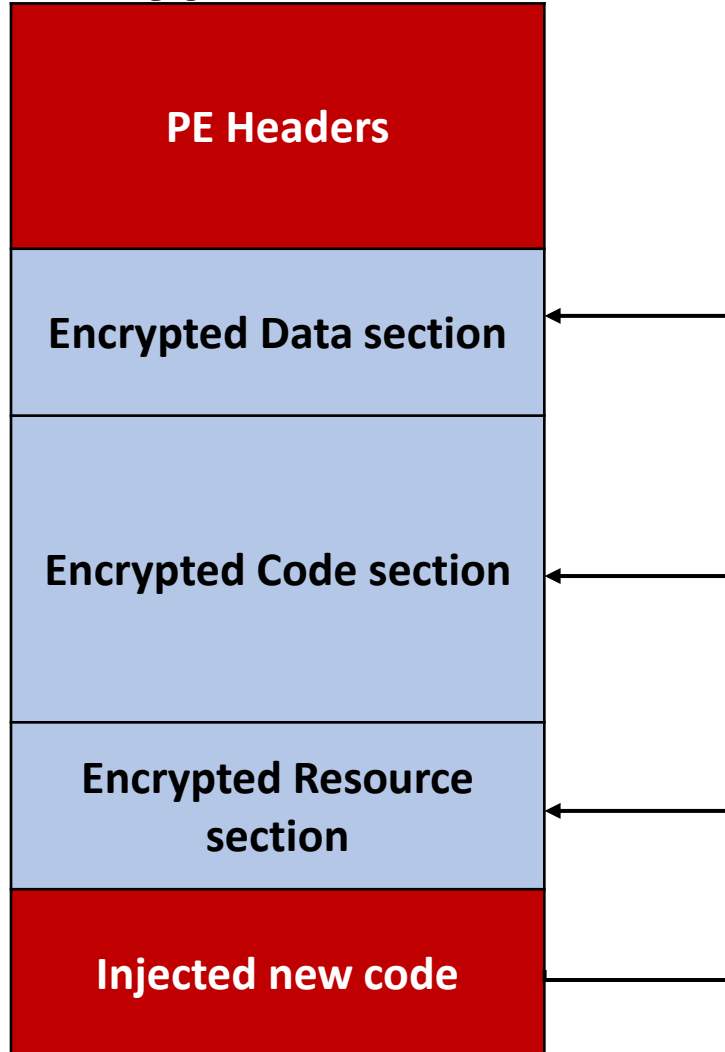
Operating System

- Execute **entry point**



Implementation – Encrypted Case

Encrypted A.exe

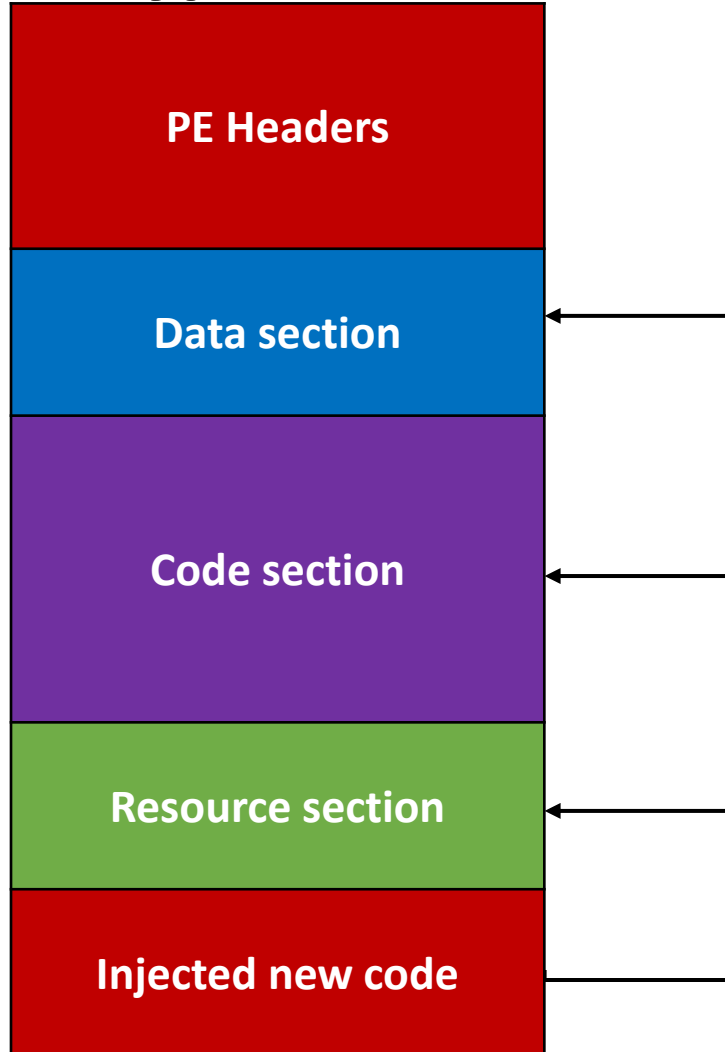


- Decrypt sections if the password is correct

**Operating
System**

Implementation – Encrypted Case

Encrypted A.exe



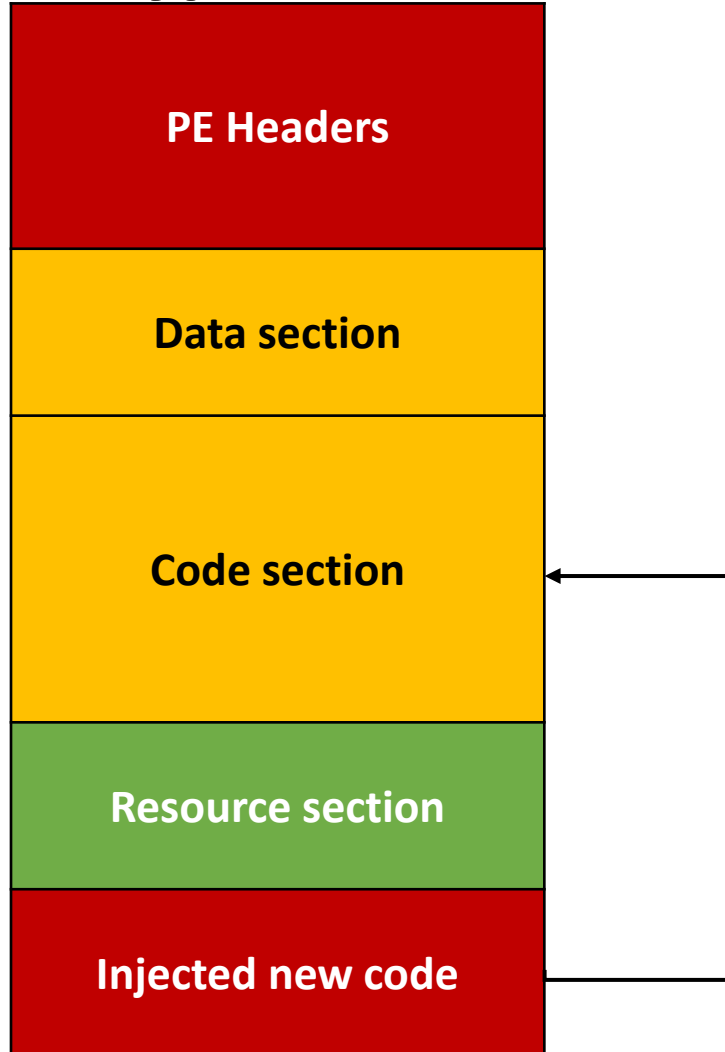
Operating
System

- Decrypt sections if the password is correct
- Get important addresses to refer to **data section**
- Refer to **data** and **code section**
- Prepare API addresses that A.exe would need for execution
- Apply relocated memory addresses to **code section**



Implementation – Encrypted Case

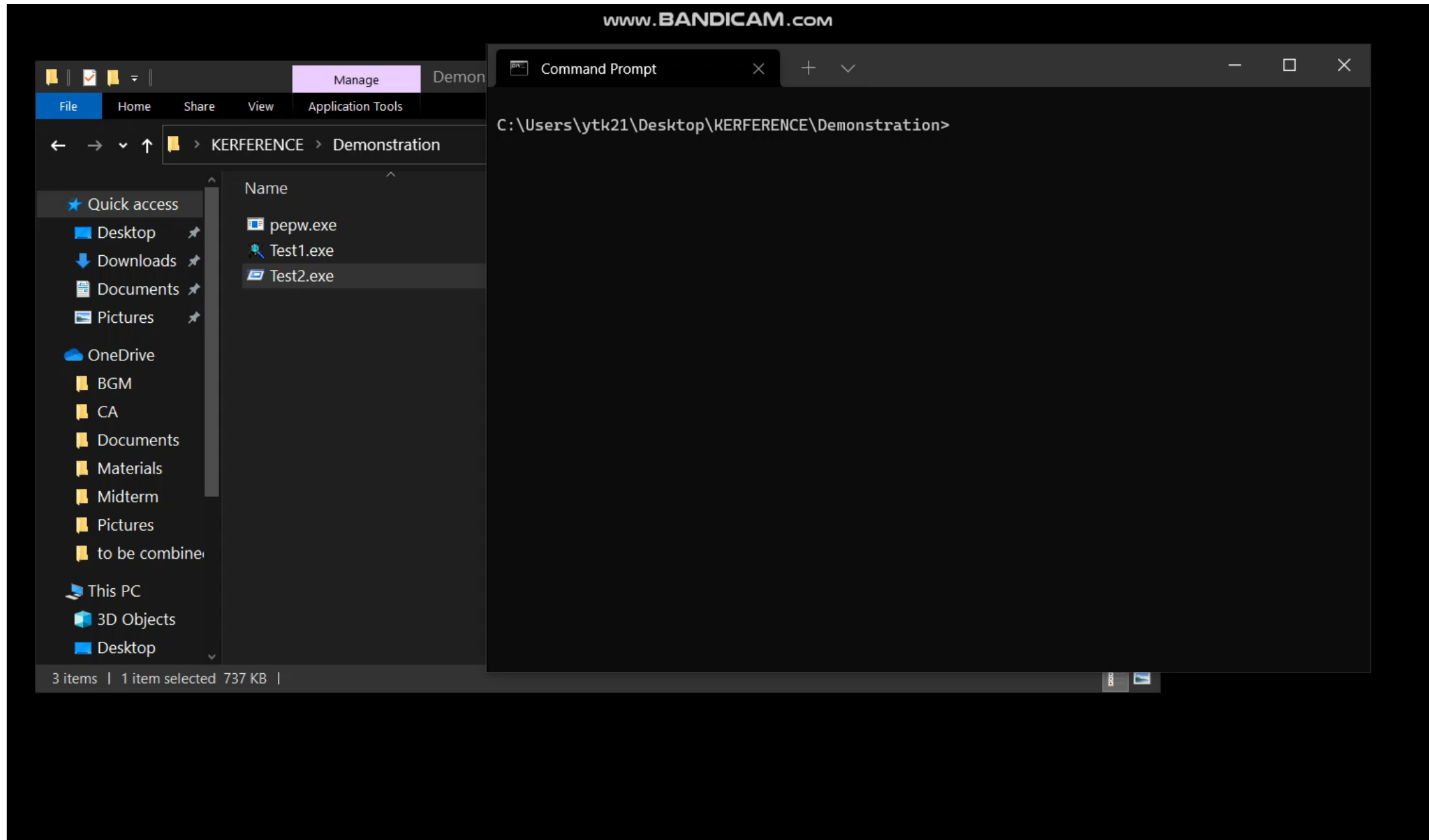
Encrypted A.exe









**Operating
System**







- Get **original entry point** address
- Jump to **original entry point**

Implementation – Demonstration



Implementation - Comparison

Test1.exe																	
Name	Virtual Size	Virtual Add...	Raw Size	Raw Address	Reloc Address	Lin											
00000208	00000210	00000214	00000218	0000021C	00000220	00											
Byte[8]	Dword	Dword	Dword	Dword	Dword	Dv											
.text	0005DC68	00001000	0005DE00	00000400	00000000	00											
.rdata	0001321C	0005F000	00013400	0005E200	00000000	00											
.data	000155A4	00073000	00001200	00071600	00000000	00											
.rsrc	0005F6C0	00089000	0005F800	00072800	00000000	00											
.reloc	000045E4	000E9000	00004600	000D2000	00000000	00											
This section contains:																	
Code Entry Point: 00013487																	
<div><div></div><div></div></div>																	
Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	Ascii
00000000	8B	FF	55	8B	EC	33	C0	85	C9	74	05	3B	4D	08	76	05	ÿUÿi3Ä Ét ;M□v
00000010	B8	57	00	07	80	5D	C2	04	00	8B	FF	55	8B	EC	53	56	,W.● jÄ .ÿUÿiSV
00000020	57	8B	D8	8B	F2	33	C0	33	FF	85	F6	74	1C	39	45	0C	W @ ö3Ä3ÿ öt 9Eÿ
00000030	74	13	8A	13	84	D2	74	0D	88	11	41	43	4E	FF	4D	0C	t!! !öt. ◀ACNÿMÿ
00000040	47	85	F6	75	E8	85	F6	75	07	49	4F	B8	7A	00	07	80	G öuè öu◦IO,z.●
00000050	C6	01	00	8B	4D	08	85	C9	74	02	89	39	5F	5E	5B	5D	Ær. M Ét 9 ^[]
00000060	C2	08	00	8B	FF	55	8B	EC	8B	4D	0C	33	C0	85	C9	74	Ä◦.ÿUÿi Mÿ3Ä Ét
00000070	08	81	F9	FF	FF	FF	7F	76	05	B8	57	00	07	80	85	C0	□ üÿÿÿ v ,W.● jÄ
00000080	7C	40	8B	45	08	53	33	DB	56	8B	F1	85	C9	74	26	8B	@ E◦S3ÜV ÿ Ét&
00000090	55	10	57	BF	FE	FF	FF	7F	2B	F9	2B	D0	8D	0C	37	85	U+Wöbÿÿ +ü+D ÿ7
000000A0	C9	74	0D	8A	0C	02	84	C9	74	06	88	08	40	4E	75	EC	Ét. ÿ Ét- □@Nui

Test1.exe		Test1.exe_out.exe															
Name	Virtual Size	Virtual Add...	Raw Size	Raw Address	Reloc Address	Lin...											
00000208	00000210	00000214	00000218	0000021C	00000220	00...											
Byte[8]	Dword	Dword	Dword	Dword	Dword	Dv...											
.text	0005DC68	00001000	0005DE00	00000400	00000000	00...											
.rdata	0001321C	0005F000	00013400	0005E200	00000000	00...											
.data	000155A4	00073000	00001200	00071600	00000000	00...											
.rsrc	0005F6C0	00089000	0005F800	00072800	00000000	00...											
.reloc	000045E4	000E9000	00004600	000D2000	00000000	00...											
.pepw	00007000	000EE000	00007000	000D6600	00000000	00...											
This section contains:																	
<div><div></div><div></div></div>																	
Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	Ascii
00000000	EE	F6	4B	51	2E	86	50	A7	D3	1D	DB	89	10	44	57	13	iöKQ. PSÓ Ü +DW!!
00000010	28	72	18	26	89	EA	72	DC	01	EC	D1	20	ED	E7	6A	EA	(r†& érÜr ñ. iöjê
00000020	BE	A6	0D	4B	97	96	57	52	F5	E1	DD	6D	46	5A	FF	3A	¾ K WRöäÿmFZÿ:
00000030	5A	43	80	69	6F	32	1D	21	9D	25	CA	AB	11	F7	37	90	ZC io2 ! %E◦◦=7.
00000040	1A	3E	1E	2B	4C	2D	5F	A1	69	DA	9C	90	6E	FE	50	C4	→> +L- _iiÜ nbPÄ
00000050	90	C6	37	8E	F0	54	5C	08	45	60	70	9F	5B	F1	A9	A3	Æ7 èT◦E`p èøè
00000060	A7	F2	FD	11	FF	1D	2B	71	C0	4A	59	52	F0	DF	0C	2F	\$öÿÿÿ +qAJYRöBÿ/
00000070	2F	98	60	03	5D	3A	5C	70	16	7F	88	D4	4D	65	6C	03	/ ` :] : \pT ÖMel L
00000080	EC	84	2A	51	24	E6	E8	9A	AD	63	89	3C	EB	99	10	0F	i *Qsæè -c ◦è +ö
00000090	A4	58	65	D2	A9	9C	25	BB	3E	5A	7A	B4	33	C7	12	F9	¼Xe0@ %>>Zz<3Clù
000000A0	45	CD	51	58	D9	0D	7F	C8	35	D7	79	5E	EF	0D	89	C2	E QXÜ. E5ÿÿ^i. Ä

Implementation - Comparison

The image displays two side-by-side screenshots of the Resource Hacker application, comparing the resource data of two executables: **Test1.exe** (left) and **Test1.exe_out.exe** (right). Both windows show the **BINRES** tree with the **RCDBGSYS64 : 1033** resource selected. The **String Table** is expanded, showing a list of strings with their offsets and lengths. A red box highlights the differences in the **RCDBGSYS64 : 1033** resource between the two files.

Left Window (Test1.exe):

- BINRES**
 - RCDBGSVC : 1033
 - RCDBGSYS : 1033
 - RCDBGSYS64 : 1033
 - Cursor
 - Bitmap
 - Icon
 - Menu
 - Dialog
 - String Table
 - 2501 : 1033
 - 2502 : 1033
 - 2503 : 1033
 - 2504 : 1033

Right Window (Test1.exe_out.exe):

- BINRES**
 - RCDBGSVC : 1033
 - RCDBGSYS : 1033
 - RCDBGSYS64 : 1033
 - Cursor
 - Bitmap
 - Icon
 - Menu
 - Dialog
 - String Table
 - 2501 : 1033
 - 2502 : 1033
 - 2503 : 1033
 - 2504 : 1033

Comparison of String Table Data:

Offset	Test1.exe	Test1.exe_out.exe
00082E48	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00	F8 F5 23 3B 64 32 68 8A 2C 72 6E 6E 10 05 70 7F
00082E58	B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00	DA F9 B0 44 B5 0F 47 69 9D 8F 40 A8 99 39 03 B1
00082E68	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	06 6F E4 EE 3A 17 C6 4C 4F D7 D1 BA 91 65 76 CA
00082E78	00 00 00 00 00 00 00 00 00 00 00 00 00 01 00 00	78 72 75 66 C0 8E E0 71 28 9E 3D 03 A9 2B 2F F9
00082E88	0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68	62 4C FF E2 D4 5E 16 F9 EC D4 2E 84 8F 7B 05 3A
00082E98	69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F	76 9E B8 72 AE 8F B2 03 02 E6 21 5E 26 EE 59 E6
00082EA8	74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20	63 AB DC 93 BF 24 6D 13 47 ED 77 05 77 4C 72 00
00082EB8	6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00	FC B9 B9 04 9D F2 83 52 B2 93 D9 C4 6F 37 AE 9C
00082EC8	CB 5C 4C DB 8F 3D 22 88 8F 3D 22 88 8F 3D 22 88	05 F3 CD BE 7A 70 98 56 94 40 7A D2 4A CA 73 6D
00082ED8	3B A1 D3 88 85 3D 22 88 3B A1 D1 88 0D 3D 22 88	58 61 41 D9 A4 B9 75 3F 5B 5C 29 29 C4 87 5C 18
00082EE8	3B A1 D0 88 97 3D 22 88 DD 55 27 89 AB 3D 22 88	2A EB BD 6B 3F A6 D6 AE AC E4 CE A7 2A E9 9C 61
00082EF8	DD 55 26 89 9E 3D 22 88 DD 55 21 89 9C 3D 22 88	C3 75 9D A3 B2 52 38 DC 68 6C 7C ED A1 4E C6 B7
00082F08	8F 3D 23 88 1A 3D 22 88 86 45 B1 88 88 3D 22 88	8C 7C 16 5B 28 55 F5 4C 6F A0 C1 4D 07 A9 8C EA
00082F18	14 54 26 89 8B 3D 22 88 14 54 DD 88 8E 3D 22 88	59 78 3B 8F BA C3 AC 43 7E 60 E6 E3 1D E5 D6 CD
00082F28	14 54 26 89 8B 3D 22 88 14 54 DD 88 8E 3D 22 88	8C 11 55 55 08 07 40 10 52 8A D0 42 8C 20 1D B4

Conclusion

Conclusion

What I learned from this project	<ol style="list-style-type: none">1. How to deal with big and complex projects2. How to make features more efficient in terms of the time complexity
What I regret the most during the project period	<ol style="list-style-type: none">1. Being lazy2. The part where I couldn't learn SEED128 ECB Algorithm more deeply due to the deadline
Future goals	<ol style="list-style-type: none">1. Releasing my library and this project on GitHub officially2. Developing a better encryptor for software security using my library

Thank you

