**Міністерство освіти і науки України**

**Національний технічний університет України**

**«Київський політехнічний інститут імені Ігоря Сікорського»**

**Факультет інформатики та обчислювальної техніки**

**Кафедра обчислювальної техніки**

**Лабораторна робота №3**

з дисципліни «Системне програмування»

на тему «Дослідження структури програм формату EXE.»

Виконав: Перевірив:

Студент 2 курсу ФІОТ доц. Павлов В. Г.

групи ІМ-22

Довженко Антон Андрійович

номер в списку групи: 9

Київ 2024

**Мета роботи:** Ознайомлення із специфікацією COFF (Common Object File Format). Вивчення прийомів дослідження структури файлів PE-формату.

Порядок виконання роботи

1. Вивчити структуру програм формату EXE.

2. Розробити програму на мові Асемблер, за допомогою якої у віконному інтерфейсі виводяться персональні дані студента – ПІБ, дата народження, номер залікової книжки тощо (див. лаб. роботу 1), але лише при вірно введеному паролі.

3. Виконати компіляцію розробленого файлу у формат EXE.

4. Перевірити роботу програми шляхом введення як правильного, так і невірного паролів.

5. Отриманий виконавчий файл дослідити за допомогою програми HEX-редактора HIEW32 (https://uk.wikipedia.org/wiki/Hiew) або WinHex (http://rainbowsky.ru/system/winhex/ - trial версия\*)[2].

6. На скріншоте перших 25 рядків вмісту файлу обвести кольоровим олівцем або фломастером області MS-DOS заголовка (DOS\_HEADER), PE заголовка (PE\_HEADER) і таблиці секцій (SECTION\_HEADERS). Скріншот привести в звіті по лабораторній роботі.

7. Відповідно до опису секцій [1] скласти таблицю, в яку занести параметри свого файлу, вказані в розділах 3.3.1, 3.4.1 і 4 (перша таблиця). ]

8. У останньому стовпчику таблиці розшифрувати виписані значення полів заголовка файлу. Таблицю привести в звіті по лабораторній роботі.

9. Провести дослідження того ж файлу за допомогою меню "PE Editor" безкоштовної програми PE Tools (http://soft.mydiv.net/win/download-PE-Tools.html\*). Все скріншоти вікон програми з даними, відповідними раніше побудованій таблиці, привести в звіті по лабораторній роботі. 10

10. Дослідити таблицю імпорту (Import Directory) даного файлу і визначити, які саме функції використовуються з бібліотек, що підключаються. Скріншоти вікон Import Directory з функціями, що імпортуються, з кожного бібліотечного файлу привести в звіті по лабораторній роботі.

11. Знайти в тексті файлу по зсуву, узятому з побудованої таблиці, секцію з даними і переконатися, що текст оригінала пароля, що міститься в тексті програми, може бути легко виявлений за допомогою HEX-редактора. Привести скріншот цього фрагмента програми у вигляді HEX - коду в звіті по лабораторній роботі.

12. Виконати шифрування пароля за допомогою функції XOR, знову скомпілювати EXE -файл і переконатися, що тепер вони не виявляються явним чином в тексті виконуваного EXE - файлу. Привести скріншоти цієї програми в режимах «Hex» і «Text» в звіті по лабораторній роботі.

13. Зробити висновки по лабораторній роботі.

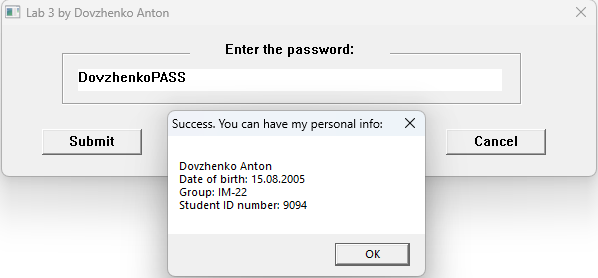
**Виконання роботи**

4. Перевірити роботу програми шляхом введення як правильного, так і невірного паролів.

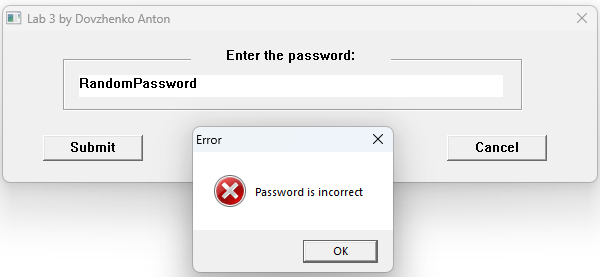
Правильний пароль – “DovzhenkoPASS”

**Програма з незашифрованим паролем**

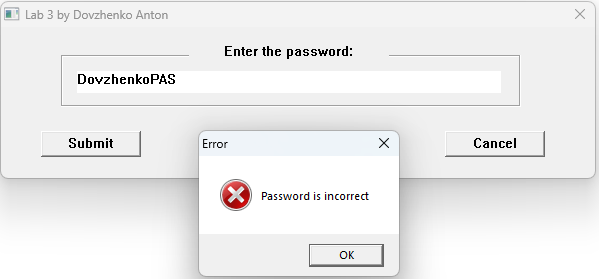
Введення правильного паролю



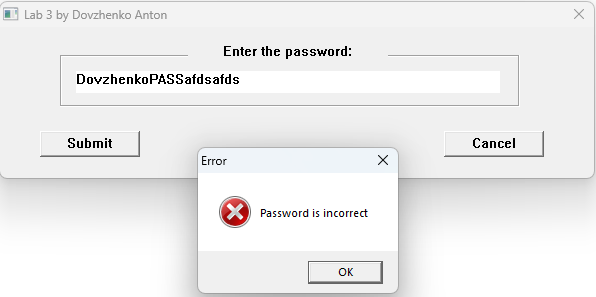
Введення неправильного паролю



Введення частини правильного паролю

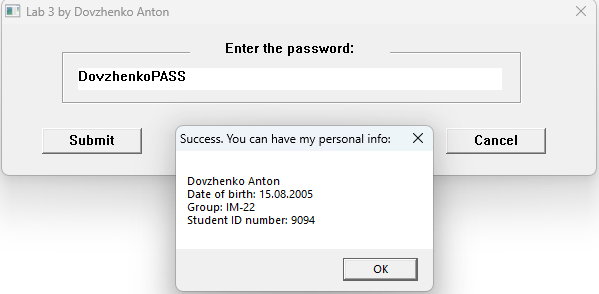


Введення паролю із зайвими літерами

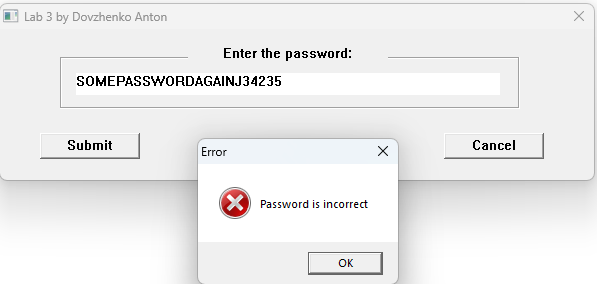


**Програма з зашифрованим паролем**

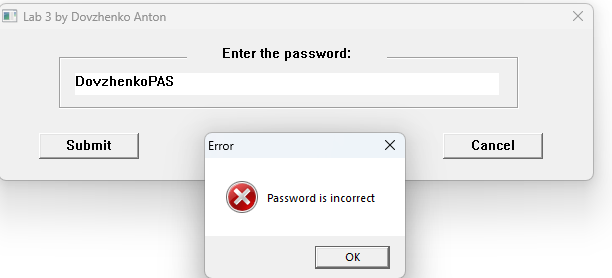
Введення правильного паролю



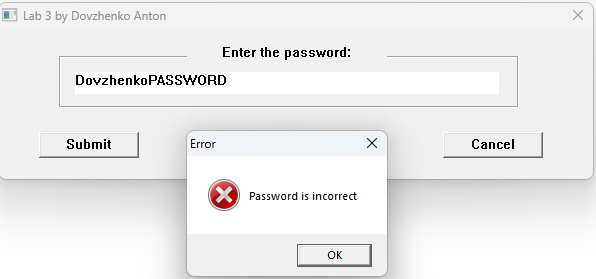
Введення неправильного паролю



Введення частини правильного паролю



Введення паролю із зайвими літерами

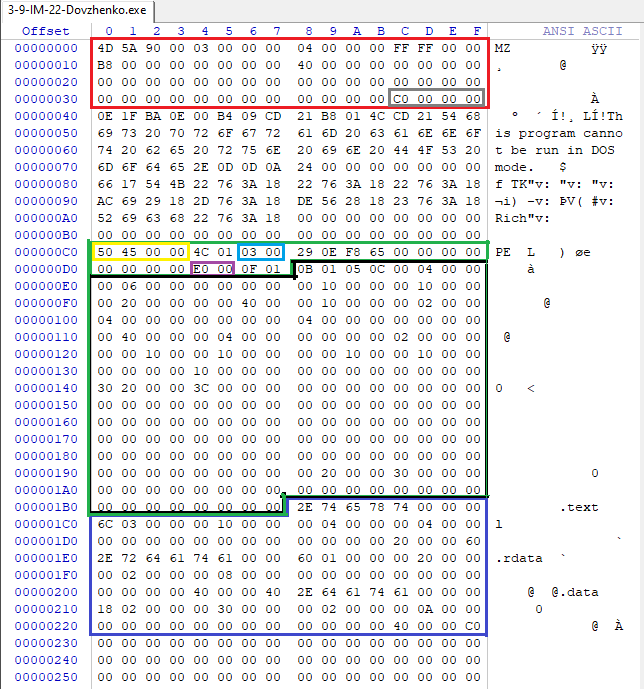


5-6. Отриманий виконавчий файл дослідити за допомогою програми HEX-редактора WinHex.  
На скріншоте перших 25 рядків вмісту файлу обвести кольоровим олівцем або фломастером області MS-DOS заголовка (DOS\_HEADER), PE заголовка (PE\_HEADER) і таблиці секцій (SECTION\_HEADERS). Скріншот привести в звіті по лабораторній роботі.

З самого верху на скріншоті виділено DOS\_HEADER червоним кольором, він займає 64 байти, також по зсуву **0x3C** було знайдено зсув від початку файлу до заголовка PE, виділено сірим кольором.

Наступним зеленим кольором виділений заголовок PE (PE\_HEADER), який починається з 4-байтовою сигнатури (виділено жовтим кольором), далі заголовок PE- файлу (складається з 20 байтів), де обведено блакитним кольором кількість секцій у файлі (NumberOfSections). Після також обведений фіолетовим кольором розмір необов’язкового заголовка (SizeOfOptionalHeaders). Далі одразу ж йде необов’язковий заголовок, який виділений **чорним кольором**.

Після необов’язкового заголовку йде таблиці секцій (SECTION\_HEADERS), які виділені синім кольором. Тут у нас, як раніше було проаналізовано, 3 секції (.text, .rdata, data), розмір кожної становить 40 байтів.



7-8. Відповідно до опису секцій скласти таблицю, в яку занести параметри свого файлу, вказані в розділах 3.3.1, 3.4.1 і 4.

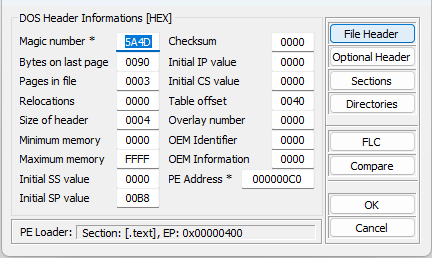
|  |  |  |
| --- | --- | --- |
| **3.3.1. Machine Types** | | |
| Назва поля | Значення | Опис |
| Machine | 01 4C | Позначення процесора. Intel 386 або пізніші процесори |

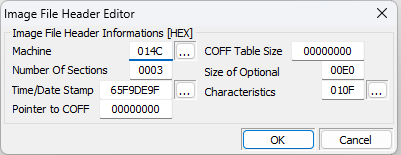
|  |  |  |
| --- | --- | --- |
| **3.4.1 Optional Header Standard Fields (Image Only)** | | |
| Назва поля | Значення | Опис |
| Magic | 01 0B | Ціле число без знаку, що ідентифікує стан файлу. |
| MajorLinkerVersion | 05 | Основний номер версії лінкера. |
| MinorLinkerVersion | 0C | Другорядний номер лінкера. |
| SizeOfCode | 00 00 04 00 | Розмір секції коду. |
| SizeOfInitializedData | 00 00 06 00 | Розмір ініціалізованого розділу даних або суми всіх розділів. |
| SizeOfUninitializedData | 00 00 00 00 | Розмір неініціалізованої розділу даних або суми всіх розділів. |
| AddressOfEntryPoint | 00 00 10 00 | Адреса точки входу. |
| BaseOfCode | 00 00 10 00 | Адреса початку коду. |
| BaseOfData | 00 00 20 00 | Адреса початку даних. |

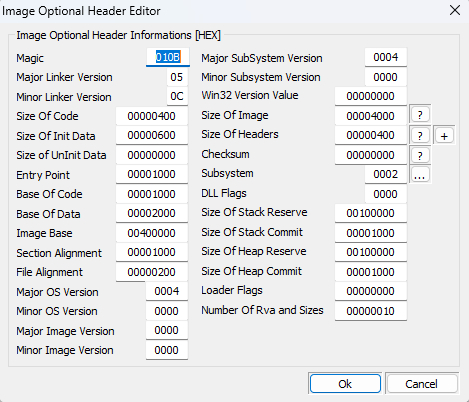
**4. Section Table (Section Headers)** (на прикладі **.text**)

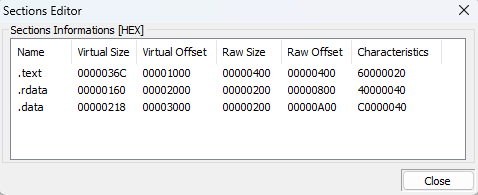
|  |  |  |
| --- | --- | --- |
| Назва поля | Значення | Опис |
| Name | .text | 8-байтовий рядок ASCII. |
| VirtualSize | 00 00 03 6С | Загальних розмір секції, що завантажується в пам’ять. |
| VirtualAddress | 00 00 10 00 | Адреса першого байту секції. |
| SizeOfRawData | 00 00 04 00 | Розмір секції. |
| PointerToRawData | 00 00 04 00 | Вказівник на сторінку номер 1 в COFF файлі |
| PointerToRelocation | 00 00 00 00 | Вказівник файлу до початку релокації для розділу. |
| PointerToLinenumbers | 00 00 00 00 | Вказівник файлу до початку номерів рядків. |
| NumberOfRelocations | 00 00 | Кількість релокаційних входжень секції. |
| NumberofLinenumbers | 00 00 | Кількість записів номерів секції. |
| Charachteristics | 60 00 00 20 | Прапорці, які описують характеристику розділу. |

9. Провести дослідження того ж файлу за допомогою меню "PE Editor"



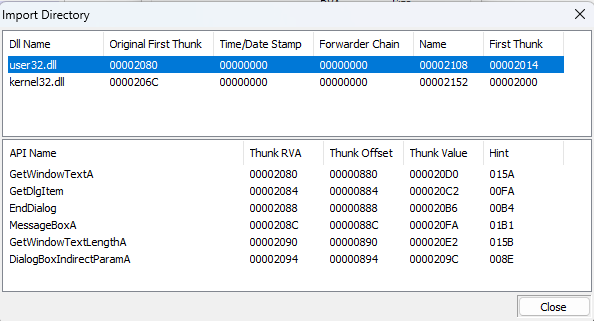




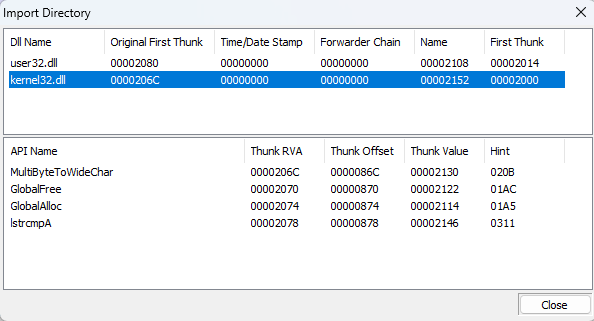


10. Дослідити таблицю імпорту (Import Directory) даного файлу і визначити, які саме функції використовуються з бібліотек, що підключаються.

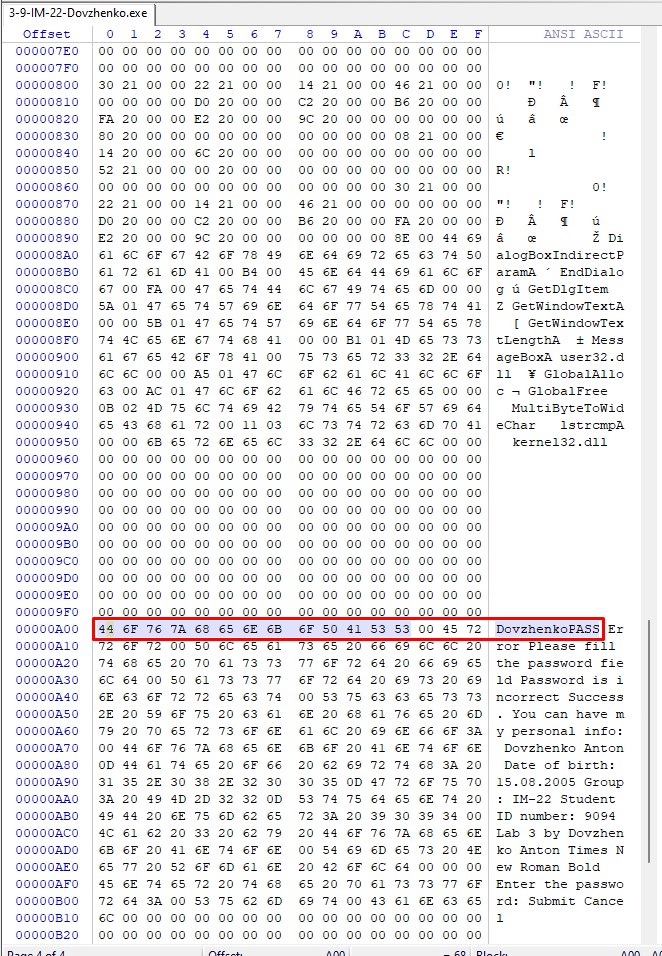
user32.dll



kernel32.dll



11. Знайти в тексті файлу по зсуву, узятому з побудованої таблиці, секцію з даними і переконатися, що текст оригінала пароля, що міститься в тексті програми, може бути легко виявлений за допомогою HEX-редактора



Бачимо явно пароль в тексті.

12. Виконати шифрування пароля за допомогою функції XOR, знову скомпілювати EXE -файл і переконатися, що тепер вони не виявляються явним чином в тексті виконуваного EXE - файлу.

Нехай ключем буде sECRETKEY1479

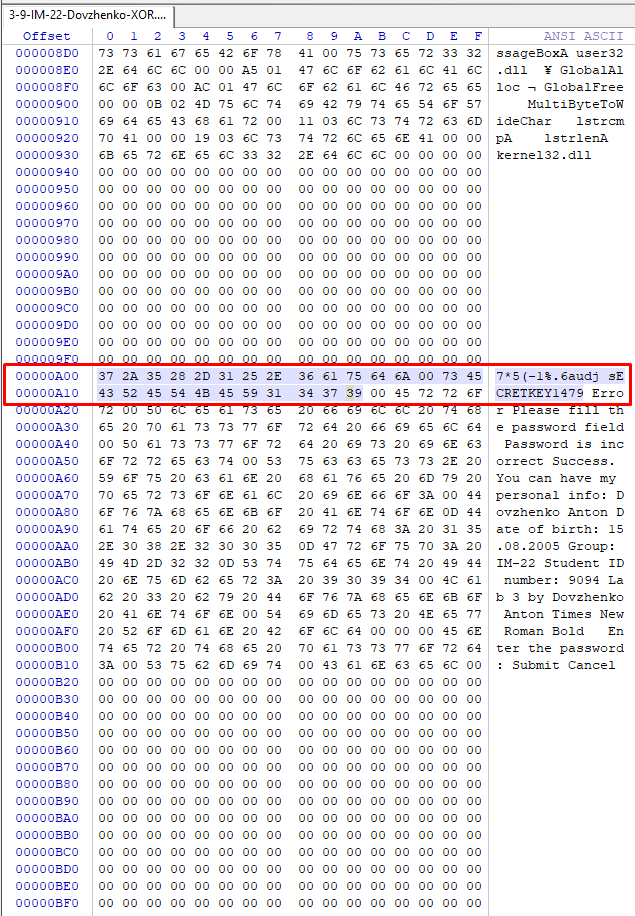
Переведення

|  |  |
| --- | --- |
| D – 010001002 – 4416 | s – 011100112 – 7316 |
| o – 011011112 - 6F16 | E – 010001012 – 4516 |
| v - 011101102 - 7616 | C – 010000112 – 4316 |
| z – 011110102 - 7A16 | R – 010100102 – 5216 |
| h - 011010002 - 6816 | E – 010001012 - 4516 |
| e – 011001012 - 6516 | T – 010101002 – 5416 |
| n - 011011102 - 6E16 | K – 010010112 – 4B16 |
| k – 011010112 - 6B16 | E – 010001012- 4516 |
| o – 011011112 - 6F16 | Y – 010110012 – 5916 |
| P – 010100002 - 5016 | 1 – 001100012 – 2116 |
| A – 010000012 - 4116 | 4 – 001101002 – 3416 |
| S – 010100112 - 5316 | 7 – 001101112 – 3716 |
| S – 010100112 - 5316 | 9 – 001110012 - 3916 |

Шифрування

D XOR s = 010001002  XOR 010100112 = 001101112 = 3716 = 7o XOR E = 011011112  XOR 010001012 = 001010102 = 2A16 = \*  
v XOR C = 011101102  XOR 010000112 = 001101012 = 3516 = 5  
z XOR R = 011110102  XOR 010100102 = 001010002 = 2816 = (  
h XOR E = 011010002  XOR 010001012 = 001011012 = 2D16 = -  
e XOR T = 011001012  XOR 010101002 = 001100012 = 3116 = 1  
n XOR K = 011011102  XOR 010010112 = 001001012 = 2516 = %  
k XOR E = 011010112  XOR 010001012 = 001011102 = 2E16 = .  
o XOR Y = 011011112  XOR 010110012 = 001101102 = 3616 = 6  
P XOR 1 = 010100002  XOR 001100012 = 011000012 = 6116 = a  
A XOR 4 = 010000012  XOR 001101002 = 011101012 = 7516 = u  
S XOR 7 = 010100112  XOR 001101112 = 011001002 = 6416 = d  
S XOR 9 = 010100112  XOR 001110012 = 011010102 = 6A16 = j

Отже, наш зашифрований пароль - 7\*5(-1%.6audj

Привести скріншоти цієї програми в режимах «Hex» і «Text» в звіті по лабораторній роботі.   
Як бачимо, явно пароль не виявляється, присутній лише зашифрований пароль та ключ.

13. Зробити висновки по лабораторній роботі

Отже, протягом виконання лабораторної роботи я опанував розуміння структури програм формату EXE. Я дізнався і вивчив детально структуру, та вигляд формату. Також практично дослідив і розглянув, на прикладі файлу програми, за допомогою спеціальних програм (WinHex, PEtools) цю ж структуру формату EXE, її секції, поля і інше.

Також не менш важливим було практичне застосування шифрування паролю, який використовується в нашій програмі. Перш за все, з метою безпеки. І як показало порівняння, шифрування дійсно допомогло нам «сховати» пароль. Якщо в версії програми без шифру через інструмент WinHex можна було явно прочитати та дізнатися пароль, то у програмі з зашифрованим паролем ми не можемо бачити явно пароль, що суттєво підвищує безпеку даних.