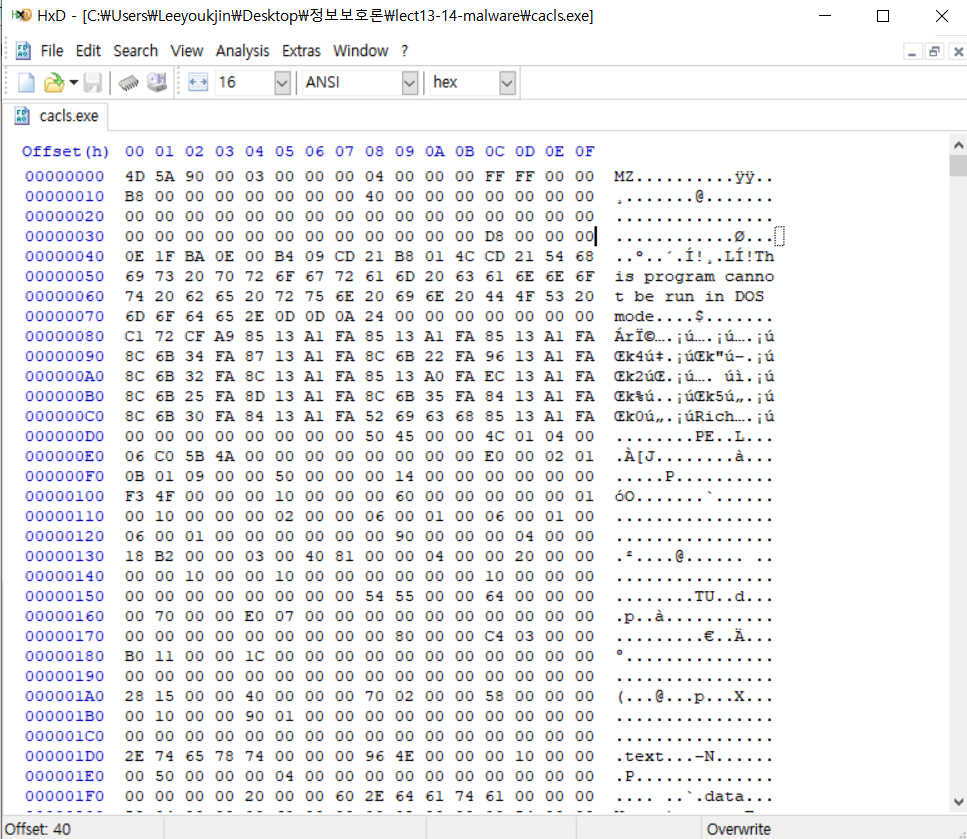
Lect13\_malware\_final

**12141163 이욱진**

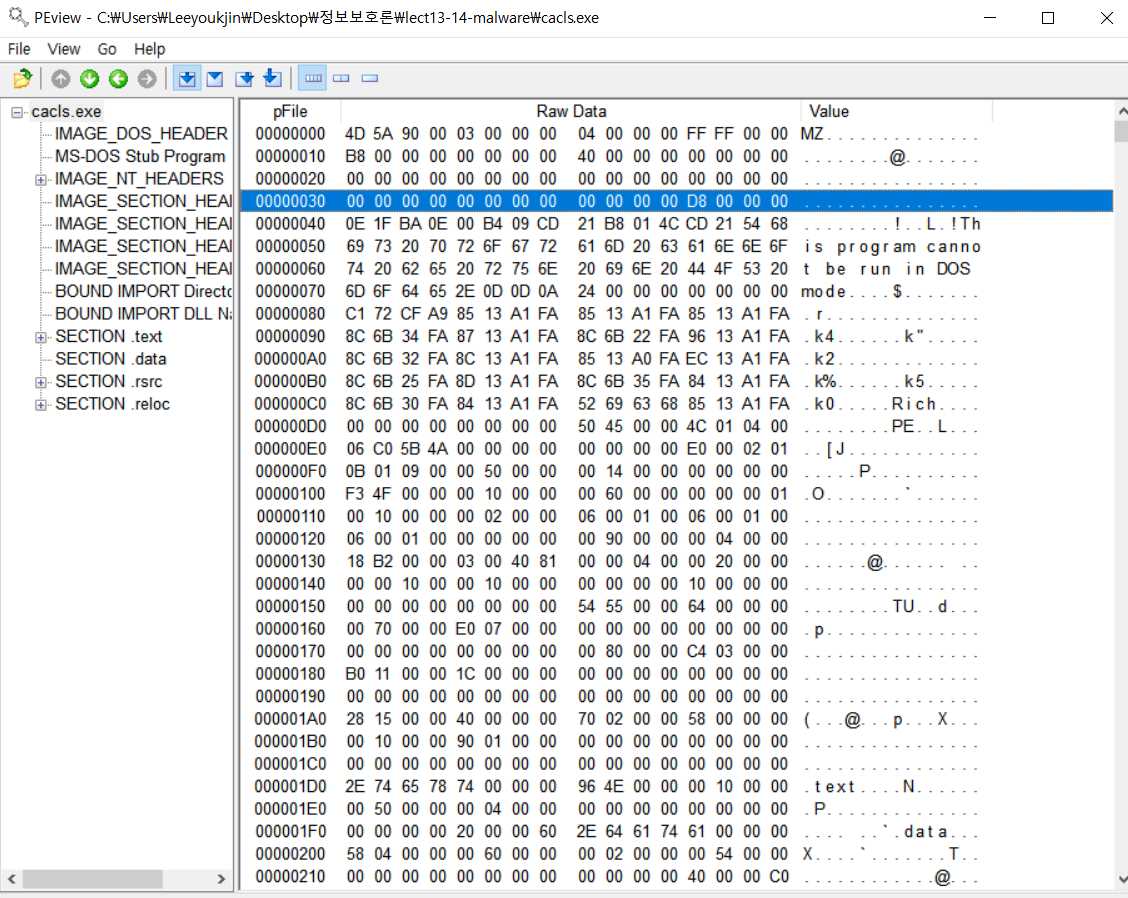
Homework

1) Open cacls.exe with HxD and extract following informtion: e\_lfanew, NumberOfSections, AddressOfEntryPoint, BaseOfCode, BaseOfData, and ImageBase. Check your answer with PEview. (If you are using MacOS, use xxd instead of HxD and skip PEview part.)

HxD를 통하여 cacls.exe 를 열어보았습니다.



이어서 PEview를 통해서도 열어보았고 PEview는 PE format에 맞게 정리되어 나타내고있었습니다.

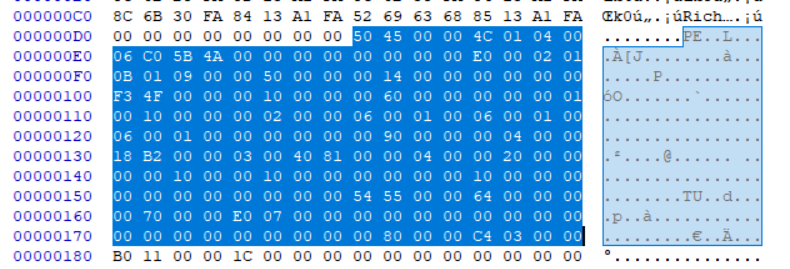


e\_lfanew : 0000003c~3f 까지 존재하였고 000000D8에 nt header가 존재하는 것을 알 수 있었습니다.

<PEview e\_lfanew>

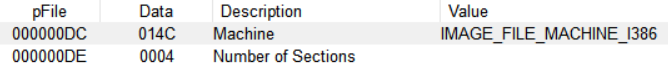


NumberofSections : 아래 사진의 드래그한 부분은 image nt header에대한 부분이고 그 중에서 첫 4바이트는 pe signature(00 00 45 50), machine(01 4C), NumberOfSections(00 04)로 시작했습니다.

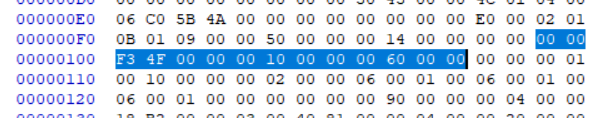


<PEview NumberofSections>





Optional header의 standard fields에서 magic(2 byte), AddressOfEntryPoint(4 byte), BaseOfCode(4 byte), BaseOfData(4 byte)는 아래사진과 같이 찾을 수 있었습니다.



Magic : 00 00

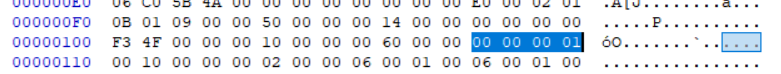
AddressOfEntryPoint: 00 00 4f f3

BaseOfCode : 00 00 10 00

BaseOfData : 00 00 60 00

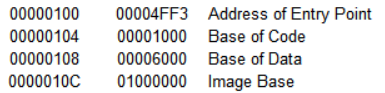
이어서 Optional header의 NT additional fields에서 ImageBase(4 byte)를 찾을 수 있었습니다.

ImageBase : 01 00 00 00



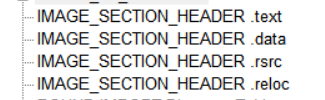
<PEview Optional header>





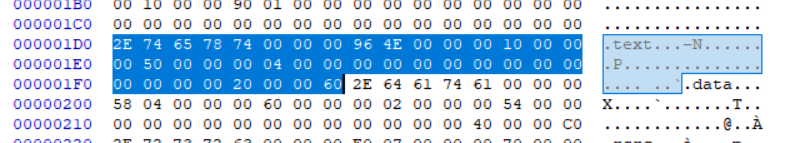
2) How many sections are there in cacls.exe? Show the name, rva, and file offset of all section.

Peview를 통하여 살펴보았을 때 section은 4개로 확인할 수 있었습니다.



이어서 section table을 찾아서 확인해보았습니다.

Section header를 찾았고 정보를 확인해보았습니다.



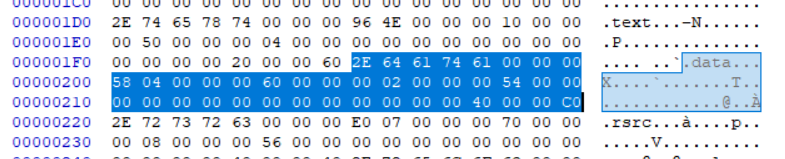
Name(8) : 2E 74 65 78 74 00 00 00 (.text)

Rel. virtual address(4) : 00 10 00 00

Size(4) : 00 50 00 00

File address(4) : 00 04 00 00

Characteristics(4) : 20 00 00 60



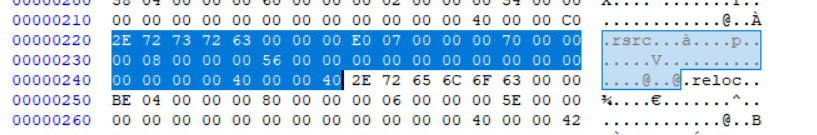
Name(8) : 2E 64 61 74 61 00 00 00 (.data)

Rel. virtual address(4) :00 60 00 00

Size(4) : 00 02 00 00

File address(4) : 00 54 00 00

Characteristics(4) : 40 00 00 C0



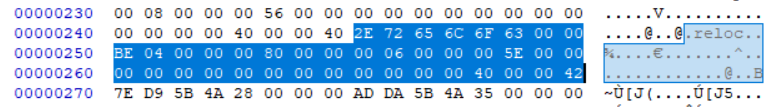
Name(8) :2E 72 73 72 63 00 00 00(.rsrc)

Rel. virtual address(4) : 00 70 00 00

Size(4) : 00 08 00 00

File address(4) : 00 56 00 00

Characteristics(4) : 40 00 00 40



Name(8) : 2E 72 65 6C 6f 63 00 00(.reloc)

Rel. virtual address(4) : 00 80 00 00

Size(4) :00 06 00 00

File address(4) : 00 5E 00 00

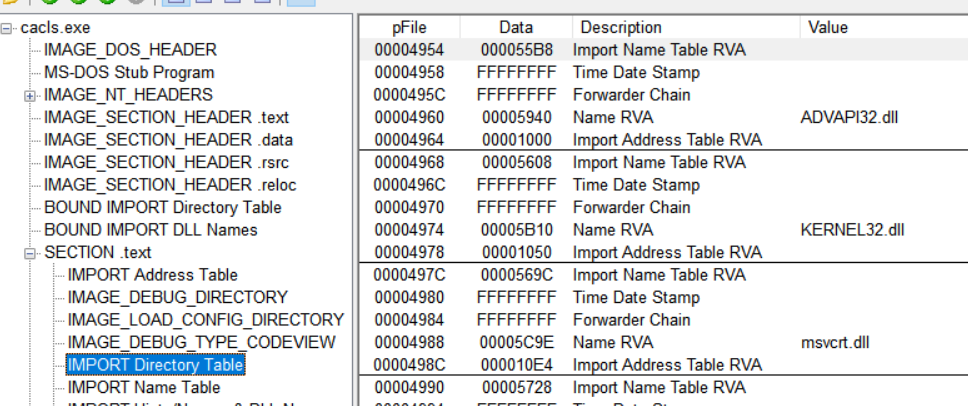
Characteristics(4) : 40 00 00 42

3) What is the file offset for IMAGE\_DATA\_DIRECTORY array? What is the file offset of the import directory? How many DLLs do you see in the import directory?

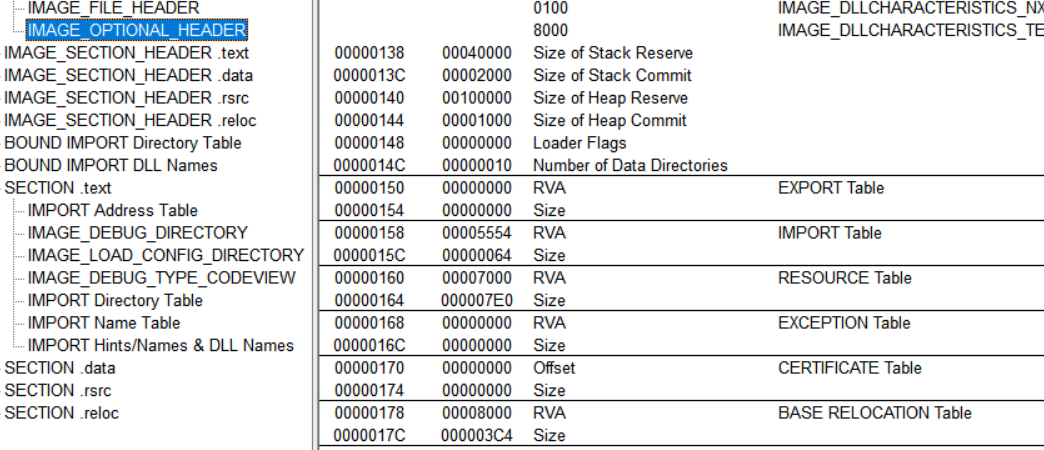
위에서 찾았던 파일 오프셋에 의하여 다음사진과 같이 00000400부터시작함을 알 수 있었습니다.



그리고 import directory table을 다음사진과같이 확인할 수 있었습니다.

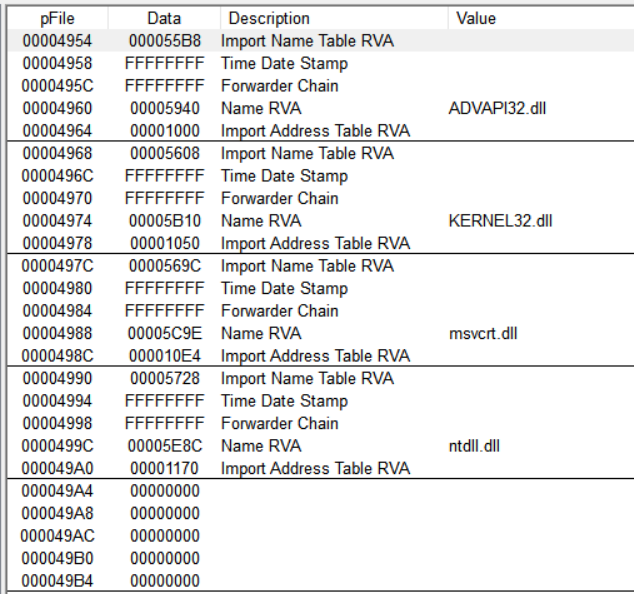


Import directory table의 address계산하는 방법은 기존의 optional Header에서 export table, import table등의 값을 찾을 수 있었습니다.



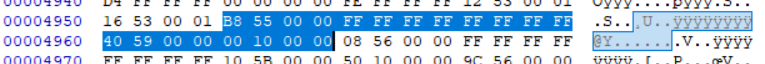
그래서 import table의 5554 에서 .text의 rva를 빼주고 file offset을 더해서 4954 주소부터 DLL이 존재한다는 것을 알 수 있었습니다.

이렇게 찾은 4954 주소를 찾아가보면 다음과같이 dll이 4개 존재하는 것을 볼 수 있었습니다.



4) Show the file offset of OriginalFirstThunk, Name, and FirstThunk for all DLLs in cacls.exe.

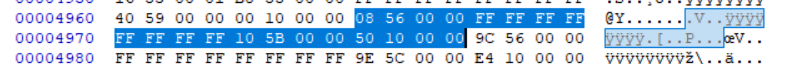
DLL1



OriginalFirstThunk(4) : B8 55 00 00

Name(4) : 40 59 00 00

FirstThunk(4) : 00 10 00 00

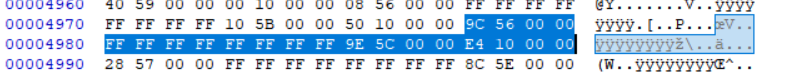


DLL2

OriginalFirstThunk(4) : 08 56 00 00

Name(4) : 10 5B 00 00

FirstThunk(4) : 50 10 00 00

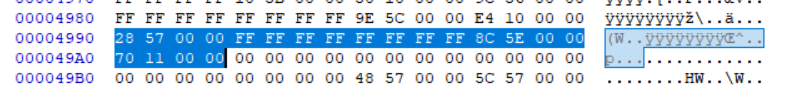


DLL3

OriginalFirstThunk(4) : 9C 56 00 00

Name(4) : 9E 5C 00 00

FirstThunk(4) : E4 10 00 00



DLL4

OriginalFirstThunk(4) : 28 57 00 00

Name(4) : 8C 5E 00 00

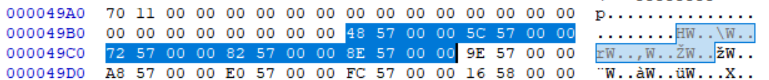
FirstThunk(4) : 70 11 00 00

5) Go to the file offset you found in 4) and show names, first 5 API names, and first 5 API function addresses (which are yet garbage addresses) for all DLLs in cacls.exe. Do not use PEview for this problem.

이어서 위에서 찾은 DLL1을 가지고 api를 찾아보았습니다.

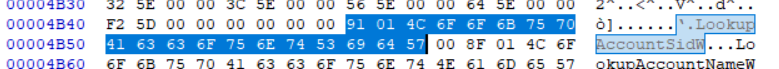
OriginalFirstThunk(4) : B8 55 00 00

55b8 – 1000 + 400 = 49b8



Api1 rva : 48 57 00 00

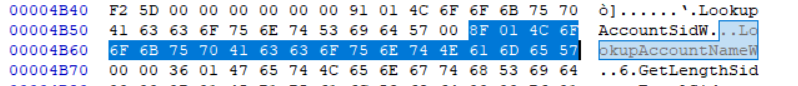
5748 – 1000 + 400 = 4b48



Api1 Name: LookupAccountSidW

Api2 rva : 5C 57 00 00

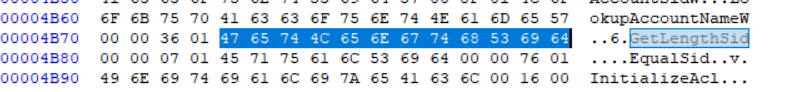
575C – 1000 + 400 = 4b5c



Api2 Name : LookupAccountNameW

Api3 rva : 72 57 00 00

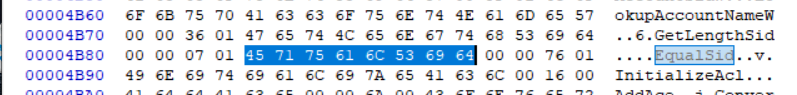
5772 – 1000 + 400 = 4b72



Api3 Name : GetLengthSid

Api4 rva : 82 57 00 00

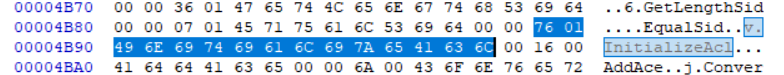
5782 – 1000 + 400 = 4b82



Api4 Name : EqualSid

Api5 rva : 8E 57 00 00

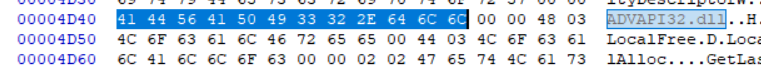
578E – 1000 + 400 = 4b8e



Api5 Name : InitializeAcl

Name(4) : 40 59 00 00

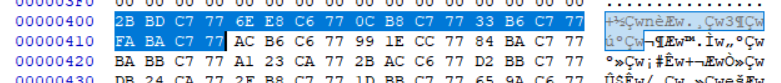
5940 – 1000 + 400 = 4d40



Name의 주소를 찾아보니 ADBAPI32.dll 이름이 확인되었습니다.

FirstThunk(4) : 00 10 00 00

1000 – 1000 + 400 = 400



Api1 va : 2B BD C7 77

Api2 va : 6E E8 C6 77

Api3 va : 0C B8 C7 77

Api4 va : 33 B6 C7 77

Api5 va : Fa Ba C7 77