ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ЗАХИСТУ ІНФОРМАЦІЇ

КАФЕДРА ІНФОРМАЦІЙНОЇ ТА КІБЕРНЕТИЧНОЇ БЕЗПЕКИ

Пояснювальна записка

до бакалаврської роботи

на тему:

«ДОСЛІДЖЕННЯ ШЛЯХІВ ТА РОЗРОБКА РЕКОМЕНДАЦІЙ ЩОДО АНАЛІЗУ ТРОЯНСЬКОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ»

Виконав студент 4 курсу, групи БСД-43
спеціальності 125 Кібербезпека
освітньо-професійної програми «Інформаційна та кібернетична безпека»
(шифр і назва спеціальності)

Яценко.Д,Д
(прізвище та ініціали)

Керівник Марченко В.В
(прізвище та ініціали)

Рецензент
(прізвище та ініціали)

Нормоконтролер Чумак Н.С.
(прізвище та ініціали)

КИЇВ – 2023

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

Інститут ННІЗІ
Кафедра Інформаційної та кібернетичної безпеки
Ступінь вищої освіти Бакалавр
Спеціальність 125 Кібербезпека
Освітньо-професійна програма Інформаційна та кібернетична безпека

ЗАТВЕРДЖУЮ
Завідувач кафедри ІКБ
_____ Гайдур Г.І.
“__”_____2023 року

З А В Д А Н Н Я
НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТУ

Яценко Денис Дмитрович
(прізвище, ім’я, по батькові)

1. Тема бакалаврської роботи: «Дослідження шляхів та розробка рекомендацій щодо аналізу троянського програмного забезпечення»
керівник бакалаврської роботи Марченко Віталій Вікторович, ст. викладач.
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від «24» лютого 2023 року № 26.

2. Строк подання студентом бакалаврської роботи 29.05.2023 р.

3. Вихідні дані до бакалаврської роботи: міжнародна та українська нормативно-правові бази, стандарти, сучасні практики та підходи щодо проведення аналізу троянського програмного забезпечення, наукові публікації вітчизняних та іноземних авторів щодо основних методів та засобів аналізу троянського програмного забезпечення, а також його властивостей, що враховуються при аналізі.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Троянське програмне забезпечення як об’єкт аналізу.
2. Дослідження методик та інструментів аналізу троянського програмного забезпечення.

3. Розробка рекомендацій щодо проведення аналізу троянського програмного забезпечення.

5. Перелік графічного матеріалу

1. Тема бакалаврської роботи.
2. Актуальність теми, мета, об’єкт та предмет, основні завдання дослідження.
3. Результати досліджень методів аналізу шкідливих програмних засобів
4. Результати досліджень інструментів статичного аналізу
5. Результати досліджень інструментів динамічного аналізу
6. Рекомендації по оптимізації аналізу троянського програмного забезпечення
7. Рекомендації по середовищу аналізу троянського програмного забезпечення
8. Практичне використання вироблених рекомендацій
9. Висновки за результатами роботи

6. Дата видачі завдання 24.02.2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ зп	Назва етапів бакалаврської роботи	Строк виконання етапів бакалаврської роботи	Примітка
1.	Аналіз наукової та технічної літератури щодо теми роботи.	23.03.2023 р.	
2.	Дослідження троянського програмного забезпечення як об’єкта аналізу.	06.04.2023 р.	
3.	Дослідження методик та інструментів аналізу троянського програмного забезпечення	20.04.2023 р.	
4.	Розробка рекомендацій щодо проведення аналізу троянського програмного забезпечення	30.04.2023 р.	
5.	Висновки.	04.05.2023 р.	
6.	Оформлення результатів дослідження.	07.05.2023 р.	
7.	Підготовка доповіді до захисту.	15.05.2023 р.	

Студент _____ Яценко.Д.Д
(підпис) прізвище та ініціали

Керівник бакалаврської роботи _____ Марченко В.В
(підпис) прізвище та ініціали

ВІДГУК РЕЦЕНЗЕНТА
на бакалаврську роботу

студента Яценка Дениса Дмитровича
на тему: «Дослідження шляхів та розробка рекомендацій щодо аналізу троянського програмного забезпечення»

Актуальність: На сьогоднішній день помітний значний зріст кількості успішних кібератак, що свідчить про недостатній рівень якості проведення реагування на кіберінциденти та недоліки в механізмах захисту. Виникає необхідність в нових рішеннях для забезпечення кібербезпеки інформаційних систем.

Аналіз шкідливого програмного забезпечення в тому числі і троянського є одним з сучасних напрямків в кібербезпеці, який можна використати для ефективної протидії кібератакам та розробці сучасних засобів захисту інформації.

Позитивні сторони:

1.На основі проведених досліджень було встановлено основні принципи проведення аналізу троянського програмного забезпечення.

2. Було досліджено методики та інструменти проведення статичного та динамічного аналізу шкідливих програмних засобів.

3. Розроблено рекомендації щодо оптимізації процесу аналізу шкідливих програмних засобів.

4. Текст викладено достатньо грамотно, послідовно. Сформульовано чіткі та змістовні висновки. Графічний матеріал оформлено якісно. Список науково-технічної літератури свідчить про вміння користуватись матеріалами за темою бакалаврської роботи.

Недоліки:

1. Застосовано недостатньо широкий набір інструментів та методик при проведенні аналізу троянського програмного засобу.

2. При виконанні практичної перевірки вироблених рекомендацій бажано було б провести порівняння з аналізом троянського програмного забезпечення до застосування розроблених рекомендацій.

Висновок: Враховуючи недоліки, бакалаврська робота заслуговує оцінку **відмінно**, а студент **Яценко Д.Д.** – присвоєння кваліфікації: бакалавр з кібербезпеки за спеціалізацією Інформаційна та кібернетична безпека.

Якість роботи	
Виконано на замовлення підприємства	
Виконано за тематикою НДР	
Виконано з макетом	
Виконано з застосуванням ЕОМ та МПТ	√
Має практичну цінність	√
Проект-частина комплексної теми	

Підпис рецензента (_____)

Підпис засвідчую
Підпис особи, що засвідчує (_____)

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

ПОДАННЯ
ГОЛОВІ ДЕРЖАВНОЇ ЕКЗАМЕНАЦІЙНОЇ КОМІСІЇ
ЩОДО ЗАХИСТУ БАКАЛАВРСЬКОЇ РОБОТИ

Направляється студент Яценко.Д.Д до захисту бакалаврської роботи
(прізвище та ініціали)

спеціальності 125 Кібербезпека
освітньо-професійної програми Інформаційна та кібернетична безпека
(шифр і назва спеціальності)

на тему: «Дослідження шляхів та розробка рекомендацій щодо аналізу троянського програмного забезпечення».

Бакалаврська робота і рецензія додаються.

Директор інституту _____ Савченко В.А.
(підпис) (прізвище та ініціали)

Довідка про успішність

Яценко Д.Д. за період навчання в інституті
(прізвище та ініціали студента)

ННІЗІ з 2019 року по 2023 рік повністю виконав навчальний план за напрямом підготовки, спеціальністю з таким розподілом оцінок за:
національною шкалою: відмінно ____%, добре ____%, задовільно ____%;
шкалою ECTS: A ____%; B ____%; C ____%; D ____%; E ____%.

Секретар інституту, факультету (відділення) _____ Берестяна Т.В.
(підпис) (прізвище та ініціали)

Висновок керівника бакалаврської роботи

Студент ***Яценко Д.Д*** обрав тему роботи, метою якої було дослідження особливостей проведення та оптимізація процесу аналізу троянського програмного забезпечення, що в свою чергу дозволить підвищити ефективність реагування та навіть попередження інцидентів ІБ пов'язаних із застосуванням троянського програмного забезпечення. Перелік використаних джерел свідчить про вміння студентом розбиратись в наукових питаннях та застосовувати їх при дослідженнях. Під час виконання бакалаврської роботи Яценко Д.Д показав добру теоретичну та практичну підготовку, вміння самостійно вирішувати питання і робити висновки. Роботу виконував сумлінно, акуратно та вчасно за планом.

Все це дозволяє оцінити виконану бакалаврську роботу студента ***Яценка Дениса Дмитровича*** на оцінку **«відмінно»** та присвоїти йому кваліфікацію: бакалавр з кібербезпеки за спеціалізацією Інформаційна та кібернетична безпека.

Керівник бакалаврської роботи _____ Марченко В.В.
(підпис) (прізвище та ініціали)

"____" _____ 2023 року

Висновок кафедри про бакалаврську роботу

Бакалаврська робота розглянута. Студент Яценко Д.Д.
(прізвище та ініціали)

Завідувач кафедри Інформаційної та кібернетичної безпеки
(назва)

_____ Гайдур Г.І.
(підпис) (прізвище та ініціали)

"____" _____ 2023 року

РЕФЕРАТ

Текстова частина бакалаврської роботи: 65 сторінки, 18 рисунків, 6 таблиць та 16 джерел.

Об'єкт дослідження – процес аналізу троянського програмного забезпечення.

Предмет дослідження – інструменти та методи аналізу троянського програмного забезпечення.

Мета роботи – визначення особливостей проведення та оптимізації процесу аналізу троянського програмного забезпечення.

Методи дослідження – опрацювання літератури за даною темою, аналізу технічної документацію, синтезу та порівняння.

В роботі проаналізовані основні види троянського програмного забезпечення та методи його аналізу, визначено основні особливості троянського програмного забезпечення, які враховуються при його проведенні.

Досліджені методики статичного та динамічного аналізу, розглянуто та порівняно основні сучасні інструменти для його проведення.

Розроблено рекомендації щодо проведення статичного та динамічного аналізу троянських програмних засобів, створенню безпечного середовища для аналізу та документування результатів.

Проведено аналіз троянського програмного забезпечення з використанням розроблених рекомендацій для перевірки їх ефективності.

Галузь використання – кібербезпека корпоративних інформаційних систем.

КЛЮЧОВІ СЛОВА: ТРОЯНСЬКІ ПРОГРАМНІ ЗАСОБИ, СТАТИЧНИЙ АНАЛІЗ, ДИНАМІЧНИЙ АНАЛІЗ, ПІСОЧНИЦІ, БІНАРНИЙ АНАЛІЗ, ЗВОРОТНА РОЗРОБКА.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ОС – операційна система

ПЗ – програмний засіб

ТПЗ – троянське програмне забезпечення

ІС – інформаційна система

API – прикладний програмний інтерфейс

DNS – система доменних імен

PE – Portable Executable

TLS – Transport Layer Security

HTTP – HyperText Transfer Protocol

SSL – Secure Sockets Layer

ВСТУП

Актуальність дослідження – на сьогоднішній день через зростання кількості та складності кібератак виникає необхідність в постійному покращені механізмів захисту, розробці методів реагування на кіберінциденти та проведення їх розслідувань. Спеціалістам з кібербезпеки необхідно постійно адаптувати системи захисту інформації під нові умови існування в кіберпросторі.

Згідно зі статистикою за 2021 рік близько 22% кібератак на приватні підприємства проводиться з використанням троянського програмного забезпечення, а втрати від програм-вимагачів виросли на 350% порівняно з 2018 роком. Отже ефективне реагування на інциденти при яких використовується шкідливе програмне забезпечення виступає одним з найбільш пріорітетів при забезпеченні кібербезпеки приватних підприємств та державних установ. На жаль сучасні механізми захисту не можуть протидіяти всім видам троянського програмного забезпечення, адже вони постійно змінюється та покращуються.

Для підвищення ефективності реагування на інциденти та покращення можливостей механізмів захисту протидіяти шкідливому програмному забезпеченню використовуються результати аналізу троянського програмного забезпечення. На жаль в наш час цей напрям в кібербезпеці недостатньо висвітлений в наукових публікаціях. Більша кількість наукової літератури по аналізу ТПЗ сконцентрована на загальних аспектах аналізу. Добре вивченими темами є класифікація троянського програмного забезпечення, використання антивірусних програних засобів, методики проведення аналізу та підбір інструментів. Проблему становить мале розкриття теми оптимізації аналізу та його ефективного проведення.

Мета роботи – визначення особливостей проведення та оптимізації процесу аналізу троянського програмного забезпечення.

Наукові завдання:

- Дослідити основні види троянського програмного забезпечення та визначити його особливості, які враховуються при проведенні аналізу.
- Проаналізувати найпоширеніші методики, а також основний інструментарій статичного та динамічного аналізу троянського програмного забезпечення.
- Виробити рекомендації щодо практичного застосування методик аналізу троянського програмного забезпечення.

Об'єкт дослідження – процес аналізу троянського програмного забезпечення.

Предмет дослідження – інструменти та методики аналізу троянського програмного забезпечення.

Практичне значення одержаних результатів полягає в можливості використання рекомендацій щодо проведення аналізу троянського програмного забезпечення при забезпеченні кібербезпеки корпоративних інформаційних систем.

1 ТРОЯНСЬКЕ ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ЯК ОБ'ЄКТ АНАЛІЗУ

1.1 Основні типи троянського програмного забезпечення та головні аспекти їх поширення

Троян - це шкідливе програмне забезпечення, яке застосовує маскування, щоб приховати справжню мету своєї діяльності. У відміну від вірусу, воно не має здатності самостійно копіювати або інфікувати файли. Щоб проникнути на пристрій жертви, така загроза використовує різні методи, такі як хитре завантаження, експлойти вразливостей, завантаження іншого шкідливого коду або соціально-інженерні методи.

Сьогодні трояни є найпоширенішою категорією шкідливого програмного забезпечення, яке використовується для відкриття бекдорів, контролю інфікованого пристрою, видалення даних користувача та їх передачі зловмисникам, завантаження та запуску інших шкідливих програм у певній системі та для інших цілей.

За типами Трояни поділяються на[1]:

- Програми-шпигуни (Mail sender, Password Trojans тощо) - найпоширеніший вид троянів. Вони відразу шукають в системі всі збережені паролі і відсилають їх своєму творцю (зазвичай поштою).
- Програми віддаленого доступу (Virus) - другий за поширеністю вид троянів. Сенс в тому, що троян відкриває який-небудь порт на комп'ютері і відсилає дані творцеві. Якщо виражатися трохи простіше, то на зараженому комп'ютері встановлюється Сервер, а у господаря стоїть Клієнт. І ось цей самий Сервер чекає сигналу від Клієнта, щоб передати йому все, що той забажає, а так само дати йому доступ до управління.
- Перехоплювачі-кейлогеры (Keyloggers) - третій за популярністю вид троянів. Виходячи з назви можна здогадатися, що вони перехоплюють. А що саме? Та все натискання на клавіатуру. А через клавіатуру вводяться дані та логіни і паролі і

листування. Все це ретельно записується у файл, який потім відсилається куди самі здогадуєтеся.

- Завантажувачі (Downloader) - завантажують з інтернету всякий непотріб на свій розсуд. Зазвичай це віруси.
- Дропперы (Droppers) - завантажують у систему інші трояни.
- Програми-жарти (Joke Programs) - виводять на екран що зараз комп буде перезавантажений або що вся інформація буде комусь передана або що комп зламаний і т. п. По суті нешкідливі і створюються для забави.
- Зломщики (Destructive Trojans) - самі злі і противні трояни. Ламають систему, шифрують дані, знищують цілі розділи жорстких дисків і т. п.

Слід ще згадати, що часто в один троян може вбудовуватися відразу кілька (наприклад та віддалене управління і кейлогер і зломщик) доходячи до десятків. Ось такий от "збірник троянів" можна встановити відкривши нешкідливу картинку із зображенням виду на море.

Часто вмість конкретного Трояна виділяють ціле сімейство програмних засобів, що відносяться до одного типу та мають спільні характеристики у функціонуванні, використанні вразливостей та активності в інформаційній системі.

Троянські програми поширюються двома способами: через завантаження зловмисниками-інсайдерами безпосередньо в комп'ютерні системи та за допомогою користувачів, яких спонукають завантажувати та/або запускати їх на своїх системах. Для досягнення останнього, зловмисники поміщають троянські програми на відкриті або індексовані ресурси, такі як файл-сервери і системи файлообміну, або використовують служби обміну повідомленнями, наприклад, електронну пошту. Вони також можуть потрапляти на комп'ютер через проломи безпеки або бути завантаженими самими користувачами з отриманих адрес. Часто використання троянів є лише однією зі ступеней багатоступінчастої атаки на конкретні комп'ютери, мережі або ресурси, включаючи треті сторони. Троянські програми маскуються, і можуть імітувати ім'я та іконку існуючої, неіснуючої або привабливої програми,

компонента або файлу даних, таких як зображення. Це дає змогу запустити їх як користувачам, так і заховати їх на системі. Троянські програми можуть частково або повністю виконувати завдання, під якими вони маскуються. У випадку останнього, шкідливий код вбудовується зловмисником в існуючу програму.

Залежно від мети зловмисника, троян може бути призначений для атаки на конкретну інформаційну систему або для поширення на велику кількість окремих систем.

1.2 Основні методи та загальні правила проведення аналізу троянського програмного забезпечення

Аналіз троянського програмного забезпечення в широкому розумінні охоплює процес встановлення принципів його роботи, джерела походження та потенційного впливу. Під час аналізу відповідного ПЗ використовуються різноманітні інструменти для розбору файлів та аналізу поведінки ПЗ в різних середовищах. Основною метою цього процесу є здобуття найбільш повної та точної інформації про відповідне програмне забезпечення з метою його подальшого використання в різних напрямках.

За допомогою аналізу троянського програмного забезпечення можна досягти наступних цілей:

- реагування на кіберінцидент базуючись на інформації отриманій в ході аналізу;
- створення мережевих сигнатур для подальшого використання в мережевих моніторах. Такі сигнатури представляють собою ряд індикаторів які вказають на активність певного ТПЗ;
- створення хостових сигнатур, які будуть використовуватись моніторами хостів мережі;
- створення ефективних інструменти для пошуку та знешкодження ТПЗ;
- знаходження додаткової інформації про зловмисників, які використовували ТПЗ;

- точне визначення інфікованих шкідливим програмним забезпечення ЕОМ;
- знаходження необхідної інформації для відновлення інформаційної системи після реалізації кібератаки.

Аналіз ТПЗ застосовується у різних сферах забезпечення кібербезпеки, таких як розслідування кіберінцидентів, реагування на кіберінциденти та розробка інструментів для забезпечення кібербезпеки.

Існує дві основні методики щодо аналізу ТПЗ – статичний аналіз та динамічний аналіз. Методики використовуються паралельно, їх ефективність залежить від самого ТПЗ, середовища в якому проводиться аналіз та інших факторів. На сьогоднішній день існує безліч інструментів які дозволяють ефективно та швидко проводити аналіз ТПЗ, розроблене програмне забезпечення для автоматичного аналізу.

Статичний аналіз ТПЗ – це методика аналізу при якому саме ТПЗ не запускається, тобто знаходиться в статичному стані. При статичному аналізі проводиться пошук рядків, аналіз оп-кодів та бінарних послідовностей. Основні методи статичного аналізу [1]:

- Створення цифрового відбитку програми (хеш-функції) для пошуку по інформації про ТПЗ;
- Аналіз антивірусним програмним забезпеченням;
- Форматування файлів;
- Аналіз на наявність «загортування» файлів ТПЗ;
- Дизасемблінг та інші методики зворотньої розробки.

Динамічний аналіз ТПЗ [2] є процесом, під час якого виконується аналіз шкідливого програмного коду або сценарію шляхом його виконання та спостереження за його поведінкою в системі. Під час такого аналізу вивчається робота ТПЗ на різних рівнях (від машинного до реєстрації інформації). Важливо попередити завдання шкоди системі або аналітичній платформі, на якій проводиться аналіз. На відміну від статичного аналізу, динамічний аналіз не зосереджений на бінарних кодах або рядках, але фокусується на фактичній діяльності ТПЗ в системі, виявляються певні шаблони

поведінки та аналізуються конкретні зміни в системі під час виконання шкідливого коду. Завдяки такій специфіці методики, динамічний аналіз стійкий до обфускації та деяких інших методів протидії аналізу.

Основні методи динамічного аналізу:

- Моніторинг процесів в системі під час роботи ТПЗ;
- Порівняння знімків реєстру;
- Перехоплення пакетів;
- Використання інструментів для налагодження програмних засобів.

Додатково статичний та динамічний аналізи поділяються на базовий статичний, базовий динамічний, розширений статичний та розширений динамічний аналіз. Подібне розподілення лише вказує на складність виконання та затрати часу на аналіз.

Повноцінний аналіз ТПЗ включає в себе як статичні, так і динамічні методики, через обмеженість результатів, які можна отримати використовуючи лише одну з них. Результат аналізу представляє собою опис функціоналу, середовище ТПЗ, його призначення. Глибина такого опису залежить від навичок спеціаліста, проводившого відповідний аналіз, підбору інструментів, затрачених ресурсів та ефективності методів протидії аналізу. Загалом можна виділити наступні рівні розуміння ТПЗ:

Таблиця 1.1

Рівні розуміння ТПЗ

Рівень розуміння ТПЗ	**Опис**
Поверхневий	Знання про загальний функціонал ТПЗ, певні його властивості
Розширений	Розуміння середовища ТПЗ, його функціоналу та можливостей
З можливістю відтворення	Розуміння всіх внутрішніх взаємодій ТПЗ, його структури, середовища з можливістю його відтворення
Повний	Повне розуміння коду ТПЗ

Під відтворенням розуміється здатність створювати інструменти для емуляції діяльності ТПЗ. Наприклад, можна відтворити роботу серверної частини програмного засобу, якщо об'єкт дослідження має клієнт-серверну архітектуру. Це надає можливість віддалено контролювати ТПЗ і отримати повну картину кіберінциденту. Однак, не завжди потрібно проводити такий глибокий рівень аналізу. Залежно від цілей аналізу можна зосередитися на певному рівні розуміння ТПЗ.

Інколи виділяють ще два типи аналізу, як похідні від статичних та динамічних методик - аналіз коду та аналіз пам’яті

Аналіз коду - це методика, яка фокусується на аналізі вихідного коду, щоб зрозуміти внутрішню роботу двійкового файлу. Ця методика розкриває інформацію, яку неможливо визначити лише за допомогою базового статичного та динамічного аналізу. Аналіз коду має схожий з аналізом ТПЗ розділ на методи – існують статичні та динамічні методи аналізу коду . Статичний аналіз коду передбачає зворотну розробку файлу ТПЗ та перегляд його коду, щоб зрозуміти принципи роботи програми, тоді як динамічний аналіз коду передбачає налагодження підозрілого двійкового файлу для розуміння його поведінки та основних властивостей.

Аналіз коду - це методологія, яка зосереджена на вивченні вихідного коду для розуміння внутрішньої роботи двійкового файлу. Цей підхід дозволяє отримати інформацію, яку неможливо встановити тільки за допомогою базового статичного або динамічного аналізу. Аналіз коду може бути здійснений за допомогою статичних або динамічних методів. Статичний аналіз коду передбачає зворотну розробку файлу ТПЗ та огляд його коду для розуміння принципів роботи програми, тоді як динамічний аналіз коду включає налагодження підозрілого двійкового файлу для вивчення його поведінки та основних властивостей.

Аналіз пам'яті - це методологія, спрямована на дослідження операційної пам'яті з метою пошуку різних інформаційних артефактів. Зазвичай цей підхід використовується в криміналістичних дослідженнях, але його інтеграція в аналіз ТПЗ може надати додаткові дані про поведінку шкідливого коду.

Етапи аналізу ТПЗ визначаються залежно від цілей дослідження, проте можна виділити оптимальну послідовність дій, яка швидко забезпечить необхідні базові дані [3].

1.3 Дослідження властивостей троянського програмного забезпечення, які враховуються при аналізі

При аналізі ТПЗ необхідно враховувати особливості шкідливого коду, його функціональні можливості, спрямованість та кінцеву мету. Враховуючи ці фактори, можна вибрати найбільш ефективні інструменти та методи аналізу. Деякі з цих факторів відомі ще до початку аналізу, а інші можна виявити лише під час докладного розгляду відповідного ТПЗ. Тому коригування напрямку роботи над аналізом є загальноприйнятою практикою.

Спочатку варто провести класифікацію ТПЗ за його типом. Тип ТПЗ надає загальну інформацію про його функціонал, призначення і особливості роботи. Часто тип ТПЗ відомий перед початком аналізу, але може статися, що в процесі аналізу будуть виявлені додаткові функції програмного засобу і засіб можна буде віднести до нового типу.

Деякі типи потребують докладнішого розгляду, оскільки ТПЗ, що відноситься до них, має функціонал, який, досліджуючи, можна отримати додаткову інформацію.

Під час свого функціонування завантажувачі звертаються до адреси, з якої буде завантажений додатковий шкідливий код. При аналізі трафіку можна віднайти цю адресу. При аналізі завантажувачів обидві частини ТПЗ - сам завантажувач та завантажена частина - представляють інтерес. Аналіз першої частини надасть інформацію про перші етапи інциденту та мережеві адреси, якими користуються зловмисники. Друга частина є самим шкідливим програмним засобом, і його аналіз є важливим для реагування на інцидент.

Якщо ми говоримо про типи ТПЗ, які створюють бекдори то зазвичай вони використовують протокол HTTP для генерації мережевого трафіку, майже завжди

мають в собі ряд функцій, таких як пошук файлів, зміни в реєстрі, створення каталогів, і не вимагають додаткового завантаження коду. Основний інтерес представляє їх клієнт-серверна архітектура, в рамках якої зловмисник отримує контроль над системою, передаючи команди з сервера на клієнтську частину ТПЗ. Зазвичай комунікація ініціюється саме клієнтом, тому під час динамічного аналізу на початкових етапах необхідно звертати увагу на вихідний трафік. Шкідливий код, який ініціює комунікацію з сервером для надання віддаленого доступу, називається зворотним кодом оболонки. Наявність такого коду дозволяє під час глибокого аналізу створити емуляцію сервера, який надає віддалені команди та керує ТПЗ на зараженій системі.

Інші типи ТПЗ також мають додаткові індикатори своєї діяльності, які можна відслідкувати та використати для отримання додаткових даних щодо принципів роботи програмного засобу. Загалом, це можуть бути виклики певних функцій, генерація мережевого трафіку, звернення до певних бібліотек і т.д.

Наступним фактором є середовище, в якому функціонує ТПЗ. Взагалі, середовище включає набір характеристик системи, в яких ТПЗ може виконувати свої функції. До таких характеристик належать операційна система, програмні засоби, бібліотеки динамічних посилань, фреймворки, певні уразливості системи та іноді апаратне забезпечення.

Визначення середовища функціонування ТПЗ є важливим етапом аналізу. Якщо не створити відповідне середовище, яке відповідає потребам програмного засобу, при динамічному аналізі можуть виникнути проблеми, які призведуть до некоректних даних. Перш за все, варто враховувати операційну систему, для якої був розроблений програмний засіб.

Операційна система Windows, на якій створюється більшість шкідливих програмних засобів, надає їм необхідні для функціонування інтерфейси за допомогою Windows API. Windows API є основним набором інтерфейсів програмування, доступних в операційних системах Microsoft Windows. Термін Windows API охоплює

кілька різних реалізацій платформи, які часто мають власні назви. Якщо програмний засіб під час своєї роботи звертається до певного інтерфейсу та використовує функції, надані Windows API, то операційна система є його середовищем. Деякі шкідливі програмні засоби можуть функціонувати на декількох операційних системах і називаються мультиплатформеними програмними засобами.

Іноді існує можливість запустити програмне забезпечення на операційній системі, для якої воно не було розроблене, за умови створення необхідного середовища виконання за допомогою інших програмних засобів. Прикладом такого явища є Wine - програмний засіб для UNIX-подібних операційних систем, який дозволяє запускати бінарні програми. Наявність цього програмного засобу створює середовище для виконання програмного забезпечення. Згідно з проведеними дослідженнями, шкідливе програмне забезпечення для Windows може успішно працювати в середовищі Linux за допомогою Wine. Проте, рівень успішності виконання шкідливого програмного забезпечення Windows в середовищі Linux виявляється відносно низьким. Факт, що деякі зразки шкідливих програм працюють успішно, свідчить про те, що використання програмного забезпечення Wine в середовищі Linux може становити загрозу безпеці систем Linux, які інакше були б захищені від шкідливого програмного забезпечення для Windows. Однак, неможливо встановити загальний зв'язок між всіма типами шкідливого програмного забезпечення або їх поведінкою та успішним виконанням шкідливого програмного забезпечення в середовищі Linux[4].

Середовищем ТПЗ також може виступати апаратне забезпечення, фреймоврки, динамічні бібліотеки тощо. Загальна схема середовища представлена на рис. 1.1. Шкідливий програмний засіб може функціонувати тільки якщо всі шари середовища починаючи з центрального придатні для цього.

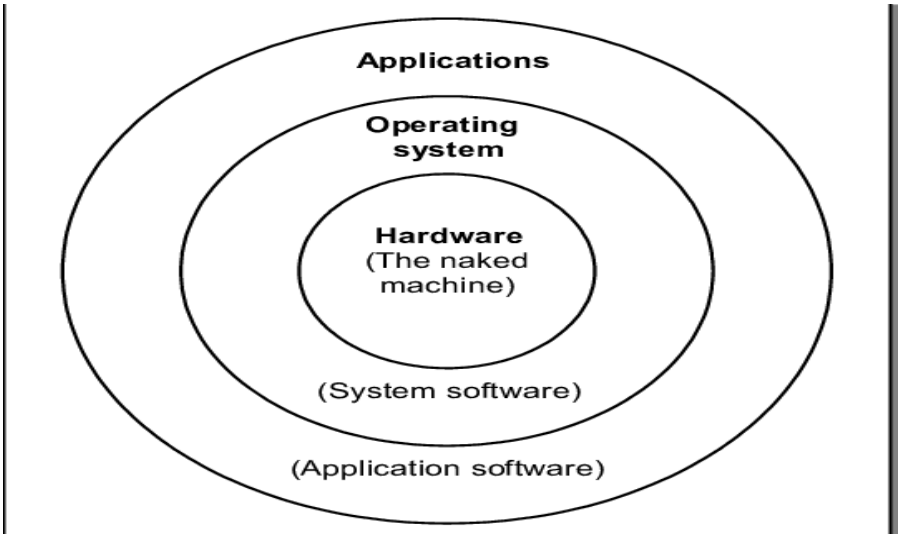


Рис 1.1. Середовище ТПЗ

Останнім важливим фактором, який враховується при аналізі ТПЗ, є наявність унікальних типів програмного забезпечення, які вимагають спеціалізованих методів аналізу. Наприклад, таким явищем є безфайлове ТПЗ. Безфайлове ТПЗ записується безпосередньо в оперативну пам'ять від імені файлу на жорсткому диску. Після успішного запису в пам'ять, ТПЗ намагається отримати привілеї в системі, звертаючись до інструментів адміністрування, та продовжує своє подальше поширення. Безфайлове шкідливе програмне забезпечення не залишає слідів, що ускладнює його виявлення та аналіз.

Також серед нових типів ТПЗ виділяється криптоджекінгове (cryptojacking) програмне забезпечення, спрямоване на використання ресурсів зараженої системи. На відміну від інших типів шкідливого програмного забезпечення, криптоджекінгові програмні засоби не завдають шкоди комп'ютерам та даним. Аналіз такого типу ТПЗ вимагає розуміння принципів його роботи та призначення.

Отже, під час аналізу ТПЗ необхідно враховувати, що деякі об'єкти аналізу мають специфічні характеристики, принципи роботи та призначення, і врахування цих параметрів є важливою частиною успішного проведення аналізу.

Висновки до розділу 1

У даному розділі розглянуто трояни, їх типи, способи поширення, основні методики аналізу та властивості, які враховуються під час аналізу. ТПЗ має багато різних типів, що відрізняються принципами роботи, цілями діяльності та впливом на інформаційну систему. Більшість ТПЗ належить до кількох типів одночасно. Вони поширюються через різні джерела та засоби, найчастіше приховуючись під легітимне програмне забезпечення. Динаміка їх поширення не є стабільною, і різні типи ТПЗ можуть зустрічатись з різною частотою в різних сферах.

Також існують спеціально розроблені ТПЗ для атак на конкретні інформаційні системи. Вони адаптовані під середовище певної ІС і важче піддаються протидії. Аналіз ТПЗ є одним із напрямків боротьби з кібератаками, його результати використовуються для реагування на кіберінциденти, розслідування та розробки механізмів захисту. Аналіз ТПЗ поділяється на статичний та динамічний. Статичний аналіз виконується без запуску ТПЗ, тобто аналізується лише сам файл. Динамічний аналіз полягає у запуску ТПЗ в спеціальному середовищі, при цьому аналізується його активність та вплив на інформаційну систему. Іноді виділяють аналіз коду та аналіз пам'яті як окремі методи аналізу, але їх можна віднести до статичного або динамічного аналізу.

У залежності від цілей аналізу, необхідний рівень розуміння принципів роботи ТПЗ може різнитись. При аналізі ТПЗ важливо враховувати різні специфічні характеристики, такі як тип ТПЗ, його відношення до певних сімейств програмних засобів, середовище, в якому воно діє, та інші специфічні особливості, якщо вони присутні. Це дозволить значно підвищити ефективність аналізу ТПЗ і правильно підібрати інструменти для проведення аналізу.

2 ДОСЛІДЖЕННЯ МЕТОДИК ТА ІНСТРУМЕНТІВ АНАЛІЗУ ТРОЯНСЬКОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1. Дослідження інструментів та методик статичного аналізу шкідливого програмного забезпечення

Статичний аналіз файлу ТПЗ дозволяє вивчити його внутрішню структуру та функціональність. Враховуючи те, що більшість ТПЗ призначені для операційних систем сімейства Windows, більшість інструментів та методів стосуються аналізу PE-файлів, які використовуються у цих операційних системах.

Для безпечного аналізу ТПЗ необхідно створити робоче середовище, в межах якого ТПЗ не зможе вийти та поширитися. Робоча обстановка повинна відповідати потребам ТПЗ для його належного виконання. Для досягнення цієї мети використовуються віртуальні машини.

Існує кілька варіантів розгортання робочого середовища. Найпростіший включає лише одну віртуальну машину без доступу до мережі. Це безпечний варіант для статичного та динамічного аналізу ТПЗ, для роботи якого не потрібна мережева активність.

Якщо ТПЗ використовує мережу під час своєї роботи, необхідно забезпечити доступ до неї. Найбезпечнішим рішенням є підключення кількох віртуальних машин через адаптер віртуального хоста (host-only adapter). На одній з віртуальних машин (де працює ТПЗ) вказуються адреси другої віртуальної машини як DNS-сервер і шлюз за замовчуванням. Таким чином, ТПЗ буде функціонувати на одній машині, а весь мережевий трафік, що генерується під час роботи, буде перенаправлятися на іншу віртуальну машину. Це дозволить контролювати мережеву активність ТПЗ.

Як альтернативний варіант, можна забезпечити підключення віртуальної машини до Інтернету через ТПЗ. Проте, варто використовувати цей підхід лише для конкретних аналітичних завдань з повним розумінням наслідків, які випливають з

надання такого доступу ТПЗ.

Перші кроки статичного аналізу включають визначення типу файлу, що підлягає аналізу, і базовий аналіз самого файлу. Під час визначення типу бінарного файлу можна встановити цільову операційну систему (Windows, Linux і т. д.) та архітектуру (32- або 64-розрядні платформи) шкідливої програми. Наприклад, якщо бінарний файл має формат Portable Executable (PE), який є форматом виконуваних файлів Windows, то можна зробити висновок, що файл призначений для операційної системи Windows. Більшість шкідливих програм для Windows мають розширення файлів, таких як .exe, .dll, .sys і т. д. Однак зазвичай вони не обмежуються лише розширеннями файлів. Розширення файлу не є єдиним індикатором типу файлу. Зловмисники використовують різні методи, щоб приховати свої файли, змінюючи їх розширення та маніпулюючи виглядом, щоб обманом надихнути користувачів на їх запуск. Замість того, щоб покладатися лише на розширення файлу, можна використовувати файловий підпис для визначення його типу. Файловий підпис - це унікальна послідовність байтів у заголовку файлу, яка вказує його тип. Для перегляду цієї послідовності можна використовувати редактори HEX.

HEX-редактори, відомі також як редактори бінарних файлів або редактори байтів, є програмами, які дозволяють редагувати вміст файлів на рівні окремих байтів, незалежно від тлумачення цього вмісту вищими програмами на більш високому рівні. Назва "HEX-редактори" походить від їх здатності відображати кожен байт у файлі в шістнадцятковому форматі. Шістнадцяткова система числення використовує основу 16 та складається з 16 унікальних символів (від 0 до 9 та від A до F). Ця система дозволяє представляти числа, використовуючи шістнадцять різних символів замість десяти, як у десятковій системі. Вона широко використовується в обчисленнях, програмуванні та низці інших галузей, особливо при роботі з пам'яттю та низькорівневими операціями.

Для ідентифікації підпису файлу необхідно відкрити його через HEX-редактор та переглянути заголовок файлу, тобто послідовність з перших двох байтів.

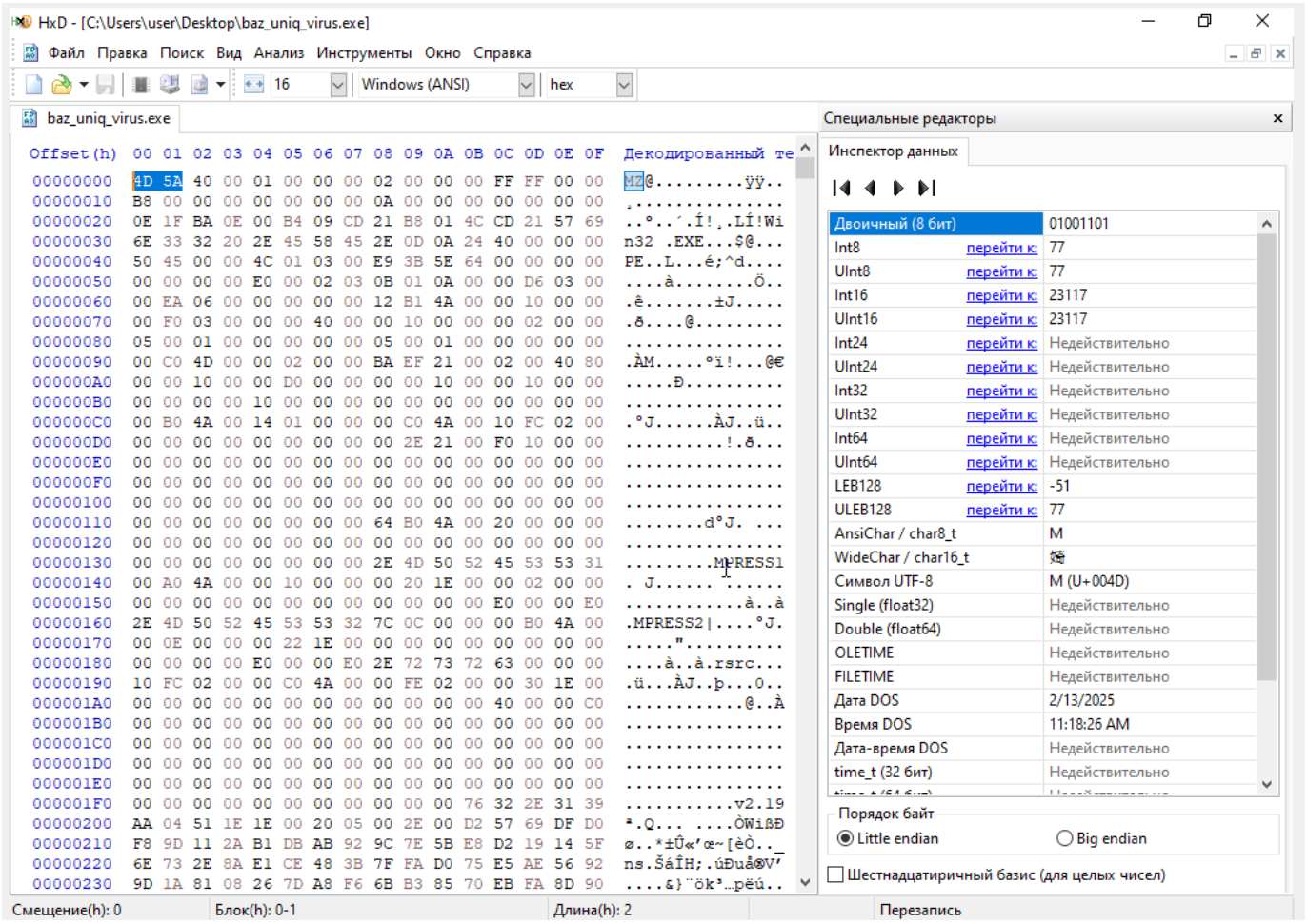


Рис 2.1. Перегляд заголовку файла через HEX-редактор

Як бачимо з прикладу, файл містить послідовність байтів 4D 5A, що відповідає формату exe-файлу. Це дозволяє визначити його тип. Проте це не єдиний спосіб визначення типу ТПЗ. Існують спеціальні інструменти призначені саме для цієї мети. Один з таких прикладів - утиліта "file".

Статичний аналіз шкідливого програмного забезпечення, яке призначене для операційних систем сімейства Windows, найчастіше виконується на PE-файлах. Однак, іноді ТПЗ може бути скриптом, і в такому випадку використовуються інші методи та інструменти для його аналізу, в порівнянні з аналізом PE-файлів.

Portable Executable (PE) є форматом для виконуваних файлів, об’єктного коду та динамічних бібліотек, що використовуються в операційних системах Microsoft

Windows у версіях 32-розрядних та 64-розрядних. В наш час існує два формати PE файлів: PE32 та PE32+. Формат PE32 використовується для систем x86, тоді як PE32+ використовується для x64. Формат PE визначає структуру даних, яка містить всю необхідну інформацію, що дозволяє завантажувачу PE відобразити файл у пам'яті. Виконуваний код містить посилання для зв'язування з динамічно завантажуваними бібліотеками, таблиці експорту та імпорту функцій API, дані керування ресурсами та потокові дані локальної пам'яті (TLS). У сімействі операційних систем Windows NT, формат PE використовується для файлів EXE, DLL, SYS (драйвери пристроїв) та інших типів виконуваних файлів.

Для поверхневого аналізу PE-файлу достатньо здійснити пошук друкованих рядків в самому файлі, проаналізувати ресурси та динамічні бібліотеки. Друковані рядки є послідовністю символів, які містяться у файлі PE. Серед цих рядків можна знайти виклики функцій, виклики динамічних бібліотек та інші дані. Для виконання цієї техніки можна використовувати звичайні утиліти, які виводять рядки. Наприклад, можна скористатися утилітою "strings", щоб вивести рядки з файлу exe.

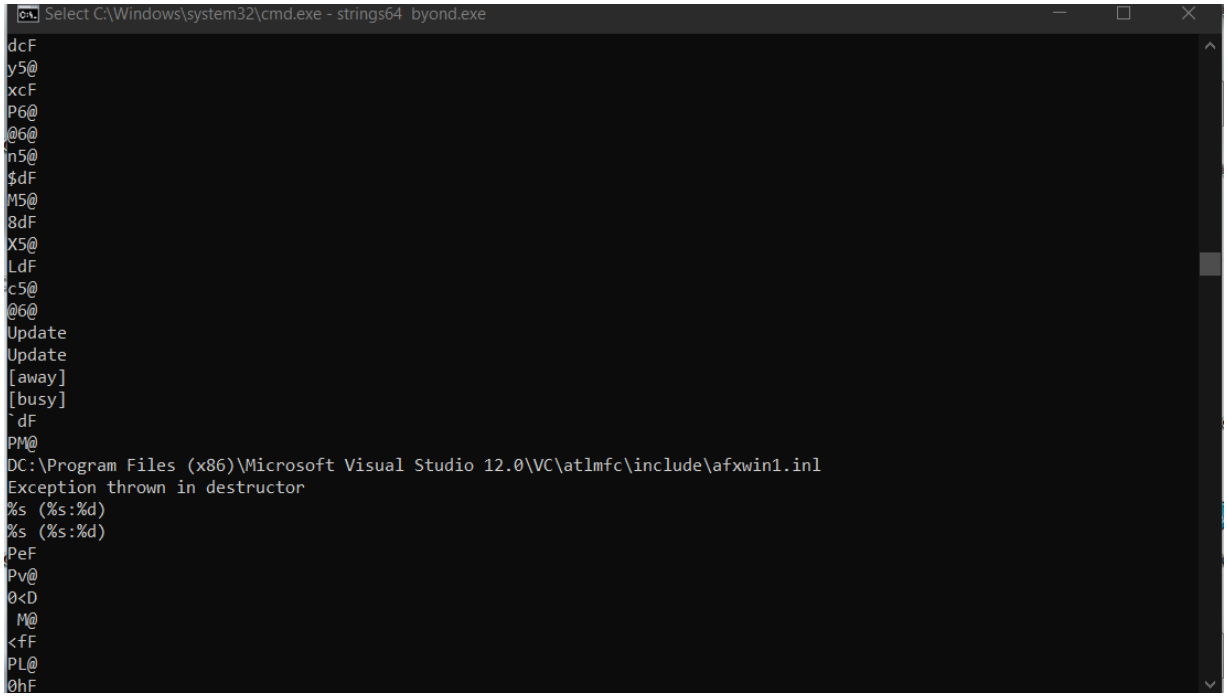


Рис 2.2. Рядки в PE-файлі

Зазвичай більшість знайдених рядків не містять корисної інформації, а представляють собою незрозумілий набір символів (наприклад, як у прикладі M@). Однак, цікавість викликають функції та виклики динамічних бібліотек, а також посилання на файлову систему, які можна легко розпізнати. У даному PE-файлі можна знайти друкований рядок, що вказує на конкретне місце у файловій системі, а також ряд функцій, які можуть містити інформацію про ТПЗ.

Існує багато інших форматів виконуваних файлів, які були розроблені для різних операційних систем та мають свої власні особливості. Наприклад, ELF (Executable and Linkable Format) є форматом виконуваних файлів для UNIX-подібних операційних систем. Він визначає структуру двійкових файлів, бібліотек і основних файлів, а специфікація цього формату дозволяє операційній системі правильно інтерпретувати машинні команди, що містяться у файлі. Файли ELF, як правило, є вихідними файлами компіляторів або компонувальників та мають двійковий формат. З використанням відповідних інструментів їх можна аналізувати та вивчати.

Окрім аналізу рядків та інших методів статичного аналізу файлів, існують різні програмні інструменти, які забезпечують більш точні результати. Серед них можна відзначити інструменти для перегляду файлових ресурсів, для пошуку експортованих та імпортованих функцій, а також для знаходження додаткових даних, таких як метадані, інформація для налагодження тощо. Усі ці інструменти належать до загальної категорії бінарного аналізу.

В окремих випадках, замість проведення самостійного аналізу файлу, інформацію про ТПЗ можна знайти у інших джерелах. Цей підхід є ефективним, особливо якщо аналізується відомий зразок ТПЗ. Для пошуку інформації необхідно обчислити хеш програмного забезпечення.

Хешування - це процес перетворення вхідних даних (таких як повідомлення, файл або довільна послідовність символів) у фіксований числовий вихідний код, який є унікальним для кожного вхідного набору даних. Цей вихідний код, відомий як хеш-значення або просто хеш, дозволяє ефективно зберігати та порівнювати дані. Під час

цього процесу можна отримати унікальну послідовність символів, яка служить ідентифікатором для конкретного ТПЗ. Ця послідовність може бути використана як підпис для пошуку інформації про це ТПЗ в різних джерелах. Вона дозволяє перевіряти наявність додаткових даних, пов'язаних з цим конкретним ТПЗ.

Також одним із способів отримання додаткової інформації є перевірка файлів антивірусними програмами. Вони можуть підтвердити, що файл дійсно містить ТПЗ, і надати додаткові відомості. Проте, головним недоліком цього підходу є можливість отримання поверхових результатів і можливість помилкового спрацювання антивірусного програмного забезпечення.

Одним із способів роботи антивірусних програм є використання сигнатур для пошуку ТПЗ. У типовому аналізі сигнатур виконується перевірка за відомими зразками ТПЗ або конкретними елементами відомого ТПЗ, такими як рядки, CRC (контрольна сума) або хеш MD5. Деякі складніші сигнатури базуються на різних характеристиках PE-заголовка, складності коду в точці входу виконуваного файлу, ентропії файлу або певного розділу/сегмента всередині виконуваного файлу. Використання таких підписів забезпечує більш ефективний аналіз.

Ще одним методом роботи антивірусних програм є використання евристичного сканування. Евристичний сканер шукає загальні шаблони в коді, структурі або поведінці програми, що є характерними для певного типу, класу або сімейства ТПЗ. Цей метод ґрунтується на припущеннях, тому можлива певна кількість помилкових спрацьовувань. [5].

З кожним роком кількість і різноманітність шкідливих програм безперервно зростає, а швидкість їх поширення є надзвичайною. Це ставить під загрозу ефективність антивірусних програм, оскільки не всі вони можуть мати достатні технологічні можливості, актуальні сигнатури або дані для евристичного сканування, щоб ефективно боротися з усіма сучасними шкідливими програмами. Крім того, використання кількох антивірусів одночасно в одній системі може призводити до конфліктів через технологічну несумісність [6]. Це ускладнює процес аналізу,

особливо в організаціях, які мають лабораторії для аналізу ТПЗ і підтримувати кілька антивірусних програм. Однак, існують сервіси, такі як VirusTotal, які надають автоматичний аналіз ТПЗ за допомогою сучасних антивірусних засобів і видають результати сканування. Використання таких сервісів дозволяє отримати максимально точну інформацію з аналізу за допомогою антивірусних програм.

VirusTotal - це служба, яка аналізує підозрілі файли та сприяє швидкому виявленню вірусів, хробаків, троянських програм та всіх видів шкідливих програм, виявлених антивірусними механізмами.

Щоб отримати більше даних про ТПЗ за допомогою статичного аналізу, необхідно використовувати більш складні методи. Один з основних методів - це зворотне проектування. Зворотне проектування - це процес аналізу програмного коду або об'єкту з метою розуміння його функціональності, структури та роботи. Цей метод використовується для розбору програмного забезпечення, апаратних засобів або інших компонентів з метою виявлення їхніх внутрішніх механізмів і алгоритмів. Зворотне проектування може включати такі процеси, як декомпіляція, дизасемблювання, відновлення вихідного коду або створення моделей для подальшого аналізу. Це дозволяє дослідникам, розробникам або безпековим експертам отримувати інформацію про функціональність і внутрішній механізм об'єкта, навіть якщо вихідний код не доступний або неясний. Декомпіляція та дизасемблювання є основними методами зворотного проектування для шкідливого програмного забезпечення.

Декомпіляція - це процес перетворення вихідного коду, який був скомпільований в машинний код, назад у вихідний код високорівневої мови програмування. Цей процес дозволяє отримати частковий або повний аналог початкового вихідного коду, що був використаний для створення програми. Декомпіляція може бути використана для аналізу або модифікації вихідного коду, відновлення алгоритмів, розуміння функціональності програми або виявлення потенційних проблем безпеки. Процес декомпіляції є складним, оскільки відновлення

точного вихідного коду з машинного коду може бути неможливим або неоднозначним, але він може надати важливу інформацію про програму для подальшого аналізу.

Декомпіляція надає можливість конвертувати двійкові послідовності у вихідний код, що полегшує аналіз і зменшує затрати часу, ресурсів та зусиль для експертного дослідження. Крім того, можлива автоматизація аналізу. Проте, декомпіляція має значні недоліки. Результати декомпіляції не завжди є точними або коректними. Зловмисники, усвідомлюючи можливість аналізу ТПЗ, можуть застосовувати методи протидії декомпіляції, вставляючи логічні помилки, беззмістовні функції або рядки, які мають вигляд, але не несуть справжнього значення [7].

Для декомпіляції використовуються спеціальні програмні засоби - декомпілятори, які здатні перетворювати виконувані файли (PE, ELF), динамічні бібліотеки та інші типи файлів у вихідний код мови програмування високого рівня. Існує багато варіантів декомпіляторів, і вибір конкретного залежить від потреб аналізу.

Дезасемблінг - це інший метод зворотної розробки, який дозволяє отримати більш детальну інформацію щодо ТПЗ. Під час дезасемблінгу виконується перетворення машинного коду на його вихідний асемблерний код, що полегшує аналіз і розуміння функцій та операцій, які виконує програма. Цей підхід дозволяє отримати докладну структуру програми і розкрити її логіку. Дезасемблінг має декілька єтапів [8]:

1). Для забезпечення точності аналізу важливо ідентифікувати конкретні частини коду, які підлягають розбору. Спроба конвертувати весь файл без відокремлення кодових інструкцій від даних може призвести до неправильного сприйняття інструкцій та даних, що негативно вплине на результати аналізу.

2). Після визначення області коду, яку необхідно аналізувати, важливо визначити значення, вказані в інструкціях, і перекласти ці значення у двійкову форму. Це дозволить правильно інтерпретувати інструкції і розбирати їх з точністю.

3). Після визначення і перекладу значень інструкцій важливо перерахувати розібрані інструкції у вибраній формі (Лістинг). Це може включати представлення інструкцій у вигляді асемблерних команд або іншого обраного формату, що сприяє зрозумінню логіки програми та подальшому аналізу.

Отже, шляхом поступового аналізу інструкцій на мові асемблера можна отримати важливу інформацію про принципи роботи ТПЗ та його особливості. Дизасемблювання, у порівнянні з декомпіляцією, надає більш точні результати, хоча для його проведення потрібно більше часу та ресурсів. Цей метод дозволяє детально розібрати процеси, функції та обробку даних, що реалізовані в ТПЗ. Проте варто зазначити, що навіть дизасемблювання не є бездоганним, оскільки зловмисники можуть застосовувати методи обфускації для унеможливлення аналізу ТПЗ та надання неправильних даних.

Отже основні методики статичного аналізу включають в себе :

Таблиця 2.1

Методики статичного аналізу

Методика	**Інструменти**	**Результат**
Визначення типу файлу	HEX-редактор	Отримання загальної інформації про середовище функціонування ТПЗ
Бінарний аналіз	Утиліта strings, PE-viewer та інші інструменти для аналізу PE-файлу	Отримання інформації про функції та динамічні бібліотеки, які використовує ТПЗ
Вирахування хешу ТПЗ	Програми для хешування	Базова сигнатура програмного засобу

Продовження таблиці 2.1

Методики статичного аналізу

Методика	**Інструменти**	**Результат**
Використання антивірусного програмного забезпечення	Антивірусні засоби, служби для перевірки файлу рядом антивірусних засобів	Підтвердження шкідливості програмного засобу, додаткові дані
Декомпіляція	Декомпілятори	Отримання вихідного коду, можливе отримання інформації про принципи роботи ТПЗ
Дезасемблінг	IDA Pro, інші дезасемблери	Отримання коду мовою асемблера з якого можна отримати дані про принципи роботи ТПЗ

2.2 Дослідження інструментів та методик динамічного аналізу троянського програмного забезпечення

Динамічний аналіз відрізняється від статичного аналізу як методами, так і інструментами, і має свої особливості та потенційні ризики. Під час динамічного аналізу ТПЗ активно працює в системі, що приносить певні ризики пошкодження системи. Тому для економії часу та ресурсів під час динамічного аналізу зазвичай створюються резервні копії, точки відновлення та знімки віртуальних машин. Перед початком динамічного аналізу враховуються особливості системи захисту та створюється відповідне середовище для проведення аналізу. Цей підготовчий етап є необхідним для успішного проведення аналізу. Дані, отримані під час статичного аналізу, використовуються для налаштування цього середовища.

Основні методи динамічного аналізу включають моніторинг викликів функцій, аналіз параметрів функції та моніторинг потоку інформації. Моніторинг викликів функцій - це процес аналізу функцій, що викликаються під час роботи ТПЗ, та запису відповідної інформації для отримання уявлення про роботу програмного засобу. Аналіз параметрів функції надає конкретну інформацію про дії програмного засобу щодо обробки даних та взаємодії з системою в цілому. Моніторинг потоку даних - це спостереження за змінами в даних під час виконання ТПЗ. За допомогою цих методів можна зробити висновки про принципи функціонування та наслідки діяльності ТПЗ, тому всі вони використовуються в методиках динамічного аналізу [9].

Основні методи статичного аналізу включають збір загальних даних про зміни в системі під час експлуатації. До цих методів належать аналіз змін у системних файлах, аналіз процесів у системі під час роботи ТПЗ, аналіз продуктивності системи, аналіз мережевої активності ТПЗ та аналіз реєстру (для ТПЗ, що працюють на операційних системах сімейства Windows). Усі ці інструменти можна розділити на три категорії [10]:

- Інструменти на основі захоплення процесів - ці інструменти використовують функції API в різних режимах для моніторингу активності ТПЗ.;
- Інструменти на основі порівняння - вони створюють певний знімок або точку відновлення системи та порівнюють стан системи до і після виконання ТПЗ, щоб виявити зміни;
- Сповіщувальні інструменти - ці інструменти сповіщають про певні зміни, які відбуваються в системі, такі як поява нового процесу, виклик функцій тощо, шляхом надсилання повідомлень.

Для аналізу змін у файловій системі використовуються програмні засоби, які записують події, пов'язані з файловою системою, у реальному часі і потім їх аналізують. Ці події можуть включати створення, видалення, модифікацію, відкриття, закриття файлів та інші операції. Основне завдання такого аналізу полягає у виокремленні подій, що пов'язані з діяльністю ТПЗ, з загального потоку подій. Для

досягнення цієї мети використовуються механізми фільтрації. Один з таких засобів для вирішення цього завдання - утиліта FileMonitor, яка є частиною ProcessMonitor. За допомогою механізмів фільтрації, що базуються на правилах і системі реєстрації подій у реальному часі, можна стежити за всіма змінами у файловій системі під час роботи ТПЗ.

наліз процесів включає безпосереднє спостереження за процесами під час роботи ТПЗ та розгляд дерева процесів. При цьому важливо враховувати той факт, що ТПЗ може намагатися приховати свою активність, і назви процесів можуть бути змінені. Для виявлення таких процесів потрібні спеціальні інструменти, які будуть моніторити появу нових процесів і відмічати їх. На сьогоднішній день існує багато програмного забезпечення для аналізу процесів, яке надає різноманітні функції та можливості для аналізу. Серед них можна виділити ProcessMonitor і Process Hacker [11].

Одним з важливих етапів при динамічному аналізі програмного забезпечення для операційної системи Windows є відстеження змін, які відбуваються в реєстрі. Реєстр Windows є централізованою базою даних, де зберігаються інформація, налаштування, параметри та інші значення, що стосуються як програмного, так і апаратного забезпечення, і використовуються у всіх версіях операційної системи Microsoft Windows. При встановленні нової програми в реєстрі створюється новий розділ, де зберігаються відповідні дані, пов'язані з цією програмою. Цей розділ містить параметри, що є характерними для даної програми, такі як її розташування, версія та основний виконуваний файл. Аналіз змін, які відбуваються в реєстрі, надає додаткову інформацію про поведінку програмного забезпечення в операційній системі. Для аналізу змін в реєстрі існують два основних підходи: аналіз журналу змін і порівняння знімків реєстру. Перший підхід схожий на аналіз процесів, оскільки він вивчає зміни, які внесені до реєстру різними програмами у хронологічному порядку. Другий підхід передбачає створення знімка стану реєстру до і після запуску програми, і порівняння конкретних змін. Для проведення аналізу змін в реєстрі можна

використовувати спеціальні утиліти для операційної системи Windows, наприклад, RegShot.

Якщо для аналізу ТПЗ необхідно відстежувати мережеву активність, то використовуються наступні методи: моніторинг DNS-запитів, прослуховування портів, перехоплення та аналіз трафіку. Аналіз DNS-запитів дозволяє з'ясувати, до яких мережевих адрес звертається ТПЗ під час його роботи, що може стати основою мережевого підпису. В даному випадку в якості інструментів можуть виступати як окремі утиліти для цього завдання, так і комплексні програми для аналізу мережевої активності. Прослуховування портів і захоплення трафіку для подальшого аналізу надає більш детальну інформацію. Fiddler, Wireshark, Netcat [12] є одними з найпоширеніших інструментів для аналізу мережевої активності ТПЗ.

Wireshark — безкоштовна програма для аналізу мережевого трафіку. Він дозволяє перехоплювати та аналізувати пакети даних, що передаються через комп’ютерну мережу. Wireshark надає детальну інформацію про кожен пакет, включаючи його джерело та призначення, тип протоколу, розмір, час передачі тощо. Основні можливості Wireshark:

Захоплення пакетів: Wireshark дозволяє захоплювати пакети даних, що передаються через мережу з різних джерел, таких як мережеві інтерфейси, бездротові мережі або записи файлів. Ви можете вибрати певний інтерфейс або фільтр для перехоплення лише певного типу трафіку.

Аналіз пакетів: Wireshark надає детальну інформацію про кожен захоплений пакет. Ви можете переглядати різні поля пакета, наприклад IP-адреси, порти, протоколи, вказаний вміст тощо. Пакети також можна розширити для детального аналізу заголовків і даних.

Фільтрування пакетів: Wireshark дозволяє використовувати фільтри для відображення лише певних пакетів, які відповідають певним критеріям. Це допоможе вам зосередитися на конкретному типі трафіку чи проблемі, над якою ви працюєте. Статистика мережі: Wireshark надає різні статистичні звіти про захоплений трафік. Ви

можете переглядати статистику використання мережевих протоколів, інтерфейсів, статусів підключень, пропускної здатності тощо.

Іншим підходом до аналізу мережі ТПЗ є використання програмного забезпечення емуляції мережі. Такі програми дозволяють проводити аналіз ТПЗ без необхідності налаштування реальної мережі, оскільки їх функції виконує програма-емулятор. Цей підхід має кілька переваг, включаючи безпеку аналізу та можливість перехоплювати й аналізувати мережевий трафік.
За допомогою програмного забезпечення для емуляції мережі ви можете створювати віртуальні середовища, які імітують реальну мережу. ТПЗ працює в цьому віртуальному середовищі, і емулятор перехоплює та аналізує його мережеву активність. Такий підхід дозволяє проводити різні тести та експерименти з ТПЗ без ризику впливу на реальну мережу та системи.

Програмне забезпечення для емуляції мережі надає зручні інструменти для аналізу активності мережі ТПЗ, наприклад захоплення пакетів, аналіз трафіку, моделювання різних умов мережі тощо. Цей підхід дозволяє вам досліджувати та тестувати ТПЗ у контрольованому середовищі, що допомагає виявити потенційні проблеми та оптимізувати продуктивність мережі.

Одним з програм-емуляторів мережі є FakeNet. Крім емуляції мережі, це програмне забезпечення також підтримує DNS, SSL і HTTP, що дозволяє більш ефективно аналізувати мережеву активність. Воно також може перенаправляти трафік на локальний хост і створювати спеціальний файл з інформацією про перенаправлений трафік.

У проведенні динамічного аналізу корисним є використання методів візуалізації. Візуалізація передає схематичне зображення певної діяльності. Вона дозволяє швидко та ефективно аналізувати вже отримані дані, хоча зазвичай не надає додаткової інформації. Візуалізація може охоплювати як мережеву активність, так і

процеси в системі. Один із інструментів візуалізації - ProcDot, який корелює інформацію, отриману з інших інструментів динамічного аналізу, і може візуалізувати її. Ця інформація включає всі процеси, зміни в файловій системі, зміни в реєстрі та мережеву активність. Використання візуалізації особливо ефективне, коли ТПЗ має складну модель поведінки, а в процесі його роботи в системі відбувається багато процесів.

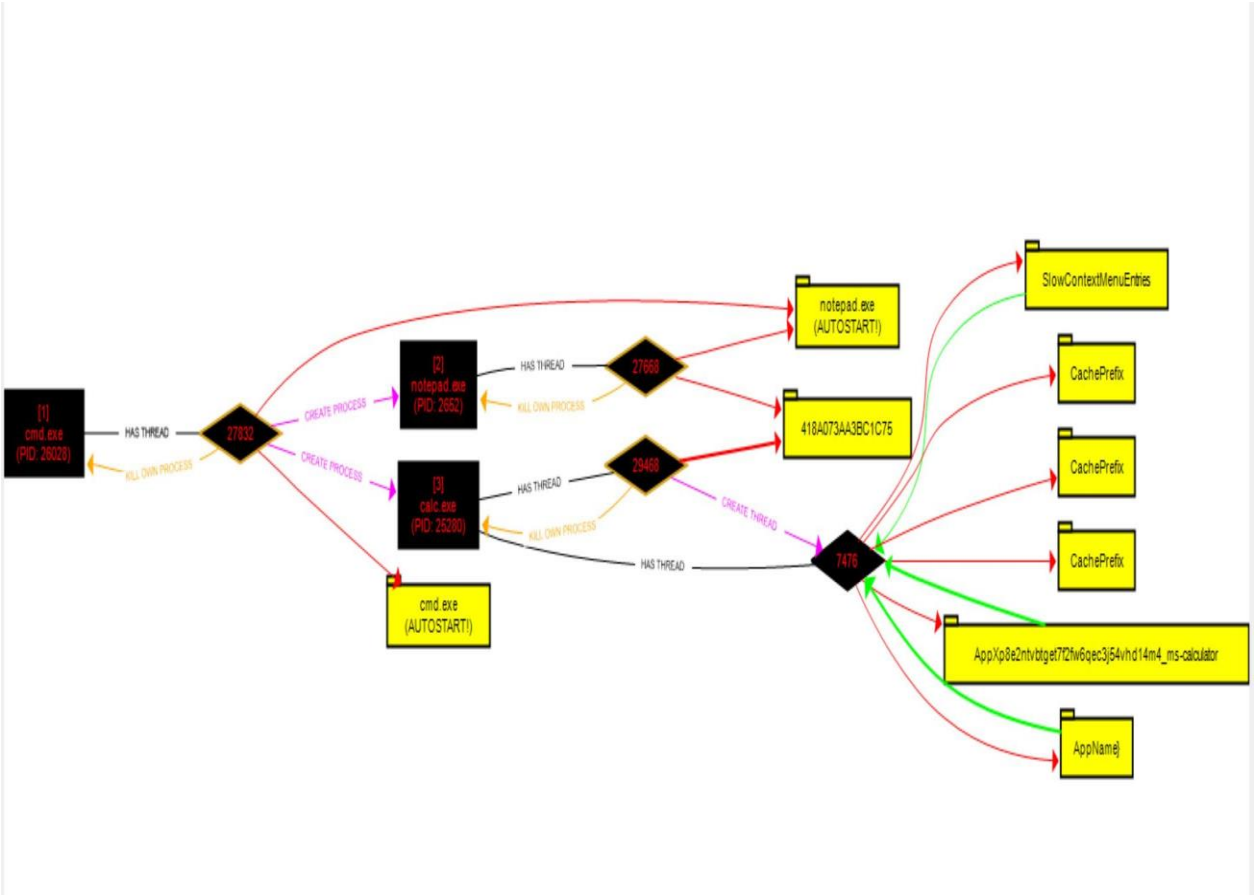


Рис 2.3. Результати візуалізації за допомогою ProcDot

Більш складним методом динамічного аналізу є використання засобів налагодження. Налагоджувачі - це програми, що використовують різні методи для тестування та налагодження іншого програмного забезпечення. Вони використовуються як для пошуку помилок у роботі програмних засобів, так і як один з методів зворотного проектування. Під час налагодження аналізуються системні

ресурси під час роботи програми, а також аналізуються дані з носіїв інформації, реєстрів та процесора. Налагоджувачі, хоча схожі на дизассемблери за своїм призначенням при аналізі ТПЗ, надають більше інформації, оскільки збирають дані під час роботи програмного забезпечення. Налагодження шкідливого програмного забезпечення здійснюється віддалено, тобто налагоджувач працює в іншій системі, ніж сам ТПЗ, оскільки його діяльність може пошкодити систему та знищити налагоджувальні дані [13].

При виконанні процесу налагодження найбільш оптимальним варіантом буде покрокове виконання ТПЗ. Більшість програм по налагодженню дозволяють проводити подібний тип виконання. При виконанні великої кількості рядків можна встановити точку переривання для продовження покрокового виконання.

При виконанні процесу налагодження найбільш оптимальним варіантом є поетапне виконання ТПЗ. Більшість програм налагодження підтримують цей тип виконання. Під час виконання великої кількості рядків можна встановити точку зупинки, щоб продовжити поетапне виконання.

При налагодженні також слід звертати увагу на винятки. Винятки виникають, коли програма потрапляє в недоступну частину пам'яті або виникають інші помилки. Зловмисники можуть намагатися створити винятки, щоб ускладнити процес налагодження. В такому випадку налагоджування ТПЗ може зайняти більше часу і ресурсів. Серед інших методів протидії може бути невиконання ТПЗ у певному середовищі. Для запобігання цьому може знадобитися модифікація виконання ТПЗ.

Деякі типи ТПЗ вимагають використання налагодження на рівні ядра. Для цього використовується спеціальне програмне забезпечення, яке може працювати безпосередньо в ядрі операційної системи. Проведення такого типу налагодження вимагає наявності мережі, тому при необхідності такого налагодження потрібно мати належні знання щодо організації робочого середовища, яке включатиме декілька систем, з'єднаних в мережу.

Альтернативним підходом до динамічного аналізу ТПЗ є використання

пісочниць як інструментів для автоматизованого аналізу. Пісочниці - це спеціальні середовища для безпечного й автоматизованого аналізу програмного забезпечення. Різні пісочниці можуть мати різні функціональні можливості, системи звітності та підтримувати різні типи файлів для аналізу.

JoeSandbox і Cuckoo Sandbox дійсно виділяються серед основних пісочниць, створених спеціально для динамічного аналізу. Поглянемо на загальні принципи роботи пісочниці на прикладі JoeSandbox.

JoeSandbox створює журнал, який включає інформацію про файлову систему, реєстр та системну активність. Ця пісочниця базується на архітектурі клієнт-сервер, де кілька клієнтських аналізаторів виконують різні типи статичного та динамічного аналізу. Усі дані аналізу збираються в єдиній системі. JoeSandbox, зосереджуючись на інструментах захоплення процесів, поєднує виклики API режиму користувача та ядро операційної системи для проведення аналізу.

Пісочниця також володіє механізмами приховування аналізу, оскільки деякі типи ТПЗ можуть виявити таку активність і зупинити свою роботу.

Аналізувавши файл у пісочниці, буде згенеровано звіт, що міститиме результати статичного та динамічного аналізу. В цьому звіті буде надано висновок про шкідливість файлу, інформацію про файл, а також записи його активності в системі, включаючи зміни в реєстрі, файловій системі та виклики функцій. Звіт також включатиме мережеву активність та список пов'язаних з діяльністю ТПЗ мережевих адрес [14].

Пісочниці мають різні типи, але основними серед них є хмарні пісочниці та пісочниці для встановлення. Хмарні пісочниці дозволяють проводити аналіз без необхідності встановлювати додаткові програмні засоби. Вам потрібно лише завантажити відповідний файл до хмари, і після завершення аналізу ви отримаєте повний звіт з отриманими результатами. Іноді хмарні пісочниці також надають віддалений доступ до віртуальної машини, де здійснюється аналіз, для спостереження в реальному часі. Проте, слід зазначити, що обмеження дій, які можна виконати над

системою за такого типу доступу, можуть існувати.

Інсталяційні пісочниці, на відміну від хмарних, вимагають додаткових налаштувань та встановлення в системі, на якій здійснюється аналіз ТПЗ. Наприклад, для використання Cuckoo Sandbox необхідно встановити додаткові програмні інструменти та налаштувати їх перед початком аналізу.

У хмарних пісочницях часто існує система публічних звітів про аналіз ТПЗ, що означає, що вже може бути доступний звіт про попередні аналізи ТПЗ, який може бути використаний для подальшого аналізу. Це дає можливість скористатися наявними результатами та звітами замість проведення повторного аналізу одного і того ж ТПЗ.

Використання пісочниць має недолік у відсутності повного контролю над середовищем ТПЗ. Хоча більшість сучасних рішень дозволяють вибирати операційну систему для аналізу ТПЗ і враховувати деякі додаткові фактори, вибір оновлень ОС або окремих програмних засобів, які використовуються в цій системі, не завжди можливий. Крім того, пісочниці не надають повного звіту про поведінку ТПЗ в системі, тому в окремих випадках використання пісочниць не може замінити звичайний динамічний аналіз, який дозволяє отримати детальну інформацію про всі процеси, зміни та інші особливості поведінки ТПЗ у контексті інформаційної безпеки. Таким чином, до методів динамічного аналізу відносяться наступні:

Таблиця 2.2

Методики динамічного аналізу

Методика	**Інструменти**	**Результат**
Аналіз змін в файловій системі	FileMonitor та його аналоги	Інформація про зміни в файловій системі під час функціонування ТПЗ

Продовження таблиці 2.2

Методики динамічного аналізу

Методика	**Інструменти**	**Результат**
Аналіз процесів в системі	ProcessMonitor, Process Hacker та їх аналоги	Інформація про процеси, в системі під час функціонування ТПЗ
Аналіз змін в реєстрі	RegShot та його аналоги	Отримання інформації про зміни в реєстрі під час функціонування ТПЗ
Аналіз мережі	Wireshark, Netcat, Fiddler та їх аналоги	Отримання інформації про мережеву аткивність ТПЗ
Візуалізація	ProcDot та його аналоги	Графічне представлення процесів та мережевої активності в системі
Налагодження	WinDeb та його аналоги	Отримання інформації про принципи роботи ТПЗ
Використання пісочниць	JoeSandbox, Cuckoo Sandbox та інші пісочниці	Повний звіт про активність ТПЗ в системі

Висновки до розділу 2

У цьому розділі розглянуті основні засоби для проведення статичного та динамічного аналізу ТПЗ. Для базового статичного аналізу потрібно спочатку визначити тип файлу, що підлягає аналізу, і провести подальший аналіз, зосереджений на цьому типі файлу. Оскільки більшість ТПЗ зазвичай виконується з PE-файлів, більшість методів спрямовані на бінарний аналіз, такий як пошук рядків, підключених бібліотек і функцій, аналіз даних і ресурсів у PE-файлі. Для отримання більш

детальних результатів статичного аналізу застосовуються методи зворотного проектування. Два основні напрямки реверс-інжинірингу, що використовуються при аналізі ТПЗ, це декомпіляція та дизасемблювання.

Декомпіляція перетворює двійковий файл у вихідний код на високорівневій мові програмування, що полегшує його аналіз. Цей метод вимагає менше часу та ресурсів, але не завжди дає чіткі результати.

Дизасемблювання перетворює двійковий код на мову асемблера, після чого отриманий код можна проаналізувати. Цей підхід є більш складним, але надає більш точні результати. Використання цих методів дозволяє отримати розширений інсайт щодо функціональності, поведінки та потенційних загроз, пов'язаних з ТПЗ.

У динамічному аналізі використовуються такі основні методи для перевірки стану ІС під час або після активності ТПЗ. Аналіз файлової системи дозволяє виявити зміни, що стосуються файлів, які використовує ТПЗ. Аналіз процесів в системі допомагає відстежити активність програм і процесів, пов'язаних з ТПЗ. Порівняння знімків реєстру дозволяє виявити зміни, що стосуються системного реєстру після активності ТПЗ. При необхідності також здійснюється моніторинг мережевої активності, який включає перегляд DNS-запитів, прослуховування портів, перехоплення та аналіз трафіку. Налагодження є ще одним прийомом, який допомагає отримати більш детальні результати. Що дозволяє аналізувати окремі частини програми під час її виконання для пошуку помилок і отримання загального уявлення про принципи її роботи. У динамічному аналізі використовуються автоматичні інструменти, такі як пісочниці, які виконують динамічний аналіз і надають звіт про дані, отримані в результаті цього аналізу. Однак варто зазначити, що аналіз в пісочниці має свої обмеження, такі як неможливість повністю налаштувати середовище і обмежена повнота отриманих результатів. Під час проведення динамічного та статичного аналізу існує широкий вибір інструментів, які мають свої аналоги. Вибір конкретного інструменту є індивідуальним для кожного окремого аналізу ТПЗ.

3 РОЗРОБКА РЕКОМЕНДАЦІЙ ЩОДО ПРОВЕДЕННЯ АНАЛІЗУ ТРОЯНСЬКОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Розробка рекомендація щодо статичного та динамічного аналізу троянського програмного забезпечення

Аналіз ТПЗ є складним та багатоетапним процесом, що включає різноманітні засоби та методи. Проте у деяких випадках необґрунтовано використовувати всі можливості аналізу. Перш за все, на початку аналізу необхідно чітко визначити цілі, які потрібно досягти за допомогою аналізу ТПЗ. Мета аналізу на загальному рівні визначатиме необхідний рівень розуміння ТПЗ, прийнятний час, що можна виділити на аналіз, а також інструменти, які слід використовувати. Аналіз може мати наступні цілі:

- Створення сигнатур.;
- Перевірка наявності інформації про ТПЗ;
- Отримання детальної інформації про поведінку ТПЗ в різних середовищах;
- Отримання детальної інформації про ТПЗ з метою розробки протидійних механізмів;
- Аналіз на потенційні збитки;
- Отримання повної інформації про ТПЗ.

Безпечне середовище є важливим фактором для аналізу ТПЗ. Для створення такого середовища часто використовуються віртуальні машини з обмеженим доступом до мережі або без доступу взагалі.

Незалежно від етапу аналізу, включаючи статичні методи, важливо забезпечити безпечне середовище. Це необхідно з огляду на потенційні загрози, пов'язані з людським фактором. Якщо ТПЗ буде запущено на основній системі, це може призвести до пошкоджень, втрати даних та інших негативних наслідків. Для забезпечення безпеки, система, на якій розташована віртуальна лабораторія для

аналізу, повинна бути розміщена у відокремленому сегменті мережі, щоб запобігти поширенню ТПЗ поза цю систему. Доступ до Інтернету повинен бути наданий ТПЗ лише після детальної оцінки можливих наслідків та у випадку, коли інші методи не забезпечують необхідну інформацію. Важливо також слідкувати за мережевою активністю ТПЗ. Зв'язок між віртуальною машиною і системою, що її підтримує, повинен бути обмеженим. Підключення носія до віртуальної машини може створити значний ризик порушення ізоляції ТПЗ, тому такий підхід не рекомендується.

Під час проведення динамічного аналізу, захищене середовище виконання повинно відповідати робочому середовищу ТПЗ (Системи процесів управління). Це означає, що воно повинно містити операційну систему, фреймворки та програмні засоби, які необхідні для коректної роботи ТПЗ. Для створення стандартного середовища можна використовувати систему з такими конфігураційними параметрами:

Таблиця 3.1

Стандартне середовище функціонування ТПЗ

Тип	Версія
Операційна система	Windows 7
Архітектура	64-бітна
Наявність DLL	Станадартні
Мережа	Емуляції доступу до глобальної мережі
Антивірусні ПЗ	Відстутні
Програмні засоби та фреймвокри	Пакет Microsoft Office 2010, Adobe Flash Player 27, Microsoft Visual C++ 2008 - Microsoft Visual C++ 2017, Java 8 Update, веб-браузер

Це середовище є підходящим для більшості сучасних ТПЗ і часто використовується в пісочницях. Це пояснюється тим, що значна частина ТПЗ

призначена для операційної системи Windows 7 [15]. Інші елементи, згадані вище, також часто зустрічаються у системах інформаційної безпеки на цій ОС. Ще одним важливим параметром є уникнення виявлення слідів віртуалізації. Сучасні ТПЗ можуть вміщувати в собі вбудовані функції для виявлення таких слідів. Якщо такі сліди будуть виявлені, ТПЗ може відмовитися від запуску, що унеможливить проведення динамічного аналізу. Слідами віртуалізації можуть бути процеси, параметри системи, назви пристроїв та інше. По-перше, слід перевірити системні параметри. Часто віртуальні машини мають обмежену кількість ядер процесора, обмежену оперативну пам'ять та невеликий об'єм дискового простору для роботи. Це створює нереалістичне виконавче середовище. Для проведення аналізу ТПЗ, віртуальна машина повинна мати параметри, які відповідають характеристикам широко використовуваних систем. Ці параметри включають, принаймні, 2 гігабайти оперативної пам'яті, 2 ядра процесора і 60 гігабайтів дискового простору.

Останнім важливим аспектом підготовки середовища є створення точок відновлення. Під час динамічного аналізу ТПЗ активна робота системи може призвести до її пошкодження або необхідності повторного проведення динамічного аналізу для збору повної інформації. У таких випадках важливо мати можливість швидкого відновлення системи до стану, що передував початку динамічного аналізу. Створення точок відновлення дозволяє ефективно відновлювати систему і продовжувати роботу. Для створення точок відновлення використовується програмне забезпечення віртуалізації, оскільки цей підхід є швидким і ефективним. Використання точок відновлення дозволяє зекономити час на повторну налаштування лабораторії та відновлення її до робочого стану. Рекомендується створювати точку відновлення після завантаження всіх необхідних інструментів аналізу до системи [16].

Правильний підбір інструментів для аналізу є важливою складовою успішного його проведення. Перш за все варто взяти до уваги універсальність інструментів. Витрата часу і отримання непотрібної інформації мають негативний вплив на ефективне використання результатів аналізу ТПЗ. Для ілюстрації можна порівняти

результати використання утиліти Strings і спеціалізованого програмного інструменту PEStudio для аналізу ТПЗ.

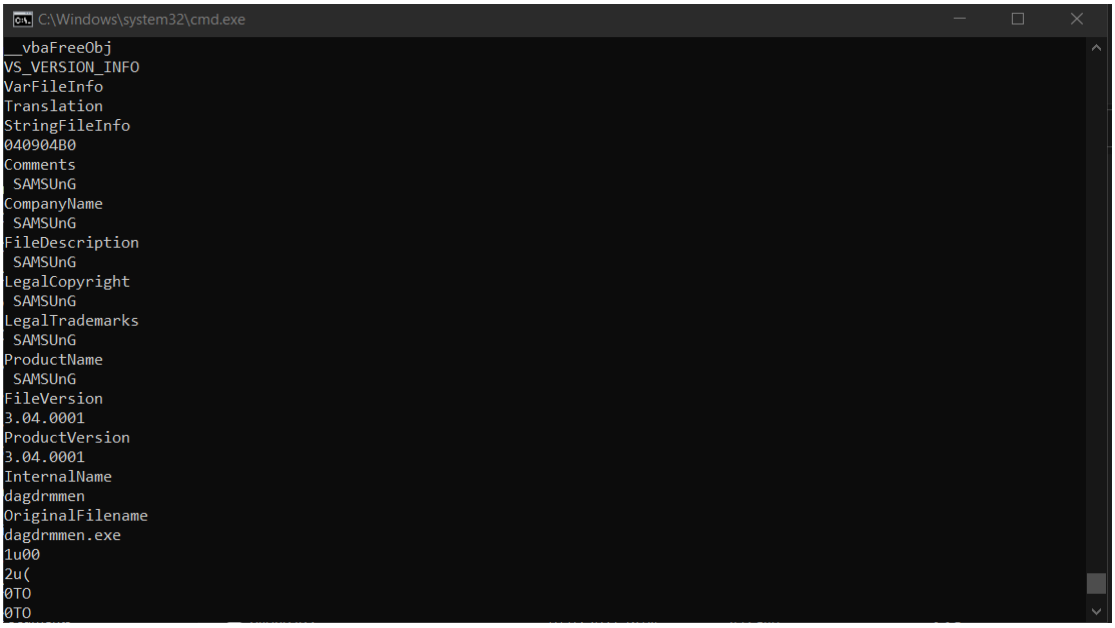


Рис 3.1. Аналіз утилітою Strings

Під час використання утиліти, порівняно зі спеціалізованим програмним забезпеченням, виникають деякі проблеми. Наприклад, утиліта виводить рядки без сортування, включаючи некорисну інформацію, що ускладнює аналіз. Крім того, двійковий аналіз файлу не завершено. Для отримання повних даних необхідно користуватися кількома додатковими програмними засобами. У той же час спеціалізоване програмне забезпечення не має цих недоліків.

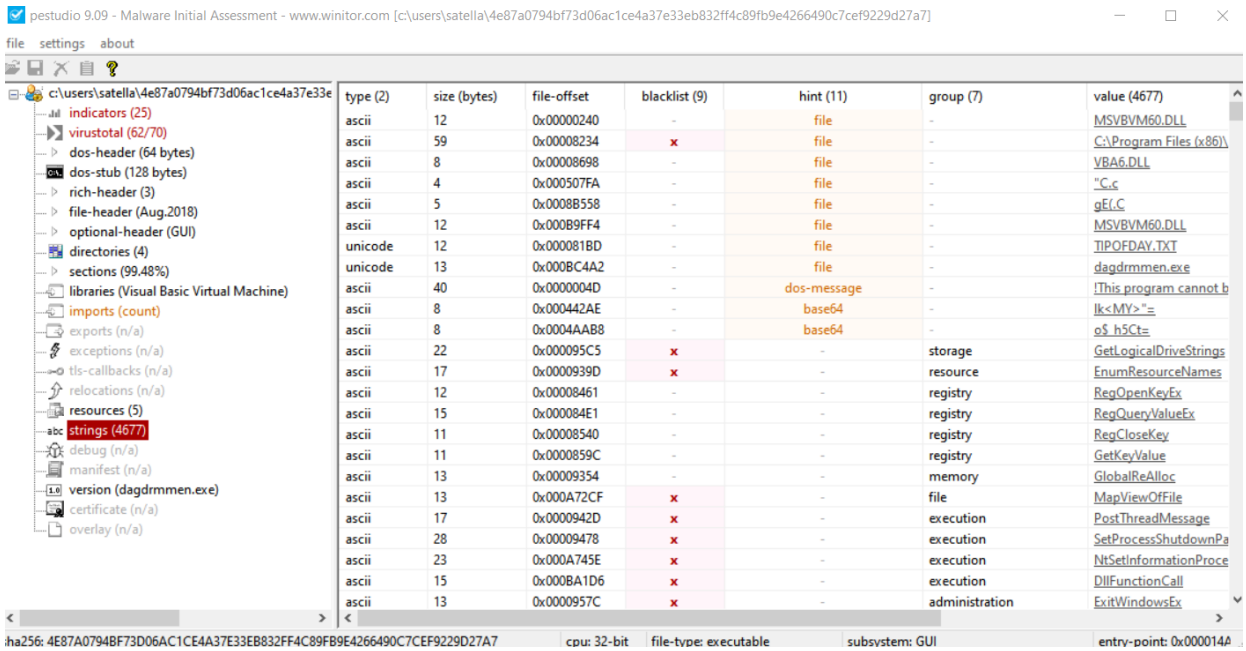


Рис 3.2. Аналіз за допомогою PEStudio

Крім виведення друкованих рядків з PE-файлу, спеціалізоване програмне забезпечення надає додаткову інформацію, структурує результати та має інший функціонал, пов'язаний з бінарним аналізом. Використання універсальних інструментів дозволяє зекономити ресурси та зменшити навантаження від непотрібних даних. Цей підхід може значно оптимізувати процес аналізу ТПЗ. Варто, однак, звернути увагу на конкретний функціонал програмного забезпечення. У деяких випадках інструменти з схожими функціями можуть надавати різні види даних, що забезпечує більше результатів. Цей підхід є ефективним як для статичного, так і для динамічного аналізу.

Для аналізу процесів в системі потрібно використовувати кілька інструментів. Перш за все, потрібне програмне забезпечення для моніторингу конкретних процесів і потоків у режимі реального часу, щоб відстежувати поведінку ТПЗ. Другий інструмент повинен фіксувати всі зміни файлової системи та реєстру. Комбінуючи інформацію з обох інструментів, можна отримати повні дані про поведінку ТПЗ, які доступні при використанні основних методів динамічного аналізу, без необхідності моніторингу мережі.

Для оптимізації аналізу ТПЗ важливо визначити послідовність етапів, що використовуються. Правильна послідовність дозволяє отримати повну інформацію про ТПЗ та ефективно використовувати кожен з методів. Першим кроком є використання автоматичних засобів, які швидко надають загальне уявлення про ТПЗ і створюють основу для подальших методів. Автоматичні засоби включають антивірусне програмне забезпечення, пісочниці та спеціальні сервіси для аналізу ТПЗ. Інформація, отримана від автоматичних засобів, може бути достатньою для досягнення аналітичних цілей. Після використання автоматичних засобів, слід провести статичний аналіз за допомогою основних методів. Цей етап включає вказівку типу файлу та необхідний аналіз, який залежить від типу файлу, що аналізується. Статичний аналіз надасть основну інформацію про ТПЗ та його оточення, якщо ця інформація не була отримана автоматичними засобами. Наступним кроком є динамічний аналіз для виявлення додаткових особливостей ТПЗ та визначення його поведінки в різних середовищах. Орієнтуючись на інформацію, отриману під час динамічного аналізу, можна розпочати зворотну розробку файлу, щоб отримати точну інформацію про принципи роботи ТПЗ. Варто враховувати, що навіть якщо автоматичний аналіз надав достатньо інформації, рекомендується пройти через наступні етапи аналізу, якщо є достатньо часу та відповідні ресурси, щоб підтвердити та розширити результати.

Таким чином формується наступна послідовність:

Рис 3.3. Етапи проведення аналізу ТПЗ

Ця послідовність аналізу є ефективною, оскільки кожен крок надає необхідну інформацію для наступного. Проте, важливо враховувати, що злочинці також використовують методи протидії аналізу ТПЗ, що може негативно вплинути на результати окремих методів аналізу. Покроковий підхід дозволяє отримати дані, які неможливо отримати за допомогою однієї методики. Тому ігнорування статичного або динамічного аналізу може призвести до втрати результатів, які можна було б використати в майбутньому. Особливо це стосується методів обфускації та упаковки ТПЗ, які ускладнюють бінарний аналіз, декомпіляцію та дезасемблінгу.

Кожен крок аналізу повинен бути належним чином задокументований. Документація має включати результати аналізу, висновки, отримані на основі цих результатів, а також припущення, що стосуються ТПЗ. За допомогою документально підтверджених даних формується загальний висновок щодо ТПЗ.

Документація результатів автоматичного аналізу повинна містити наступну

інформацію:

- Основні дані про ТПЗ, отримані під час використання автоматичних засобів;
- Початкові припущення або висновки, якщо отримана інформація відповідає цілям аналізу;

Документація результатів основного статичного аналізу повинна містити такі елементи:

- Дані, отримані під час аналізу файлу ТПЗ, за винятком незначущої інформації для аналізу ТПЗ;
- Інформацію про середовище, в якому працює ТПЗ, якщо це не було визначено на попередньому етапі.;
- Припущення щодо поведінки ТПЗ в робочому середовищі;
- Висновок на підставі результатів аналізу.

Документація результатів динамічного аналізу має включати таку інформацію:

- Дані про поведінку ТПЗ в робочому середовищі;
- Дані про мережеву активність;
- Висновок на підставі отриманих результатів.

Наступна документація має описувати отриману під час зворотної розробки інформацію про принципи роботи ТПЗ та містити висновки. Важливо відзначити, що визначення даних, які не мають цінності для аналізу, залежить від експертної оцінки спеціаліста з аналізу ТПЗ.

3.2. Практичне використання рекомендації щодо використання статичних та динамічних методик аналізу

Відповідно до рекомендацій, що виникли під час проведення дослідження, для аналізу був обраний зразок відомого ТПЗ. Головною метою цього аналізу є визначення даних для створення підпису та аналізу поведінки ТПЗ під час його запуску в відповідному середовищі. З метою досягнення максимально ефективного аналізу, ми розпочнемо з автоматичного аналізу. В якості інструментів для цього

аналізу будемо використовувати сервіс Virus Total та пісочницю Hybrid Analysis. Спочатку ми використаємо Virus Total, щоб підтвердити наявність шкідливого програмного забезпечення та отримати додаткову інформацію.

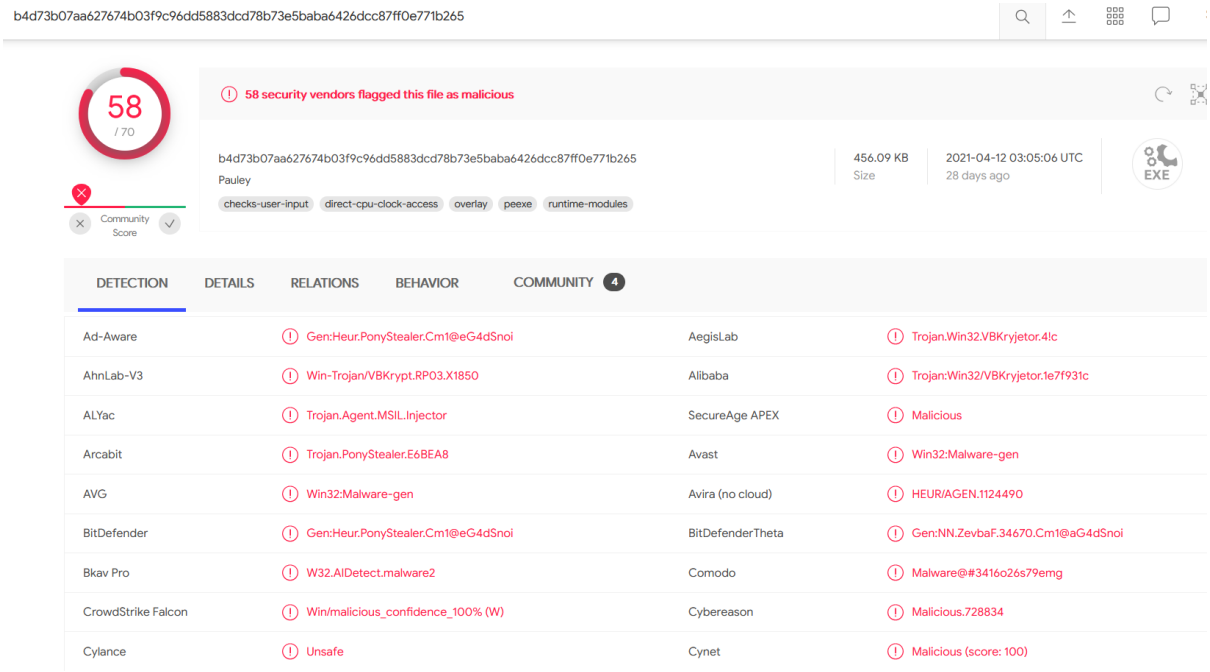


Рис 3.4. Результати аналізу Virus Total

Більшість антивірусних програм підтвердили, що файл є шкідливим і належить до троянських програм. Крім того, було обчислено хеш-суму файлу, яку можна використати як підпис, і отримані результати бінарного аналізу. Далі перейдемо до аналізу в пісочниці, щоб отримати додаткову інформацію. Для цього використаємо пісочницю зі середовищем Windows 7.

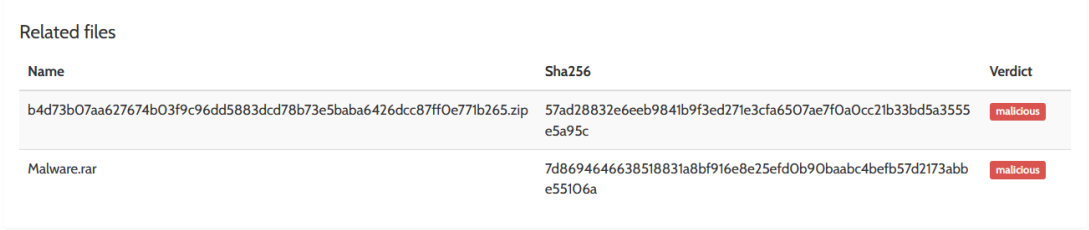


Рис 3.5. Результати аналізу через пісочницю : Пов’язані файли

Пісочниця виділила два шкідливих файли, які були пов’язані з атакою. Ці дані

можна віднести до результатів аналізу для подальшого використання.

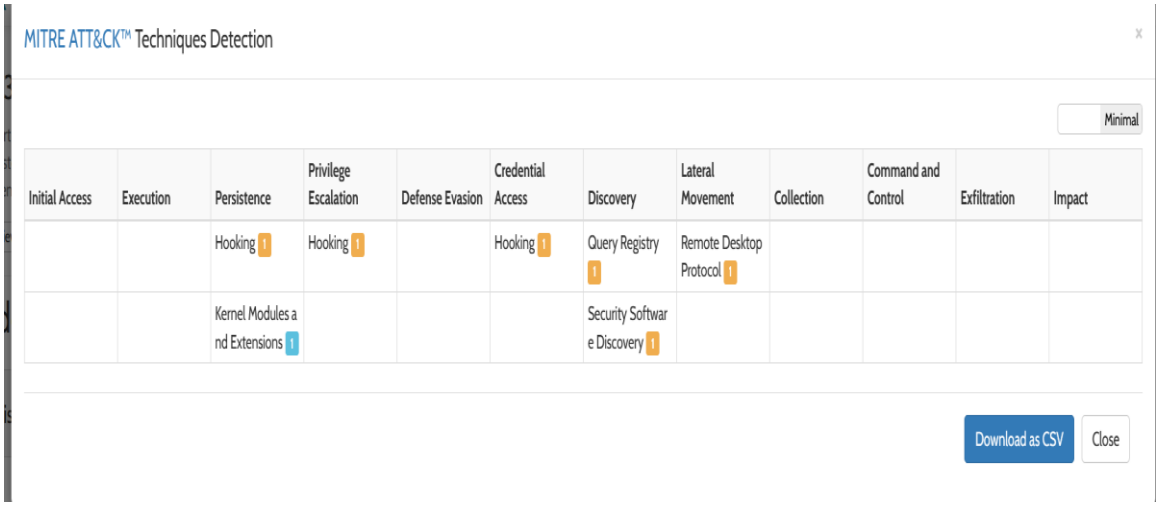


Рис 3.6. Результати аналізу через пісочницю : Шкідливі функції ТПЗ

Як бачимо з отриманих даних під час аналізу в пісочниці, це конкретне ТПЗ спрямовує свої дії на отримання привілегій в системі з метою виконання функцій бекдору та шпигунського програмного забезпечення. Крім того, була надана інформація про тип файлу - PE. Загалом, під час автоматичного аналізу були отримані такі результати: підтвердження шкідливості файлу, хеш-сума файлу, інші файлів, пов'язаних з атакою, а також виявлені шкідливі функції файлу.

Отримана інформація достатня для досягнення цілей аналізу, проте для підтвердження та отримання більш детальної карти ТПЗ необхідно пройти додаткові етапи аналізу. З урахуванням відомого середовища та типу файлу, ми переходимо до основного статичного та динамічного аналізу самого файлу.

Для проведення динамічного аналізу файлу необхідно створити віртуальну машину, яка відповідає параметрам, зазначеним у таблиці 3.1. Вона має два ядра та 4 гігабайти оперативної пам'яті, а також 100 гігабайт вільного дискового простору. Віртуальна машина буде ізольована від мережі, а для перевірки мережевої активності використовуватиметься емуляція мережі. У цьому середовищі також буде проведено статичний аналіз.

Оскільки тип файлу є PE, основним інструментом для динамічного аналізу буде

PEStudio. Спочатку розглянемо друковані рядки, що містяться в ТПЗ.

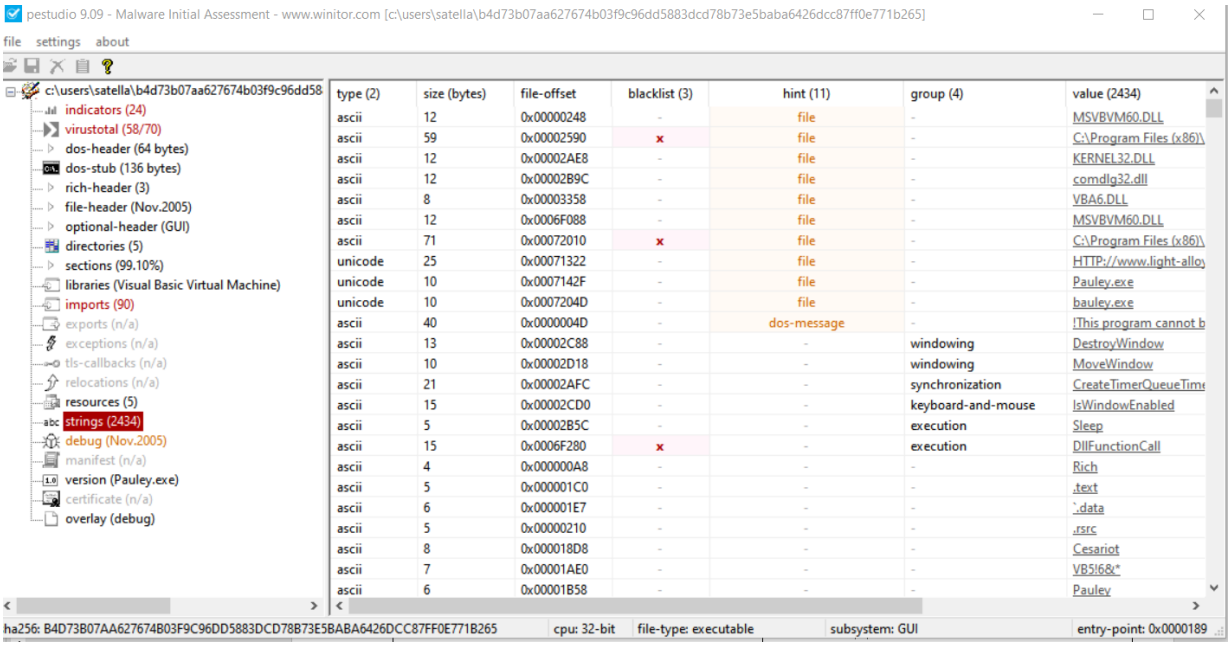


Рис 3.7. Аналіз друкованих рядків

Для повного урахування отриманих результатів необхідно задокументувати виділені рядки, які можуть бути важливими для аналізу. Багато з результатів мають незрозумілу послідовність символів, що не має значення. Також є рядки, які, незважаючи на чіткий формат, не надають додаткової інформації і мають характер коментарів. У таблиці 3.2 наведений перелік рядків, які можна використовувати в аналізі. В таблиці показано лише певні функції, які присутні у друкованих рядках.

Таблиця 3.2

Список друкованих рядків отриманих з PE-файлу

Рядок	**Примітки**
MSVBVM60.DLL	Динамічна бібліотека
C:\Program Files (x86)\Microsoft Visual Studio\VB98\VB6.OLB	Посилання на файл
comdlg32.dll	Динамічна бібліотека

Продовження таблиці 3.2

Список друкованих рядків отриманих з PE-файлу

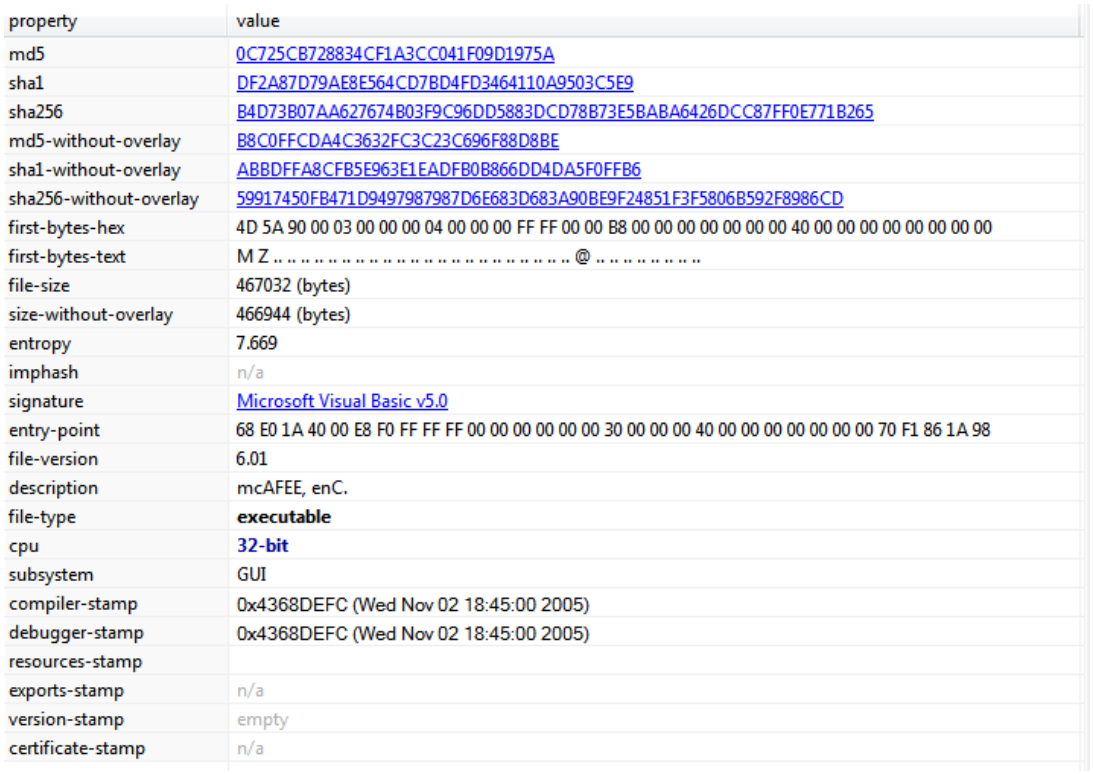
Рядок	**Примітки**
DestroyWindow	Функція ОС
IsWindowEnabled	Функція ОС
VBA6.DLL	Динамічна бібліотека
__vbaRecDestruct	Функція ОС
__vbaNew2	Функція ОС
__vbaLenVar	Функція ОС
__vbaR8Sgn	Функція ОС
_adj_fptan	Функція ОС
HTTP://www.light-alloy.ru	Доменна ім’я
Pauley.exe	Коментар з назвою файлу
C:\Program Files (x86)\AdministratorWIN-46S82Q2PKTK\Projects\Pauley.pdb	Посилання на файл

Після документування результатів можна перейти до більш детального аналізу. Перш за все, варто звернути увагу на проблему динамічних бібліотек. У цьому об'єкті аналізу дві викликані бібліотеки (MSVBVM60.DLL і VBA6.DLL), пов'язані з роботою віртуальної машини Visual Basic. Представлені функції належать до цих динамічних бібліотек.

Також є два посилання на файлову систему, одне з них згадує Microsoft Visual Studio. Інше посилання стосується формату файлу PDB, який використовується для зберігання налагоджувальної інформації.

Останнім цікавими результатами є доменне ім'я, яке потрібно перевірити під час подальшого аналізу, та ім'я виконуваного файлу, яке, ймовірно, представляє оригінальне ім'я об'єкта аналізу. Далі можна розглянути загальну інформацію про

файл PE. Оскільки PEStudio є універсальним інструментом, немає потреби використовувати інше програмне забезпечення для цього завдання.



property	value
md5	0C725CB728834CF1A3CC041F09D1975A
sha1	DF2A87D79AE8E564CD7BD4FD3464110A9503C5E9
sha256	B4D73B07AA627674B03F9C96DD5883DCD78B73E5BABA6426DCC87FF0E771B265
md5-without-overlay	B8C0FFCDA4C3632FC3C23C696F88D8BE
sha1-without-overlay	ABBDFFA8CFB5E963E1EADFB0B866DD4DA5F0FFB6
sha256-without-overlay	59917450FB471D9497987987D6E683D683A90BE9F24851F3F5806B592F8986CD
first-bytes-hex	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00
first-bytes-text	M Z .. .. @ .. ..
file-size	467032 (bytes)
size-without-overlay	466944 (bytes)
entropy	7.669
imphash	n/a
signature	Microsoft Visual Basic v5.0
entry-point	68 E0 1A 40 00 E8 F0 FF FF FF 00 00 00 00 00 00 30 00 00 00 40 00 00 00 00 00 00 00 70 F1 86 1A 98
file-version	6.01
description	mcAFEE, enC.
file-type	**executable**
cpu	**32-bit**
subsystem	GUI
compiler-stamp	0x4368DEFC (Wed Nov 02 18:45:00 2005)
debugger-stamp	0x4368DEFC (Wed Nov 02 18:45:00 2005)
resources-stamp	
exports-stamp	n/a
version-stamp	empty
certificate-stamp	n/a

Рис 3.8. Загальна інформація та хеші об’єкту аналізу

Один з хеш-форматів ТПЗ вже був відомий, а значення для аналізу представлених даних надає інформацію про підсистему GUI та підпис Visual Basic. Крім того, з даних видно, що ТПЗ був створений для 32-розрядної архітектури. Цікавим параметром є ентропія файлу. Високе значення ентропії може свідчити про можливу обфускацію, що треба враховувати при використанні методів зворотного проектування. Усю цю інформацію слід задокументувати. Наступним кроком є перегляд ресурсів, які завантажує PE-файл, а також розгляд деяких додаткових параметрів.

type (3)	name	file-offset (5)	signature (3)	non-standard	size (3476 bytes)	file-ratio (0.74%)
version	1	0x00071150	version	-	748	0.16 %
icon-group	1	0x0007143C	icon-group	-	48	0.01 %
icon	30001	0x0007187C	icon	-	1640	0.35 %
icon	30002	0x00071754	icon	-	296	0.06 %
icon	30003	0x0007146C	icon	-	744	0.16 %

Рис 3.9. Ресурси PE-файлу

Файлові ресурси містять інформацію про версію та групу значків, але ці дані не є цінними для аналізу і не потребують документування. Їх можна витягнути з PE-файлу для окремого розгляду, але в даному випадку це необов'язково.

Для завершення базового статичного аналізу перейдемо до розгляду індикаторів - це властивості PE-файлу, які саме програмне забезпечення визначило як підозрілі.

xml-id	indicator (24)	detail
1430	The file references string(s) tagged as blacklist	count: 3
1120	The file is scored by virustotal	score: 58/70
1266	The file imports symbol(s) tagged as blacklist	count: 2
1262	The file imports anonymous function(s)	count: 18
1055	The file runs in the Visual Basic Virtual Machine (VBVM)	status: yes
1019	The file contains a rich-header	status: yes
1424	The original name of the file has been found	name: Pauley.exe
1152	The file references debug symbols	file: c:\program files (x86)\a
1634	The file references a group of API	api: synchronization, count:
1634	The file references a group of API	api: execution, count: 2
1634	The file references a group of API	api: windowing, count: 2
1634	The file references a group of API	api: keyboard-and-mouse, c
1633	The file references a group of hint	hint: dos-message, count: 1
1633	The file references a group of hint	hint: file, count: 10
1633	The file references a group of hint	hint: url-pattern, count: 0

Рис 3.10. Індикатори PE-файлу

Серед цінних для аналізу індикаторів можна виділити наявність підозрілих рядків, виклик API для клавіатури та миші, імпортування підозрілих символів. Загалом з даних, які б доповнили результати автоматичного аналізу можна виділити

посилання на доменне ім’я, використання DLL-файлів пов'язаних з Visual Basic (подібну інформацію можна виділити як додаток до сигнатури файлу), високу ентропію, що вказує на обфускацію та індикатори, що вказують на приналежність ТПЗ до типу кейлогерів.

Під час динамічного аналізу припускається, що мережевий ТПЗ буде використовуватись для передачі отриманої інформації та дистанційного керування системою. З урахуванням досягнутих цілей аналізу, проводитиметься поверхневе спостереження за поведінкою ТПЗ. Для детального перегляду процесів ТПЗ буде використано інструмент Process Hacker.

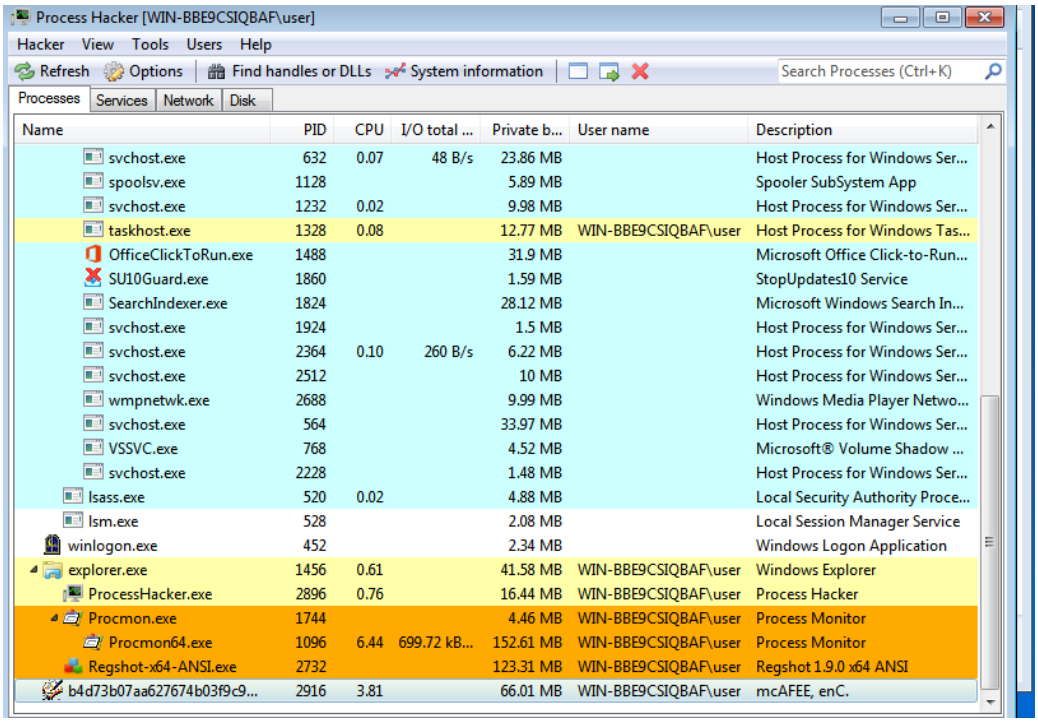


Рис 3.11. Процес ТПЗ

ТПЗ успішно запущено і працює в системі, що підтверджує правильний вибір середовища для його виконання. Не було запущено жодних додаткових процесів. Для отримання додаткової інформації необхідно переглянути деталі процесу.

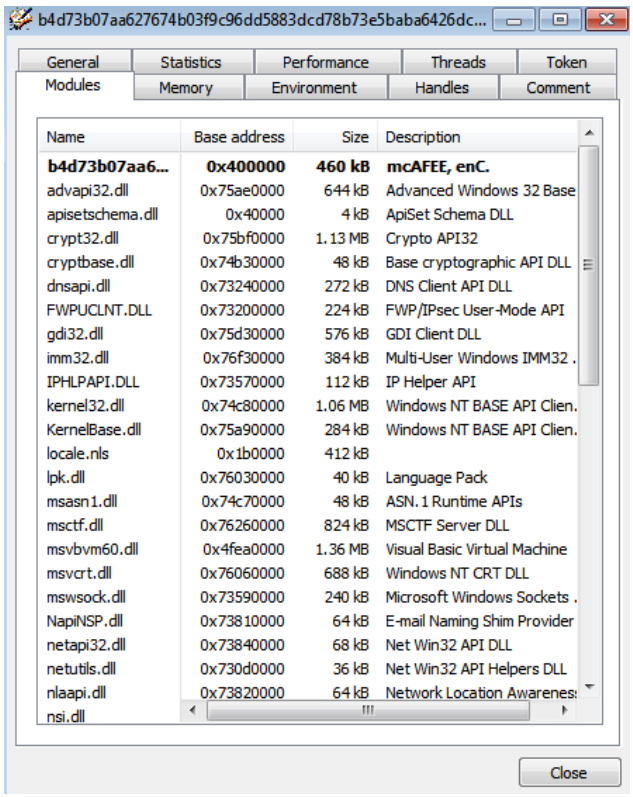


Рис 3.12. Модулі процесу ТПЗ

З інформації, яку надає Process Hacker, видно, що під час виконання ТПЗ використовуються додаткові DLL, які не були вказані в рядках PE-файлу. Серед цих DLL можна виділити netapo32.dll для роботи з мережею та crypt32.dll, що містить криптографічні функції. Усі бібліотеки DLL, які використовуються процесом, слід додати до документації щодо поведінки мережі ТПЗ для подальшого розгляду.

Наявність мережевих DLL чітко вказує на те, що ТПЗ використовує мережу у своїй діяльності, що підтверджує припущення про його тип як бекдора з додатковими шпигунськими функціями, призначеним для дистанційного керування системою. Якщо віртуальна лабораторія не дотримувалася рекомендацій щодо організації безпечного середовища для інформаційної системи, можливі порушення конфіденційності інформації. Для отримання більш детальної інформації про ТПЗ рекомендується використовувати інше програмне забезпечення для моніторингу процесів в системі, таке як Process Monitor. Це допоможе отримати відомості про

взаємодію з мережею, файловою системою та реєстром. Важливим фактором у цьому випадку є використання фільтрації для ефективного відбору інформації, що стосується ТПЗ, з потоку подій. Без використання фільтрації було б практично неможливо знайти потрібні дані серед обсягу інформації. У даному випадку буде використана проста фільтрація, застосовуючи правила, що відбирають події з назвою, пов'язаною з ТПЗ.

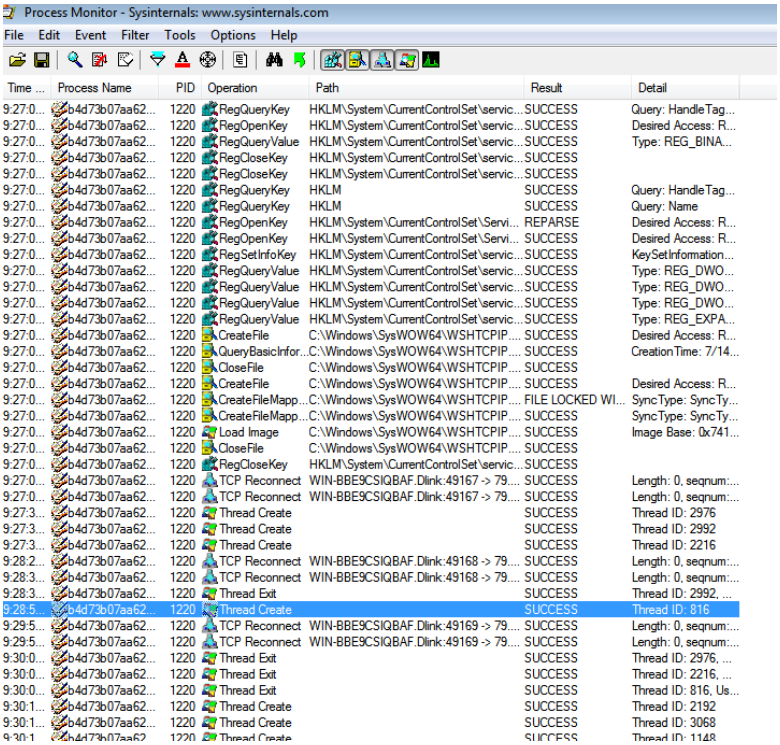


Рис 3.13. Перегляд процесів через Process Monitor

За допомогою Process Monitor ви можете відстежувати активність ТПЗ в системі, включаючи створення файлів та взаємодію з реєстром. Інструмент також підтвердив припущення про мережеву активність ТПЗ. Щоб завершити динамічний аналіз, варто перевірити, чи ТПЗ запускається разом з операційною системою. Для цього використовується утиліта Autoruns, яка виявляє програми, які автоматично запускаються під час завантаження системи. Автозавантаження є типовим для ТПЗ, тому огляд цієї інформації може дати додаткові дані і допомогти виявити інші процеси, які були налаштовані на автоматичне запуск без відома користувача.

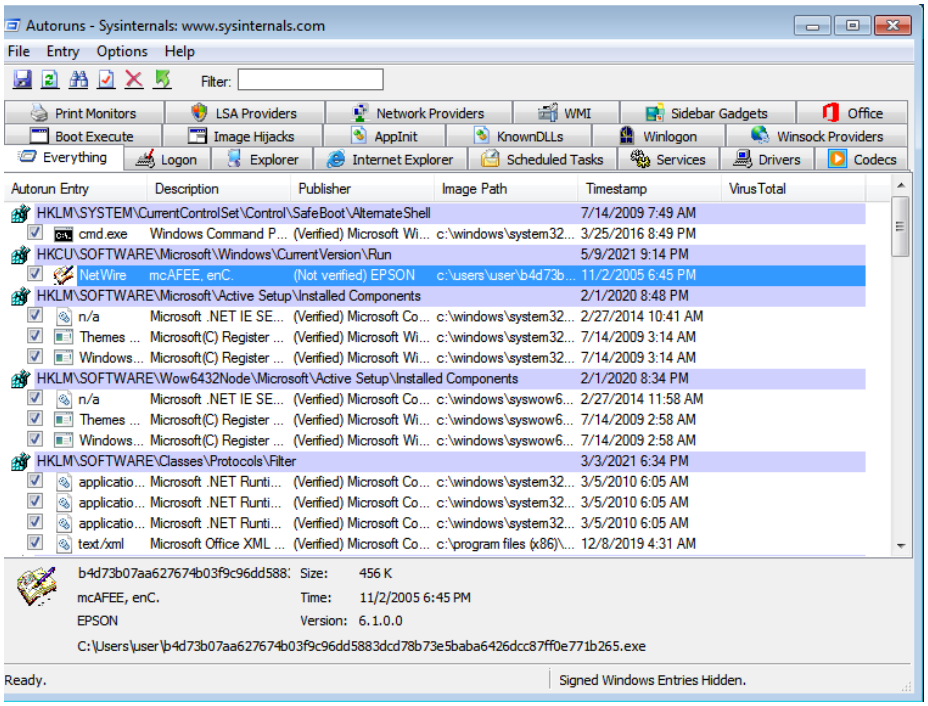


Рис 3.14. Перевірка ТПЗ на автозавантаження

Під час перевірки через Autoruns було підтверджено, що ТПЗ працюватиме разом з операційною системою, але жодних інших підозрілих елементів в списку автозавантаження не було виявлено.

Під час динамічного аналізу були виявлені основні особливості поведінки ТПЗ під час його функціонування. Було додано до даних список DLL-файлів, які ТПЗ використовує під час роботи, і підтверджено його належність до категорії кейлогерів, бекдорів і троянських програм. Інформація, отримана під час автоматичного аналізу, узгоджується з даними, отриманими на попередніх етапах, але без цих даних було б складно отримати такий рівень деталізації щодо поведінки ТПЗ в системі. Загальна інформація з усіх етапів аналізу дозволяє зробити остаточні висновки щодо ТПЗ та задокументувати їх.

На основі всієї отриманої в ході аналізу інформації можна виділити наступні характеристики ТПЗ :

Таблиця 3.3

Характеристики ТПЗ

Характеристика	Значення
Тип	Бекдор, кейлогер, троянська програма
Середовище	Операційні системи сімейства Windows
Мережева активність	Наявна
Особливості	Використання віртуальної машини Visual Basic
Протидія аналізу ТПЗ	Обфускація PE-файлу
Хеш (SHA256)	b4d73b07aa627674b03f9c96dd5883dcd78b73e5baba6426dcc87ff0e771b265

У результаті успішного аналізу ТПЗ була отримана повна інформація, необхідна для досягнення поставлених цілей. Використовуючи розроблені рекомендації, вдалося отримати всю необхідну інформацію про ТПЗ. Важливо зауважити, що під час функціонування ТПЗ не виникли пошкодження операційної системи і його поширення за межі віртуальної машини. Зібрані дані можуть бути використані в майбутньому для розробки захисних механізмів проти цього конкретного зразка ТПЗ. Результати статичного та динамічного аналізу можуть служити як сигнатури для виявлення його активності в області інформаційної безпеки. Проте, для цього необхідно детально проаналізувати отримані результати та визначити загальну сигнатуру поведінки ТПЗ, яка була б унікальною для нього. Зазначено, що зворотне проектування, таке як декомпіляція або розбирання коду ТПЗ, не було потрібним для досягнення цілей аналізу. Документована інформація може бути використана в майбутньому для продовження аналізу, якщо це буде потрібно.

Висновки до розділу 3

В ході проведення досліджень були розроблені рекомендації щодо оптимізації та безпечного проведення аналізу ТПЗ. Виявлено, що для ефективного аналізу необхідно чітко визначити цілі аналізу, на підставі яких обираються відповідні інструменти та методи. Згідно з отриманими результатами, найбільш ефективним і безпечним методом аналізу ТПЗ є використання автоматичних засобів або середовища віртуальної машини з обмеженими мережевими підключеннями. При використанні віртуальної машини важливо налаштувати параметри таким чином, щоб вони відповідали сучасним методам виявлення віртуалізації. Для оптимізації процесу аналізу також рекомендується створити точки відновлення, які дозволять повернутися до попереднього стану системи у разі необхідності.

Під час аналізу найбільш ефективним підходом є використання універсальних інструментів, які можуть виконувати кілька завдань одночасно. Це дозволяє забезпечити ефективність аналізу і швидше виявляти специфічні характеристики ТПЗ. Іноді для отримання повної інформації необхідно використовувати кілька інструментів з подібним функціоналом.

Оптимальна послідовність етапів аналізу, яка забезпечує ефективність процесу, включає автоматичний аналіз, базовий статичний аналіз, динамічний аналіз та зворотне проектування. Кожен етап має свої особливості, і дані, отримані на кожному етапі, можуть впливати на вибір методів і інструментів наступного етапу. Важливо докладно задокументувати всі етапи аналізу, включаючи дані, висновки та припущення щодо ТПЗ. Така документація допоможе зберегти і систематизувати отриману інформацію для подальшого використання.

В результаті проведення аналізу відомого ТПЗ з використанням розроблених рекомендацій підтвердилися їх ефективність і вдале досягнення цілей аналізу на послідовних етапах.

ВИСНОВКИ

Під час виконання бакалаврської роботи отримано наступні науково-практичні результати:

1) Встановлюються основні характеристики ТПЗ, які враховуються під час аналізу, такі як операційне середовище, тип програмного забезпечення, його приналежність до певного сімейства програмного забезпечення, методи розповсюдження та різні специфічні характеристики, якщо вони є доступний. Це дозволить значно підвищити ефективність аналізу ТПЗ та підібрати правильний інструментарій для аналізу.

2) Досліджено два основних методи проведення аналізу ТПЗ: статичний та динамічний. Статичний аналіз виконується без запуску самого ТПЗ, а динамічний аналіз передбачає спостереження за поведінкою ТПЗ в певному середовищі. За результатами дослідження було встановлено, що через обмеження кожного методу окремо для отримання повної інформації про ТПЗ необхідно використовувати обидва методи, доповнюючи дані, отримані під час застосування одного методу, даними, отриманими за допомогою іншого методу.

3) Підлягаючи аналізу, були визначені основні переваги та недоліки статичного та динамічного методів аналізу ТПЗ. Також було досліджено можливості основних інструментів, що використовуються для проведення аналізу ТПЗ.

4) На основі проведеного дослідження були сформульовані рекомендації щодо оптимізації процесу аналізу ТПЗ. Ці рекомендації включають в себе вказівки щодо середовища, у якому слід проводити аналіз, його параметрів та особливостей роботи, вибору оптимальних інструментів, послідовності етапів виконання аналізу ТПЗ, а також рекомендації щодо документування отриманих результатів.

5) Було успішно проведений аналіз відомого зразка ТПЗ. Під час аналізу були досягнуті всі поставлені цілі, а сама інформаційна система не зазнала жодних пошкоджень від об'єкта аналізу, на якому проводилися дослідження.

ПЕРЕЛІК ПОСИЛАНЬ

1. Sihwail R., Khairuddin O., Akram K. A Survey on Malware Analysis Techniques: Static, Dynamic, Hybrid and Memory Analysis. International Journal on Advanced Science, Engineering and Information Technology. 2021. No.8. P.1664.
2. Sikorski M. Practical Malware Analysis / M. Sikorski, A. Honig. – San Francisco: No Starch Press, 2012. 733 с.
3. A M. K. Learning Malware Analysis: Explore the concepts, tools, and techniques to analyze and investigate Windows malware: Packt Publishing, 2018. 510 p.
4. Duncan R., Schreuders Z. C. Security implications of running windows software on a Linux system using Wine: a malware analysis study. Journal of Computer Virology and Hacking Techniques. 2018. Vol. 15, no. 1. P. 39–60.
5. Koret J., Bachaalany E. Antivirus Hacker's Handbook: Wiley & Sons, Incorporated, John, 2015. 384 p.
6. Касперский Е. В. Современная антивирусная индустрия и её проблемы [Електронний ресурс] Securelist. – 2005. – Режим доступу: https://securelist.ru/sovremennaya-antivirusnaya-industriya/711/.
7. K.Yakdan, S. Dechand, E. Gerhards-Padilla, E. Smith. Helping Johnny to Analyze Malware: A Usability-Optimized Decompiler and Malware Analysis User Study. 2016. 164–166 p.
8. Eagle C. The IDA Pro Book. No Starch Press, 2008. 656 p.
9. Kaur N., Kumar A. A Complete Dynamic Malware Analysis. International Journal of Computer Applications. 2016. Vol. 135, no. 4. P. 20–25.
10. M.Hale Ligh, S. Adair, B. Hartstein, M. Richard. Malware Analyst’s Cookbook – Indiana: Wiley Publishing, 2011. 183 -185 p.
11. Bencherchali N. Hunting Malware with Windows Sysinternals—Process Monitor 2020. [Електронний ресурс] – Режим доступу: https://nasbench.medium.com/hunting-malware-with-windows-sysinternals-process-monitor-e67476f44514.

12. Fox N. 11 Best Malware Analysis Tools and Their Features 2021. [Електронний ресурс] – Режим доступу до ресурсу: https://www.varonis.com/blog/malware-analysis-tools/.

13. Dang B. Practical Reverse Engineering: X86, x64, ARM, Windows Kernel, reversing tools, and obfuscation. Indianapolis: John Wiley & Sons, Inc., 2014. 355 p.

14. A survey on automated dynamic malware-analysis techniques and tools / M. Egele et al. ACM Computing Surveys. 2012. Vol. 44, no. 2. P. 1–42.

15. Shanhong L. Global market share held by operating systems for desktop PCs, from January 2013 to December 2020 2021. [Електронний ресурс] – Режим доступу: https://www.statista.com/statistics/218089/global-market-share-of-windows-7/

16. Zeltser L. Your 5-Step Malware-Analysis Toolkit 2018. [Електронний ресурс] – Режим доступу до ресурсу: https://campustechnology.com/articles/2010/01/01/your-5-step-malware-analysis-toolkit.aspx

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ (Презентація)