

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ПОЛІСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет інформаційних технологій, обліку та фінансів

Кафедра комп'ютерних технологій

і моделювання систем

Кваліфікаційна робота

на правах рукопису

Дмитрук Данило Вікторович

УДК 004.056.55:004.42

КВАЛІФІКАЦІЙНА РОБОТА

Інформаційна технологія захисту від програм-вимагачів

(тема роботи)

126 «Інформаційні системи та технології»

(шифр і назва спеціальності)

Подається на здобуття освітнього ступеня бакалавр
кваліфікаційна робота містить результати власних досліджень.
Використання ідей, результатів і текстів інших авторів мають посилання
на відповідне джерело

_____ Дмитрук Д. В.

(підпис, ініціали та прізвище здобувача вищої освіти)

Керівник роботи

Молодецька Катерина Валеріївна

(прізвище, ім'я, по батькові)

доктор технічних наук, професор

(науковий ступінь, вчене звання)

Житомир – 2025

Висновок кафедри _____

за результатами попереднього захисту: _____

Протокол засідання кафедри _____

№ _____ від «_____» _____ 20____ р.

Завідувач кафедри _____

(науковий ступінь, вчене звання)

(підпис)

(прізвище, ім'я, по батькові)

«_____» _____ 20____ р.

Результати захисту кваліфікаційної роботи

Здобувач вищої освіти _____ захистив (ла)

(прізвище, ім'я, по батькові)

кваліфікаційну роботу з оцінкою:

сума балів за 100-бальною шкалою _____

за шкалою ECTS _____

за національною шкалою _____

Секретар ЕК

(науковий ступінь, вчене звання)

(підпис)

(прізвище, ім'я, по батькові)

АНОТАЦІЯ

Дмитрук Д. В. Інформаційна технологія захисту від програм-вимагачів – Кваліфікаційна робота на правах рукопису.

Кваліфікаційна робота на здобуття освітнього ступеня бакалавра за спеціальністю 126 – Інформаційні системи та технології. – Поліський національний університет, Житомир, 2025.

У роботі розглянуто актуальну проблему зростання кількості атак програм-вимагачів. Було запропоновано підходи для розробки інформаційної технології, яка забезпечить відповідний рівень захисту від даного типу атак.

У першому розділі було проаналізовано поняття та класифікацію програм-вимагачів, їхній механізм дії, а також найбільш розповсюджені приклади шкідливого програмного забезпечення цього типу

Другий розділ присвячено архітектурі розробленої інформаційної технології захисту, яка включає кілька взаємодіючих компонентів, зокрема графічну програму, утиліту для сканування файлів за правилами YARA та фонову програму на C++ для моніторингу загроз. Розроблено UML-діаграми, контекстну модель, діаграми активності, сценарії використання та функціональну декомпозицію системи.

Реалізацію всіх компонентів інформаційної технології описано в третьому розділі. Наведено скріншоти інтерфейсу користувача, приклади коду, використані бібліотеки, фреймворки та сервіс API VirusTotal. Розроблено інструкцію для кінцевого користувача, що дозволяє легко налаштувати систему захисту відповідно до власних потреб

Ключові слова: програми-вимагачі, інформаційна безпека, захист даних, інформаційна технологія, кібератаки.

SUMMARY

Dmytruk D. V. Information Technology for Protection Against Ransomware – Qualification work on manuscript rights.

Qualification thesis for the degree of Bachelor in specialty 126 – Information Systems and Technologies. – Polissia National University, Zhytomyr, 2025.

This work addresses the urgent issue of the growing number of ransomware attacks. It proposes approaches for the development of an information technology system that provides an appropriate level of protection against this type of threat.

The first chapter analyzes the concept and classification of ransomware, their operational mechanisms, and the most widespread examples of this type of malicious software.

The second chapter is devoted to the architecture of the developed protection technology, which includes several interacting components: a graphical application, a file scanning utility using YARA rules, and a background C++ application for threat monitoring. UML diagrams, a context model, activity diagrams, usage scenarios, and the system's functional decomposition are provided.

The third chapter describes the implementation of all components of the information technology system. It includes user interface screenshots, code samples, libraries and frameworks used, and integration with the VirusTotal API. A user guide was developed to enable easy configuration of the system according to individual needs.

Keywords: ransomware, information security, data protection, information technology, cyberattacks.

Т

АНОТАЦІЯ.....	3
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ.....	7
ВСТУП.....	8
РОЗДІЛ 1 АНАЛІЗ ПРОБЛЕМАТИКИ РОЗРОБЛЕННЯ ТЕХНОЛОГІЇ ЗАХИСТУ ВІД ПРОГРАМ-ВИМАГАЧІВ.....	10
1.1 Поняття та класифікація ransomware.....	10
1.2 Аналіз існуючих платформ.....	14
1.3 Огляд шкідливого програмного забезпечення.....	16
1.4 Постановка завдання.....	17
Висновки до першого розділу.....	19
РОЗДІЛ 2 ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ДЛЯ ЗАХИСТУ ВІД ПРОГРАМ-ВИМАГАЧІВ.....	20
1.1 Розроблення архітектури інформаційної технології.....	20
1.2 Функціональне моделювання інформаційної технології.....	21
Висновки до другого розділу.....	23
РОЗДІЛ 3 РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ.....	24
1.1 Розроблення інтерфейсу користувача.....	24
3.2 Технічна реалізація інструментів для проактивного та сигнатурного захисту.....	26
3.3 Інструкція для користувача.....	30
Висновки до третього розділу.....	31
ВИСНОВКИ.....	32
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	33
ДОДАТКИ.....	36

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ПЗ – програмне забезпечення;

ІТ – інформаційні технології;

ОС – операційна система;

GUI – Graphical User Interface;

RaaS – Ransomware-as-a-Service;

RAT – Remote Access Trojan;

URL –Uniform Resource Locator;

TTP – Tactics, Techniques and Procedures;

UML – Unified Modeling Language;

API – Application Programming Interface;

GDPR – General Data Protection Regulation;

WinAPI – Windows API;

IDEF0 – Icam DEFinition for Function Modeling;

IDEF3 – Integrated DEFinition for Process Description Capture Method

ВСТУП

Актуальність теми дослідження. Сучасне суспільство дедалі більше залежить від інформаційних систем, що робить їх потенційними цілями для кіберзагроз, зокрема шкідливого програмного забезпечення. Програми-вимагачі (ransomware) шифрують дані або блокують доступ до системи з метою вимагання викупу [1]. Такі атаки спричиняють значні втрати: від зупинки критичних сервісів до витоку конфіденційної інформації й прямих фінансових збитків. Часто витрати на простої в кілька разів перевищують суму викупу, що стимулює зловмисників до вдосконалення тактик і активного поширення подібного ПЗ.

Згідно з дослідженням компанії SpaceLift, у 2023 році середня сума викупу становила 1,85 млн доларів, а вартість повного відновлення інфраструктури — 2,73 млн доларів; у 2024 ці показники зросли до 2,74 млн доларів [2]. Крім того, за даними Ponemon Institute, понад 88% організацій зазнали принаймні однієї ransomware-атаки [3]. Це зумовлює необхідність розробки проактивних, доступних та ефективних рішень, здатних адаптуватись до нових викликів у сфері кібербезпеки.

Мета кваліфікаційної роботи полягає у створенні інформаційної технології захисту від програм-вимагачів, яка поєднує сигнатурний, поведінковий і проактивний підходи, забезпечує моніторинг системних процесів і надає користувачеві візуальні підказки щодо зміцнення кіберзахисту.

Предметом дослідження є процес проєктування, розроблення та впровадження інформаційної технології, орієнтованої на підвищення рівня кіберстійкості в середовищі ОС Windows, включаючи підтримку застарілих систем (зокрема Windows XP і Server 2003).

Об'єктом дослідження є процес проектування та реалізації інформаційної технології, орієнтованої на підвищення рівня кібербезпеки ОС Windows, з урахуванням підтримки застарілих версій [4].

Наукова новизна роботи полягає в інтеграції кількох підходів до захисту (сигнатурного, поведінкового, проактивного) в межах єдиної технології, адаптованої до широкого діапазону операційних систем, що супроводжується функціональністю візуального контролю безпеки та інтеграцією з сервісом VirusTotal і механізмом локального сканування за YARA-правилами.

Практичне значення отриманих результатів полягає у створенні прикладної, гнучкої та доступної технології, яка може бути використана як у домашньому, так і в корпоративному середовищі. Розроблене рішення здатне ефективно працювати на обмежених ресурсах, забезпечує базовий рівень захисту та дає змогу користувачеві своєчасно виявляти й мінімізувати потенційні ризики.

Результати дослідження були апробовані на всеукраїнській та міжнародній конференціях:

Дмитрук Д.В. Дослідження методів виявлення та нейтралізації аномальної активності в інформаційних системах. *Цифрові технології в управлінні та адмініструванні*. 2025. С.88-90.

Дмитрук Д.В. Впровадження YARA-правил як інструменту блокування підозрілих файлів під час ransomware-атак. *The Future of Science: Emerging Research and Technological Innovations*. 2025. С.84-87.

РОЗДІЛ 1 АНАЛІЗ ПРОБЛЕМАТИКИ РОЗРОБЛЕННЯ ТЕХНОЛОГІЙ ЗАХИСТУ ВІД ПРОГРАМ-ВИМАГАЧІВ

1.1 Поняття та класифікація ransomware

На рисунку 1.1, наведеному в додатку А, на основі відкритих аналітичних даних Panda Security представлено найнебезпечніші типи кіберзагроз, які прогнозуються як ключові у 2025 році [5].

Згідно з даними Palo Alto Network [6], сучасні програми-вимагачі використовують складні методи шифрування, такі як AES-256, що ускладнює відновлення даних без відповідного ключа. Крім того, деякі з них застосовують тактику "подвійного вимагання", коли перед шифруванням дані викрадаються, і зловмисники погрожують їх оприлюдненням у разі несплати викупу.

Роблячи висновки зі статей CrowdStrike та Palo Alto Networks [7, 8], програми-вимагачі можна класифікувати за кількома основними ознаками. За способом блокування вони поділяються на шифрувальники (crypto-ransomware), які шифрують файли користувача з метою вимагання викупу за їх розшифрування, та блокувальники (locker-ransomware), що не шифрують файли, але блокують доступ до системи або інтерфейсу, позбавляючи користувача можливості працювати з комп'ютером.

З огляду на методи поширення, ransomware може проникати в систему через індивідуальний підхід — найчастіше за допомогою фішингових листів або шкідливих вебсайтів. Часто достатньо одного зараженого комп'ютера в корпоративній мережі, щоб атака розповсюдилася далі. Іншим поширеним способом є експлуатація відомих уразливостей у програмному забезпеченні, які дозволяють зловмисникам автоматично проникати в систему без участі користувача.

Також програми-вимагачі класифікують за рівнем таргетування. Масові атаки спрямовані на широку аудиторію без конкретної мети — зазвичай це автоматизовані кампанії, які розповсюджуються одночасно великою кількістю

каналів. У свою чергу, цільові атаки націлені на конкретні організації або осіб і часто супроводжуються ретельною розвідкою, складним проникненням та персоналізованим шифруванням даних для максимального тиску на жертву.

Шифрувальники (crypto-ransomware) — це найбільш поширений тип ransomware. Вони шифрують файли на пристрої користувача за допомогою криптографічних алгоритмів (наприклад AES, ChaCha20, RSA, poly1305) і вимагають викуп за ключ дешифрування.

З технічної точки зору, механізм роботи програм-вимагачів є таким: кожен файл у комп'ютерній системі шифрується за допомогою симетричного криптографічного алгоритму, найчастіше AES, ChaCha20 або Salsa20. Для кожного окремого файлу генерується унікальний симетричний ключ, який, у свою чергу, шифрується із застосуванням асиметричного алгоритму, такого як RSA або алгоритми на основі еліптичних кривих. Розширення зашифрованого файлу зазвичай змінюється на задане в програмі-вимагачі, а в самому файлі записані данні (як правило зашифрований ключ шифрування та розширення) для повернення в розшифрований вид. Також аби повідомити про викуп зазвичай в кожній директорії або лише на робочому столі створюється файл з інструкцією як отримати програму для дешифрування.

У наш час оператори програм-вимагачів дедалі частіше діють як «тіньові бізнеси», орієнтовані на максимізацію прибутку. Це проявляється у все більш «клієнтоорієнтованому» підході до вимагання викупу. В інструкції зазвичай вказані канали для зв'язку як-от електронна пошта, чати на веб-ресурсах «тіньового інтернету», Telegram-боти. Такий підхід створює ілюзію «надійності» угоди з кіберзлочинцями, що схиляє компанії до сплати викупу.

Крім того, все частіше застосовується стратегія «подвійного вимагання» (double extortion) [9]: оператори не лише шифрують файли, але й крадуть конфіденційні дані перед цим. Погроза оприлюднення або продажу викрадених даних слугує додатковим важелем тиску на потерпілу сторону. У ряді випадків зафіксовано публікацію конфіденційних корпоративних документів, внутрішнього листування та фінансової звітності на спеціалізованих веб-

ресурсах, так званих leak-sites, доступ до яких зазвичай здійснюється через мережу Tor.

Яскравим прикладом такої стратегії є група LockBit, яка на момент 2023 року була однією з найактивніших груп і опублікували дані понад 1700 компаній на власному leak-сайті, коли жертви відмовлялись платити викуп [10]. На рисунку 1.2 додатка А показана кількість жертв у 2023 році від найпоширеніших операторів.

Варто згадати також такі шифрувальники масового розповсюдження як WannaCry та Petya. Ці види шкідливого ПЗ в свою чергу полягались лише на вразливості EternalBlue [11] і за думкою експертів в кібербезпеці дана стратегія не принесла фінансового успіху їх творцям, хоча й кількість жертв була колосальною: один лише WannaCry зміг заразити понад 200 000 систем у 150 країнах [12]. На рисунку 1.3 додатка А зображено приклад зараження комп'ютера вірусом WannaCry.

Існують також такі вимагачі як блокувальними (locker-ransomware), вони блокують доступ до інтерфейсу користувача (GUI), не шифруючи файли. Найчастіше блокують екран або виводять фальшиві повідомлення від поліції або державних органів.

WinLock блокував робочий стіл Windows і показував фальшиві повідомлення про порушення закону, вимагав SMS-платежі для розблокування [13]. Він розповсюджувався іншим вірусом-вимагачем який імітував повідомлення про активацію Windows Product Activation. В 2010 році угруповання яке стояло за ним було затримано в Москві і загалом було зароблено більш ніж 16 мільйонів доларів на даному типі шкідливого ПЗ (див. рис. 1.4, Дод.А).

Police Ransomware відображав підроблені повідомлення від ФБР або поліції ЄС, звинувачуючи жертв у перегляді забороненого контенту. На рисунку з додатка К зображено приклад вікна з вимаганням викупу WinLock.

Вразливість EternalBlue свого часу стала ключовим чинником масового поширення шифрувальників. Її варіації, зокрема EternalRomance,

EternalSynergy та EternalChampion, експлуатували протокол SMB (Server Message Block) [14], що широко використовувався в критичній інфраструктурі. Хоча Microsoft оперативно усунула ці вразливості, важливість своєчасного оновлення ОС залишається надзвичайно актуальною.

Схожа більш нова вразливість SMBGhost [15] також надавала можливість атакувати мережу, але вже SMB версії 3. Вразливість була присутня в Windows 10 версії 1903 та 1909 і в Windows Server тих самих версій. Хоча практики масового зараження комп'ютерних мереж, як в атаках WannaCry, ця вразливість не набула, але вона й досі активно використовується на застарілих ОС операторами кібератак.

Найбільш актуальним методом зараження на сьогодні є цільові атаки (targeted attacks), що дедалі частіше використовуються операторами програм-вимагачів для проникнення в інфраструктуру організацій. Найпоширенішими варіаціями таких атак є фішингові листи та фішингові веб-ресурси.

Найбільш поширеним методом зараження сьогодні залишаються цільові атаки (targeted attacks), зокрема фішингові листи та веб-ресурси. Такі листи імітують офіційну комунікацію, змушуючи жертву відкрити шкідливе вкладення або перейти за посиланням, що веде до інфікування системи (через infostealer, RAT тощо) або до підробленого сайту для збору облікових даних.

Сучасні браузерери значно покращили захист від подібних завантажень, зокрема за допомогою технологій Safe Browsing, цифрових підписів і репутаційних фільтрів. Це призвело до зниження ефективності прямих шкідливих завантажень. У відповідь зловмисники почали використовувати більш витончені методи: наприклад, імітацію перевірки Captcha під час якої користувач несвідомо активує JavaScript-скрипт із вбудованим зловмисним кодом [16, 17]. Розповсюдження таких веб-ресурсів в наші дні в основному реалізується через вищезгадані фішингові листи, SEO-оптимізацію веб-сайтів, а також через зловмисні рекламні компанії. Такі підходи свідчать про постійне вдосконалення технік соціальної інженерії та гнучкість зловмисників у

боротьбі з сучасними методами захисту. На рисунку 1.5 додатка А зображено приклад атаки ClickFix за допомогою імітації перевірки Captcha.

1.2 Аналіз існуючих платформ

На ринку ІТ в галузі забезпечення кібербезпеки існує низка комерційних та відкритих рішень для боротьби з програмами-вимагачами. До найбільш відомих належать антивірусні продукти, які використовують евристичний аналіз та поведінкові алгоритми, інструменти резервного копіювання і сервіси багатомодульного аналізу файлів.

Розглянемо такі антивірусні продукти як Microsoft Windows Defender (Windows Security) та Bitdefender Total Security.

Microsoft Windows Defender є одним із найпоширеніших антивірусів завдяки вбудованості в ОС Windows. Він забезпечує захист у реальному часі, контроль доступу до папок, виявлення експлойтів, інтеграцію з SmartScreen тощо. До його переваг належать безкоштовність і невелике споживання ресурсів, однак функціональність щодо аналізу інцидентів обмежена без корпоративної EDR-версії.

Bitdefender Total Security демонструє високу ефективність у боротьбі з ransomware-загрозами. Цей антивірус має модуль Ransomware Remediation, який автоматично створює резервні копії критичних файлів перед потенційною атакою. Також присутні функції поведінкового аналізу, захисту від мережесих атак, а також інтеграція із sandbox-оточенням для перевірки підозрілих файлів. До основних переваг можна віднести потужну багаторівневу фільтрацію загроз, ефективність захисту та автоматичне реагування на інциденти. Недоліком є дещо складне налаштування в корпоративному середовищі та потреба в постійному оновленні системних компонентів для підтримки актуальності захисту.

Одним з ефективних рішень для резервного копіювання є Acronis Cyber Protect. Це багатофункціональний інструмент, який поєднує в собі резервне копіювання, кіберзахист та управління системами. Серед основних функцій можна відзначити створення образів дисків, резервне копіювання на локальні носії, хмару або мережеві ресурси, автоматичне створення резервних копій за розкладом, перевірку резервних копій на цілісність та можливість швидкого відновлення навіть на інше обладнання.

Серед рішень для резервного копіювання доцільно виділити Acronis Cyber Protect, який об'єднує резервне копіювання, кіберзахист і централізоване управління. Програма підтримує різні ОС, зокрема Windows, Linux і віртуальні середовища, а також включає модуль виявлення ransomware у режимі реального часу. Основним недоліком є потреба в ресурсах і висока вартість для індивідуального використання..

VirusTotal є одним із найвідоміших і широко використовуваних сервісів для багатомодульного сканування файлів та URL-адрес. Цей інструмент дозволяє користувачам безкоштовно перевіряти підозрілі файли чи веб-ресурси за допомогою десятків антивірусних рушіїв та систем виявлення загроз. Після завантаження файлу або вставлення посилання, VirusTotal запускає одночасне сканування за допомогою більш ніж 70 антивірусів. На рисунку 1.6 додатка А зображено результати сканування програми-шифрувальника Babyk на VirusTotal.

Проте більшість існуючих рішень або вимагають значних ресурсів системи, або не сумісні зі старими версіями ОС, такими як Windows XP чи Server 2003. Крім того, багато утиліт мають вузький функціонал і не забезпечують комплексного підходу, зокрема не поєднують моніторинг системи, перевірку наявності ознак шкідливого ПЗ та рекомендації для покращення захисту.

1.3 Огляд шкідливого програмного забезпечення

LockBit - це один із найактивніших представників сучасного ransomware-as-a-service (RaaS), який постійно модифікується. LockBit 3.0 ("LockBit Black") підтримує функції видалення тіньових копій, обходу антивірусного ПЗ, шифрування локальних та мережевих ресурсів, та вивантаження даних перед шифруванням [18, 19]. Оператори активно використовують подвійне вимагання — і шифрування даних, і погрози оприлюднити їх на спеціальних leak-сайтах.

На своєму веб-сайті група LockBit заявляє що має найшвидшу швидкість шифрування серед списку інших шифрувальників на ринку. На рисунку 1.7 додатка А зображено таблицю порівняння швидкості шифрувальників на сайті LockBit.

Розпочавши виконання шифрувальник LockBit видаляє важливі записи та резервні копіювання з пристрою аби унеможливити відновлення даних з боку жертв.

Команда «vssadmin delete shadows /all /quiet» - видаляє всі тіньові копії з системи.

Команда «bcdedit /set {default} recoveryenabled no» - не дає можливості користувачеві увійти до recovery boot ОС Windows 10.

LockBit генерує SID і використовує його для додавання себе до списку автозавантаження за допомогою запису реєстру в «HKCU\Software\Microsoft\Windows\CurrentVersion\Run». На рисунку 1.8 додатка А зображено створення запису реєстру LockBit.

Також дане ПЗ асоціює зображення кожного зашифрованого файлу з власним, має можливість перечислення кожного з хостів мережі за допомогою протоколу SMB.

Babuk (або Babyk) — це зразок програми-вимагача, що з'явився на початку 2021 року та був націлений переважно на корпоративні мережі. Babuk використовував модель Ransomware-as-a-Service (RaaS), де оператори платформи надавали її афілійованим учасникам, які здійснювали атаки. Відомий здебільшого завдяки використанню сильного шифрування (Elliptic Curve Cryptography), цей зразок був здатен обходити звичайні методи захисту та швидко поширювався в локальних мережах. У травні 2021 року сталося внутрішнє розділення в угрупованні, після чого вихідний код Babuk був опублікований у відкритому доступі на форумі xss.is [20]. На рисунку 1.9 додатка А зображено оригінальний пост на форумі xss.is від автора вимагача.

Цей витік став значною подією, адже код пізніше був використаний іншими кіберзлочинцями для створення похідних версій. Babuk також є одним з перших, хто активно атакував VMware ESXi-сервери, що стало новим вектором у сфері корпоративних атак. В додатку А візуально розглянуто код шифрувальника (див. рис.1.10, Дод.А) та наведено фрагмент коду C++ відповідального за використання еліптичної кривої в ransomware (див. рис.1.11, Дод.А).

Ці приклади демонструють різноманітність тактик, технік та процедур (TTP) програм-вимагачів. Сучасні зразки часто поєднують функціонал бекдорів, інфостілерів, експлуатаційне ПЗ для вразливостей та компонентів для автономного поширення. Вони здатні обходити традиційний захист і вимагають більш складних проактивних та поведінкових методів виявлення і реагування. Саме це зумовлює необхідність створення нових захисних технологій, адаптованих до змін в кіберпросторі.

1.4 Постановка завдання

В приведених прикладах вище було розглянуто велику кількість кібератак із застосуванням програм-вимагачів, які завдають значної шкоди як

приватним користувачам, так і організаціям, виникає об'єктивна потреба у створенні ефективної, доступної та проактивної інформаційної технології захисту. Більшість наявних рішень зосереджуються на реактивному виявленні загроз після проникнення системи, в той час як значна частина шкідливого ПЗ встигає заподіяти шкоду ще до моменту спрацювання класичних сигнатурних захистів. Актуальність розробки полягає також у тому, що велика частина існуючих захисних рішень має обмеження: високе споживання ресурсів, недоступність для старіших версій Windows, недостатня гнучкість або надмірна складність у використанні для пересічного користувача.

У межах даного проекту необхідно вирішити низку завдань, серед яких — аналіз існуючих рішень у сфері протидії програмам-вимагачам з метою виявлення їхніх переваг та недоліків. Наступним етапом є формування архітектури нової інформаційної технології з обов'язковим урахуванням підтримки операційних систем Windows, включаючи застарілі версії, такі як Windows XP та Windows Server 2003. Передбачається реалізація десктопного застосунку на базі WinForms (.NET Framework 4.5), який забезпечуватиме візуалізацію рівня захисту системи та керування параметрами безпеки. Також необхідно розробити Python-утиліту для вибіркового сканування файлів за YARA-правилами, що дозволить виявляти потенційно шкідливі об'єкти за характерними шаблонами. Одним з ключових компонентів є низькорівневий агент безпеки, створений на мові C++, який працює у фоновому режимі, здійснює моніторинг директорії завантажень, перевіряє процеси на підозрілу активність, взаємодіє з API VirusTotal та, за необхідності, автоматично запускається при старті системи згідно з параметрами конфігураційного файлу settings.json. Крім того, важливо забезпечити простоту налаштування системи для кінцевого користувача та провести тестування працездатності розробленого рішення в експериментальному середовищі з подальшим аналізом його ефективності.

Таким чином, кінцевий продукт має на меті надати користувачеві інструмент не лише для виявлення та нейтралізації загроз програм-вимагачів, але й для проактивної оцінки кібергігієни системи, що дозволить знизити ймовірність успішного зараження.

Висновки до першого розділу

У першому розділі проаналізовано основні аспекти проблеми захисту від програм-вимагачів: класифікацію, механізми дії, методи поширення та сучасні тактики зловмисників, зокрема «подвійне вимагання». Розглянуто приклади відомих загроз (LockBit, Babuk, WannaCry тощо) та ключові вразливості (EternalBlue, SMBGhost), що сприяли масштабним атакам.

Також проведено огляд сучасних захисних рішень — антивірусів, резервного копіювання та сервісів аналізу файлів. Встановлено їх обмеження щодо ресурсомісткості, підтримки старих ОС та комплексності захисту.

На підставі цього сформульовано завдання створити адаптивну інформаційну технологію, здатну поєднувати кілька підходів до виявлення загроз і забезпечити практичну користь для кінцевого користувача.

РОЗДІЛ 2 ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ДЛЯ ЗАХИСТУ ВІД ПРОГРАМ-ВИМАГАЧІВ

1.1 Розроблення архітектури інформаційної технології

Розроблена інформаційна технологія захисту від програм-вимагачів складається з кількох взаємопов'язаних модулів, які виконують специфічні функції. Архітектура системи реалізована на основі принципів модульності, масштабованості та сумісності із старими версіями Windows, що забезпечується використанням .NET Framework 4.5 та C++. Архітектура містить три основні частини:

- Інтерфейс користувача — реалізований на C# WinForm. Відображає стан захисту системи, надає рекомендації, дозволяє змінювати параметри захисту та вибірково сканувати за допомогою python-утиліти.
- Фоновий агент — реалізований мовою C++. Здійснює проактивний захист: моніторинг процесів, сканування файлів, комунікацію з VirusTotal API.

На основі створеної UML-діаграми описано такі основні сценарії використання:

- Перегляд стану безпеки та рекомендацій;
- Керування конфігураціями;
- Схвалення або блокування підозрілих процесів;
- Перевірка стану системи та активних загроз;
- Моніторинг завантажених файлів і взаємодія з VirusTotal API.

На рисунку 1.1 додатка Б міститься зображення UML-діаграми розробленої ІС.

Актори System та VirusTotal API взаємодіють із системою автоматично. Основна логіка контролюється користувачем, але частина функцій виконується автономно у фоновому режимі.

Умовно, робота системи розбивається на такі активності:

- Початкове зчитування конфігурацій;
- Перевірка стану системи (антивірус, тіньові копії, оновлення тощо);
- Паралельний моніторинг процесів і файлів;
- Обробка інцидентів та зворотній зв'язок із користувачем.

Станова модель включає основні стани: Очікування, Моніторинг, Попередження, Реакція, Завершення.

1.2 Функціональне моделювання інформаційної технології

Функціональне моделювання є одним із ключових етапів проєктування інформаційної технології, оскільки дозволяє чітко визначити функціональні блоки, сценарії їх взаємодії та процеси, що забезпечують досягнення цілей системи. У цьому розділі представлено пояснення до контекстної діаграми, декомпозиції IDEF0, IDEF3, а також сценарії використання системи (діаграми UML типу Use Case).

На схемі наведеній в рисунку 1.2 додатка Б було змодельовано типову послідовність дій системи під час запуску, моніторингу, реакції на загрози та формування рекомендацій для користувача. IDEF3 дозволяє представити поведінку системи у часі, враховуючи паралельні й послідовні процеси, а також взаємозв'язки між модулями.

Згідно з діаграмою IDEF3, основна логіка функціонування системи включає кілька послідовних етапів. На початковому етапі запуск системи супроводжується зчитуванням базової конфігурації (2), після чого виконується

ініціалізація основних модулів (3) та відображення графічного інтерфейсу користувача (4).

Далі реалізується механізм постійного моніторингу системи, який передбачає перевірку появи нових файлів (7) та запуску нових процесів (5), а також отримання атрибутів активних процесів (6) із подальшим порівнянням результатів із сервісом VirusTotal (9). Окремо здійснюється перевірка наявності активного антивірусного захисту (16), актуальності оновлень (15), стану резервного копіювання (18), функціонування шифрування дисків (17) та загального стану системи (14).

У разі виявлення потенційної загрози система може реагувати відповідно: дозволяється додавання підозрілих об'єктів до білого списку (11), видалення шкідливих компонентів (12), а також фіксація всіх виконаних дій у логах (13). Завершальним етапом є формування рекомендацій, яке базується на оцінці поточного стану системи та виявлених ризиків (19).

Для опису функціональної структури інформаційної технології захисту від програм-вимагачів була побудована контекстна діаграма IDEF0. Основна функція верхнього рівня — це «Захист системи від ransomware-атак». На цій діаграмі зазначено, що до системи надходить низка вхідних даних: поточний стан системи, результати попередніх перевірок, файли (зокрема потенційно шкідливі виконувані файли), а також конфігураційні параметри, що задаються користувачем у файлі «settings.json».

Забезпечення виконання основної функції відбувається за участі кількох ключових механізмів. Це, зокрема, агент захисту (UAgent), модуль перевірки процесів із реалізацією хукінгу, модуль рекомендацій, графічний інтерфейс користувача, API сервісу VirusTotal, а також вбудовані механізми захисту, що надаються операційною системою Windows. Суттєвий вплив на логіку роботи системи мають політики безпеки, які діють на рівні системи.

Робота системи обмежується чинними стандартами і політиками. Серед них — вимоги GDPR, стандарти ISO 27001, встановлені правила моніторингу, технічні обмеження самої системи, стратегія резервного копіювання та політика реагування на загрози. Всі ці фактори визначають межі та напрямки функціонування інформаційної технології.

В результаті виконання функцій система наведена в рисунку 1.3 додатка Б генерує вихідні дані, до яких належать попередження про виявлені загрози, рекомендації з підвищення рівня захисту, звіти про виконані перевірки, лог-файли з записами подій, оновлені налаштування безпеки, а також зібрані або змінені дані з системного реєстру.

Висновки до другого розділу

Другий розділ присвячено проектуванню інформаційної технології захисту від програм-вимагачів, що поєднує сигнатурний, поведінковий і хмарний аналіз загроз. Архітектура системи враховує сумісність зі старими версіями Windows, модульність та зручність для користувача.

За допомогою UML-моделей сформовано структурну модель взаємодії між інтерфейсом, службами моніторингу та зовнішніми сервісами (VirusTotal API), що формалізує сценарії використання та забезпечує узгоджену роботу системи в реальному часі.

Функціональне моделювання за IDEF0 і IDEF3 дозволило детально проаналізувати логіку процесів, обробку подій, обмін даними і взаємодію з користувачем. Побудовано контекстну діаграму та декомпозицію ключових процесів: перевірка стану, сканування, генерація рекомендацій і реагування на загрози.

Таким чином, на основі розробленої архітектури сформовано комплексну інформаційну технологію, яка здатна виявляти шкідливу активність, оцінювати

рівень безпеки середовища та надавати кінцевому користувачеві релевантні поради для запобігання зараженню системи програмами-вимагачами. Отримані результати закладають основу для реалізації прототипу системи, що буде представлено у наступному розділі. Враховуючи методи виявлення аномальної активності в ІС було також написано публікацію [21].

РОЗДІЛ 3 РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ

1.1 Розроблення інтерфейсу користувача

При розробці інтерфейсу ключову увагу було приділено головному вікну звідки і відбувається керування іншими компонентами інформаційної системи. На рисунку 3.1 додатка В зображено головне вікно USecure в IDE Visual Studio.

При натисканні на підкомпоненти «Поради», «Проактивний захист», «Просканувати файл» та «Про ПЗ» компоненту Windows Forms «ToolStripMenuItem» відкриваються нові вікна з відповідним функціоналом. Відбувається це за допомогою стандартного методу ініціалізації об'єкту форми та виконання методу Show() ініціалізованого об'єкту. На рисунку 3.2 додатка В зображено реалізацію відкриття нових вікон в USecure.

При завантаженні форми (метод Form2_Load()) спочатку виконується перевірка на предмет впровадження порад в систему, а потім перевірка на те скільки компонентів з налаштувань проактивного захисту було активовано. На основі цих даних формується висновок в відсотках. Важливе зауваження: ніяка система не може бути на 100% захищена, тому висновок про захищеність системи, який формується в змінній SystemProtectionPercentage завжди буде менший за 100. На рисунку 3.3 додатка В зображено частину коду перевірки порад та налаштувань проактивного захисту в методі Form2_Load(). А на рисунку 3.4 додатка В зображено частину коду відповідального за формування висновку і присвоювання його в елементи форми label2 та progressBar1.

Також для дизайну вікна було обрано відповідне зображення Icon, назву вікна обрано як «USecure». Для отримання фіксованого по розмірам, розташованого в центрі вікна було обрано відповідні параметри «MinimizeBox», «MaximizeBox», «FormBorderStyle» та «StartPosition» (для центрування вікна на екрані). Надалі відповідні параметри будуть використані

в інших вікнах з точки зору їх оптимальності для користувацького досвіду (див. рис. 3.5, Дод.В).

З суб'єктивної точки зору було створено також вікно з етичною порадою (див. рис. 3.6, Дод.В). Дане вікно можливо вимкнути натиснувши на `checkBox1`. При вході в інформаційну систему перевіряється відповідний змінений параметр в `settins.json` («helpzsu»).

При натисканні на кнопку `button1` («Допомогти ЗСУ») викликається метод `button1_Click()`, що відкриває в браузері веб-сайт «<https://savelife.in.ua/>» за допомогою методу `Process.Start()`.

При натисканні на кнопку `button2` («Продовжити») відкривається головне вікно `Form2` раніше описаним методом з ініціалізацією об'єкта `Form2` і при активності `checkBox1` записує параметр «helpzsu» до файлу налаштувань.

Форма порад складається з елементу `dataGridView1` та `label1`, `label2`, `label3`.

При ініціалізації елементу `dataGridView1` (на більш слабких пристроях можливо необхідно буде дочекатись кінця ініціалізації) за допомогою методів `IsLatestUpdatesInstalled()`, `IsAntivirusInstalled()`, `IsFirewallInstalled()`, `IsBitLockerUsed()` висновки до відповідних порад щодо встановлення останніх оновлень, встановлення антивірусу, встановлення фаєрволу, використання `BitLocker` («Виконано» або «Не виконано»). `GetLastBackupShadowCopy()` в свою чергу відповідає за пораду на рахунок зроблених тінювих копій: виводить інформацію щодо останньої тінювої копії або виводить повідомлення «Тінювих копій не знайдено» якщо ПЗ їх не вдалось знайти (див. рис. 3.7, Дод.В). В свою чергу `label1`, `label2`, `label3` дають незалежні текстові поради, які не перевіряються та фіксуються програмним методом. На рисунку 3.8 додатка В зображено запущене вікно порад.

Вікно проактивного захисту складається з заголовку (`label1`), декількох `checkbox` та кнопки `button1` («Примініти налаштування») відповідальних за редагування налаштувань даного типу захисту (див. рис. 3.9, Дод.В).

При завантаженні цієї форми (метод `ProactiveProtectionForm_Load()`) перевіряється наявність файлу налаштувань і при його наявності відповідні налаштування відображаються як активні чи неактивні відміткою в відповідальному за пункт налаштувань `checkbox` (див. рис. 3.10, Дод.В).

При натисканні на `button1` («Примінити налаштування») відповідні налаштування зчитуються з `checkbox` і записуються в файл налаштувань.

Вікно вибіркової перевірки файлів було додано на більш пізніх етапах розробки як інструмент для гнучкішого сканування виконуваних файлів. Воно складається з кнопки для обрання файлу (`button1`), відображення шляху до обраного файлу (`label1`), кнопки для початку виконання перевірки (`button3`), `label2` для заголовку «Результат», `richTextBox1` для відображення результатів сканування `python`-утилітою та кнопки (`button2`) для копіювання результату до буферу обміну. (див. рис. 3.11, Дод.В)

При натисканні на `button1` відкривається діалогове вікно для вибору виконуваного файлу і після обрання текст `label1` змінюється на шлях (див. рис. 3.12, Дод.В).

При натисканні на `button3` за допомогою об'єкту `Process` виконується `python`-скрипт `pyaradetector` і вивід з даного інструменту записується в `richTextBox1`.

Вікно «Про ПЗ» в свою чергу складається лише з тексту з інформацією про програмне забезпечення (див. рис. 3.13, Дод.В).

3.2 Технічна реалізація інструментів для проактивного та сигнатурного захисту

Модуль `UAgent` відповідає за проактивний та сигнатурний захист інформаційної системи, він не має графічного інтерфейсу і працює в фоновому режимі. Написаний на `C++` та складається з виконуваного файлу `UAgent.exe` та динамічних бібліотек які знаходяться на стадії РОС (`proof-of-concept`): `agent32.dll`, `agent64.dll`.

На початку за допомогою класу `fstream` бібліотеки `std` зчитується зміст файлу налаштувань `settings.json` (див. рис. 3.14, Дод.В). Потім за допомогою класу `regex` бібліотеки `std` виконується пошук параметрів і в глобальні змінні заносяться дані про параметри аби згодом вжити необхідних користувачеві дій (див. рис. 3.15, Дод.В).

Якщо активний параметр `«agent_autostart»`, то виконується функція `AddSelfToAutostart`. В ній отримується повний шлях виконуваного модуля `UAgent.exe` за допомогою WinAPI функції `GetModuleFileNameW` [22] та створюється ключ `«UAgent»` зі значенням повного шляху в реєстрі зі шляхом `«HKCU\ Software\Microsoft\Windows\CurrentVersion\Run»` за допомогою функцій `RegOpenKeyExW` (для отримання дескриптора ключа), `RegSetValueExW` (для створення ключа зі значенням) та `RegCloseKey` (для закриття дескриптора ключа). На рисунку 3.16 додатка В зображено код функції `AddSelfToStart`.

При активності параметру `«virustotal_downloads_send»` створюється окремий потік для функції `VTCheckDownloads` (перевірка завантажених файлів на сервісі `VirusTotal`). Якщо ще не було проведено ін'єкції динамічної бібліотеки `UAgent`, то вона проводиться.

У функції `VTCheckDownloads` ми отримуємо шлях до директорії завантажень, а потім за допомогою `FindFirstFile` / `FindNextFile` / `FindClose` [23] виконуємо ітерацію по файлам та директоріям. Кожен файл відправляється через API `VirusTotal` за допомогою бібліотеки `curl` [24], при цьому до полів форми `HTTP POST`-запиту додається поле для API ключа користувача (параметр `«apikey»`) та поле для самого файлу (параметр `«file»`).

Якщо встановлений параметр `«agent_background_check»` (перевірка підозрілих файлів та процесів в фоновому режимі), то виконується ін'єкція динамічної бібліотеки, оскільки даний функціонал не залежить від виконуваного файлу `UAgent`.

Параметр `«agent_processes_access_check»` відповідає за перевірку підозрілих процесів у системі. Потокова функція `SuspiciousProcessesCheck`

нескінченно кожні пів секунди (500 мілісекунд) виконує ітерацію по процесам і перевіряє їх відкриті дескриптори за допомогою іншої функції – LoopForProcesses (див. рис. 3.17, Дод.В).

Всередині функції LoopForProcesses виконується отримання snapshot за допомогою WinAPI функції CreateToolhelp32Snapshot [25] процесів на момент виконання коду і його ітерація за допомогою функцій Process32First та Process32Next [26, 27]. При цьому процеси які раніше проходили перевірку або мають назву зі списку дозволених пропускаються, в інакшому випадку – створюється окремий потік для сканування процесу задля збільшення швидкості сканування. Код даного потоку знаходиться в функції CheckOpenedFilesOfProcess. Всередині цієї функції за допомогою NtQuerySystemInformation отримується загальний список дескрипторів і якщо дескриптор належить до обраного для сканування процесу дублікується його дескриптор та отримується шлях до відкритого файлу аби згодом перевірити чи не містить він слів із списку зазначеного в коді SuspiciousProcessesCheck.cpp (змінна bannedSubStrings): якщо містить, то користувачу запропонується усунути загрозу (див. рис. 3.18, Дод.В).

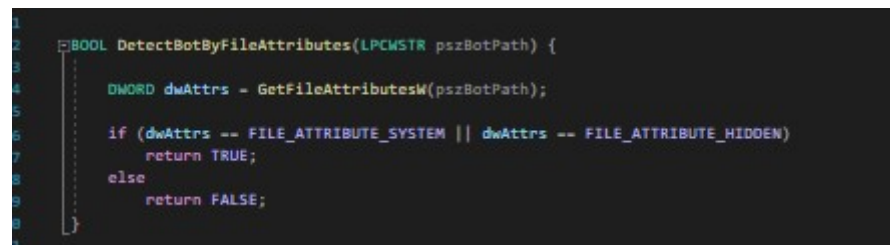
Загалом принцип роботи чітко розглядається в точці входу WinMain в програмі UAgent зображено на рисунку 3.19 додатка В.

Окремої уваги вартує реалізація динамічної бібліотеки UAgent (agent32.dll та agent64.dll). Існує можливість побудувати 32 та 64 бітні версії цього елементу ПЗ.

На початку роботи проходить вищеописане отримання параметрів, хоча в даному випадку отримуються лише два параметри необхідні для роботи ПЗ: «agent_background_check» та «virustotal_downloads_send». В відповідності до їх активності створюються два потоки за які відповідають функції BackgroundRoutine для параметру «agent_background_check» та HookingRoutine для «virustotal_downloads_send» відповідно (див. рис. 3.20 Дод.В).

Всередині функції потоку BackgroundRoutine в ще одному окремому потоці (в планах додати більше механізмів в кожному окремому потоці)

викликається один з механізмів протидії malware віддаленого доступу функцією BotKillerThread. В ній вищезгаданим методом ітерації процесів перевіряється чи не має виконуваний файл процесу підозрілий шлях (а саме CSIDL_APPDATA) і чи при цьому процес не має назву системного файлу (наприклад dllhost.exe чи explorer.exe) або чи не має при цьому виконуваний файл процесу підозрілі параметри (системний чи прихований). Якщо ж процес не проходить перевірку на підозрілість то користувачеві ІС пропонується усунути загрозу. На рисунку 3.21 додатка В зображено код функції перевірки назви процесу.



```

1  2  3  4  5  6  7  8  9
2  BOOL DetectBotByFileAttributes(LPCWSTR pszBotPath) {
3      DWORD dwAttrs = GetFileAttributesW(pszBotPath);
4      if (dwAttrs && FILE_ATTRIBUTE_SYSTEM || dwAttrs && FILE_ATTRIBUTE_HIDDEN)
5          return TRUE;
6      else
7          return FALSE;
8  }
9

```

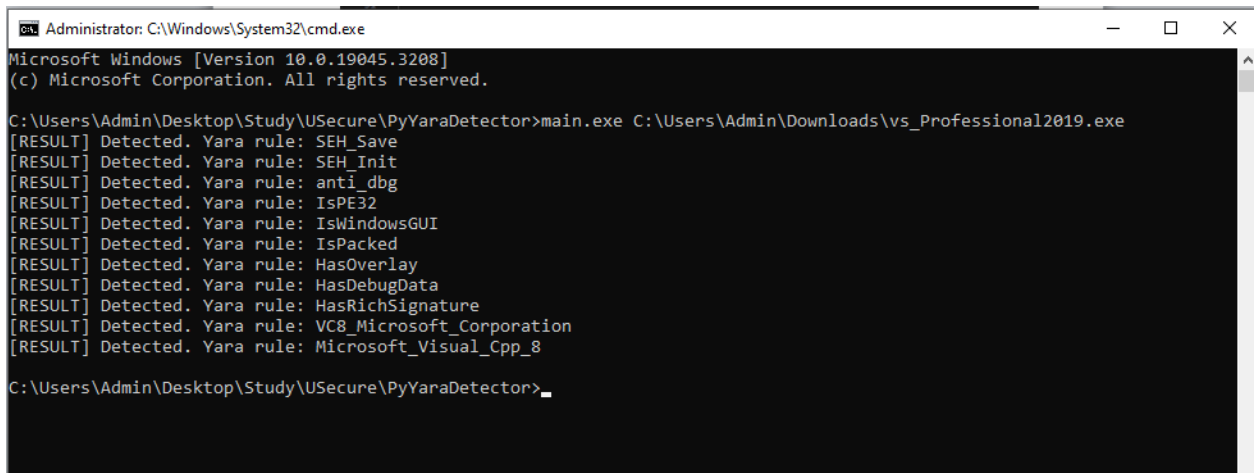
Рисунок 3.22 - Код функції перевірки атрибуту виконуваного файлу процесу за допомогою WinAPI функції GetFileAttributesW

Оскільки динамічна бібліотека знаходиться на стадії РОС з погляду на великий об'єм розробки комплексного продукту ІС, то функція відповідальна за сканування сервісом VirusTotal (HookingRoutine), раніше реалізована в виконуваний програмі UAgent, планується бути реалізована в наступних релізах дещо видозміненим чином і в динамічній бібліотеці. Планується перехоплення функції створення процесів CreateProcess і аналіз кожного виконуваного файлу за допомогою сервіса VirusTotal..

Одним з найновіших компонентів на момент написання став python-скрипт для вибіркового сканування за правилами YARA. На цю тему було написано публікацію в конференції The Future of Science: Emerging Research and Technological Innovations [28].

Отримує як аргумент шлях до виконуваного файлу і у відповідь виводить результат сканування обраними правилами YARA. При цьому правила рекурсивно шукаються у обраних директоріях зі списку rule_dirs, що дозволяє

додавати власні правила, і скануються за допомогою бібліотеки yara [29]. В додатку Д міститься повний код описаного скрипта.



```
Administrator: C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19045.3208]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin\Desktop\Study\USecure\PyYaraDetector>main.exe C:\Users\Admin\Downloads\vs_Professional2019.exe
[RESULT] Detected. Yara rule: SEH_Save
[RESULT] Detected. Yara rule: SEH_Init
[RESULT] Detected. Yara rule: anti_dbg
[RESULT] Detected. Yara rule: IsPE32
[RESULT] Detected. Yara rule: IsWindowsGUI
[RESULT] Detected. Yara rule: IsPacked
[RESULT] Detected. Yara rule: HasOverlay
[RESULT] Detected. Yara rule: HasDebugData
[RESULT] Detected. Yara rule: HasRichSignature
[RESULT] Detected. Yara rule: VC8_Microsoft_Corporation
[RESULT] Detected. Yara rule: Microsoft_Visual_Cpp_8

C:\Users\Admin\Desktop\Study\USecure\PyYaraDetector>_
```

Рисунок.3.23 - Результат роботи скрипта

3.3 Інструкція для користувача

Розроблена інформаційна технологія захисту від програм-вимагачів є досить простою та гнучкою у використанні, хоча для максимальної ясності бажано ознайомитись з інструкцією для користувача.

Після встановлення шляхом розархівування архіву останнього релізу необхідно розпочати з запуску WinForms-застосунку USecure.exe.

На головному екрані користувач може ознайомитись з прогрес-баром рівня безпеки, який відображає умовну оцінку стану системи (в процентах) та обрати в стрічці меню зверху вікно з відповідним функціоналом.

При необхідності перевірити стан захисту ОС і вжити заходів відповідно до сформованих порад необхідно натиснути на «Поради» в верхній стрічці меню і в відкритому вікні будуть відображені поради та їх статуси.

Якщо існують сумніви щодо певного виконуваного файлу і існує необхідність його просканувати необхідно натиснути відповідно на

«Просканувати файл» і у відкритому вікні обрати файл для сканування та натиснути на кнопку «Перевірити»; за необхідності можна скопіювати результат сканування до буферу обміну за допомогою кнопки «Копіювати».

Аби редагувати активні налаштування проактивного захисту, які при старті будуть відключені необхідно натиснути на «Проактивний захист» у стрічці меню зверху головного вікна і відповідно до потреб активувати налаштування в checkbox та підтвердити натисканням кнопки «Примінити налаштування». Після цього, при необхідності тогочасної роботи модуля проактивного захисту UAgent, необхідно запустити виконуваний файл UAgent.exe. Модуль можна запустити і пізніше якщо нагальної необхідності не існує і додати його до запуску при старті системи (checkbox «Додання USecure Agent в автозавантаження»).

Аби ознайомитись з короткою інформацією щодо розробленої системи необхідно натиснути на пункт стрічкового меню «Про ПЗ» в головному вікні і у відкритому вікні буде представлена інформація.

Висновки до третього розділу

У третьому розділі було реалізовано функціональну інформаційну технологію захисту від програм-вимагачів, яка включає графічний застосунок USecure (на базі WinForms .NET Framework 4.5), Python-утиліту для сканування файлів за YARA-правилами, фоновий агент UAgent на C++ для моніторингу директорій, перевірки процесів і взаємодії з VirusTotal, а також систему конфігурації через файл settings.json. Додатково розроблено інструкцію користувача, що забезпечує простоту експлуатації. Система забезпечує сигнатурний і проактивний захист, візуалізацію рівня безпеки, а також підтримує застарілі ОС, що підвищує її універсальність і практичну цінність.

ВИСНОВКИ

У цій кваліфікаційній роботі було розроблено інформаційну технологію захисту від програм-вимагачів, яка поєднує сучасні підходи до кібербезпеки з практичними вимогами щодо зручності, гнучкості та сумісності з різними версіями операційних систем, зокрема застарілими.

Перший розділ присвячено аналізу загроз ransomware: охарактеризовано їхню класифікацію, методи дії та вектори поширення, а також наведено приклади реальних інцидентів. Встановлено, що наявні засоби захисту мають обмеження, які знижують їхню ефективність у нових умовах.

У другому розділі здійснено проєктування архітектури інформаційної технології. За допомогою UML-діаграм, моделей IDEF0 та IDEF3 визначено взаємодію між графічним інтерфейсом, фоновими модулями та зовнішніми сервісами, такими як VirusTotal.

У третьому розділі було реалізовано кожен із модулів проєкту, а саме десктопна WinForms-програма USecure, фоновий агент на C++ (UAgent), Python-утиліта та гнучкий конфігураційний механізм через settings.json.

Крім того, було створено інструкцію користувача, яка дає змогу експлуатувати систему без необхідності глибоких технічних знань.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Програма-вимагач - що таке вірус-вимагач, як його видалити *ESET* : веб-сайт. URL: <https://www.eset.com/ua/support/information/entsyklopediya-zahroz/prohrama-vymahach>
2. 50+ Ransomware Statistics for 2025 *Spacelift* : веб-сайт. URL: <https://spacelift.io/blog/ransomware-statistics>
3. Study Reveals 88% of Companies Experienced a Ransomware Attack Last Year *HIPAA Journal* : веб-сайт. URL: <https://www.hipaajournal.com/cost-of-a-ransomare-attack-study-2024/>
4. DanyloDmytruk/USecure *GitHub* : веб-сайт. URL: <https://github.com/DanyloDmytruk/USecure>
5. 9 Key Cybersecurity Trends to Know in 2025 *Panda Security Mediacenter* : веб-сайт. URL: <https://www.pandasecurity.com/en/mediacenter/cybersecurity-trends/>
6. What Is Ransomware? *Palo Alto Networks* : веб-сайт. URL: <https://www.paloaltonetworks.com/cyberpedia/what-is-ransomware>
7. 5 Most Common Types of Ransomware *CrowdStrike* : веб-сайт. URL: <https://www.crowdstrike.com/en-us/cybersecurity-101/ransomware/types-of-ransomware/>
8. What Are the Most Common Types of Ransomware? *Palo Alto Networks* : веб-сайт. URL: <https://www.paloaltonetworks.com/cyberpedia/what-are-the-most-common-types-of-ransomware>
9. What Is Double Extortion Ransomware? *Zscaler* : веб-сайт. URL: <https://www.zscaler.com/resources/security-terms-glossary/what-is-double-extortion-ransomware>
10. Understanding Ransomware Threat Actors: LockBit *CISA* : веб-сайт. URL: <https://www.cisa.gov/news-events/cybersecurity-advisories/aa23-165a>

11. Експлойт EternalBlue атакує з новими силами - дослідження ESET *ESET* : веб-сайт. URL: <https://www.eset.com/ua/about/newsroom/press-releases/malware/uyazvimost-v-sluzhbach-udalennogo-rabochego-stola-novaya-opasnost-dlya-organizatsiy-wannacryptor/>
12. Вірус WannaCry заблокував 200 тис. комп'ютерів у 150 країнах *Портал новин LB.ua* : веб-сайт. URL: https://lb.ua/world/2017/05/14/366220_virus_wannacry_zablokiroval_200_tis.html
13. WinLock Ransomware *KnowBe4* : веб-сайт. URL: <https://www.knowbe4.com/ransomware-knowledgebase/winlock>
14. IBM Documentation *IBM* : веб-сайт. URL: <https://www.ibm.com/docs/en/aix/7.2.0?topic=management-smb-protocol>
15. SMBGhost – Analysis of CVE-2020-0796 *Trellix* : веб-сайт. URL: <https://www.trellix.com/blogs/research/smbghost-analysis-of-cve-2020-0796/>
16. Fake DocuSign, Gitcode Sites Spread NetSupport RAT via Multi-Stage PowerShell Attack *The Hacker News* : веб-сайт. URL: <https://thehackernews.com/2025/06/fake-docusign-gitcode-sites-spread.html>
17. Microsoft Warns of ClickFix Phishing Campaign Targeting Hospitality Sector via Fake Booking[.]com Emails *The Hacker News* : веб-сайт. URL: <https://thehackernews.com/2025/03/microsoft-warns-of-clickfix-phishing.html>
18. Malware Evolution - Analyzing LockBit 2.0 *Cynet* : веб-сайт. URL: <https://www.cynet.com/attack-techniques-hands-on/malware-evolution-analyzing-lockbit-2-0/>
19. LockBit Analysis *Truesec* : веб-сайт. URL: <https://www.truesec.com/hub/blog/lockbit-analysis>
20. Babuk ransomware's full source code leaked on hacker forum *BleepingComputer* : веб-сайт. URL: <https://www.bleepingcomputer.com/news/security/babuk-ransoms-ware-s-full-source-code-leaked-on-hacker-forum/>
21. Дмитрук Д.В. Дослідження методів виявлення та нейтралізації аномальної активності в інформаційних системах. *Цифрові технології в управлінні та адмініструванні*. 2024. С.88-90

22. GetModuleFileNameW function (libloaderapi.h) *MSDN* : веб-сайт. URL: <https://learn.microsoft.com/en-us/windows/win32/api/libloaderapi/nf-libloaderapi-getmodulefilenamew>

23. Listing the Files in a Directory *MSDN* : веб-сайт. URL: <https://learn.microsoft.com/en-us/windows/win32/fileio/listing-the-files-in-a-directory>

24. curl *curl* : веб-сайт. URL: <https://curl.se/>

25. CreateToolhelp32Snapshot function (tlhelp32.h) *MSDN* : веб-сайт. URL: <https://learn.microsoft.com/en-us/windows/win32/api/tlhelp32/nf-tlhelp32-createtoolhelp32snapshot>

26. Process32First function (tlhelp32.h) *MSDN* : веб-сайт. URL: <https://learn.microsoft.com/en-us/windows/win32/api/tlhelp32/nf-tlhelp32-process32first>

27. Process32Next function (tlhelp32.h) *MSDN* : веб-сайт. URL:

<https://learn.microsoft.com/en-us/windows/win32/api/tlhelp32/nf-tlhelp32-process32next>

28. Дмитрук Д.В. Впровадження YARA-правил як інструменту блокування підозрілих файлів під час ransomware-атак. *The Future of Science: Emerging Research and Technological Innovations*. 2025. С.84-87

29. Using YARA from Python *yara 4.4.0 documentation* : веб-сайт. URL: <https://yara.readthedocs.io/en/stable/yarapython.html>

ДОДАТКИ

ДОДАТОК А



Рисунок 1.1 - Найнебезпечніші види кіберзагроз за 2025 рік

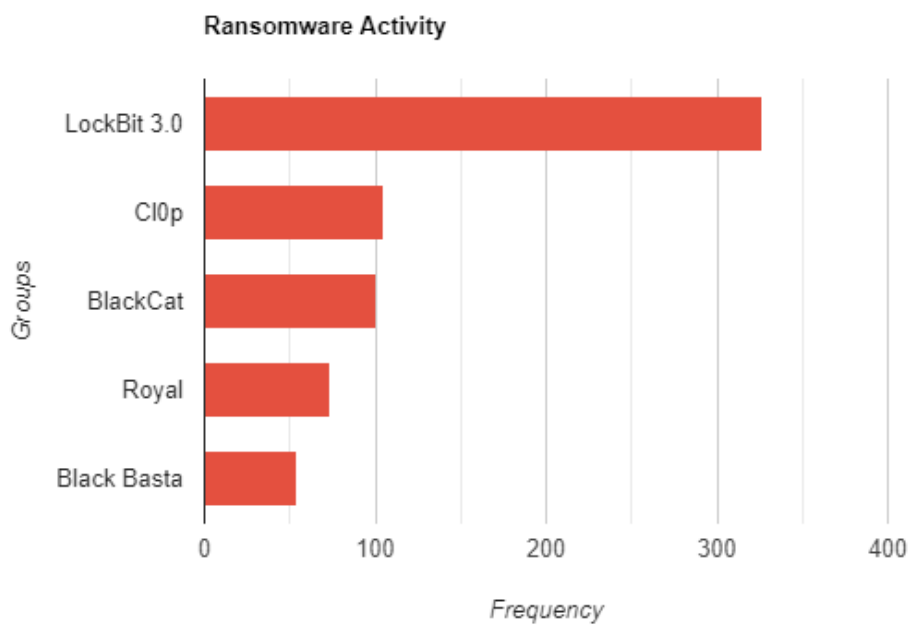


Рисунок 1.2 - Кількість жертв у 2023 році



Рисунок 1.3



Рисунок 1.4

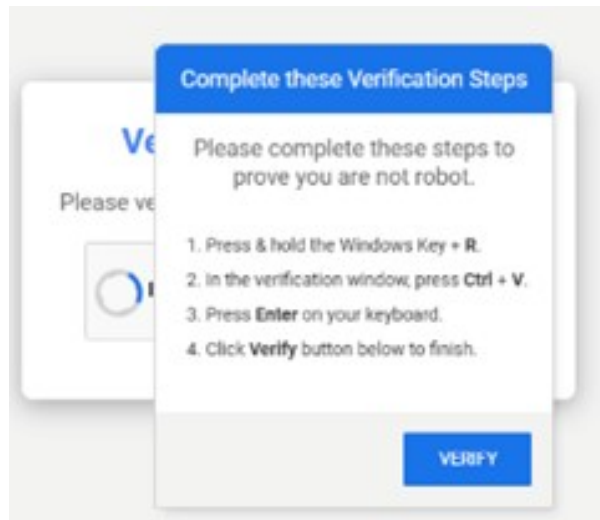


Рисунок 1.5

Search: c4282e9040cdc1df92b722568a8b4c42ce9f6533fed0bd34b7fdbae264947784

66 / 74 Community Score -96

66/74 security vendors flagged this file as malicious

c4282e9040cdc1df92b722568a8b4c42ce9f6533fed0bd34b7fdbae264947784

Babuk_20_04_2021_79KB.exe

Size: 79.00 KB | Last Analysis Date: 11 months ago

peexe direct-cpu-clock-access runtime-modules

DETECTION DETAILS RELATIONS BEHAVIOR COMMUNITY 15+

Join our Community and enjoy additional community insights and crowdsourced detections, plus an API key to automate checks.

Popular threat label: ransomware.babuk/babuk Threat categories: ransomware trojan Family labels: babak babyk ransomx

Security vendors' analysis

AhnLab-V3	Ransomware.Win.Babuk.R428564	Alibaba	Ransom:Win32/Babuk.af7
AliCloud	RansomWare:Win/Babyk	ALYac	Trojan.Ransom.Filecoder
Antiy-AVL	Trojan[Ransom]/Win32.Generic	Arcabit	Trojan.Ransom.Babuk.A
Arctic Wolf	Unsafe	Avast	Win32:RansomX-gen [Ransom]
AVG	Win32:RansomX-gen [Ransom]	Avira (no cloud)	HEUR/AGEN.1316053
BitDefender	Trojan.Ransom.Babuk.A	BitDefenderTheta	Gen:NN.ZexaF.36808.euW@aWBloug
Bkav Pro	W32.AIDetectMalware	ClamAV	Win.Ransomware.Packer-7473772-1
CrowdStrike Falcon	Win/malicious_confidence_90% (D)	Cybereason	Malicious.ef9aba

Рисунок 1.6

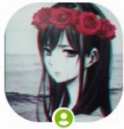
Encryption speed comparative table for some ransomware - 02.08.2021 (added BlackMatter)							
PC for testing: Windows Server 2016 x64 \ 8 core Xeon E5-2680@2.40GHz \ 16 GB RAM \ SSD							
Name of the ransomware	Date of a sample	Speed in megabytes per second	Time spent for encryption of 100 GB	Time spent for encryption of 10 TB	Self spread	Size sample in KB	The number of the encrypted files (All file in a system 287472)
LOCKBIT 2.0	5 Jun, 2021	373 MB/s	4M 28S	7H 26M 40S	Yes	855 KB	109964
LOCKBIT	14 Feb, 2021	266 MB/s	6M 16S	10H 26M 40S	Yes	146 KB	110029
Cuba	8 Mar, 2020	185 MB/s	9M	15H	No	1130 KB	110468
BlackMatter	2 Aug, 2021	185 MB/s	9M	15H	No	67 KB	111018
Babuk	20 Apr, 2021	166 MB/s	10M	16H 40M	Yes	79 KB	109969
Sodinokibi	4 Jul, 2019	151 MB/s	11M	18H 20M	No	253 KB	95490
Ragnar	11 Feb, 2020	151 MB/s	11M	18H 20M	No	40 KB	110651
NetWalker	19 Oct, 2020	151 MB/s	11M	18H 20M	No	902 KB	109892
MAKOP	27 Oct, 2020	138 MB/s	12M	20H	No	115 KB	111002
RansomEXX	14 Dec, 2020	138 MB/s	12M	20H	No	156 KB	109700
Pysa	8 Apr, 2021	128 MB/s	13M	21H 40M	No	500 KB	108430
Avaddon	9 Jun, 2020	119 MB/s	14M	23H 20M	No	1054 KB	109952
Thanos	23 Mar, 2021	119 MB/s	14M	23H 20M	No	91 KB	81081

Рисунок 1.7

Class: Registry
Operation: RegSetValue
Result: SUCCESS
Path: HKCU\Software\Microsoft\Windows\CurrentVersion\Run\{D568D8EB-2424-4974-C3B9-C312099CCE75}
Duration: 0.0000837

Type: REG_SZ
Length: 72
Data: "C:\Users\John\Desktop\Lockbit.exe"

Рисунок 1.8



dyadka0220
(L3) cache
Пользователь

Регистрация: 18.06.2020
Сообщения: 242
Реакции: 48

Вчера в 16:31

Карма или нет по ую уже, жить мне ещё не долго, но пожить как человек успею)
<https://www.sendspace.com/file,>

babuk, babYk, babak - слава админу и дядьке)

Пароль sha256 от: Енотик полоскун, полоскает свой пи чюн)

Последнее редактирование: Вчера в 17:55

🗨 Жалоба

👍 Like + Цитата 🗨 Ответ

👍 petroglyph, Bidon, X-DDoS и ещё 8

Рисунок 1.9

Babyk	5/26/2023 3:53 PM	File folder	
builder	5/26/2023 3:53 PM	File folder	
Decryptor	5/26/2023 3:53 PM	File folder	
Release	4/5/2025 1:13 PM	File folder	
test	4/5/2025 1:13 PM	File folder	
keygen.exe	9/3/2021 3:19 PM	Application	22 KB
Proj.sln	9/3/2021 3:19 PM	Visual Studio Solu...	5 KB
s.txt	9/3/2021 3:19 PM	Text Document	1 KB

Рисунок 1.10 - Код шифрувальника Babyk

```

File Edit Selection View Go Run
curve25519-donna.cpp x
C: > Users > Admin > Desktop > Malware > Main > Ransomware > Babyk > ecc > curve25519-donna.cpp
1 /* Copyright 2008, Google Inc.
40 *
41 * djb's sample implementation of curve25519 is written in a special assembly
42 * language called qhasm and uses the floating point registers.
43 *
44 * This is, almost, a clean room reimplementation from the curve25519 paper. It
45 * uses many of the tricks described therein. Only the crecip function is taken
46 * from the sample implementation. */
47
48 #include <string.h>
49 #include <stdint.h>
50
51 #include "../memory.h"
52 #include "curve25519-donna.h"
53
54 #ifdef _MSC_VER
55 #define inline __inline
56 #endif
57
58 /* Field element representation:
59 *
60 * Field elements are written as an array of signed, 64-bit limbs, least
61 * significant first. The value of the field element is:
62 * x[0] + 2^26·x[1] + x^51·x[2] + 2^102·x[3] + ...
63 *
64 * i.e. the limbs are 26, 25, 26, 25, ... bits wide. */
65
66 /* Sum two numbers: output += in */
67 static void fsum(limb *output, const limb *in) {
68     unsigned i;
69     for (i = 0; i < 10; i += 2) {
70         output[0+i] = output[0+i] + in[0+i];
71         output[1+i] = output[1+i] + in[1+i];
72     }

```

Рисунок 1.11 - Фрагмент коду відповідальний за використання еліптичної кривої в Babyk

ДОДАТОК Б

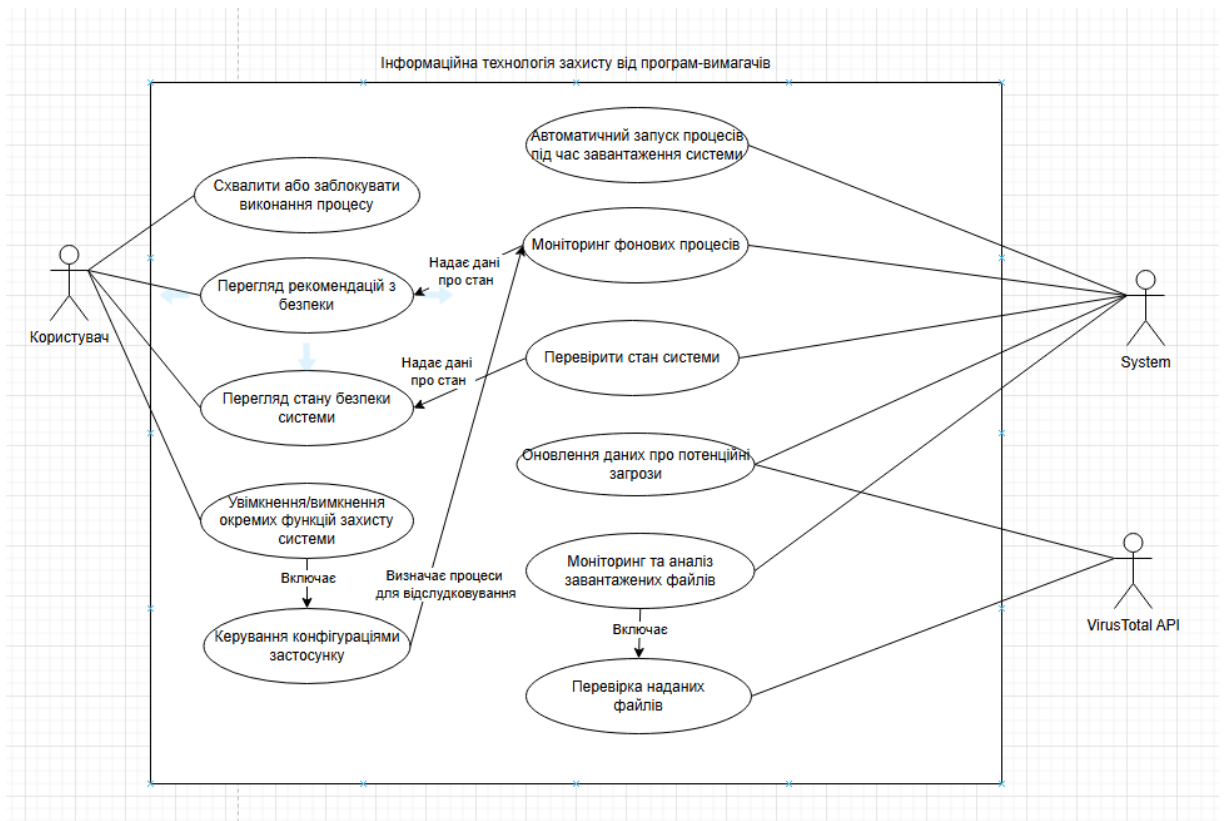


Рисунок 1.1

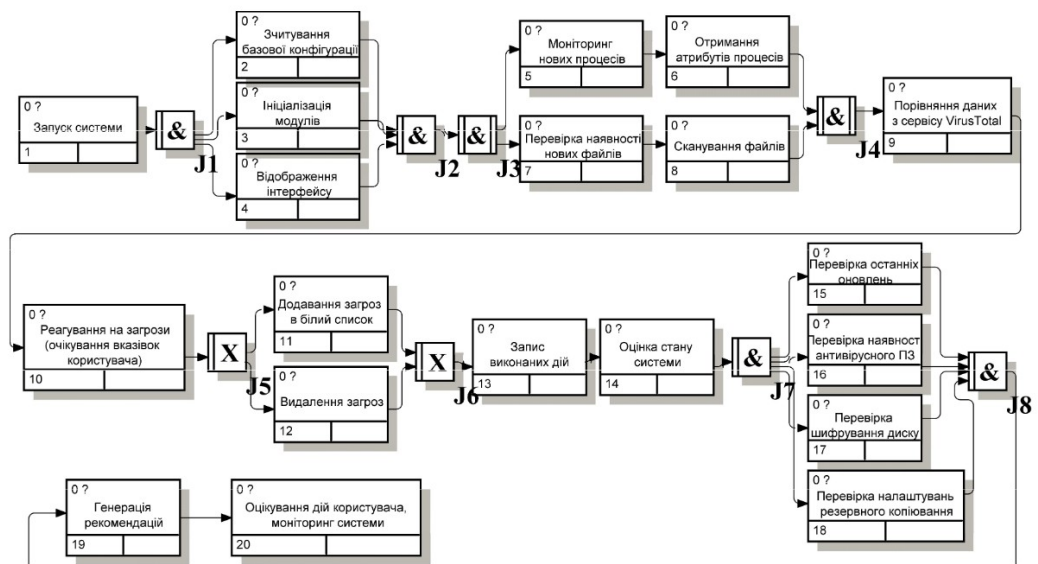


Рисунок 1.2 - IDEF3 діаграма USecure

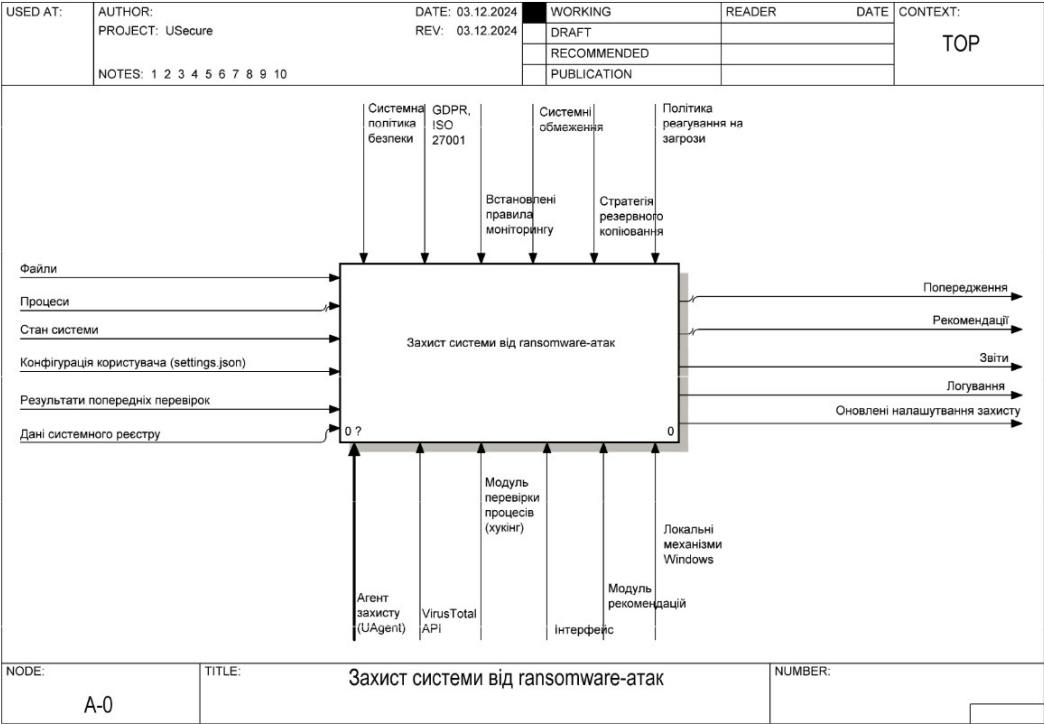


Рисунок 1.3 - IDEF0 діаграма USecure

ДОДАТОК В

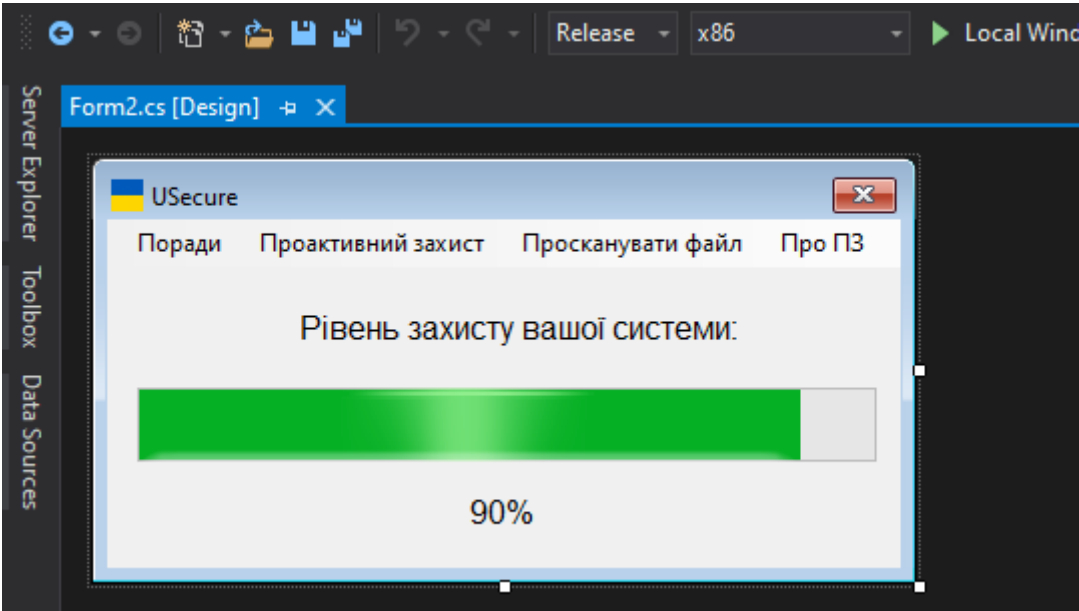


Рисунок 3.1

```

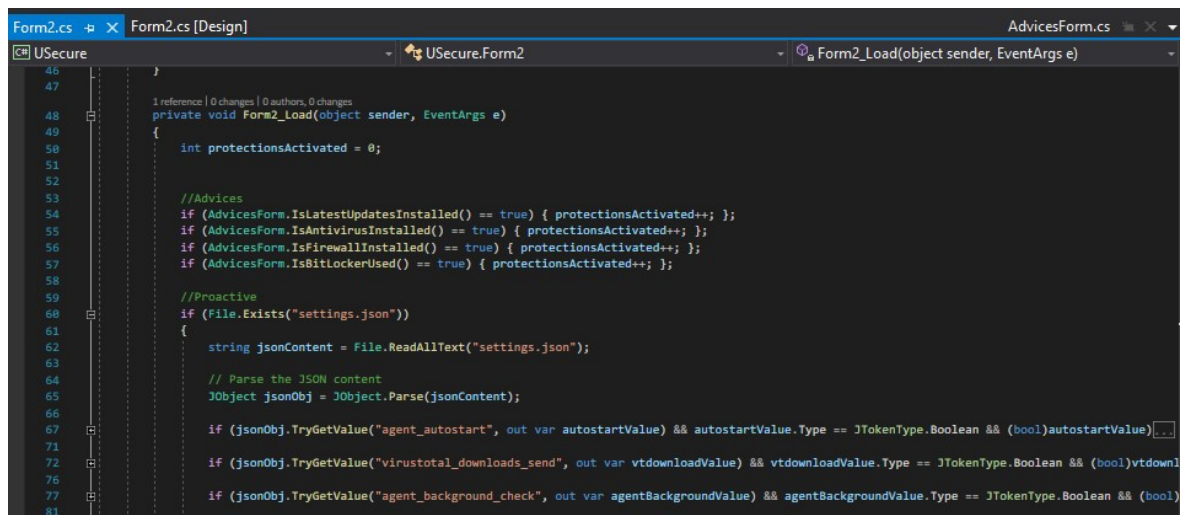
1 reference | 0 changes | 0 authors, 0 changes
private void порадиToolStripMenuItem_Click(object sender, EventArgs e)
{
    AdvicesForm advicesForm = new AdvicesForm();
    advicesForm.Show();
}

1 reference | 0 changes | 0 authors, 0 changes
private void проактивнаЗащитаToolStripMenuItem_Click(object sender, EventArgs e)
{
    ProactiveProtectionForm proactiveProtectionForm = new ProactiveProtectionForm();
    proactiveProtectionForm.Show();
}

1 reference | 0 changes | 0 authors, 0 changes
private void пропнToolStripMenuItem_Click(object sender, EventArgs e)
{
    AboutForm aboutForm = new AboutForm();
    aboutForm.Show();
}

```

Рисунок 3.2



```

46
47
48 1 reference | 0 changes | 0 authors, 0 changes
49 private void Form2_Load(object sender, EventArgs e)
50 {
51     int protectionsActivated = 0;
52
53     //Advices
54     if (AdvicesForm.IsLatestUpdatesInstalled() == true) { protectionsActivated++; }
55     if (AdvicesForm.IsAntivirusInstalled() == true) { protectionsActivated++; }
56     if (AdvicesForm.IsFirewallInstalled() == true) { protectionsActivated++; }
57     if (AdvicesForm.IsBitLockerUsed() == true) { protectionsActivated++; }
58
59     //Proactive
60     if (File.Exists("settings.json"))
61     {
62         string jsonContent = File.ReadAllText("settings.json");
63
64         // Parse the JSON content
65         JObject jsonObj = JObject.Parse(jsonContent);
66
67         if (jsonObj.TryGetValue("agent_autostart", out var autostartValue) && autostartValue.Type == JTokenType.Boolean && (bool)autostartValue)
68         {
69             // ...
70         }
71
72         if (jsonObj.TryGetValue("virustotal_downloads_send", out var vtdownloadValue) && vtdownloadValue.Type == JTokenType.Boolean && (bool)vtdownloadValue)
73         {
74             // ...
75         }
76
77         if (jsonObj.TryGetValue("agent_background_check", out var agentBackgroundValue) && agentBackgroundValue.Type == JTokenType.Boolean && (bool)agentBackgroundValue)
78         {
79             // ...
80         }
81     }

```

Рисунок 3.3

```

//Summary:
int SystemProtectionPercentage = (int)Math.Round((((float)protectionsActivated / 8.0) * 100));

if(SystemProtectionPercentage == 100)
{
    SystemProtectionPercentage--; //There's no absolutely protected system
}

label2.Text = SystemProtectionPercentage.ToString()+"%";
progressBar1.Value = SystemProtectionPercentage;
}

```

Рисунок 3.4

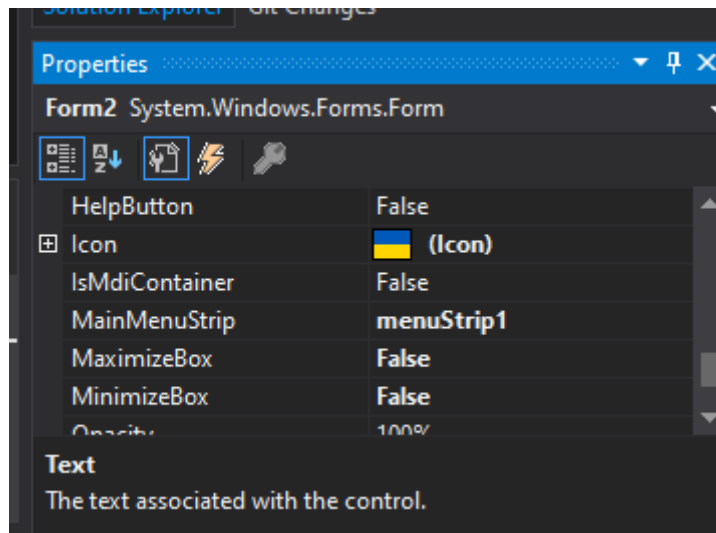


Рисунок 3.5 - Вікно параметрів головного вікна Form2

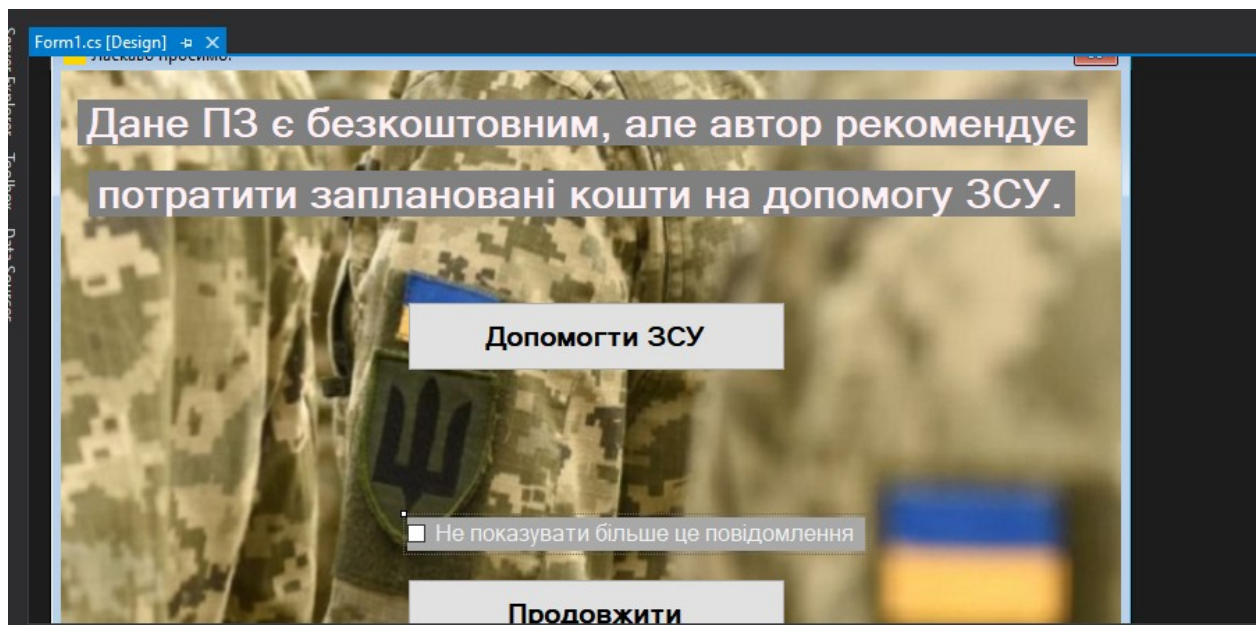


Рисунок 3.6

```
1 reference | 0 changes | 0 authors, 0 changes
private void InitializeDataGridView()
{
    // Create columns
    DataGridViewTextBoxColumn column1 = new DataGridViewTextBoxColumn();
    column1.HeaderText = "Порада";
    column1.Name = "AdviceColumn";

    DataGridViewTextBoxColumn column2 = new DataGridViewTextBoxColumn();
    column2.HeaderText = "Статус";
    column2.Name = "StatusColumn";

    // Add columns to DataGridView
    dataGridView1.Columns.Add(column1);
    dataGridView1.Columns.Add(column2);

    // Add sample data
    dataGridView1.Rows.Add("Встановлення останніх оновлень системи", (IsLatestUpdatesInstalled()) ? "Виконано ✓" : "Не виконано ✗");
    dataGridView1.Rows.Add("Встановлення антивірусного ПЗ", (IsAntivirusInstalled()) ? "Виконано ✓" : "Не виконано ✗");
    dataGridView1.Rows.Add("Використання фаєрвола", (IsFirewallInstalled()) ? "Виконано ✓" : "Не виконано ✗");
    dataGridView1.Rows.Add("Шифрування диску (BitLocker)", (IsBitlockerUsed()) ? "Виконано ✓" : "Не виконано ✗");
    dataGridView1.Rows.Add("Резервне копіювання (остання тіньова копія)", GetLastBackupShadowCopy());

    // Set DataGridView properties
    dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;
    dataGridView1.AllowUserToAddRows = false; // Optional: Disable the extra "new row" at the bottom
}
```

Рисунок 3.7 - Реалізація ініціалізації елементу dataGridView1

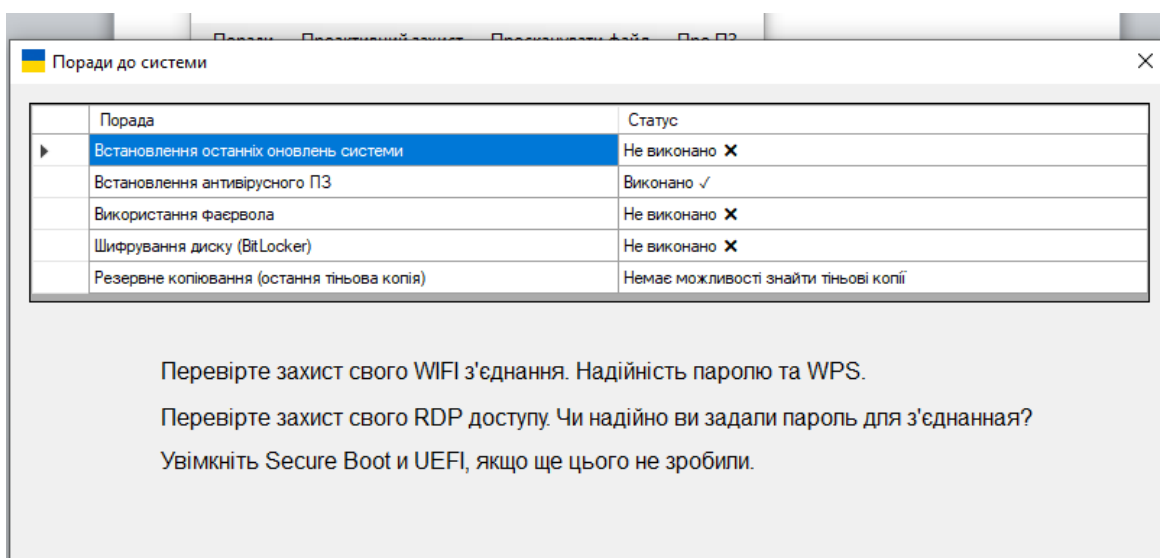


Рисунок 3.8 - Запущене вікно порад

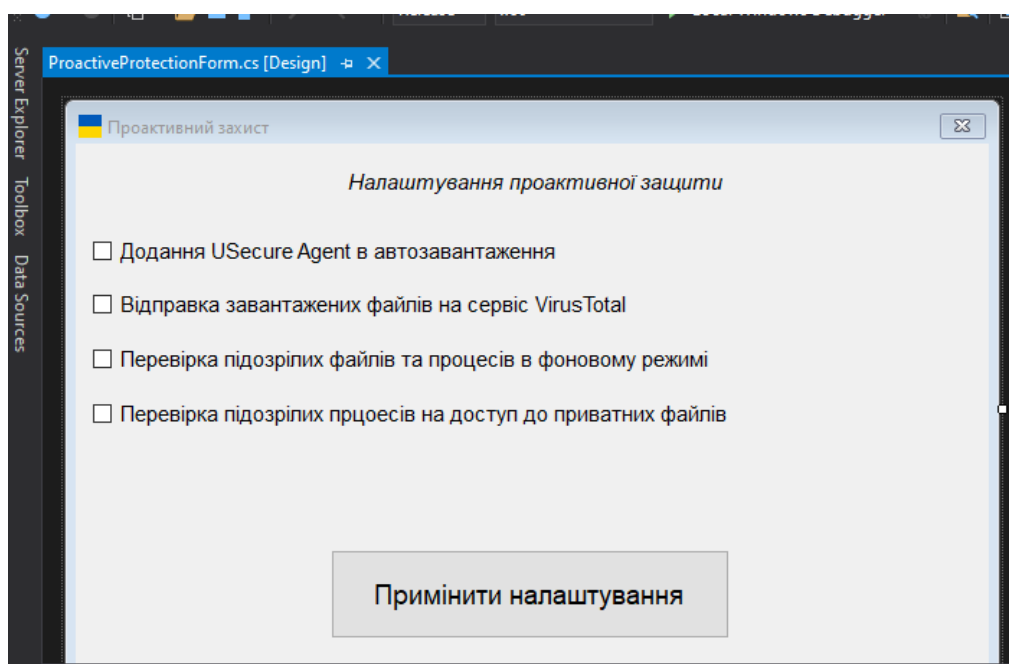


Рисунок 3.9 - Вікно проактивного захисту ProactiveProtectionForm

```
ProactiveProtectionForm.cs [Design]
USecure.ProactiveProtectionForm

1 reference | 0 changes | 0 authors, 0 changes
private void ApplySettings()
{
    try
    {
        // Read JSON from file and deserialize it
        string json = File.ReadAllText("settings.json");
        dynamic settings = JsonConvert.DeserializeObject(json);

        // Set CheckBox states based on the loaded settings
        checkBox1.Checked = settings.virustotal_downloads_send;
        checkBox2.Checked = settings.agent_autostart;
        checkBox3.Checked = settings.agent_background_check;
        checkBox4.Checked = settings.agent_processes_access_check;
    }
    catch (Exception ex) { }
}

1 reference | 0 changes | 0 authors, 0 changes
private void ProactiveProtectionForm_Load(object sender, EventArgs e)
{
    if (File.Exists("settings.json"))
    {
        // Load and apply settings
        ApplySettings();
    }
}
```

Рисунок 3.10 - Код відповідальний за відображення активних налаштувань

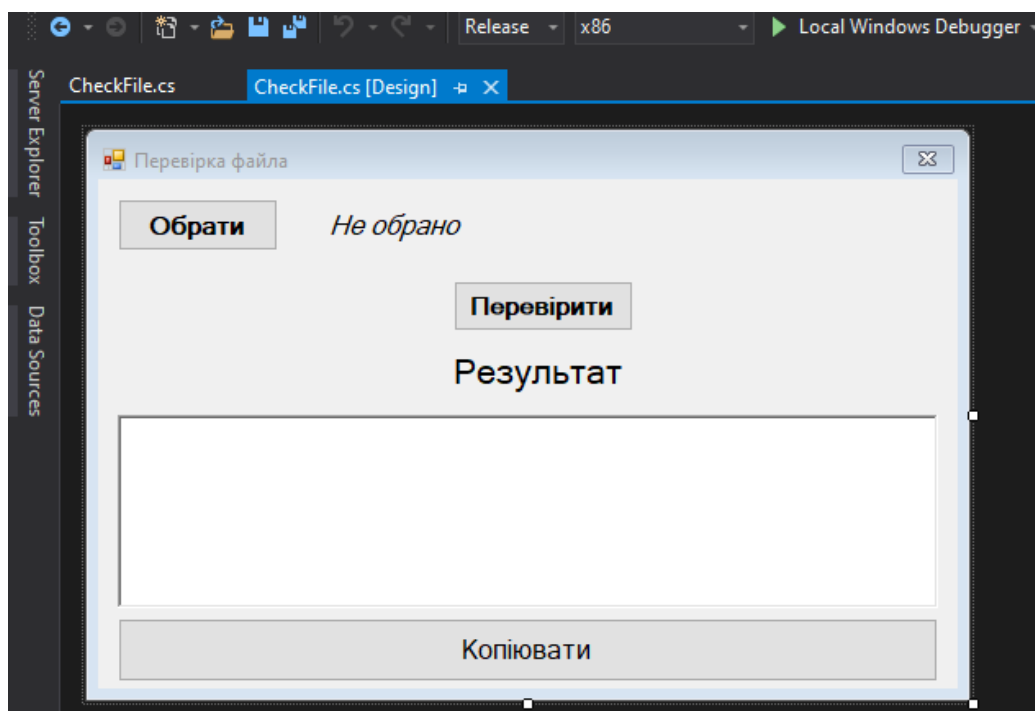


Рисунок 3.11 - Вигляд вікна вибіркового сканування

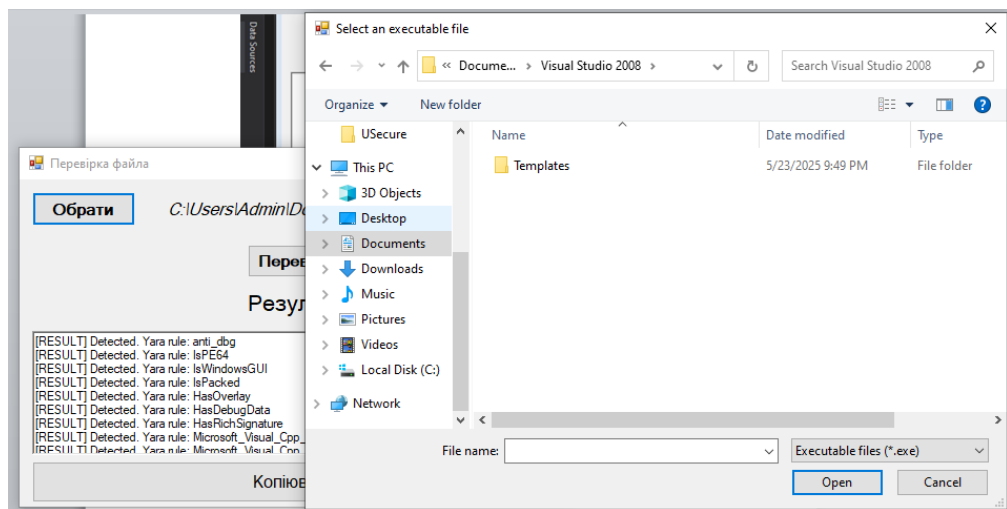


Рисунок 3.12 - Вигляд вікна вибіркового сканування з вибором виконуваного файлу через діалогове вікно та результатами сканування

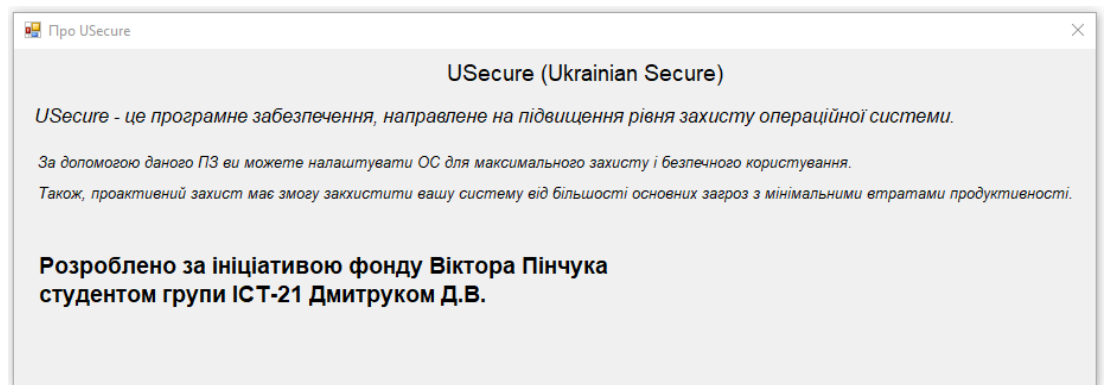


Рисунок 3.13 - Вигляд вікна «Про ПЗ»

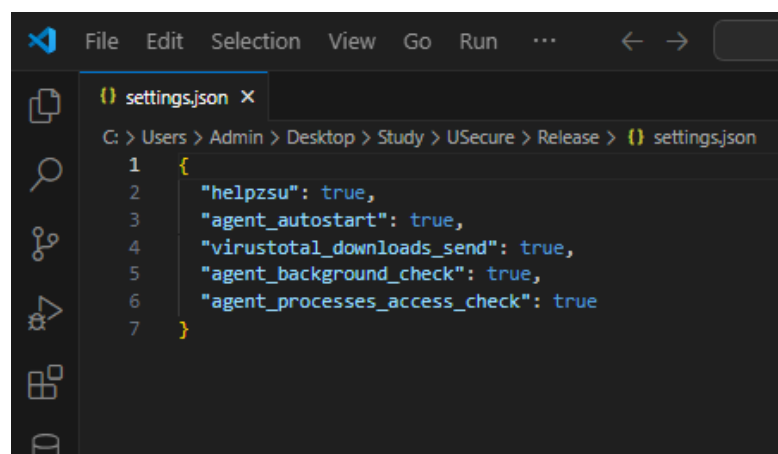


Рисунок 3.14 - Зміст файлу налаштувань

```

BOOL ParseSettings(BOOL* agent_autostart, BOOL* virustotal_downloads_send, BOOL* agent_background_check, BOOL* agent_processes_access_check) {
    // Read the content of the JSON file
    std::ifstream file("settings.json");
    if (!file.is_open()) {
        return FALSE;
    }

    std::string jsonContent((std::istreambuf_iterator<char>(file)), std::istreambuf_iterator<char>());
    file.close();

    // Define the regex pattern to match boolean values
    std::regex pattern(R"([^\s]+)\s*(true|false)");

    // Match boolean values in the JSON content
    std::smatch matches;
    std::string::const_iterator searchStart(jsonContent.cbegin());

    while (std::regex_search(searchStart, jsonContent.cend(), matches, pattern)) {
        // Extract parameter name and boolean value
        std::string paramName = matches[1].str();
        bool paramValue = matches[2].str() == "true";

        if (paramName == "agent_autostart") {
            *agent_autostart = paramValue;
        }
        else if (paramName == "virustotal_downloads_send") {
            *virustotal_downloads_send = paramValue;
        }
        else if (paramName == "agent_background_check") {
            *agent_background_check = paramValue;
        }
        else if (paramName == "agent_processes_access_check") {

```

Рисунок 3.15 - Код функції ParseSettings, яка відповідає за зчитування параметрів

```

BOOL AddSelfToAutostart()
{
    // Get the path to the current executable
    wchar_t exePath[MAX_PATH];
    GetModuleFileNameW(nullptr, exePath, MAX_PATH);

    // Open the registry key
    HKEY hKey;
    LONG result = RegOpenKeyExW(HKEY_CURRENT_USER, L"Software\\Microsoft\\Windows\\CurrentVersion\\Run", 0, KEY_SET_VALUE, &hKey);

    if (result == ERROR_SUCCESS) {
        // Set the registry value
        result = RegSetValueExW(hKey, L"UAgent", 0, REG_SZ, reinterpret_cast<const BYTE*>(exePath), (wcslen(exePath) + 1) * sizeof(wchar_t));

        if (result != ERROR_SUCCESS) {
            return FALSE;
        }

        // Close the registry key
        RegCloseKey(hKey);
    }
    else {
        return FALSE;
    }
}

```

Рисунок 3.16

```

DWORD WINAPI SuspiciousProcessesCheck(LPVOID lpParam) {
    allowedPIDs = (DWORD*)malloc(sizeof(DWORD)*4096);
    memset(allowedPIDs, 0, sizeof(DWORD) * 4096);

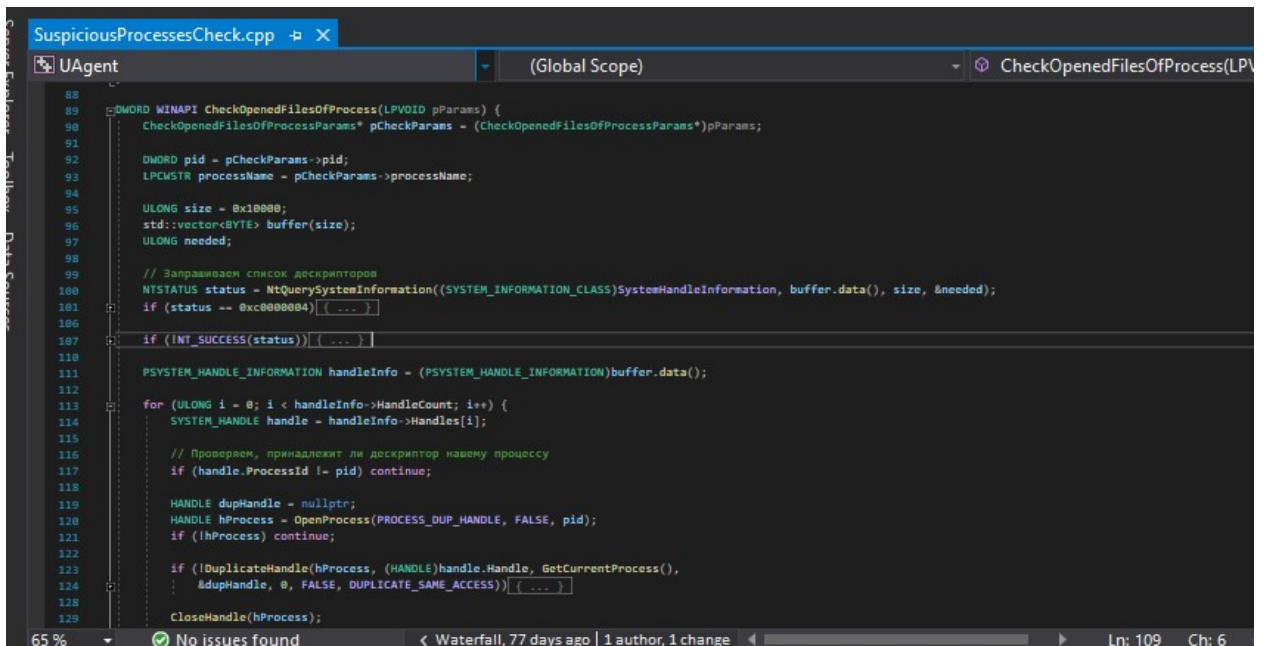
    while (true)
    {
        try {
            //Check process
            LoopForProcesses();
            Sleep(500); // 0.5 sec of delay
        }
        catch (...) {
        }
    }

    free(allowedPIDs);

    return TRUE;
}

```

Рисунок 3.17 - Код функції SuspiciousProcessesCheck

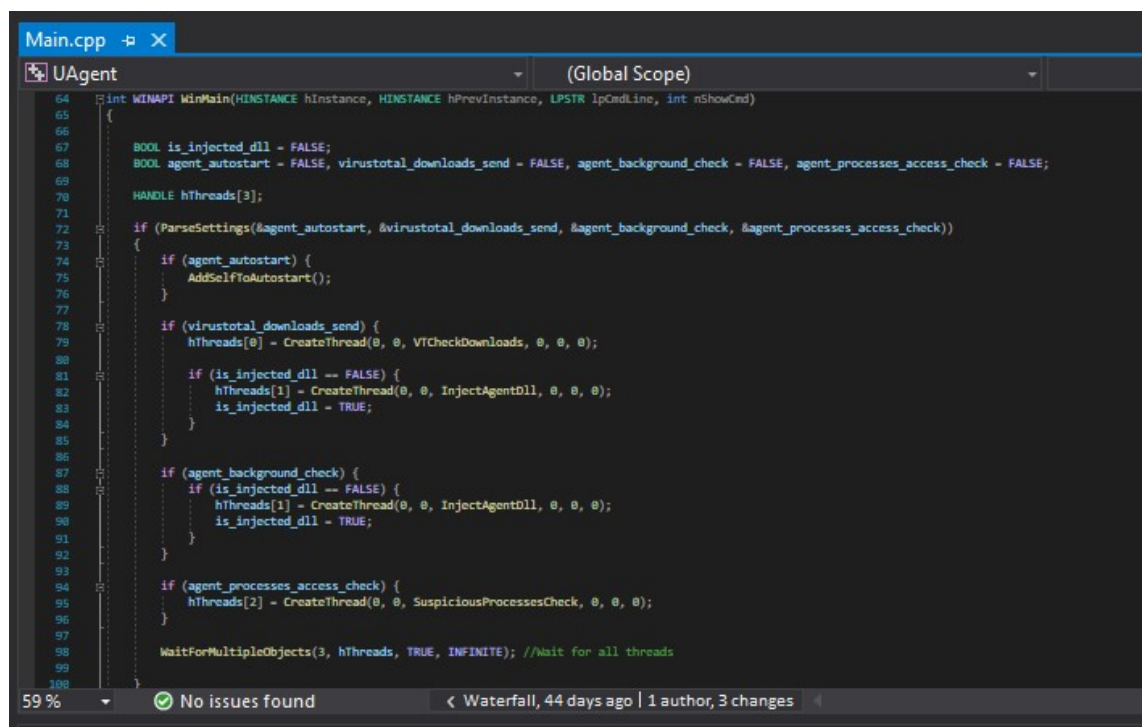


```
SuspiciousProcessesCheck.cpp
UAgent
(Global Scope)
CheckOpenedFilesOfProcess(LPV...
```

```
88 DWORD WINAPI CheckOpenedFilesOfProcess(LPVOID pParams) {
89     CheckOpenedFilesOfProcessParams* pCheckParams = (CheckOpenedFilesOfProcessParams*)pParams;
90
91     DWORD pid = pCheckParams->pid;
92     LPCWSTR processName = pCheckParams->processName;
93
94     ULONG size = 0x10000;
95     std::vector<BYTE> buffer(size);
96     ULONG needed;
97
98     // Запрашиваем список дескрипторов
99     NTSTATUS status = NtQuerySystemInformation((SYSTEM_INFORMATION_CLASS)SystemHandleInformation, buffer.data(), size, &needed);
100     if (status == 0xc0000004) { ... }
101
102     if (NT_SUCCESS(status)) { ... }
103
104     PSYSTEM_HANDLE_INFORMATION handleInfo = (PSYSTEM_HANDLE_INFORMATION)buffer.data();
105
106     for (ULONG i = 0; i < handleInfo->HandleCount; i++) {
107         SYSTEM_HANDLE handle = handleInfo->Handles[i];
108
109         // Проверяем, принадлежит ли дескриптор нашему процессу
110         if (handle.ProcessId != pid) continue;
111
112         HANDLE dupHandle = nullptr;
113         HANDLE hProcess = OpenProcess(PROCESS_DUP_HANDLE, FALSE, pid);
114         if (!hProcess) continue;
115
116         if (!DuplicateHandle(hProcess, (HANDLE)handle.Handle, GetCurrentProcess(),
117             &dupHandle, 0, FALSE, DUPLICATE_SAME_ACCESS)) { ... }
118
119         CloseHandle(hProcess);
120     }
121 }
```

65 % No issues found Waterfall, 77 days ago | 1 author, 1 change Ln: 109 Ch: 6

Рисунок 3.18 - Фрагмент коду функції потоку CheckOpenedFilesOfProcess



```
Main.cpp
UAgent
(Global Scope)
```

```
64 int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nShowCmd)
65 {
66     BOOL is_injected_dll = FALSE;
67     BOOL agent_autostart = FALSE, virustotal_downloads_send = FALSE, agent_background_check = FALSE, agent_processes_access_check = FALSE;
68
69     HANDLE hThreads[3];
70
71     if (ParseSettings(&agent_autostart, &virustotal_downloads_send, &agent_background_check, &agent_processes_access_check))
72     {
73         if (agent_autostart) {
74             AddSelfToAutostart();
75         }
76
77         if (virustotal_downloads_send) {
78             hThreads[0] = CreateThread(0, 0, VTCheckDownloads, 0, 0, 0);
79
80             if (is_injected_dll == FALSE) {
81                 hThreads[1] = CreateThread(0, 0, InjectAgentDll, 0, 0, 0);
82                 is_injected_dll = TRUE;
83             }
84
85             if (agent_background_check) {
86                 if (is_injected_dll == FALSE) {
87                     hThreads[2] = CreateThread(0, 0, InjectAgentDll, 0, 0, 0);
88                     is_injected_dll = TRUE;
89                 }
90
91                 if (agent_processes_access_check) {
92                     hThreads[3] = CreateThread(0, 0, SuspiciousProcessesCheck, 0, 0, 0);
93                 }
94
95                 WaitForMultipleObjects(3, hThreads, TRUE, INFINITE); //Wait for all threads
96             }
97         }
98     }
99 }
```

59 % No issues found Waterfall, 44 days ago | 1 author, 3 changes

Рисунок 3.19 - Точка входу WinMain ПЗ UAgent інформаційної системи

USecure

```

14
15
16 VOID MainRoutine()
17 {
18     HANDLE hThreads[2];
19
20     LPSTR SettingsFilePath = (LPSTR)malloc(MAX_PATH);
21     GetSettingsFile(&SettingsFilePath);
22
23     BOOL agent_autostart = FALSE, virustotal_downloads_send = FALSE, agent_background_check = FALSE, agent_processes_access_check = FALSE;
24
25     if (ParseSettings(SettingsFilePath, &agent_autostart, &virustotal_downloads_send, &agent_background_check, &agent_processes_access_check))
26     {
27         if (agent_background_check)
28         {
29             hThreads[0] = CreateThread(0, 0, BackgroundRoutine, 0, 0, 0);
30         }
31
32         if (virustotal_downloads_send)
33         {
34             hThreads[1] = CreateThread(0, 0, HookingRoutine, 0, 0, 0);
35         }
36     }
37
38     WaitForMultipleObjects(2, hThreads, TRUE, INFINITE); //Wait for all threads
39
40     free(SettingsFilePath);
41 }
42

```

Рисунок 3.20 - Код функції MainRoutine, яка викликається після під'єднання dll

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2
```

```
import yara

import os

import sys

import glob


def load_yara_rules_from_dirs(directories):

    rule_files = []

    for dir_path in directories:

        rule_files.extend(glob.glob(os.path.join(dir_path, '*.yar')) +
glob.glob(os.path.join(dir_path, '*.yara')))


    rules_dict = {}

    for idx, rule_file in enumerate(rule_files):

        try:

            rules_dict[f'rule_{idx}'] = rule_file

        except Exception as e:

            print(f'Failed to include {rule_file}: {e}')

    return yara.compile(filepaths=rules_dict)
```

```
def main():

    if len(sys.argv) != 2:

        print("Usage: python scan.py path_to_exe")

        sys.exit(1)

    exe_path = sys.argv[1]

    if not os.path.isfile(exe_path):

        print(f"File not found: {exe_path}")

        sys.exit(1)

    rule_dirs = [

        "rules/antidebug_antivm",

        "rules/cve_rules",

        "rules/malware",

        "rules/packers"

    ]

    try:

        try:
```

```
rules = load_yara_rules_from_dirs(rule_dirs)

except Exception as e:

    print(f"[ERROR LOAD] Can not load yara rules {e}")

    return


matches = rules.match(exe_path)

if matches:

    for match in matches:

        print(f"[RESULT] Detected. Yara rule: {match.rule}")

    else:

        print("No detection.")

except yara.Error as e:

    print(f"YARA error: {e}")


if __name__ == "__main__":

    main()
```