Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
Факультет інформаційних технологій
Кафедра безпеки інформації та телекомунікацій

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня магістра

студента *Залевського Максима Владиславовича*

академічної групи *125-22-1*

спеціальності *125 Кібербезпека*

спеціалізації[1]

за освітньо-професійною програмою *Кібербезпека*

на тему *Методи виявлення та аналізу зловмисного програмного забезпечення в ОС Linux*

Керівники	Прізвище, ініціали	Оцінка за шкалою		Підпис
		рейтинговою	інституційною	
кваліфікаційної роботи	проф. Котух Є.В.			
розділів:				
спеціальний	ас. Мілінчук Ю.А.	85	Добре	
економічний	к.е.н., доц. Пілова Д.П.	92	Відмінно	

Рецензент				

Нормоконтролер	ст. вик. Мєшков В.І.			

Дніпро
2023

ЗАТВЕРДЖЕНО:
завідувач кафедри
безпеки інформації та телекомунікацій
_____ д.т.н., проф. Корнієнко В.І.

«_____»_____2023року

ЗАВДАННЯ
на кваліфікаційну роботу ступеня магістра

студенту *Залевському М.В.* **академічної групи** *125-22-1*
(прізвище та ініціали) (шифр)

спеціальності *125 Кібербезпека*

спеціалізації

за освітньо-професійною програмою *Кібербезпека*

на тему *Методи виявлення та аналізу зловмисного програмного забезпечення в ОС Linux*

Затверджену наказом ректора НТУ «Дніпровська політехніка» від 09.10.23 № 1227-с

Розділ	**Зміст**	**Термін виконання**
Розділ 1	Постановка задачі. Визначення актуальності питання.	10.11.2023
Розділ 2	Методи виявлення та аналізу зловмисного програмного забезпечення в ОС Linux	25.11.2023
Розділ 3	Розрахунок економічних показників запропонованих рішень	02.12.2023

Завдання видано _____ Євген КОТУХ
(підпис керівника) (ім'я, прізвище)

Дата видачі завдання:

Дата подання до екзаменаційної комісії:

Прийнято до виконання _____ Максим ЗАЛЕВСЬКИЙ
(підпис студента) (ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 92с., 23 рис., 2 табл., 4 додатка, 46 джерел.

Об’єкт розробки: операційна система Linux.

Предмет розробки: методи виявлення, аналізу та протидії шкідливому програмному забезпеченню у ОС Linux.

Мета роботи: метою цієї роботи буде вивчення та практичне застосування методів виявлення та аналізу зразків ЗПЗ в операційній системі на базі Linux.

У першій частині розглянуті можливі види атак з використанням шкідливого програмного забезпечення та проаналізовано методи виявлення цього програмного забезпечення з відокремленням їх переваг та недоліків.

У спеціальній частині проаналізовані методи виявлення та аналізу ЗПЗ для ОС Linux на прикладах програмного забезпечення яке використовується у комерційному середовищі провідними фахівцями з кібербезпеки а також комерційні готові рішення для виявлення та протидії ЗПЗ для ОС Linux.

В економічному розділі визначено економічну ефективність впровадження розглянутих комерційних рішень для виявлення та протидії шкідливому програмному забезпеченню для ОС Linux.

Практичне значення роботи полягає в розробці рекомендацій щодо виявлення, аналізу та протидії ЗПЗ для систем під керування ОС Linux.

Результати роботи можуть бути використані для поліпшення безпеки інформації будь-яких установ або звичайних користувачів ПК, які використовуюють ОС Linux.

ШКІДЛИВЕ ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, ОПЕРАЦІЙНА СИСТЕМА LINUX, ДИЗАСЕМБЛІНГ, ДЕКОМПІЛЯЦІЯ, IDS, IPS, ADS, ЗВОРОТНЕ ІНЖЕНЕРУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.

ABSTRACT

Explanatory note: 92 pages, 23 figures, 2 table, 4 appendices, 46 sources.

Object of research: Linux operating system.

Subject of development: methods of malware detection, analysis and prevention for Linux OS.

Objective: study and practical application of malware detection, analysis and prevention for Linux OS based systems.

The first part analyses possible types of attacks on Linux OS with the use of malware and methods of detection of such attacks, separating their advantages and disadvantages.

In the special part, the methods of detection and analysis of malware for Linux OS are analyzed using examples of software used in the commercial environment by leading cyber security specialists, as well as commercial ready-made solutions for detecting and countering malware for Linux OS.

The economic section identifies the cost-effectiveness of implementing the proposed measures to address these vulnerabilities.

The practical significance of the work consists in the development of recommendations for detection, analysis and countermeasures against malware for systems running the Linux OS.

The results of the work can be used to improve the information security of any institutions or ordinary PC users using the Linux OS.

MALWARE, LINUX OPERATING SYSTEM, DISASSEMBLING, DECOMPLILING, IDS, IPS, ADS, SOFTWARE REVERSE ENGINEERING.

СПИСОК УМОВНИХ СКОРОЧЕНЬ

ЗПЗ – зловмисне програмне забезпечення

ОС – операційна система

ПЗ – програмне забезпечення

ADS – Anomaly Detection System (системи виявлення аномалій)

ARP – Address Resolution Protocol (протокол розпізнавання адрес)

C2 – Command and Control Infrastructure (інфраструктура команд та контролю)

DNS – Domain Name System (система доменних імен)

DoS – denial of service (відмова у обслуговувані)

IDS – Intrusion Detection System (система виявлення вторгнень)

IoT – Internet of Thing (інтернет речей)

IPS – Intrusion Prevention System (система запобігання вторгненням)

RaaS - Ransomware as a Service (програми вимагачі як послуга)

SRE – software reverse engineering (зворотне інженерування програмного забезпечення)

UFW – uncomplicated firewall (нескладний брандмауер)

ЗМІСТ

с.

ВСТУП

В сучасному цифровому світі, де інформаційні технології відіграють важливу роль у всіх сферах життя, кібербезпека стає все більш важливою. Зростаюча популярність операційної системи Linux, що базується на відкритому вихідному коді, призвела до появи нових викликів у сфері кібербезпеки, зокрема зростання кількості та складності зловмисного програмного забезпечення (ЗПЗ), спрямованого на цю платформу.

Метою цієї роботи буде вивчення та розробка методів виявлення та аналізу зразків ЗПЗ в операційній системі на базі Linux. Ціль буде полягати у тому, щоб знайти та перевірити на практиці нові ефективні методи та інструменти виявлення різних видів ЗПЗ, які можуть бути використані при атаках на системи під керування ОС Linux та засобів протидії цім атакам.

Об’єктом дослідження буде безпека та архітектура операційної системи Linux, а саме найпопулярнішого її дистрибутиву на базі Debian – Ubuntu. Дослідження буде стосуватися методів виявлення та аналізу ЗПЗ у середовищі цієї ОС а також інших потенційних небезпек цієї платформи, проти яких не передбачено вбудованого захисту.

Робота складається з розділів, кожен з яких розглядає певну ділянку дослідження. У першому розділі дається короткий огляд поточного стану безпеки Linux та існуючих методів виявлення зловмисних програм. Практичне застосування сигнатурних та аномалійних методів виявлення ЗПЗ та їх аналізу розглядаються в другому розділі. У третій частині розглядається економічна доцільність застосування ПЗ для виявлення та протидії ЗПЗ на прикладі коллцентра банківської установи.

1 СТАН ПИТАННЯ ПОСТАНОВКА ЗАДАЧІ

1.1 Визначення проблеми та актуальність дослідження.

Незважаючи на те що більшість користувачів у світі досі використовує ОС Widnows, все більше компаній та домашніх користувачів переходять до Unix базованих операційних систем, завдяки їх надійності, простоті налаштування та безпеці у вигляді вбудованого контроля прав доступу для користувачів у додаток до цього активно розробляються програмні шари типу Crossover та Proton які дозволяють запускати ПЗ призначене виключно для ОС Windows на машинах під керування ОС Linux. Якщо говорити про комерційне середовище, то більш ніж 90% [1] усіх серверних машин у світі використовують Linux. Це робить актуальною проблему шкідливого програмного забезпечення актуальною майже для кожного комерційного підприємства з серверними машинами, а перехід від використання ОС Windows до ОС Linux для робочих станцій як у комерційному середовищі, так і для персонального використання є дуже привабливою можливістю через те що більшість шкідливого програмного забезпечення розробляється виключно для атаки на системи які використовують ОС Windows просто через те що Windows займає 70% ринку серед персональних комп'ютерів. Тобто просто використовуючи Linux можна у десятки разів знизити вірогідність компрометації системи.

Актуальність питання виявлення, аналізу та захисту від шкідливого програмного забезпечення у цілому обумовлена тим що кількість атак з використанням ЗПЗ з кожним роком тільки зростає, як можна побачити на рисунку 1.1 [2].

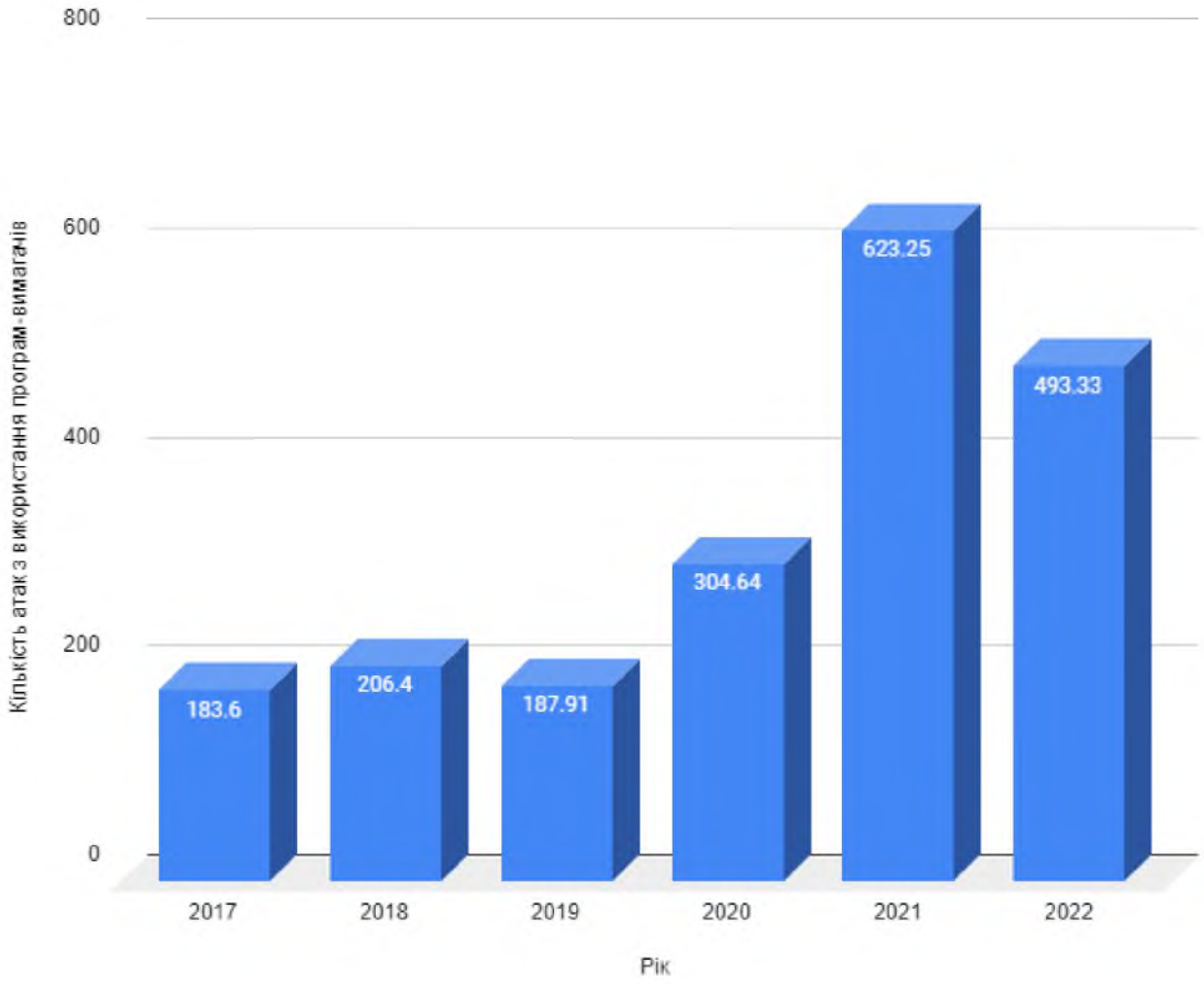


Рисунок 1.1 – Кількість атак з використання програм-вимагачів за рік

У випадку саме Linux кількість атак ЗПЗ у 2022 році виросла на 75% у зрівнянні з 2021 роком. [3], а з нещодавніх атак можна назвати масовану атаку на сервери віртуалізації ESXI з використанням програми-вимагача ESXiArgs у лютому 2023 року, атаку мультіплатформної програми-вимагача Luna яку виявили на початку 2022 року, та атаки ЗПЗ RansomEXX яку виявили ще у 2020 року але остання виявлена атака була на прикінці 2022 року. [4]

1.2 Аналіз об’єкту

Загальний огляд сучасних загроз у сфері кібербезпеки для операційної системи Linux. У цьому розділі будуть розглянути різні види потенційних загроз для систем на базі ОС Linux.

1.2.1 Загрози пов'язані з розповсюдженням зловмисного пз – віруси та інші види ЗПЗ, завдяки яким зловмисники намагаються експлуатувати вразливості самої операційної системи для отримання нелегітимного доступу до системи, перехопити чутливу інформацію або запустити виконання команд які можуть нанести невідворотної шкоди файлам користувача або самій операційній системі загалом.

1. Віруси та вірусні атаки

Програми, які можуть самостійно розповсюджуватися та вбудовуватися і інші виконувані файлі. Зазвичай спрямовані на завдання шкоди або через знищення/шифрування/блокування роботи системи або через витік конференційної або чутливої інформації користувача.

Malware – шкідливе програмне забезпечення у виді троянських конів, черв'яків та шпигунське пз, які зазвичай використовуються для крадіжки конфіденційної інформації користувача.

Ransomware – атаки які використовують шифрування файлів системи що приводить до блокування доступу користувача до вмісту цих файлів. Проводяться або для того щоб вимагати грошову нагороду за розблокування файлів або просто нанесення невідворотної шкоди системі без можливості її відновлення.

2. Експлуатування вразливостей

Використання вразливостей операційної системи для залучення шкідливого ПЗ/інжекції шкідливого коду або отримання привілейованого доступу до системи.

3. Бекдор доступ

Використання бекдор механізмів задля отримання несанкціонованого доступу до системи та збереження прав доступу для подальших атак або інших злочинних дій.

4. Вразливості нульового дня (Zero-day вразливості)

Нові, ще невідомі більшості користувачів вразливості, які злочинці можуть використовувати для атак на систему уникаючи виявлення.

1.2.2 Атаки через мережевий стек: атаки які використовують вразливості мережевого програмного забезпечення системи для отримання несанкціонованою

доступу до системи, впливу на комунікацію та викрадення конфіденційної інформації.

Атаки що спрямовані на вразливості у різних рівнях мережевого стеку:

1. ARP спуфінг

Атаки що підроблюють таблиці ARP (Address Resolution Protocol – протокол розпізнавання адрес), відправляючи підроблені зловмисником ARP відповіді, що приводить до перенаправлення мережевого трафіку, включаючи конфіденційну інформацію, через мережевий пристрій або систему злочинника.

2. DNS спуфінг

Атаки що підроблюють DNS (Domain Name System – Система Доменних Імен) відповіді та у наслідок змінюють DNS таблиці у системі користувача, що призводить до перенаправлення користувача з реального інтернет ресурсу, до якого користувач намагається отримати доступ, на підроблену зловмисником копію цього ресурсу.

3. TCP/IP Hijacking

«Угон»/Викрадення TCP/IP з'єднання задля захоплення активного сеансу користувача та використання цього сеансу для отримання доступу до конфіденційної інформації цього користувача або використання його привілеїв.

4. Маніпуляція мережевими пакетами

Зміна вмісту/аналіз мережевих пакетів для впливу на комунікацію користувача, перенаправлення трафіку що у наслідок може призвести до отримання несанкціонованого доступу до системи користувача.

5. Спуфінг MAC-адрес

Підробка фактичної фізичної адреси мережевої карти яке може дозволити видачу підробленого трафіку машині користувача та у наслідок може призвести до отримання зловмисником доступу до системи користувача

6. SYN/ACK Flooding

Атаки перевантаження «затоплення» SYN/ACK (механізм узгодження TCP протоколу) призводять до переповнення мережевого стеку що у свою чергу викликає DoS (denial of service) –відмову від обслуговування.

7. ICMP Redirect Attacks

Атаки перенаправленная ICMP – змінюють шляхи маршрутизації, що призводить до перенаправлення трафіку з машини користувача на машину зловмисника, або у інше місце.

8. Спуфінг IP-адрес

Зміна IP адреси у пакетах або на машині зловмисника для перехоплення трафіку з машини користувача.

1.2.3 Атаки через недоліки ядра Linux: атаки що використовують вразливості ядра самої операційної системи та дозволяють отримати повний доступ до системи і контроль над нею.

1. Експлуатування вразливостей ядра

Вразливості у коді ядра Linux, які пов'язані із недоліками у роботі системних викликів, обробці пам'яті або іншими аспектами функціонування ядра та можуть бути використані для отримання несанкціонованого доступу до системи.

2. Підняття привілеїв

Атаки що використовують вразливості у системі задля отримання вищих рівнів привілеїв, ніж зазвичай має користувач, наприклад, отримання звичайним користувачем прав адміністратора.

3. Kernel Shellcode

Запуск шкідливого коду безпосередньо у середовищі ядра до запуска самої системи, що надає зловмиснику доступу до всієї системи та робить дуже складним виявлення зловмисника із середовища системи.

4. Утиліти для експлуатації ядра

Готові автоматизовані утілити для атак що використовують вразливості ядра, що полегшує роботу зловмисника та дозволяє робити атаку навіть технічно недосвідченим зловмисникам.

5. Використання вразливостей нульового дня

Використання вразливостей про які розробникам ще невідомо, що може призвести до швидкого та непомітного вторгнення у систему користувача зловмисниками.

6. Атаки на динамічні системні виклики

Модифікація або виклик механізму взаємодії між програмами та ядром системи - системних викликів, що дозволяє програмам звертатися до функції ядра для виконання таких завдань як керування файловою системою, мережею, процесами у системі та системними ресурсами.

7. Denial-of-Service (DoS) атаки

Використання вразливостей ядра для виклику атак на відмову у обслуговувані, що приводе до відмови у роботі системи чи її окремих компонентів.

1.2.4 Атаки на віртуальне середовище: Оскільки використання віртуалізації в сучасних системах зростає, атаки, спрямовані на отримання доступу до віртуальних середовищ і порушення їхньої ізоляції, стають більш актуальними.

1. Використання вразливостей гіпервізора

Атаки з використанням вразливостей гіпервізора (пз яке керує віртуальними машинами) для отримання доступу над віртуальними машинами, або над самим гіпервізором у цілому.

2. Атаки на інтерфейси віртуальних мереж

Зміна налаштувань віртуальної мережі задля перехоплення трафіку, його перенаправлення або вилучення віртуальних машин з мережі.

3. Escape атаки

Атаки «втечі» завдяки яким зловмисник може виходити за межі віртуального середовища та атакувати систему у якій безпосередньо ця віртуальна машина розвернута.

4. Маніпуляція ресурсами віртуальної машини

Зміна конфігурації ресурсів виділених для віртуальної машини (кількості оперативної пам’яті, процесорного часу та інших) що може призвести до обмеження роботи віртуальних машин або витоку інформації.

5. Side-Channel атаки

«Атаки на побічний канал» - такі як атаки на кеш пам’ять можуть дозволити зловмисникам отримати доступ до чутливої інформації з інших віртуальних машин які розвернуті у той самій системі.

6. Впровадження шкідливого коду в гостьову систему

Запуск шкідливого коду у гостьовій системі завдяки вразливостям віртуального середовища.

7. Недостатня ізоляція

Через порушення ізоляції віртуальних машин злочинці можуть отримати доступ до конфіденційної інформації або взаємодіяти з іншими машинами в хост-системі.

8. Отримання доступу до конфіденційної інформації з пам'яті

Вразливості в обробці пам'яті можуть дозволити отримати конфіденційні дані з віртуальної пам'яті машини.

1.2.5 Соціальний інженеринг: Атаки, які використовують соціальний інженеринг, спрямовані на отримання даних від користувачів або спонукання їх до недбалості, стають все більш поширеними.

1.2.6 Розповсюдження атак за допомогою IoT-пристроїв: коли IoT-пристрої підключаються до мережі, вони можуть створювати нові вектори атак і створювати небезпеку для Linux-систем через вразливості цих пристроїв.

1. Брутфорс-атаки на пристрої Інтернету речей:

Атаки на підбор паролів, також відомі як брутфорс, дозволяють злочинцям отримати незаконний доступ до IoT-пристроїв, які зазвичай використовують стандартні паролі або мають слабкі захисні механізми.

2. Використання пристроїв Internet of Things для DDoS-атак:

Заражені пристрої Інтернету речей (IoT) можуть бути використані великими мережами ботів для здійснення розподілених атак на відмову в обслуговуванні (DDoS), які порушують роботу мережевих сервісів.

3. Витік конфіденційної інформації з пристроїв Internet of Things:

Атаки на пристрої, пов’язані з Інтернетом речей (IoT), можуть призвести до витоку конфіденційної інформації, яка зберігається на цих пристроях, таких як дані датчиків, відео- та аудіопотоки та інші.

4. Використання пристроїв Інтернету речей для запобігання фізичним нападам:

Заражені пристрої Інтернету речей можуть бути використані для фізичних атак, таких як відключення електроенергії або вплив на інші системи.

5. Впровадження шкідливого коду через пристрої Internet of Things:

Зловмисники можуть намагатися впровадити шкідливий код на пристрої Інтернету речей, використовуючи вразливість цих пристроїв для подальших атак або створення ботнетів.

6. Зміна функцій пристроїв Інтернету речей:

Атака яка змінює функції IoT пристроїв що може призвести до використання цього пристрою для майбутніх атак, скомпрометувати інші пристрої у мережі або вивести його з ладу.

7. Зіткнення з роботою мережі:

Атаки на мережеві пристрої Інтернету речей можуть спричинити до збоїв в роботі мережі, що впливає на зв'язок між пристроями та роботу цілісних систем.

1.2.7: Загрози від інсайдерів: Інсайдери можуть мати знання про внутрішню систему та отримати доступ до важливих даних, що робить атаки від внутрішніх джерел ще більш небезпечними.

1.3 Огляд існуючих методів виявлення та аналізу зловмисного програмного забезпечення в операційній системі Linux.

1.3.1 Сигнатурний метод – ґрунтується на визначенні унікальних відбитків – сигнатур, існуючого ЗПЗ, характеристик вже відомих вірусних загроз та шкідливого коду. Після чого створюється база цих сигнатур та система порівнює файл із відомими сигнатурами із цієї бази та робить висновок щодо до того чи є ПЗ шкідливим. Цей метод зазвичай є основним для антивірусного пз на будь-якій ОС.

Переваги сигнатурного методу:

1. Простота реалізації та обслуговування – є відносно простим у реалізації і обслуговувані через те що це просто є базою хешей шкідливого пз, тобто її легко створити маючи зразки ЗПЗ та системі легко оновлювати свою базу сигнатур, включаючи нові відомі зразки, забезпечуючи актуальність бази.

2. Ефективність виявлення загроз – сигнатурний метод є дуже ефективним у виявленні загроз, при умові що сигнатура ЗПЗ вже є у базі, цей метод дозволяє негайно виявити загрозу.

3. Швидкість виявлення загроз – сигнатурний метод працює у реальному часі та не вимагає значних обчислювальної потужності від системи. Що робить його актуальним для критичних систем де є важливим негайне виявлення загроз.

4. Низький ризик помилкових спрацювань – помилкові спрацювання зазвичай є дуже рідкісними через те що метод порівнює пз із конкретною сигнатурою вже відомої загрози, що дозволяє мінімізувати ризик помилкового блокування звичайного не шкідливого ПЗ.

5. Висока точність для відомих загроз – сигнатурний метод має високу точність виявлення відомих загроз та шкідливого ПЗ, що робить його особливо ефективним для захисту машині від відомих широко розповсюджених загроз.

Обмеження сигнатурного методу:

1. Неефективність проти нових загроз – основа сигнатурного методу це база сигнатур вже відомих загроз, тобто нові загрози яких в цієї базі немає, сигнатурним методу виявлені не будуть. У наслідок система може пропустити загрози нові відносно останньої версії бази сигнатур у системі.

2. Помилкові спрацювання (помилково позитивні) – незважаючи на те що помилково позитивні спрацювання є досить рідкісними для сигнатурного методу, вони все ще можуть трапляться через те що злоякісне програмне забезпечення може використовувати частину коду із безпечного програмного забезпечення чи однаковість коду в безпечному пз та вірусному пз, яке має сигнатуру у базі.

3. Необхідність частих оновлень – через те що сигнатурний метод базується на порівнянні пз із сигнатурами із бази, часте оновлення цієї бази сигнатур є критичною для працездатності методу. Якщо користувач не оновлює своє антивірусне ПЗ, це зробить його ефективно марним.

4. Постійно зростаючий обсяг сигнатурних баз – незважаючи на те що сигнатури вірусного пз самі по собі не займають багато місця, кількість вірусів та шкідливого пз з кожним днем тільки зростає, відповідно і розмір баз сигнатур цього

пз буде теж тільки зростати. Це може призвести до через мірного використання ресурсів систем для збереження та оброблення цих баз.

5. Вразливість перед поліморфними вірусами. Поліморфні віруси здатні змінювати свою структуру – відповідно і сигнатуру, не змінюючи свого призначення. Тобто такі віруси маскують свою сигнатуру що робить їх неможливими для виявлення сигнатурним методом.

1.3.2 Евристичний аналіз – виявлення шкідливого ПЗ що базується на постійному моніторингу системі та визначенні підозрілої та незвичайної активності спираючись на загальні правила та шаблони поведінки.

Переваги евристичного аналізу:

1. Виявлення нових та невідомих загроз – на відміну від сигнатурного методу, головною перевагою евристичного аналізу є виявлення нового ще невідомого шкідливого програмного забезпечення, сигнатури якого відповідно ще не має у базі відомих загроз.

2. Адаптивність до постійних змін – евристичний аналіз є дуже гнучким у постійно мінливих умовах кібербезпеки, оскільки він не обмежений базою лише відомих сигнатур, що дозволяє швидше адаптуватися під нові загрози та розвивати стратегії виявлення шкідливого ПЗ.

3. Врахування контексту та поведінки – евристичній аналіз враховує контекст та поведінку системи при виявленні аномалій, що зменшує кількість помилкових спрацювань для типових сценаріїв поведінки ПЗ у системі.

4. Виявлення атак за сценаріями – евристичний аналіз виявляє атаки за їхнім сценарієм а не лише завдяки сигнатурі, що дозволяє виявляти більш комплексні загрози, які використовують різні методи атаки або їх комбінацію.

5. Навчання та вдосконалення – евристичний аналіз може бути навчений на основі нових або штучних загроз, що робить його більш ефективним з часом та дозволяє безпеці системи залишатися актуальною та ефективною.

Обмеження евристичного аналізу:

1. Велика кількість помилкових (помилково позитивних) спрацювань – через те що евристичний аналіз покладається на аналіз підозрілої поведінки за

загальними правилами, на відміну від точної бази сигнатур загроз у сигнатурному методі, він може буте схильний до великої кількості помилково позивних спрацювань на типову але безпечну активність.

2. Менш ефективний проти вже відомих загроз – евристичний аналіз буде повільніше реагувати або може взагалі не зреагувати на вже відомі загрози у зрівняні з більш точними та швидкими методами виявлення вже відомого ЗПЗ такими як сигнатурний метод виявлення загроз. Тобто він буде менш ефективним проти вже відомих та розповсюджених загроз.

3. Неспроможність визначати точний тип загрози – оскільки евристичний аналіз виявляє тільки аномальну поведінку ПЗ, він не завжди може визначити точний тип загрози та її джерело, що робить складним усунення цих загроз та створення стратегії відповіді та протидії їм у майбутньому.

4. Потреба сталого нагляду та налаштування – оскільки евристичний аналіз працює завдяки визначеним правилам та шаблонам поведінки, він вимагає постійного нагляду та налаштування для підтримки його актуальності та ефективності, тобто система потребує постійного оновлення евристичних правил для адаптації до нових загроз та вилучення можливостей помилково позивних спрацювань.

5. Ризик уникнення виявлення нових загроз – так як евристичний аналіз базується на заданих правилах та шаблонах, є ризик того що нова неочікувана загроза не буде покриватися цими правилами та уникне виявлення системою.

1.3.3 Аналіз поведінки – спрямований на виявлення шкідливого ПЗ шляхом аналізу фактичної поведінки системи або програми. На відміну від евристичного аналізу, не спирається на попередньо визначені правила, а аналізує звичайні події та мережеві дії у системі і виявляє аномалії або незвичайні патерни, які можуть свідчити про атаку на цю систему.

Переваги аналізу поведінки:

1. Виявлення нових та невідомих загроз – так як, на відміну від сигнатурного та евристичного аналізу, аналіз поведінки не обмежений базою сигнатур або заздалегідь заданими правилами він може ефективно виявляти нові та невідомі

загрози, сигнатура яких відсутня у базі та які не перекриваються попередньо визначеними сценаріями завдяки виявленню аномалій при аналізі фактичної активності системи.

2. Висока адаптивність – через те що аналіз поведінки виявляє загрози завдяки аномальній поведінки у системі, він буде краще адаптований до нових загроз аніж статична база сигнатур або набір правил.

3. Виявлення атак за сценарієм – аналіз поведінки не спирається на базу сигнатур або шаблони поведінки, що дозволяє йому виявляти атаки за їх сценарієм, що особливо важливо для виявлення складних комплексних або комбінованих атак.

4. Здатність виявляти нові загрози без оновлення набору правил або бази сигнатур – оскільки аналіз поведінки не спирається на набір правил або набор сигнатур, системі не потрібно оновлювати цю базу або набір правил для того щоб виявляти нові загрози.

5. Врахування контексту та емпіричних даних – при аналізі поведінки система буде враховувати унікальні особливості та специфіку конкретної системи та інфраструктури у якій вона знаходиться, а також використовувати зібрану інформацію для створення шаблону нормальної поведінки системи, що зменшує вірогідність помилкових спрацювань.

6. Висока точність виявлення загроз – аналіз поведінки має високу точність виявлення загроз оскільки він враховує реальну поточну активність системи, а не спирається на статичні ознаки атак, що покращує кількість реальних спрацювань та зменшує кількість помилкових.

Обмеження аналізу поведінки:

1. Висока складність налаштування – аналіз поведінки вимагає значну кількість часу та ресурсів для налаштування, оцінки та адаптації до конкретних проблем системи. Для забезпечення ефективності роботи та уникнення помилкових спрацювань, він потребує ретельного налаштування параметрів.

2. Велика кількість помилкових спрацювань – незважаючи на те що аналіз поведінки є дуже ефективним для виявлення загроз на основі аномальної поведінки, він також може бути схильним до великої кількості помилково

позитивних спрацювань через нормальну активність системи, яка буде виглядати аномальною але є безпечною та нормальною для системи.

3. Неефективність проти вже відомих загроз – як і у випадку з евристичним аналізом, такі методи як сигнатурний аналіз будуть значно швидші та ефективніші для виявлення вже відомих загроз.

4. Важкість визначення контексту – чим складніша буде система та інфраструктура у якій вона знаходиться, тим складніше буде визначення що є аномалією у цьому контексті та тим більша вірогідність помилкових спрацювань.

5. Потреба у великій кількості обчислювальних ресурсів – для безперервного та ефективного аналізу поведінки може бути потрібна значна кількість обчислювальних ресурсів, особливо у великих та комплексних системах.

6. Можливість обходу заходів безпеки – деякі складні атаки можуть бути схильними до обходу заходів безпеки на основі аналізу поведінки, якщо зловмисники мають уявлення як працює система та розроблять шкідливе пз робота якого не буде виглядати аномальною.

7. Необхідність постійного моніторингу – аналіз поведінки вимагає постійного моніторингу для виявлення аномалій, що може потребувати додаткову кількість людських ресурсів або автоматизованих рішень у порівнянні з іншими методами.

8. Неможливість точного визначення атак – визначення точного типу та джерела атаки базуючись тільки на аналіз поведінки є важкою задачею, що негативно вплине на швидкість ліквідації загрози та впровадження засобів протидії.

1.3.4 Сканування пам’яті – метод виявлення потенційних загроз або некоректних областей оперативної пам’яті завдяки аналізу антивірусною програмою оперативної пам’яті комп’ютера. Може працювати як за заданими правилами і шаблонами, так із базою сигнатур відомих загроз.

Переваги сканування пам’яті:

1. Виявлення невідомих загроз – сканування пам’яті може допомогти виявляти як відомі загрози по сигнатурам, так і невідомі нові загрози.

2. Аналіз аномалій у роботі програм – аналіз пам'яті дозволяє у реальному часі виявляти аномалії у роботі програм та виділяти програми які підозріло себе поводять. Особливо ефективно для виявлення шкідливого пз яке не взаємодіє із файловою системою напряму.

3. Виявлення вразливостей та експлойтів – завдяки скануванню пам'яті можна виявити потенційні вразливості у системі через які може буде виконана атака.

4. Аналіз взаємодії програм – сканування пам'яті дозволяє вивчати як програми та процеси взаємодіють між собою, що може бути важливим для виявлення атак чи аномальної поведінки у системі.

5. Виявлення динамічних атак – аналіз пам'яті дозволяє виявляти динамічні атаки, які змінюють свою поведінку в ході атаки.

6. Можливість виявлення витоків конфіденційної інформації – сканування пам'яті дозволяє побачити витоки конфіденційної інформації у оперативній пам'яті та захищати систему від таких загроз.

7. Швидкість реагування – сканування пам'яті відбувається у реальному часі, що дозволяє швидко та ефективно реагувати на потенційні загрози.

Обмеження сканування пам'яті:

1. Велика вимоги до обчислювальних ресурсів – в залежності від об'єму оперативної пам'яті, сканування пам'яті може бути дуже витратним у плані системних ресурсів, особливо для систем типу серверів з дуже великим об'ємом оперативної пам'яті, що приведе до збільшення навантаження на систему та погіршення продуктивності.

2. Можливість виникнення помилкових спрацювань – через те що аналіз пам'яті не обмежується тільки сигнатурами для виявлення шкідливого пз, можливі часті випадки коли нормальна активність системи буде інтерпретуватися як аномальна, що приведе до негативного впливу на роботу системи та користувачів.

3. Складність реалізації – впровадження ефективного механізму сканування пам'яті є дуже складним та трудомістким завданням, особливо у системах де доступ до пам'яті обмежений з міркувань безпеки.

4. Великий обсяг даних для обробки – сканування пам’яті вимагає аналізу великої кількості даних що будуть вимагати відповідної кількості ресурсів для їх зберігання та обробки.

5. Не є ефективним для усіх типів загроз – аналіз пам’яті не може виявляти усі типі загроз, на кшталт потенційно небезпечних але легальних дії, ЗПЗ яке захищене від аналізу пам’яті завдяки шифруванню або обфускації коду, загрози на низькому рівні пам’яті які не можуть бути виявлені шляхом аналізу вищого рівня пам’яті.

6. Можливість витоку конфіденційної інформації – так як сканування пам’яті напряму може взаємодіяти із конфіденційною інформацією яка знаходиться у оперативній пам’яті, це може призвести до її витоку.

7. Несумісність з іншими заходами безпеки – сканування пам’яті може бути не сумісне з іншими заходами безпеки у системі, особливо тими що обмежують доступ до оперативної пам’яті для запобігання вразливостям.

1.3.5 Виявлення нуль-денних атак – критичне завдання для забезпечення безпеки системи як полягає у виявленні атак нульового дня, тобто атак які використовують вразливості про які ще публічно невідомо, а у деяких випадках невідомо нікому крім зловмисників які планують провести атаку.

Може буте реалізовано завдяки:

1. Моніторингу несподіваних дії – аналіз несподіваних або невластивих для нормальної роботи системи дії у самій системі або мережі.

2. Аналіз системних журналів – виявлення атак завдяки аналізу системних журналів на наявність інформації про невідомі або підозрілі дії.

3. Використання систем виявлення вторгнень (IDS) - IDS системи аналізують системну активність та мережевий трафік і виявляють аномалії або підозрілі зразки, які можуть бути пов’язані з атаками нульового дня.

4. Облік патернів атак – для виявлення характерних сигнатур атак нульового дня можуть бути використані бази даних шаблонів атак, та регулярне оновлення таких баз дозволяє запобігати новим вразливостям та атакам.

5. Аналіз трафіку та поведінки – моніторинг аномальної поведінки у мережі та системі по типу змін у рівнях доступу, пересилання даних або несподівані модифікації системних файлів, може дозволяти виявляти атаки нульового дня.

6. Хмарні рішення для виявлення загроз – для виявлення загроз нульового дня можна залучати хмарні рішення з великою базою подібних загроз, які швидко оновлюються у реальному часі та можуть виявити нуль-денні атаки завдяки аналізу актуальних даних та статистики загроз.

7. Системи аналізу вмісту пакетів – для виявлення атак нульового дня можна використовувати системи що аналізують мережеві пакети, це дозволить виявити підозрілих патернів у мережі або атаки на рівні протоколів.

8. Взаємодія з виробниками пз – зв'язок та співпраця з розробниками програмного забезпечення дозволить отримувати актуальну інформацію про нові вразливості та вчасно отримувати виправлення та оновлення які ці вразливості виправляють.

Важливість виявлення атак нульового дня:

1. Мінімізація впливу атак – чим швидше буде виявлена атака нульового дня та вразливість через яку вона реалізована, тим менше буде розмір шкоди завданої зловмисником та менше у нього буде можливостей для атак.

2. Забезпечення реагування на нові загрози – виявлення атак нульового дня дозволяє організаціям бути готовими до реагування на нові невідомі загрози. Швидке реагування дозволяє вчасно вживати заходи безпеки та розробити оновлення яке усуне вразливість через яку було проведено атаку.

3. Розвиток стратегії безпеки – виявлення атак нульового дня надає важливу для розробки та удосконалення стратегії безпеки інформацію. Це дозволяє підприємствам адаптуватися до змін у актуальних загрозах та ефективно захищати свої системи.

Обмеження виявлення атак нульового дня:

1. Відсутність відомостей про вразливості – оскільки атаки нульового дня спираються на ще невідомі та невиявлені вразливості, виявлення таких атак на ранніх стадіях є дуже важкою задачею.

2. Неоднорідність зразків атак – зловмисники постійно змінюють методи своїх атак, задля запобігання виявленню, це приводить до того що зразки атак нульового дня стають занадто неоднорідними для відокремлення конкретних схем та шаблонів атак.

3. Бездіяльність атак до виявлення – зловмисники мають можливість утримувати запуск атаки нульового дня до тих пір поки вони не будуть впевнені у тому, що атака залишиться непоміченою. Це приводить до того що атака буде виявлена тільки після того як вона було успішна.

4. Висока імовірність помилково позитивних спрацювань – багато систем типу IDS можуть допомогти виявити атаки нульового дня опираючись на аномальну поведінку у системі та несподівані патерни, але це може привести до високої ймовірності помилково позитивних спрацювань.

5. Правові обмеження – виявлення атак нульового дня може зіткнутися з правовими обмеженнями через розголошення інформації про нові вразливості, що ускладнює обмін інформації та співпрацю між організаціями.

1.3.6 Виявлення атак у реальному часі – постійний моніторинг та аналіз активності у системі або мережі для негайного виявлення атак або потенційно шкідливих дій.

Може бути реалізовано завдяки:

1. Системам моніторингу та журналювання – системи які аналізують мережевий трафік та активність у реальному часі, що дозволяє виявляти аномальну або підозрілу поведінку та вразливості. До таких систем належать системи виявлення вторгнень (IDS) та системи виявлення аномалій (ADS), які виявляють незвичайні патерни поведінки або ознаки атак для моментального реагування.

2. Аналіз поведінки користувачів – моніторинг та аналіз поведінки користувачів системи щодо підозрілої або незвичайної поведінки, яка може свідчити про компрометацію облікових записів або несанкціонований доступ.

3. Використання технологій машинного навчання – навчання сучасних моделей машинного навчання для розпізнання атак на систему, що допоможе покращити розпізнавати атаки у реальному часі.

4. Автоматизовані системи реагування – системи які у автоматичному режимі вживають заходів у разі виявлення атаки чи підозрілої або небезпечної активності. До таких заходів відносяться блокування атак, відключення скомпрометованих облікових записів або робочих станцій чи ізолювання частини мережі.

5. Мережеві та системні архітектури з розподіленими сенсорами – розташування сенсорних елементів у різних точках мережі або інформаційної системи для впровадження комплексного та розподіленого моніторингу.

6. Інтеграція з threat intelligence (розвідка щодо загроз) – уявляє собою збір, аналіз та розповсюдження інформації про потенційні загрози, дані про нові види атак, вразливості програмного забезпечення, техніки зловмисників та інше.

Важливість виявлення атак у реальному часі:

1. Негайна реакція – можливість негайно реагувати на загрози у системі, що мінімізує час у який атака може нанести шкоду.

2. Мінімізація збитків – при виявленні атак у реальному часі, система може негайно впроваджувати заходи безпеки для мінімізації збитків від атак, що дозволяє або повністю запобігти або обмежити вплив атаки на інформаційну систему.

3. Виявлення нових атак – моніторинг, аналіз та реагування на підозрілу і аномальну активність дозволяє виявляти невідомі атаки які ще не мають сигнатур.

4. Адаптивність до змін сценаріїв атак – система виявлення загроз у реальному часі може змінюватися і адаптуватися до нових типів і сценаріїв атак за потребою.

5. Спрощення відновлення – негайне виявлення і реагування на атаки дозволяє швидше їх зупинити та почати процес відновлення що мінімізує час простою системи та вплив на бізнес-процеси.

6. Можливість визначення кореляції між подіями – виявлення атак у реальному часі дозволяє знаходити кореляції між різними подіями та виявляти потенційно небезпечні сценарії.

7. Використання розумних алгоритмів – можливість використовувати розумних алгоритмів та технологій машинного навчання для автоматизації виявлення, аналізу та відповіді для атак у реальному часі.

Обмеження систем виявлення атак у реальному часі:

1. Вартість реалізації та обслуговування – впровадження та підтримка систем виявлення атак у реальному часі є дуже трудомісткою та витратною задачею, особливо для великих інформаційних систем. Це включає вартість обладнання, програмного забезпечення, навчання персоналу та постійне оновлення цих систем.

2. Великий обсяг даних – оскілки системі виявлення атак у реальному часі необхідно займатися моніторингом та аналізом як усього мережевого трафіку, так і усіх подій у системі, це створює величезний об'єм даних для аналізу з яким система може просто не впоратися та відповідно не зможе забезпечити високу ефективність.

3. Помилково позитивні спрацювання – одне з головних обмежень систем виявлення загроз у реальному часі, є велика кількість помилково позитивних спрацювань у тому числі просто через неймовірний обсяг даних для аналізу. Також кількість помилково позитивних спрацювань буде тільки збільшуватися, якщо ціллю є виявлення ще невідомих атак так як види і схеми атак постійно змінюються і часто їх дуже легко сплутати з аномальною та незвичайною але цілком безпечною та нормальною роботою системи.

4. Залежність від якості даних – незважаючи на великий об'єм даних для аналізу, часто дані можуть бути неповними або неточними що буде приводити до помилкових висновків та впровадження марних або навіть шкідливих засобів безпеки.

5. Проблеми з конфіденційністю – збір та обробка великої кількості даних у системі в реальному часі часто може порушувати приватність користувачів так як їх конфіденційні дані теж можуть збиратися та аналізуватися системою, тому є також важливим враховувати юридичні та етичні аспекти при роботі систем виявлення атак у реальному часу в таких випадках.

6. Атака на систему виявлення атак – зловмисники можуть проводити атаки саму на систему виявлення атак та скомпрометувати її, що не тільки дозволить проведення подальших атак, але зробить їх непомітними та надасть почуття хибної безпеки системи.

7. Необхідність постійного оновлення – підтримка системи виявлення загроз у реальному часі у ефективному стані вимагатиме постійного оновлення сигнатур загроз, шаблонів атак та правил у системі для боротьби з постійно та швидко мінливими загрозами безпеки.

Для максимізації виявлення та аналізу ЗПЗ в операційній системі Linux ці методи можна використовувати як окремо, так і в поєднанні. У кожного методу є переваги та недоліки, і як їх використовувати, залежить від потреб і вимог системи кібербезпеки.

1.4 Завдання дослідження:

1. Аналіз методів виявлення ЗПЗ для операційних систем Linux, включаючи переваги та недоліки.

2. Дослідження методів аналізу шкідливого програмного забезпечення для ОС Linux.

3. Практичний огляд інструментів для виявлення і аналізу ЗПЗ у системах під керуванням ОС Linux.

4. Порівняння та оцінка ефективності інструментів виявлення ЗПЗ в операційних системах Linux, визначення їх практичності.

5. Дослідження ПЗ для виявлення та протидії ЗПЗ для ОС Linux та визначення економічної доцільності використання цього ПЗ.

Виконання цих завдань дозволить систематизувати та розширити знання щодо виявлення та аналізу ЗПЗ в середовищі ОС Linux, а також внести практичний внесок у покращення засобів та методів протидії їм для усіх користувачів цієї операційної системи.

1.5 Висновок до першої частини

У першому розділу було розглянуто актуальність використання ОС Linux та загроз для неї у вигляді атак з використанням шкідливого програмного

забезпечення. Також були детально розглянути можливі вектори атак на систему та методи виявлення цих атак з відокремленням їх індивідуальних переваг та недоліків.

Актуальність питання обумовлена тим що 90% всіх серверних машин використовує ОС Linux, а використання ОС Linux на робочих станціях та персональних комп'ютерах на є привабливою можливістю через те що більшість шкідливого програмного забезпечення створюється для атак на системи з використанням ОС Windows, тобто просто використання ОС Linux значно зменшує можливість атаки зі сторони зловмисників. Актуальність загрози ЗПЗ у цілому обумовлено тим що щорічна кількість атак з використанням ЗПЗ виросла у 3 рази з 2017 по 2022 роки, а конкретно для ОС Linux кількість атак у 2022 року виросла на 75% у зрівнянні з 2021 роком.

У другому розділі будуть розглянуті методи виявлення та аналізу шкідливого програмного забезпечення з практичним аналізом на прикладах конкретного програмного забезпечення яке використовується та у деяких випадках розробляється ведучими фахівцями з кібербезпеки.

2 СПЕЦІАЛЬНА ЧАСТИНА

Методи виявлення та аналізу зловмисного програмного забезпечення.

2.1 Середовище для тестування та зразки шкідливого ПЗ.

Для перевірки працездатності вірусного та потенційно шкідливого програмного забезпечення буде використовуватися ОС Ubuntu 20.04, яка буде встановлена та запущена як віртуальна машина на платформі VirtualBox. Гостьова система після налаштування та встановлення необхідного ПЗ, не буде мати доступу до мережі та хост системи, включаючи буфер обміну, файлову систему та периферію, задля забезпечення безпеки та запобіганню зараження хост системи.

Для перевірки працездатності антивірусного ПЗ буде використовуватися публічно доступні зразки шкідливого або потенційно шкідливого ПЗ.

Tsunami – шкідливе ПЗ для створення бекдор доступу до системи яке використовує канал IRC (Internet Relay Chat) щоб слухати та отримувати інструкцій від зловмисників і у наслідок для запуску DDoS атак, використовуючи інфікований комп'ютер як один з вузлів для атаки. [5, 6] Уперше з'явилось ще у 2009 році і було відомо під ім'ям Kaiten, це ЗПЗ з початку було націлено на Linux машини, але пізніше функціонал було розширено та адаптовано для використання на системах під керуванням ОС MacOS. Після цього вихідний код цього ПЗ виклали у публічний доступ, що дозволяє будь-кому його відредагувати та скомпілювати. Незважаючи на те що ЗПЗ є достатньо старим та відомим, на початку 2023 року було виявлену масову атаку з використанням цього ЗПЗ на погано захищені Linux машини, у який був відкритий SSH канал у інтернет. [7]. Окрім бекдор доступу для проведення DDoS атак завдяки Tsunami, зловмисники також встановлювали інше ЗПЗ в тому числі майнери криптовалюти. Для перевірки у роботі буде використовуватися одна з варіації Tsunami з SHA-256: 6c6888a75d6a62dc7414dd22d0b6a70456a108a14889b8406f7aeb8b61b34633.

Hive – вид програми-вимагача уперше виявлений у 2021 році та створений групою зловмисників Hive яке шифрує файлі в системі та після цього погрожує тим що уся важлива інформація попаде в даркнет якщо не буде надано компенсації. Окрім цього ЗПЗ Hive видаляє усі бекапи, точки відновлення, вимикає та видаляє

антивірусне та анти шпигунське програмне забезпечення та залишає у кожній теці текстовий файл з вимогами. Група Hive надавала послуги Ransomware as a Service (RaaS) – програми вимагачі як послуга по всьому світу, надаючи можливість бажаючим замовляти атаки. Їх атаки в основному були націлені на заклади охорони здоров'я та інші не комерційні або державні підприємства. Шкідливе програмне забезпечення потрапляло у мережу закладу завдяки фішінговим атакам або соціальної інженерії, після чого або сам співробітник запускав ЗПЗ, або група зловмисників отримувала облікові дані співробітника та віддалено від'єднувалась та запускала атаку. Усі скомпрометовані мережі, які вдалося відновити до попереднього стану згодом були повторно заражені цим же ЗПЗ. У січні 2023 ФБР успішно усунула групу зловмисників Hive та їх сайт, отримала доступ до їх серверних машин та протягом семи місяців до цього проникла у цю групу та надавала ключи дешифрування жертвам ЗПЗ Hive. [8] По підрахункам група отримала більш ніж сто мільйонів доларів завдяки своїм атакам, а їх усунення запобігло втраті ще ста тридцяти мільйонів. [9, 10] Для перевірки у роботі буде використовуватися одна з варіації Hive з SHA-256: 6c6888a75d6a62dc7414dd22d0b6a70456a108a14889b8406f7aeb8b61b34633.

HelloKitty – шкідливе програмне забезпечення-вимагач яке вперше було помічено у 2020 році та отримало свою назву через процес "HelloKittyMutex" який з'являвся при виконанні атаки. Від початку це ЗПЗ було націлене на системи під керуванням ОС Windows. Зловмисне програмне забезпечення розповсюджувалось завдяки фішинговим поштовим листам та програмним забезпеченням для тесту на проникнення Cobalt Strike, яке встановлює у систему вузол Beacon (маяк) який дозволяє виконувати скріпти PowerShell, робити скріншоти, завантажувати та запускати файли у системі, та створювати інші точки бекдору для доступу до системи.[11] У 2022 році державні організації України зазнали кібератаки з використанням пз Cobalt Strike. [12] Після запуску пз HelloKitty буде намагатися вимкнути усі процеси, які можуть перешкодити шифруванню диска завдяки taskkill.exe та net.exe і, якщо це не вдалось, використовує Windows Restart Manager API – інтерфейс менеджеру перезавантаження Windows, який вимикає усі процеси

окрім критичних. Шифрування диску проводиться з використанням комбінації AES-256 та RSA-2048 або NTRU та AES-128 і відбувається дуже щвидко. [13] Найвідоміша атака сталась у 2021 році на компанію CD Projekt Red, у наслідок якої за заявою зловмисників вони отримали вихідний код усіх ігор які розробляють CD Projekt Red. Атака відбувається дуже швидко та не намагається бути схованою, викликаючи велику кількість вікон терміналу Windows. Після проведення шифрування зловмисники залишають текстове повідомлення різне для кожної жертви. [14]. Після 2021 була розроблена версія цього шкідливого пз націлена на системи під керуванням ОС Linux, а саме серверів віртуалізації які використовують популярну платформу для віртуалізації ESXI. [15] Для перевірки у роботі буде використовуватися одна з варіації HelloKitty з SHA-256: 556e5cb5e4e77678110961c8d9260a726a363e00bf8d278e5302cb4bfccc3eed

eCh0Raix – шкідливе програмне забезпечення-вимагач яке було виявлено вперше у 2019 році. Це ЗПЗ націлене на мережеві NAS пристрої для збереження даних та виконує шифрування документів, текстових файлів, архівів, баз даних та мультимедійних файлів. ЗПЗ вимагає 0,05-0,06 Bitcoin для розблокування (близько $600 США на момент перших атак у 2019 році) та автоматично видаляє себе і не проводить атаку якщо мережевий пристрій знаходився у певних країнах СНД, а саме Беларусь, Україна та Росія. [16] Шкідливе ПЗ розповсюджувалось завдяки брутфорсу мережевих NAS пристроїв з поганою безпекою та застарілими версіями мережевих протоколів. ЗПЗ використовує алгоритм AES для шифрування та залишає усі файли з розширенням .encrypted (.зашифровано). ЗПЗ почало бути активним з 2019 та перша велика хвиля атак відбулась у 2021 році жертвами якої були NAS пристрої Тайванських компанії QNAP та Synology. Виробники почали попереджувати своїх клієнтів про атаки з використанням цього ПЗ з кінця весни 2021 року, у разі QNAP атаки відбувались через вразливість нульового дня, а у випадку Synology – через слабкі паролі на пристроях користувачів. Після попередження від виробників, у тому ж місяці було випущено оновлення, яке усуває вразливість завдяки якої проводилась атака. [17] Для перевірки у роботі буде використовуватися одна з варіації eCh0Raix з SHA-256:

24b5cdfc8de10c99929b230f0dcbf7fcefe9de448eeb6c75675cfe6c44633073 та 3d8d25e2204f25260c42a29ad2f6c5c21f18f90ce80cb338bc678e242fba68cd.

WinnTI – ЗПЗ китайської групи хакерів відомої як WinnTI або APT41 яка існує як мінімум з 2010 року та атаки якої в основному були націлені на державні підприємства США, Індії та Тайваню, а також у випадку Об'єднаного Королівства, Ірландії, Тайваню, Монголії, Індії, Бангладешу, Індонезії та Гонг Конгу – було націлене на різноманітні мережеві ресурси включаючи освітні, медійні, логістичні та охорони здоров'я. Розповсюдження ЗПЗ відбувалось як за допомогою фішингу та соціальної інженерії, так і з використанням SQL-інжекцій, supply chain attack (атак на ланцюг поставок – атаки на малі організації які є частиною або надають послуги більш великій компанії), watering hole attack (напад на водопій – напад не на саму організацію, а на мережеві ресурси які компанія найчастіше використовує у своїй роботі), та атак з використанням спеціалізованого ПЗ для атак такого як Acunetix, Nmap, SQLmap, OneForAll, subdomain3, subDomainsBrute, Sublist3r, та відомого пз для тестування на проникнення Cobalt Strike, який часто використовується зловмисниками. [18] Атаки групи WinnTI є дуже складними та комплексними у виконанні і являють собою дуже довгий ланцюг послідовних замаскованих атак, це приводить до того що атака може легко провалитися якщо не спрацює одна з ланок ланцюга, але також робить атаки майже неможливими для виявлення. WinnTI використовують HTTP, HTTPS, TLS, UDP та TCP протоколи через які вони відправляють жертві пакети з RAW даними які зашифровані досі невідомим алгоритмом з динамічним та станичним ключами і закодовані у BASE64, які у подальшому дешифруються та запускаються. Одним з наслідків атаки є шифрування машини жертви алгоритмом AES. [19] Для перевірки у роботі буде використовуватися одна з варіації WinnTI з SHA-256: 2f1321c6cf0bc3cf955e86692bfc4ba836f5580c8b1469ce35aa250c97f0076e.

XorDDoS – шкідливе ПЗ з rootkit можливостями (можливість зберігати контроль над машиною жертви впродовж довгого часу без усвідомлення власника машини). XorDDoS отримав свою назву через те що це шкідливе програмне забезпечення яке використається для проведення DDoS атак та при цьому у великої

кількості використовує XOR шифрування для більшості своїх дій. ЗПЗ націлене на машини під керування ОС Linux з архітектурами ARM, x86 та x64. У перше виявлене у 2014 як Shell скріпт для розповсюдження DDoS механізму з використанням виконавчого ELF файлу. При зараженні ЗПЗ створює безліч своїх копій у всіх теках та додає механізм автозапуску себе ж. Rootkit доступ надає зловмиснику доступ до файлів у системі та до запуску команд, що використовується для запуску DDoS атак. [20] ЗПЗ розповсюджуються як троянській кінь з інтернет ресурсів, або запускається у системі іншим ЗПЗ, які при зараженні інфікують машину купою іншого шкідливого ПЗ. Наступна хвиля атак почалась у липні 2023 року та зазнала свого піку у серпні 2023 року використовуючи C2 (Command and Control Infrastructure – атака на систему через мережеві протоколи такі як HTTP, HTTPS, DNS та DHCP). Перед запуском зловмисником проводиться сканування машини на мережеві вразливості для атаки, в особливості на наявність HTTP служб на машині які дозволяють провести атаку directory traversal (атаку обходу каталогу) – атака що дозволяє зловмиснику отримати доступ до системних файлів на мережевому сервері, у тому числі для того щоб отримати доступ до файлу /etc/passwd завдяки якому можна дізнатися логін користувача та отримати доступ до машини через SSH тунель завдяки брутфорс атаками з відомим логіном. Після запуску впродовж перших трьох днів ЗПЗ проводить сканування машини та збирає усю інформацію про машину, включаючи індетіфікатор машини, версію ОС, інформацію про компоненти машини та її оточення, після чого починає встановлювати вірусне ПЗ на машину та інфікує інші пристрої з вразливостями у мережі, якщо такі є. Після чого ЗПЗ використовує машину для DDoS атак. ЗПЗ використовує декілька механізмів для забезпечення своєї роботи – створює задачу на автозапуск себе кожні 3 хвилини та додає себе у службу автоматичного запуску для того щоб ЗПЗ запускалося разом із системою. Щоб запобігти виявлення ЗПЗ робить свій процес фоновий процесом, який працює окремо від поточної сесії користувача для того запобігти сигналам усунення процесу від користувача та замаскуватися під реальний не шкідливий процес. Також ПЗ робить велику кількість виконавчих ELF файлів з трохи зміненим кодом

щоб відрізнятися від оригіналу та запобігти масовому видаленню. Між груднем 2022 року та серпнем 2023 року було ідентифіковано близько таких 26000 мутованих зразків які з'явились з близько 800 початкових зразків. Атаки зазнали багато індустрій на глобальному рівні включаючи виробництва напівпровідників, компаній телекомунікацій, транспортування, фінансів, страхування та роздрібненого продажу. [21] Для перевірки у роботі буде використовуватися одна з варіації WinnTI з SHA-256:
d920dec25946a86aeaffd5a53ce8c3f05c9a7bac44d5c71481f497de430cb67e.

Gafgyt – також відоме як Bashlite або Mirai (через те що ЗПЗ використовує велику кількість запозиченого вихідного коду з Mirai) це ЗПЗ яке націлене на машини під керуванням ОС Linux, зокрема на IoT пристрої. У перше виявлено у 2004 році та являє собою ботнет який інфікує IoT пристрої для виконання C2 команд та запуску масованої DDoS атаки. Спочатку для зараження ЗПЗ використовувало вразливості у мережевих пристроях такі як CVE-2017-17215 (Huawei), CVE-2017-18368 (ZYXEL), та CVE-2014-8361 (Realtek). У більш сучасних варіантах до ЗПЗ було додано функціоналу для приховування шкідливого мережевого трафіку завдяки мережі TOR та шифруванню команд які передаються на інфіковані пристрої. Для зараження використовуються брутфорс атаки на мережеві пристрої зі слабким Telnet паролем та більш сучасні вразливості у мережевих пристроях такі як CVE-2019-16920 (D-Link RCE), CVE-2019-19781 (Citrix), та CVE-2020-7961 (Liferay Portal RCE) і досі підтримує усі старі вразливості, що дозволяє проводити атаки на пристрої які не оновлялись. Атака зазвичай відбувається у такій послідовності: знаходиться вразливий IoT пристрій, зловмисник експлуатує вразливість у пристрої, використовуючи брутфорс зламуються слабкі або взагалі стандартні облікові дані користувача пристрою, дані відправляються на C2 сервер, сервер відправляє інфіковані мережеві пакети пристрою та запускає їх, зловмисник отримує повний бекдор доступ до пристрою та може виконувати на ньому будь-які команди. [22] Незважаючи на те що це ЗПЗ було вперше виявлено ще у 2004 році і використовує вразливості які вже були виправлені новими оновленнями, у серпні 2023 році розпочалася масова хвиля атак

цього ЗПЗ на маршрутизатори виробника Zyxel з використанням вразливості CVE-2017-18368 – тобто вразливості 2017 року, через те що користувачі до сих пір використовують застарілу версію ПЗ на цих пристроях, у тому числі на маршрутизаторах які були вироблені у 2019 році та які у тому числі знаходяться у державних установах. У серпні 2023 року було зареєстровано близько семи тисяч запусків неавторизованих команд на таких пристроях у день. [23] І ці пристрої скоріш за все будуть і далі використовуватися як ланки для DDoS атаки, доки власники цих маршрутизаторів не оновлять їх ПЗ. Для перевірки у роботі буде використовуватися одна з варіації Gafgyt з SHA-256:
c7961eb366717e1d1641c478849513b207b36b685c97a59160a9e970a27ca8e6

Mirai – шкідливе програмне забезпечення яке націлене на IoT пристрої з процесором ARC, машин з цим процесором використовують урізану версію ОС Linux та включають у себе транспортні засоби, мережеві маршрутизатори, радіоняні, сільськогосподарські пристрої, медичні пристрої, побутову техніку, відео регістратори, камери та інше . Для отримання доступу до них та використання цих пристроїв у бот мережі для проведення DDoS атак. Mirai інфікує IoT пристрої завдяки підбору облікових даних пристрою з таблиці стандартних заводських облікових даних або брутфорсу а також завдяки розсилці його у виді фішінгових листів або троянських конів. ЗПЗ було у перше виявлено у 2016 році та було створено двадцяти однорічним Парас Джа разом з його приятелями, які у подальшому створили компанію Protraf Solutions, яка пропонувала послуги захисту від DDoS атак через це саме ЗПЗ Mirai та тім самим компаніям які зазнавали атаки від засновників Protraf Solutions. Найбільшу атаку автори ЗПЗ запустили у 2016 році на цей момент у бот мережі Mirai нараховувалося приблизно триста тисяч пристроїв що було на той час найбільшою бот мережею у історії, після чого виклали вихідний код Mirai у відкритий доступ, потенційно для того щоб скрити сліди атаки, та що дозволило великій кількісті кіберзлочинців почати відтворювати це ЗПЗ та створювати свої варіації і призвело до атаки невідомого хакера на DNS провайдера Dyn, що у наслідок ефективно позбавило інтернету для мільйонів користувачів у північній Америці та Європі. [24] Та у подальшому породило ще

більше атак які у тому числі вивели з ладу дев'ятсот тисяч маршрутизаторів у Німеччині та позбавили доступу до інтернет мережі усю країну Ліберію. Незважаючи на те що Парас Джа та його колеги були піймані у 2017 році, і в подальшому почали співпрацю з ФБР, доступ до вихідного коду Mirai дозволив не тільки модифікувати та підтримувати актуальним саме ЗПЗ але й привело до появлення інших сімей ЗПЗ таких як Gafgyt, Okiru, Satori, Masuta та PureMasuta та нещодавно виявленого ботнету IoTrooper або Reaper, який здатний набагато швидше компрометувати IoT девайси аніж Murai. [25] Для перевірки у роботі буде використовуватися одна з варіації Mirai з SHA-256:
c5cc057458e5563c563f874a84bb2992c82ff44b043913a7ce2080c9c187ea97

Sodinokibi – також відоме як REvil програма вимагач яка шифрує файли користувача та після цього вимагає платежу у Bitcoin для дешифрування файлів з погрозою подвоїти суму платежу якщо користувач буде намагатися обійти ЗПЗ. ЗПЗ у перше було виявлено у 2019 році приблизно у той самий час як дуже схоже за методом дії ЗПЗ GangCrab, яке відповідальне за 40% усіх атак програм-вимагачів у світі, вийшло з експлуатації через то ще для нього був розроблений універсальний дешифратор, що скоріш за все свідчить про то що за цими двома ЗПЗ стоять одні і ті самі групи зловмисників. Та те що ЗПЗ перед початком атаки перевіряє мовні ідентифікатори системи і не проводить атаку якщо користувач інфікованої машини знаходиться у східній Європі, скоріш за все також свідчить про те що розробниками цього ЗПЗ є групи зловмисників з Росії. [26] Sodinokibi є ще одним прикладом моделі ЗПЗ-як-сервіс – група зловмисників платять розробникам ЗПЗ, які надають їм останню версію програми та забезпечують усю необхідну інфраструктуру для атаки. Для запобіганню виявленню ЗПЗ дінамічно генерує таблиці з даними для імпорту своїх файлів, що робить його складним для статичного аналізу та під час стартової фази воно дешифрує зашифрований файл своєї конфігурації використовуючи RC4, який містить інформацію про C2 домен та один із публічних ключів шифрування який буде використовуватися ЗПЗ для шифрування системи. Також майже весь виконуваний код ЗПЗ є шифрованим та дешифрується під час виконання програми. Після цього ЗПЗ перевіряє розкладку клавіатуру та мову які

використовує користувач. Як що вони не знаходять у «безпечних» зонах вказаних у конфігураційному файлі, ЗПЗ починає атаку з генерації публічного та приватного ключів використовуючи алгоритм Еліптичної кривої Діффі-Хеллмана. ЗПЗ зберігає ключі та іншу важливу інформацію у реєстрі для використання при подальших запусках. Більш нові версії ЗПЗ також перезавантажують ОС у безпечний режим що запобігає запуску антивірусного ПЗ. Після чого Sodinokibi відправляє інформацію про інфіковану машину та приватний ключ зашифровані алгоритмом AES на С2 сервер та проводить шифрування системи алгоритмом Salsa20. [27] У 2021 році компанії які займаються кібербезпекою у співпраці з правоохоронними органами у США створили безкоштовний універсальний дешифратор, який дозволить відновити зашифровані файли для усіх жертв цього ЗПЗ, які постраждали раніше ніж липень 2021 року. Але через постійні модифікації до ЗПЗ, для актуальних версій Sodinokibi цей універсальний дешифратор не працює, що робить загрозу атаки від цього ПЗ досі актуальною незважаючи на розробку дешифратора. ЗПЗ розповсюджувалося завдяки фішинговим атакам а найбільші атаки включають JBS Food у травні 2019 року через яку повністю зупинилися декілька заводів з обробки їжі, Travelex у грудні 2019 року через яку зупинилась робота систем глобальної валютної компанії на декілька тижнів і під час якої зловмисники вимагали викуп у розмірі шести мільйонів доларів та Acer у березні 2021 року під час якої зловмисники вимагали платіж у розмірі п’ятдесяти мільйонів, вплинули на ланцюг поставок компанії та порушила її роботу. [28] Для перевірки у роботі буде використовуватися одна з варіації Sodinokibi з SHA-256: c31f2aad3e0e00d5fe09571d3a82663025da6715974807b7763bb22d6907f1df

Conti – ЗПЗ-вимагач який базується на вихідному коді ЗПЗ Ryuk відповідального за масивні атаки на державні заклади, заклади охорони здоров’я та заклади освіти. Розробники Conti також використовували модель ЗПЗ-як-послуга, надаючи зловмисникам доступ до готових інструментів та ЗПЗ для атаки в обмін на частину викупу після вдалої атаки. Conti у перше било виявлено у 2019 році під час атаки на заклади охорони здоров’я, мережі швидкого реагування, правоохоронні ограни у США та більш ніж 400 компаній по всьому світі. Сума

викупу підлаштовувалась під жертв ЗПЗ – у травні 2021 року при атаці на вендора ПЗ для бекапу ExaGrid зловмисники вимагали сім мільйонів доларів та потім компанія домовилась на три мільйона доларів і заплатила зловмисникам викуп, і також у травні 2021 року при атаці на Ірландське Health Services Executive (HSE) зловмисники вимагали двадцять мільйонів доларів але компанія відмовилась платити. За даними ФБР вимоги Conti досягають двадцяти п'яти мільйонів доларів, що робить це ЗПЗ найприбутковішим та найагресивнішим ЗПЗ-вимагачем. У лютому 2022 року група зловмисників Conti висловила свою повну підтримку Російської влади та вторгнення у Україну, після чого один із членів групи, який висловив свою підтримку Україні, виклав у мережу близько шістдесяти тисяч повідомлень із внутрішніх журналів чатів ПЗ Jabber разом з вихідним кодом Conti та інших внутрішніх проектів групи зловмисників які вони використали від час атак. У квітні 2021 року на ОС Linux було виявлено ЗПЗ, яке було як версія ЗПЗ Conti код якої скомпільований для роботи в Linux та націлений на сервера віртуалізації ESXI. [29] З інструкцій по використанню ЗПЗ можна побачити що воно здатне шифрувати файли по вказаному шляху, вбивати усі процеси які заважають шифруванню, вимикає усі віртуальні машини з можливістю задавати виключення у разі якщо це ESXI, та працювати у фоновому режимі окремо від терміналу. [30] З витоку повідомлень 2022 року можна побачити що у групи зловмисників були проблеми з дешифратором файлів через що жертви та самі зловмисники у деяких випадках не могли дешифрувати файли після атаки ЗПЗ. Через що під час однієї з атак на Linux машини компанії Conti з початку вимагали викуп у двадцять мільйонів доларів, але потім погодились на один мільйон доларів через те що дешифратор ЗПЗ працював не так як треба а також деякі жертви просто відмовлялися від дешифратору зловмисників через те що їм вдалося дешифрувати файли самостійно. Остання велика жертва Conti була виявлена у січні 2022 року що свідчить про те що ЗПЗ досі активне і використовується для атак. Для шифрування ЗПЗ використовує бібліотеку OpenSSL та вбудований у неї алгоритм симетричний Salsa20, генеруючи новий випадковий ключ для кожного файлу що шифрується. Після чого файл з усіма ключами Salsa20 шифруються публічним ключем RSA

зловмисників. [31] Для перевірки у роботі буде використовуватися одна з варіації Conti з SHA-256:
8b57e96e90cd95fc2ba421204b482005fe41c28f506730b6148bcef8316a3201

2.2 Аналіз сигнатурного методу виявлення на прикладі антивірусних рішень для Linux.

Для аналізу сигнатурного методу виявлення шкідливого ПЗ буде використано ClamAV. ClamAV – це найпопулярніше та найпоширеніше антивірусне рішення для ОС Linux, яке підтримується Cisco та використовується у їх платних комплексних рішеннях як Cisco Endpoint Protection. Його популярність обумовлена тим що це безкоштовне пз з відкритим кодом, яке пропонує надійний захист з використанням сигнатурного виявлення завдяки швидкому скануванню та базою сигнатур котра постійно оновлюється та також модерується Cisco. [32]

Переваги:

- Безкоштовність та відкритий код – ClamAV є безкоштовним ПЗ з відкритим вихідним кодом, що робить його доступним рішенням для широкого кола користувачів.

- Ефективність – призначений спеціально для систем під керування ОС Linux демонструє високу ефективність виявлення загроз для цієї системи.

- Розроблений з головним фокусом для ОС Linux, але також працює та може використовуватися для інших ОС включаючи Windows та MacOS.

- Недоліки:

- Обмеженість у виявленні нових загроз – ClamAV спирається виключно на сигнатурний метод виявлення загроз, відповідно він обмежений у виявленні нових загроз, сигнатури яких ще не потрапили у базу.

- Потреба у постійному оновленні – ефективність ClamAV напряму залежить від регулярних оновлень сигнатур для виявлення нових загроз. Також часто користувачі встановлюють рекомендовану версію із стандартного репозиторію системи, який зазвичай містить застарілу версію програми, що призводить до проблем.

- Довге та трудомістке налаштування та відсутність дії за замовчуванням – ClamAV буде виявляти загрози та шкідливі файли, але за замовчуванням він з ними нічого не робить та не попереджає користувача, що є дуже небезпечним для звичайного користувача бо це не тільки не захищає його, але й надає хибне відчуття безпеки.

Налаштування, використання та перевірка:

Компоненти ClamAV:

ClamAV Daemon (clamd) – багатопотоковий демон для сканування на наявність зловмисного ПЗ, налаштовується через файл конфігурації clamd.conf та використовує локальну базу сигнатур. Відповідає за аналіз файлів на запити у реальному часі.

ClamAV Scanner (clamscan) – інтерфейс командного рядка, що надає можливість користувачу вручну сканувати файли або директорії для виявлення потенційних загроз. Також налаштовується при плануванні регулярних сканувань.

ClamAV Update (freshclam) – компонент який відповідає за оновлення бази даних сигнатур яку буде використовувати ClamAV. Періодично отримує оновлення з серверів ClamAV (якщо була вибрана ця база сигнатур) і забезпечує актуальність бази сигнатур, а відповідно і ефективність самого антивірусного ПЗ. Налаштовується файлом конфігу freshclam.conf.

База сигнатур – основний елемент для забезпечення захисту системи. Це безпосередньо сама база сигнатур яка містить відомі сигнатури вірусів та іншого шкідливого програмного забезпечення. Сигнатури зберігаються у виді CVD (ClamAV Virus Database) – це контейнер з цифровим підписом, який інкапсулює сигнатури вірусів, що забезпечує захист бази сигнатур від стороннього редагування. А також підтримує YARA формат.

ClamAV Milter (clamav-milter) – компонент який дозволяє інтегрувати ClamAV з серверами електронної пошти. Milter (mail filter – поштовий фільтр) сканує поштові листи на наявність вірусів та може опціонально фільтрувати спам та фішинг повідомлення. Налаштовується файлом конфігу clamav-milter.conf.

ClamAV GUI (clamtk) – графічний інтерфейс який надає користувачам спосіб налаштування та використання ClamAV без консольних команд у графічній оболонці.

Перевірка роботи ПЗ:

Для першої перевірки буде використана версія та налаштування ClamAV які надаються у стандартному репозиторію Ubuntu 20.04 та скоріш за все будуть встановлені більшістю користувачів. Це версія ClamAV 0.103.9.

Одразу при спробі оновлення бази сигнатур за допомогою freshclam виникає повідомлення про те що версія є застарілою, як можна побачити на рисунку 2.1.

```
sadmin@UbuntuClient:~$ sudo freshclam
Mon Dec  4 00:35:33 2023 -> ClamAV update process started at Mon Dec  4 00:35:33 2023
Mon Dec  4 00:35:33 2023 -> ^Your ClamAV installation is OUTDATED!
Mon Dec  4 00:35:33 2023 -> ^Local version: 0.103.9 Recommended version: 0.103.11
Mon Dec  4 00:35:33 2023 -> DON'T PANIC! Read https://docs.clamav.net/manual/Installing.html
Mon Dec  4 00:35:33 2023 -> daily.cvd database is up-to-date (version: 27112, sigs: 2048202,
r: raynman)
Mon Dec  4 00:35:33 2023 -> main.cvd database is up-to-date (version: 62, sigs: 6647427, f-le
igmgr)
Mon Dec  4 00:35:33 2023 -> bytecode.cvd database is up-to-date (version: 334, sigs: 91, f-le
nvilleg)
```

Рисунок 2.1 – Повідомлення про застарілу версію ClamAV у репозиторію

Тобто звичайний користувач за замовчуванням буде використовувати застарілу версію ПЗ.

Після запуску активного сканування, ClamAV відчутно навантажив систему, використовуючи 30% усіх 4х виділених віртуальній системі ядер процесора, як видно на рисунку 2.2.

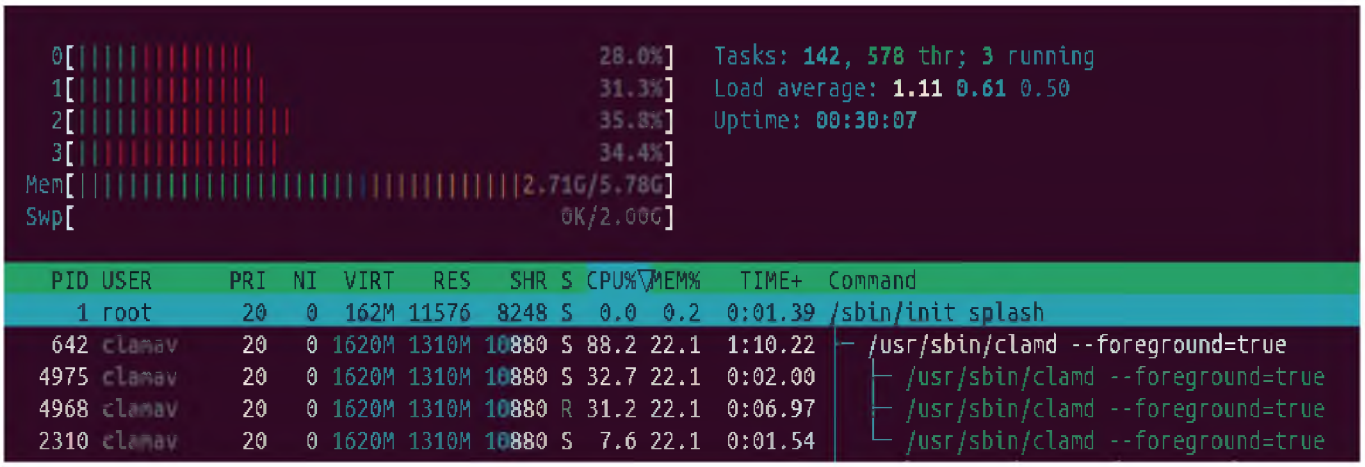


Рисунок 2.2 – Навантаження на систему при скануванні ClamAV

Як можна побачити на рисунку 2.3 та у таблиці 2.1, скануванню однієї директорії зайняло 2 секунди, з та з 10 відомих шкідливих або потенційно шкідливих виконавчих файлів, було виявлено 7.

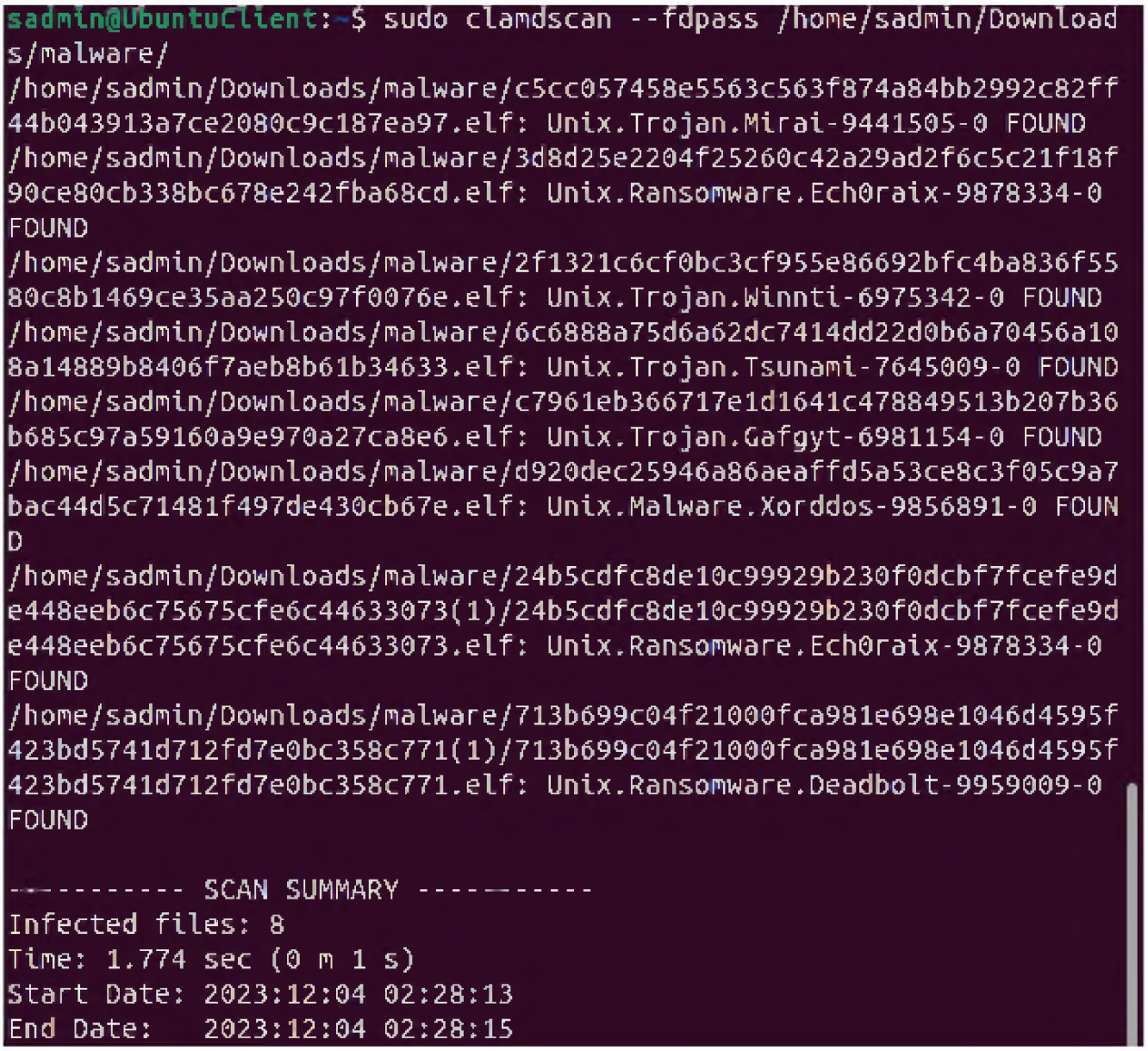


Рисунок 2.3 – Результати сканування ClamAV з репозіторію

Таблиця 2.1 – Результати сканування ClamAV з репозиторію

Sha-256	Група вірусу	Тип вірусу	Виявлен
6c6888a75d6a62dc7414dd22d0b6a70456a108a14889b8406f7aeb8b61b34633	Tsunami	Ransomware	Так
713b699c04f21000fca981e698e1046d4595f423bd5741d712fd7e0bc358c771	Hive	Ransomware	Так
556e5cb5e4e77678110961c8d9260a726a363e00bf8d278e5302cb4bfccc3eed	HelloKitty	Ransomware	Ні
24b5cdfc8de10c99929b230f0dcbf7fcefe9de448eeb6c75675cfe6c44633073	eCh0raix	Ransomware	Так
3d8d25e2204f25260c42a29ad2f6c5c21f18f90ce80cb338bc678e242fba68cd	eCh0raix	Ransomware	Так
2f1321c6cf0bc3cf955e86692bfc4ba836f5580c8b1469ce35aa250c97f0076e	Winnti	Malware	Так
d920dec25946a86aeaffd5a53ce8c3f05c9a7bac44d5c71481f497de430cb67e	XorDDoS	Trojan	Так
c7961eb366717e1d1641c478849513b207b36b685c97a59160a9e970a27ca8e6	Gafgyt	Trojan	Так
c5cc057458e5563c563f874a84bb2992c82ff44b043913a7ce2080c9c187ea97	Mirai	Ransomware	Так
c31f2aad3e0e00d5fe09571d3a82663025da6715974807b7763bb22d6907f1df	Sodinokibi	Ransomware	Ні
8b57e96e90cd95fc2ba421204b482005fe41c28f506730b6148bcef8316a3201	Conti	Ransomware	Ні

Тепер для порівняння буде використовуватися ClamAV останньої версії 1.2.1 встановлений вручну із пакету.

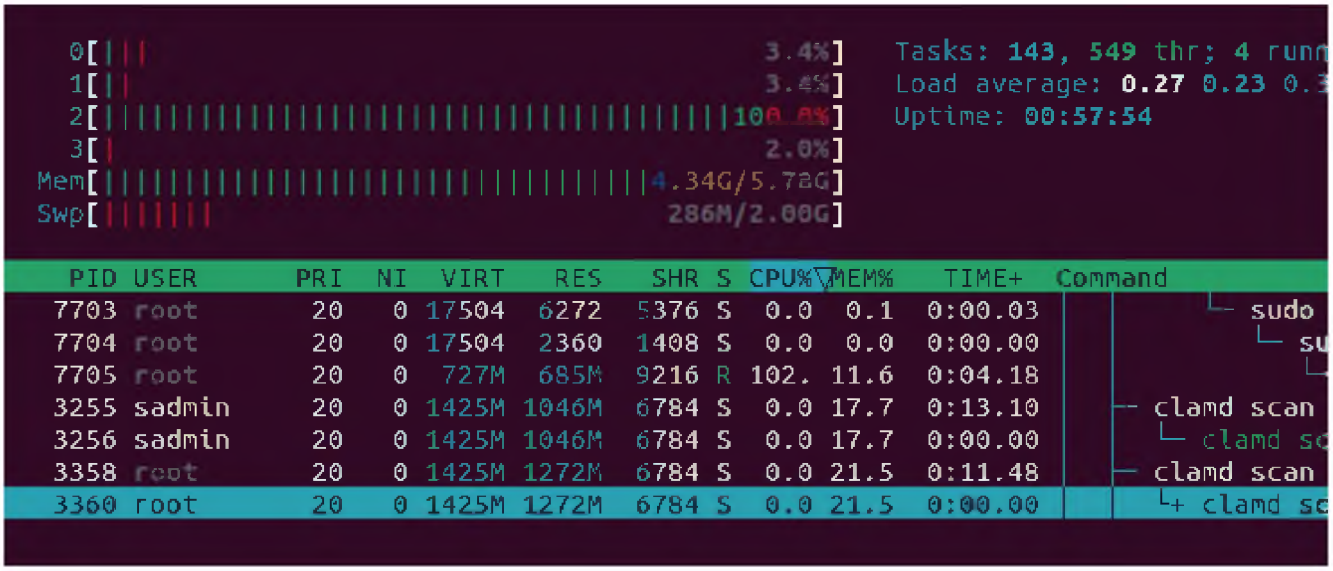


Рисунок 2.4 – Навантаження системи при скануванні новим ClamAv

Як можна побачити на рисунку 2.4, нова версія ClamAV також досить суттєво навантажує систему але, на відміну від старої версії, тільки одне ядро процесора.

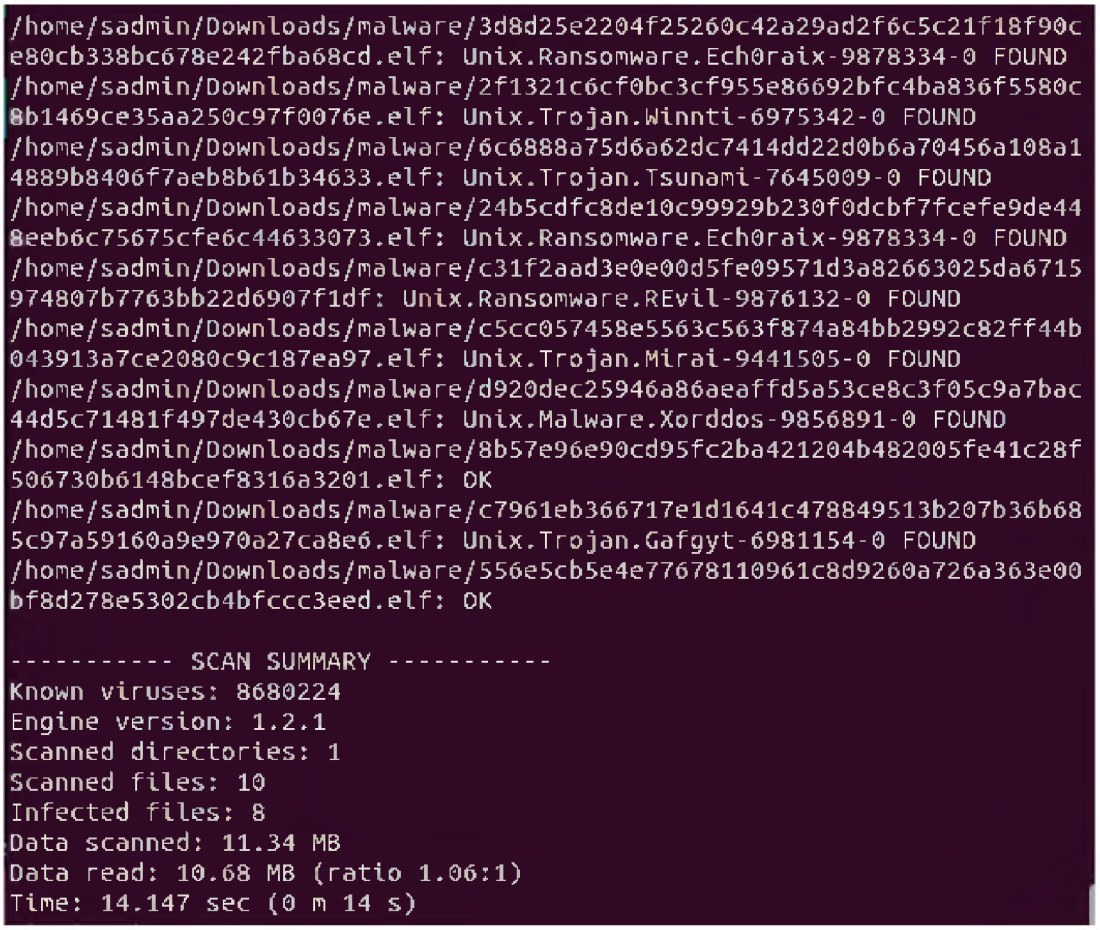


Рисунок 2.5 – Результати сканування нового ClamAV

Як можна побачити на рисунку 2.5 та у таблиці 2.2, сканування одної теці зайняло 14 секунд та з 10 відомих шкідливих або потенційно шкідливих виконавчих файлів, було виявлено 8.

Таблиця 2.2 – Результати сканування нового ClamAV

Sha-256	Група вірусу	Тип вірусу	Виявлен
6c6888a75d6a62dc7414dd22d0b6a70456a108a14889b8406f7aeb8b61b34633	Tsunami	Ransomware	Так
713b699c04f21000fca981e698e1046d4595f423bd5741d712fd7e0bc358c771	Hive	Ransomware	Так
556e5cb5e4e77678110961c8d9260a726a363e00bf8d278e5302cb4bfccc3eed	HelloKitty	Ransomware	Ні
24b5cdfc8de10c99929b230f0dcbf7fcefe9de448eeb6c75675cfe6c44633073	eCh0raix	Ransomware	Так
3d8d25e2204f25260c42a29ad2f6c5c21f18f90ce80cb338bc678e242fba68cd	eCh0raix	Ransomware	Так
2f1321c6cf0bc3cf955e86692bfc4ba836f5580c8b1469ce35aa250c97f0076e	Winnti	Malware	Так
d920dec25946a86aeaffd5a53ce8c3f05c9a7bac44d5c71481f497de430cb67e	XorDDoS	Trojan	Так
c7961eb366717e1d1641c478849513b207b36b685c97a59160a9e970a27ca8e6	Gafgyt	Trojan	Так
c5cc057458e5563c563f874a84bb2992c82ff44b043913a7ce2080c9c187ea97	Mirai	Ransomware	Так
c31f2aad3e0e00d5fe09571d3a82663025da6715974807b7763bb22d6907f1df	Sodinokibi	Ransomware	Так
8b57e96e90cd95fc2ba421204b482005fe41c28f506730b6148bcef8316a3201	Conti	Ransomware	Ні

У порівнянні з більш старою версію, новий ClamAV виявив на 1 загрозу більше аніж стара версія. Два шкідливих файла, які були помічені як безпечні це тіж самі файли, які були також помічені як безпечні старою версією програмою/

База сигнатур ClamAV.

Для того щоб виявляти шкідливі файли ClamAV використовує базу сигнатур яка містить сигнатури вже відомих загроз. За стандартом використовується база сигнатур самого ClamAV, яку підтримує та регулярно оновляє Cisco, але також є можливість підключення не офіційних баз сигнатур та створення власних. База ClamAV представлена у виді CVD (ClamAV Virus Database) файлу який являє собою контейнер з цифровим підписом, що захищає файл від редагування зловмисниками або шкідливим ПЗ.

Для роботи с файлом бази сигнатур використовується утиліта sigtool, яка дозволяє працювати з файлом. У першу чергу як на рисунку 2.6, завдяки утиліти sigtool можна перевірити статус локальної бази сигнатур – дату побудови, версію, кількість сигнатур, хеш суму, цифровий підпис та статус.

```
sadmin@UbuntuClient:/tmp/clamsigs$ sigtool --info /usr/local/share/clam
av/main.cvd
File: /usr/local/share/clamav/main.cvd
Build time: 16 Sep 2021 08:32 -0400
Version: 62
Signatures: 6647427
Functionality level: 90
Builder: sigmgr
MD5: 137eccce31aacb21b5a98bb8c21cefd6
Digital signature: twaJBls8V5q64R7QY10AatEtPNuPWoVoxTaNO1jpBg7s5jIMMXpi
tgG1000YLp6rb0TWkEKjRqxneGTxuxWaWm7XBjsgwX2BRWh/y4fhs7uyImdKRLzQ5y8e2Ek
SChegF/i8clqfn+1qetq9j4gbktJ3JZpOXPoHlyr2Dv9S/Bg
Verification OK.
```

Рисунок 2.6 – Дані про бази сигнатур ClamAV

Утиліта sigtool дозволяє розпакувати файл бази сигнатур та перевірити його зміст. Як видно на рисунку 2.7, база складається з купи різних файлів:

- main.fp, main.sfp – файли зі списками дозволених файлів;
- main.hdb, main.hsb – файли з хеш сумами шкідливих файлів;
- main.mdb, main.msb – файли з хеш сумами PE (portable executable) файлів;
- main.crb – сигнатури довірених PE файлів;

- main.ldb – логічні сигнатури (об'єднання сигнатур завдяки логічним операндам);

- main.cdb – сигнатури для перевірки файлів у контейнері (наприклад zip архив);

- main.info – файл з інформацією про базу сигнатур;

- COPYING – файл з угодою користувача.

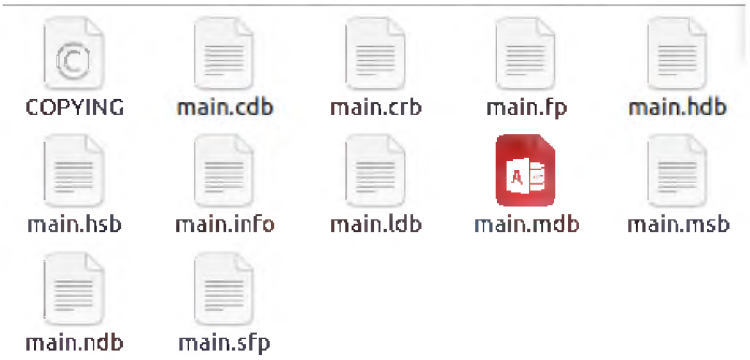


Рисунок 2.7 – Зміст CVD файлу бази сигнатур

Після розпакування можна відкрити у якому зберігаються хеш суми вірусного ПЗ – рисунок 2.8, та вручну його переглянути і виконати пошук за сигнатурою.

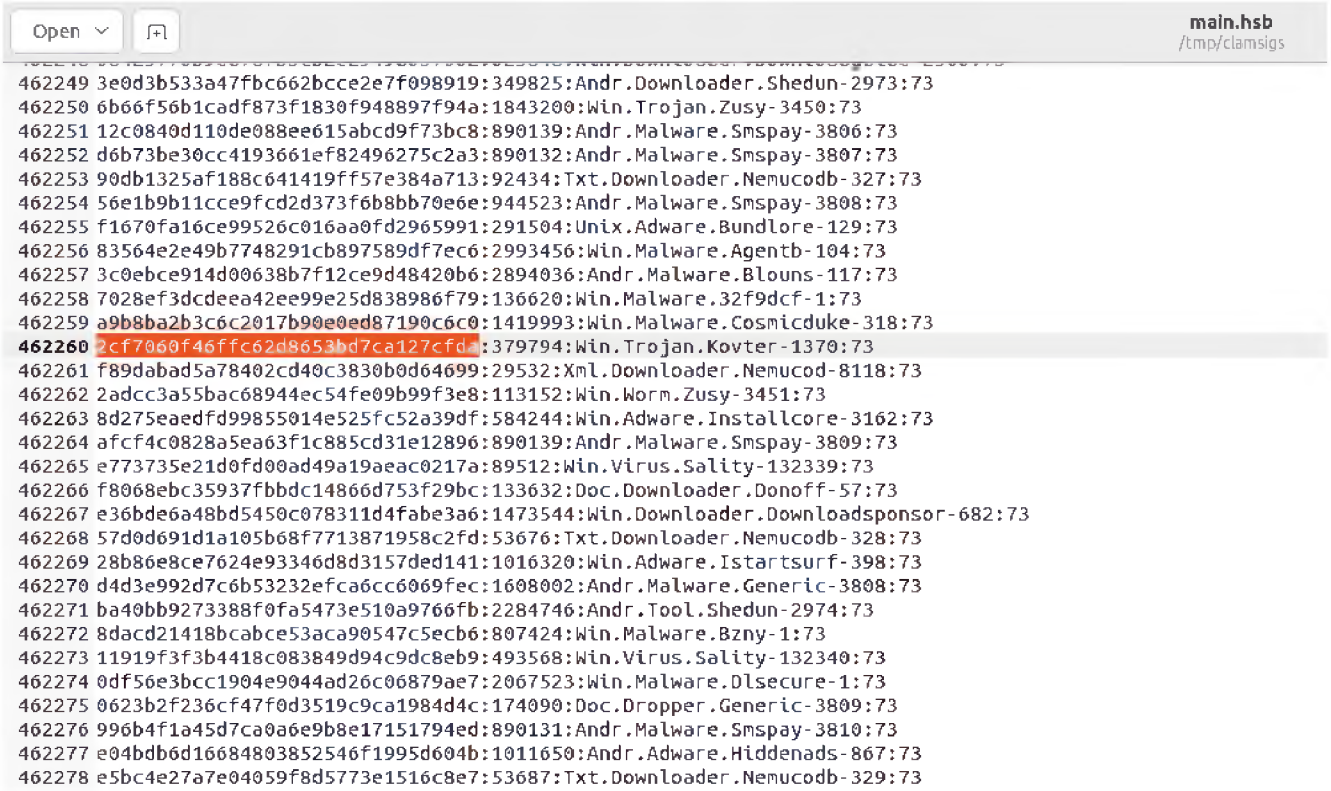


Рисунок 2.8 – Зміст hsb файлу сигнатур ЗПЗ

Також утиліта sigtool дозволяє генерувати файли сигнатур з локальних файлів для використання їх у локальній базі сигнатур. На прикладі одного з виконавчих файлів ЗПЗ, якого не має у базі сигнатур ClamAV – HelloKitty, за допомогою утиліти sigtool можна згенерувати .hdb файл – рисунок 2.9, який буде містити md5 хеш суму файлу, розмір файлу та назву шкідливого ПЗ яке за стандартом буде просто іменем файлу.

Рисунок 2.9 – Сигнатура виконавчого файлу ЗПЗ HelloKitty

Після оновлення та налаштування ClamAV його можна перемкнути у режим антивірусного ПЗ змінивши налаштування у його конфігураційному файлі:

ScanOnAccess true

OnAccessRemove yes

Що змусить його автоматично видаляти виявлені шкідливі чи потенційно шкідливі файли, або

ScanOnAccess true

OnAccessPass /тека_карантіну/

Що змусить його переміщати виявлені шкідливі чи потенційно шкідливі файли до теки карантину та ізолювати їх.

2.3 Вивчення методів аномалійного виявлення зловмисного програмного забезпечення в Linux. Системи IDS/IPS.

Одним з прикладів аномалійного виявлення ЗПЗ це системи IDS/IPS – системи виявлення проникнення та системи запобігання проникненню. Обидва типу систем виявляють незвичайну або потенційно шкідливу активність та у випадку системи виявлення відправляють сигнал тривоги користувачу або на централізований сервер, а у випадку систем запобігання намагається або

ліквідувати загрозу, або від'єднує систему від мережі та ізолює її для запобіганню розповсюдження атаки або шкідливого програмного забезпечення. [33]

Системи IDS та IPS можуть виявлять вторгнення двома методами:

- Виявлення на основі сигнатури – система постійно сканує систему та при виявлені події що співпадають з відомими сигнатурами атак, надсилає попередження адміністратору або застосує дії для попередження атаки. Але на відміну від антивірусного ПЗ, системи IDS та IPS виконують сканування не виконавчих файлів, а мережевого трафіку, файлів журналювання та змін у системних або користувацьких файлах. Виявлення відбувається завдяки порівнянню с базою відомих сигнатур, тому воно зазвичай є точним та швидким, але не працює для нових загроз, сигнатур яких немає у базі.

- Виявлення на основі аномалій – система визначає вторгнення завдяки виявленню нестандартної поведінки або змін у системі, що дозволяє виявляти нові невідомі загрози, на відміну від сигнатурного методу. Для роботи виявлення на основі аномалій, спочатку визначається стандартні неаномальні стан системи та поведінка користувача та потім проводиться постійно зрівняння з цим базовим неаномальним станом для виявлення відхилень, що може бути як і незвичайно великий об'єм мережевого трафіку, так і вхід у систему користувача на вихідних при тому що він працює тільки по будніх днях.

Безпосередньо методи якими IDS та IPS системи визначають загрози можна поділити на:

1. NIDS (network intrusion detection system) – система виявлення вторгнення у мережу. Це системи що займаються моніторингом мережевого трафіку завдяки апаратним або програмним сенсорам встановлених у саму мережу, та аналізують весь трафік що через ці сенсори проходить.

2. HIDS (host intrusion detection system) – система виявлення вторгнення хосту, яка розгорнута у самій системі для аналізу мережевого трафіку, що надає більше контролю та гнучкості для системного адміністратора, але може зробити важким централізоване спостереження за великою кількістю робочих станцій.

3. PIDS – protocol-based intrusion detection system – система виявлення вторгнень на основі протоколу. Ця система встановлюється на вході до серверу та аналізує увесь трафік що йде з серверу до робочих станції та з робочих станцій на сервер по визначеному мережевому протоколу, такому як HTTP або HTTPS.

4. APIDS – application protocol-based intrusion detection system - система виявлення вторгнень на основі прикладного протоколу – по суті роботи теж саме що й PIDS, але виконує моніторинг для кластеру серверів.

5. Гібридна система виявлення вторгнень – системи які зустрічаються найчастіше та які є комбінацією усіх або декількох з попередніх методів виявлення.

Систему IDS/IPS можна розглянути на прикладі ПЗ Suricata. Suricata це монітор мережевої безпеки який використовує набір сигнатур та правил створеними користувачами та спільнотою для того щоб аналізувати мережевий трафік. Suricata може генерувати події журналювання, підіймати тривогу та зупиняти трафік у разі виявлення підозрілих пакетів або запитів до різних служб, що працюють на сервері. За замовчуванням ПЗ працює як IDS, але також може бути налаштована як IPS.

Правила Suricata подаються у виді “ACTION HEADER OPTIONS” – Дія Заголовок Опції. Дія – це команди що виконуються у разі виявлення трафіку що співпадає з правилом, Заголовок – описує хости, IP адреси, порти, протоколи та напрямок трафіку, Опції – вказують такі речі як ID правила, повідомлення для журналювання, регулярні вираження які підходять під зміст пакетів, класифікацію та інші модифікатори які полегшують виявлення легітимного та підозрілого трафіку.

Після встановлення та налаштування набору правил, Suricata буде створювати вести свій журнал дії у системі. Для перевірки у системі буде запущено скрипт який емулює витік даних облікового запису рут користувача, що змусить Suricata створити подію тривоги у журналі, через те що це порушить правило. Файл журналу можна як намагатися читати у сирому виді як він генерується, або використовувати утілити як jq, які роблять файл журналу набагато легшим для розуміння людиною.

```
{
  "timestamp": "2023-12-11T16:54:21.594319+0200",
  "flow_id": 1526330698778296,
  "in_iface": "enp0s3",
  "event_type": "alert",
  "src_ip": "10.0.2.15",
  "src_port": 54216,
  "dest_ip": "108.157.4.121",
  "dest_port": 80,
  "proto": "TCP",
  "pkt_src": "wire/pcap",
  "community_id": "1:/oax6FxhNLQuT1KJOOksNvQmb4o=",
  "tx_id": 0,
  "alert": {
    "action": "allowed",
    "gid": 1,
    "signature_id": 2013028,
    "rev": 7,
    "signature": "ET POLICY curl User-Agent Outbound",
    "category": "Attempted Information Leak",
    "severity": 2,
    "metadata": {
      "created_at": [
        "2011_06_14"
      ],
      "updated_at": [
        "2022_05_03"
      ]
    }
  },
```


Рисунок 2.10 – Подія тривоги у журналі Suricata

Як видно на рисунку 2.10, скрипт який емулює витік даних облікового запису визвав подію тривоги у Suricata. У події можна побачити тип події - тривога, вхідну та вихідну адресу, протокол - TCP, порт - 80, id правила - 2100498, опис сигнатури – правила що стосуються утілити curl , категорію – спроба витоку інформації та дату створення події.

Правило 2100498 у повному обсязі виглядає так:

alert ip any any -> any any (msg:"GPL ATTACK_RESPONSE id check returned root"; content:"uid=0|28|root|29|"; classtype:bad-unknown; sid:2100498; rev:7; metadata:created_at 2010_09_23, updated_at 2010_09_23;)

Перша частина правила alert – тривога це дія що виконується коли правило порушено. Замість тривоги Suricata також може виконувати наступні дії:

- Pass – пропуск – Suricata зупинить сканувати пакети такого типу та буде їх дозволяти без створення події у журналі.

- Drop – сброс – ПЗ негайно припинить аналіз пакету при виявленні та згенерує подію тривоги у журналі та при застосуванні TCP протоколу, TCP протокол отримує тайм аут, тобто час очікування завершиться.

- Reject – відказ – ПЗ відправить TCP reset packet (RST), який сигналізує про те що відправник пакету не буде більше приймати та отримувати дані, що запобіжить цьому шкідливому чи потенційно шкідливому з'єднанню.

Саме режими Drop та Reject перетворюють Suricata з системи виявлення вторгнення у систему запобігання вторгненню, так як ПЗ починає запобігати небажаному мережевому трафіку.

Заголовок правила представлено у виді:

<PROTOCOL> <SOURCE IP> <SOURCE PORT> -> <DESTINATION IP> <DESTINATION PORT>

Де <PROTOCOL> відповідає за протокол що використовується - TCP, UDP, ICMP, IP або інші протоколі. А <SOURCE IP> <SOURCE PORT> та <DESTINATION IP> <DESTINATION PORT> це вхідні та вихідні порти та IP адреси. Це дозволяє використовувати правила для конкретного діапазону IP адрес чи одного конкретного IP адреса або завдяки ключовому слову any – для усіх адрес.

Аргументи у дужках відповідають за опції правила та дозволяють задавати повідомлення для журналу, id події, дату та інше. Одної з головних опції є content – наповнення, що відповідає саме за зміст пакету який аналізується. У цьому випадку це content:"uid=0|28|root|29|" який відповідає за перевірку мережевих пакетів на наявність даних про кореневий обліковий запис які поступають або відправляються на будь які мережеві адреси з будь-яких портів, бо це може свідчити про виток облікових даних кореневого запису.

Після налаштування серверу Suricata, на робочих станціях встановлюється на налаштовується брандмауер, якій буде відправляти увесь, або обраний мережевий трафік на сервер з Suricata, що дозволить Suricata аналізувати та фільтрувати мережевий трафік з робочих станції у режимі IDS або IDS в залежності від потреби.

Одним з рішень для налаштування брандмауеру на робочих станціях є UFW (uncomplicated firewall - нескладний брандмауер) який являє собою зручний інтерфейс для керування вбудованою утилітою iptables, та постачається за замовчуванням у ОС Ubuntu та Debian.

Для цього буде використовуватися режим ядра nfqueue у iptables, що значить Netfilter Queue (черга мережевого фільтру) який дозволяє аналізувати пакети, фільтрувати трафік та формувати мережевий трафік за потребою. NFQUEUE засовується як і безкоштовним IDS/IPS рішеннями з відкритим кодом у випадку Suricata, так і комерційними рішеннями з закритим кодом.

Для налаштування Suricata у режимі IDS, у конфігураційному файлі Suricata ПЗ встановлюється у режим прослуховування nfqueue: LISTENMODE=nfqueue, замість стандартного LISTENMODE=af-packet, де af-packet це порт мережевих пакетів, якій встановлює міст між двома хоста та копіює мережевий трафік з одного хоста на інший. А на робочій станції у конфігураційному файлі брандмауера UFW задаються правила:

-I INPUT 1 -p tcp --dport 22 -j NFQUEUE --queue-bypass

-I OUTPUT 1 -p tcp --sport 22 -j NFQUEUE --queue-bypass

-I FORWARD -j NFQUEUE

-I INPUT 2 -j NFQUEUE

-I OUTPUT 2 -j NFQUEUE

Де перші два правила обходять сервер Suricata по 22 порту, що дозволяє підключитися до робочої станції використовуючи SSH у разі якщо сервер Suricata не працює, бо у цьому випадку увесь трафік з робочою станції буде відправлятися у нікуди. FORWARD правило гарантує, що якщо сервер діє як шлюз для інших систем, то увесь цей трафік також буде надходитиме до серверу Suricata для обробки. А останні два правила відповідають за те що увесь інший не SSH трафік було направлено до Suricata.

Застосувавши ці зміни та змінивши правила для підозрілого трафіку з режиму попередження alert на режим блокування зі створенням події тривоги drop, можна перевірити роботу Suricata запустивши той самий скрипт, якій емулює виток

облікових даних кореневого користувача, після чого на робочій станції буде отримано помилку перевищення часу очікування:

curl: (28) Operation timed out after 5000 milliseconds with 0 out of 39 bytes received

А у журналі Suricata було створено повідомлення тривоги про потенційно шкідливий трафік, але на відміну від попереднього тестування замість дії allow – дозволити, була виконана дія blocked – заблоковано як можна побачити на рисунку 2.11.


```
{
  "timestamp": "2023-12-11T17:30:46.595940+0200",
  "flow_id": 1526330698778296,
  "in_iface": "enp0s3",
  "event_type": "alert",
  "src_ip": "108.157.4.121",
  "src_port": 80,
  "dest_ip": "10.0.2.16",
  "dest_port": 54216,
  "proto": "TCP",
  "pkt_src": "wire/pcap",
  "community_id": "1:/oax6FxhNLQuT1KJOOksNvQmb4o=",
  "alert": {
    "action": "blocked",
    "gid": 1,
    "signature_id": 2100498,
    "rev": 7,
    "signature": "GPL ATTACK_RESPONSE id check returned root",
    "category": "Potentially Bad Traffic",
    "severity": 2,
    "metadata": {
      "created_at": [
        "2010_09_23"
      ],
      "updated_at": [
        "2010_09_23"
      ]
    }
  }
```


Рисунок 2.11 – Подія тривоги у журналі Suricata працюючого у режимі IPS

Після налаштування серверу Suricata та робочих станції для можливості зручного перегляду файлів журналювання та можливості якось їх аналізувати окрім консольного перегляду, необхідно встановити додаткове ПЗ яке це дозволить.

Для простого зручного перегляду журналу подій буде використовуватися EveBox – інструмент оповіщення та керування подіями Suricata, який можна використовувати з існуючим набором компонентів ELK або як окремий менеджер подій Suricata, що буде використовувати його вбудовану базу даних SQLite для

невеликих розгортань або Elasticsearch/Opensearch для більших розгортань. [34] Застосування EveBox відбувається через веб інтерфейс, що також дозволяє переглядати події і тривоги у журналі Suricata з інших систем просто підключившись до веб інтерфейсу серверу Suricata як можна побачити на рисунку 2.12.

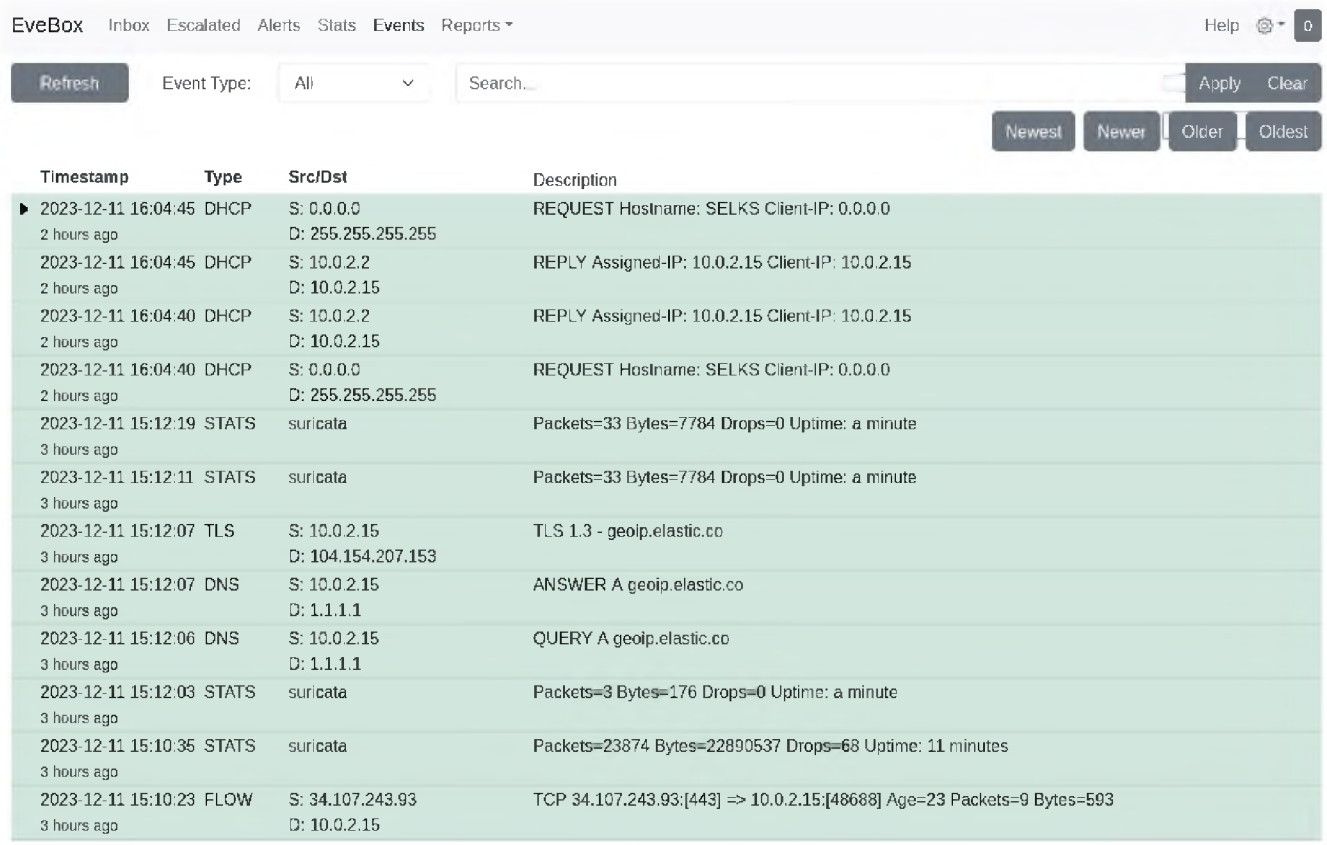


Рисунок 2.12 – Відображення подій на централізованому сервері

Для зручного перегляду та налаштування правил Suricata можна використовувати Scirius, який також є безкоштовною утилітою з відкритим кодом яка являє собою веб інтерфейс для перегляду та керування правилами Suricata. Завдяки тому що це веб інтерфейс, його також можна використовувати з інших робочих станції просто підключившись до нього з браузеру. Scirius дозволяє будувати набір правил Suricata з різних джерел і автоматичному режимі буде їх будувати та засовувати зміни, він також надає статистику правил дозволяє вимикати/вмикати правила окремо або по категоріям, як можна побачити на рисунку 2.13 та ставити пороги шуму для правил які спрацьовують надто часто щоб запобігти засміченню файлів журналу.

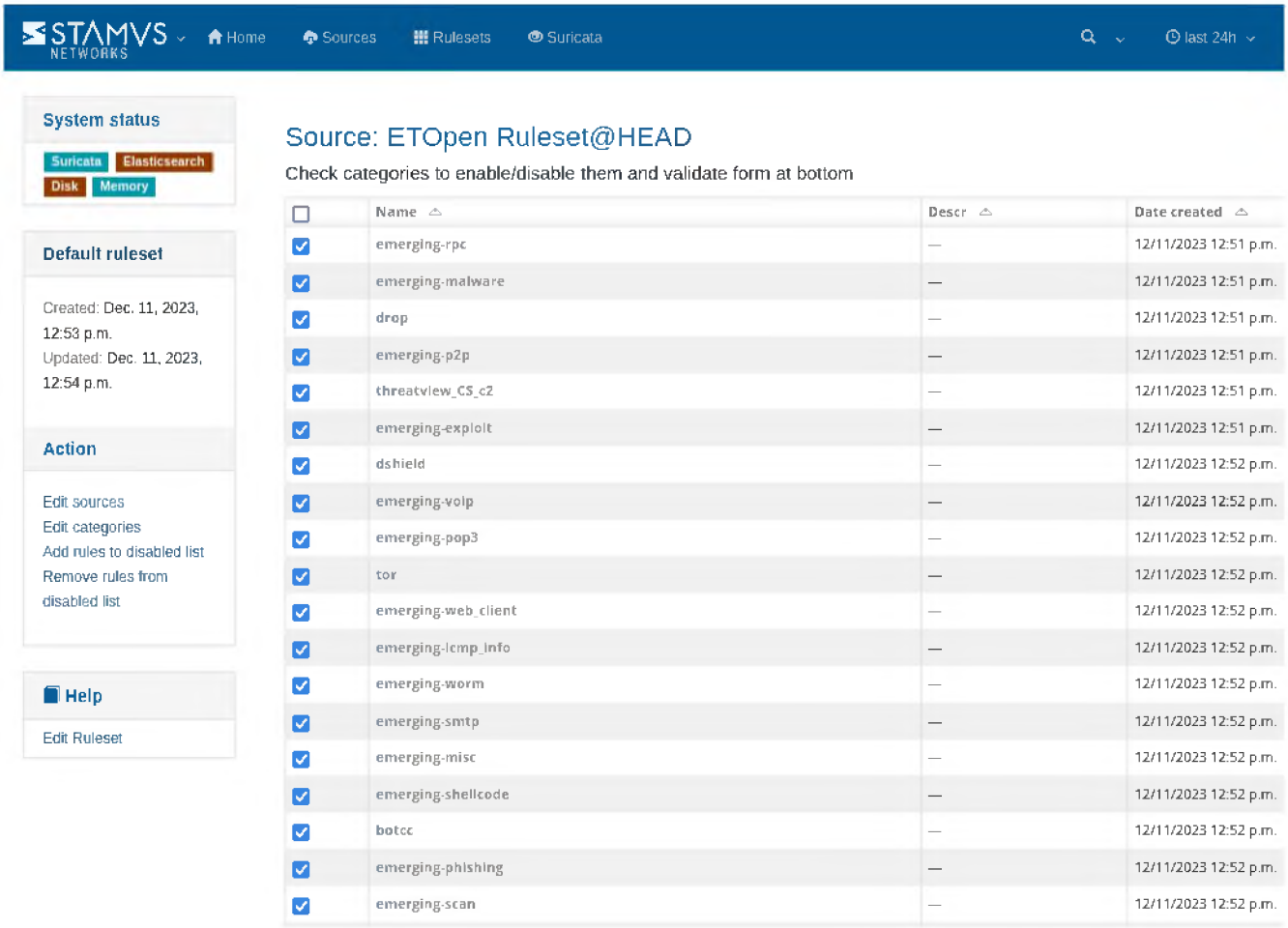


Рисунок 2.13 – Управління правилами Suricata через ПЗ Scirius

2.4 Методи аналізу зловмисного програмного забезпечення

2.4.1 Статичний аналіз. Використання технік декомпіляції та дизасемблювання для розкриття внутрішньої логіки зловмисного коду.

Статичний аналіз це практика вивчення бінарного коду зловмисного програмного забезпечення не запускаючи його. Техніки статичного аналізу дозволяють зрозуміти як працює ЗПЗ, його можливості і який тип шкоди він може нанести.

Статичний аналіз включає у себе три основні підходи:

- Дизасемблювання – процес перетворення бінарного коду програми у код асемблеру який може бути прочитаний людиною та подальша його реверс інженерія для дешифрування структури коду. Наприклад, якщо у зловмисного програмного забезпечення невідоме корисне навантаження, то завдяки дизасемблерам таким як IDA Pro або Ghidra можливо проаналізувати двійковий код

програми та знайти точку входу в програму, після чого аналіз коду дозволить побачити як корисне навантаження програми себе веде. Дизасемблювання також дозволяє виявити пакети зловмисного коду до яких зловмисники застосовують техніки обфускації та стиснення коду, які роблять вихідний код складним для розуміння та аналізу. Після чого можливо створення сигнатури для майбутньої ідентифікації цього шкідливого ПЗ.

- Декомпіляція – процес перетворення скомпільованого машинного коду програми назад у мови програмування високого рівня такі як C або C++ для того щоб було легше аналізувати логіку та структуру програми. Декомпіляція зазвичай проводиться разом з дизасемблюванням у тих самих програмах як IDA Pro або Ghidra, і так само допомагає виявити пакети зловмисного коду до яких було застосовано техніки обфускації та стиснення коду.

- Символьне виконання – техніка що використовує змінні та символічні значення для імітації виконання програми, що може бути корисним для розуміння того як шкідливий код взаємодіє з системою та змінює її поведінку. Техніка символьного виконання також може дозволяти виявляти вразливості нульового дня у програмного забезпеченні завдяки ідентифікації вхідних даних, які призводять до певних шляхів коду, якими можуть скористуватися зловмисники. [35]

Для демонстрування технік декомпіляції та дизасемблювання буде використано ПЗ Ghidra – це SRE (software reverse engineering - зворотного інженерування програмного забезпечення) фреймворк який створений та підтримується Управлінням Досліджень Агентства Національної Безпеки США. Цей фреймворк містить набір повноцінних високоякісних інструментів аналізу програмного забезпечення, які дозволяють аналізувати скомпільований код для різних платформ включаючи Windows, MacOS та Linux. До можливостей входять розбирання, збирання, декомпіляцію, побудову графіків, створення сценаріїв та інші. Підтримує широкий спектр наборів інструкції процесора та форматів виконуваних файлів і може працювати як в інтерактивному, так і в автоматизованому режимах. Також має підтримку розробки доповнень та сценаріїв на мовах Java та Python.

Ghidra була створення у підтримку місії АНБ у сфері кібербезпеки для вирішення проблем масштабування та об'єднання у комплексних зусиллях SRE, а також для забезпечення настоюваної та розширюваної дослідницької платформи SRE. АНБ засовувала можливості Ghidra для вирішення різноманітних проблем, пов'язаних з аналізом шкідливого коду та отримання глибокої інформації для SRE аналітиків, які прагнуть краще розуміти потенційні вразливості у мережах та системах. [36]

Для аналізу буде використовуватися нескладний Crackme – програми розроблені для тренування та перевірки навиків реверсивної інженерії. Вони створені іншими ентузіастами або спеціалістами реверсивної інженерії як легальний спосіб зламу програмного забезпечення для того щоб жодна компанія та реальне програмне забезпечення не зазнали шкоди. Але техніки які використовуються для обфускації та стиснення коду а також техніки їх виявлення та зламу ПЗ є реальними та застосовуються у існуючому програмного забезпеченні. [37]

Буде використовуватися легкий crackme з SHA-256: 5b8a37a433c5d45fc286ad83, створений користувачем cbm-hackers на мові C/C++ для Unix/Linux на архітектурі x86-64. [38]

Після інсталяції та запуску Ghidra, створюється новий проект у який імпортується файл Crackme:

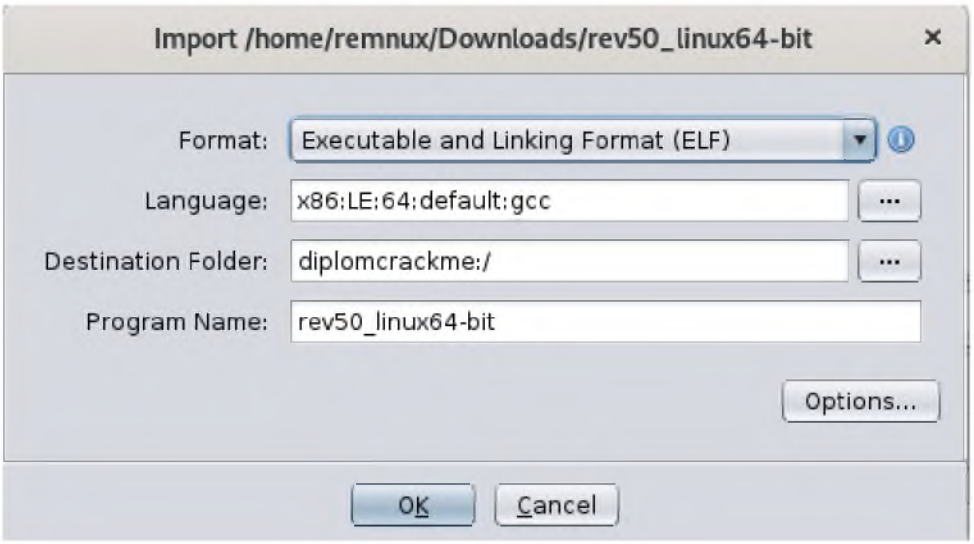


Рисунок 2.14 – Процес імпорту файлу у Ghidra

При імпорті обирається формат файлу, язик та архітектура для якої файл розроблений. Як можна побачити на рисунку 2.14, у цьому випадку було визначено формат файлу ELF для архітектури x86-64 з компілятором gcc.

Після імпорту файлу відображається уся технічна інформація про нього, як можна побачити на рисунку 2.15:

```
Project File Name:          rev50_linux64-bit
Last Modified:              Wed Dec 13 08:44:08 EST 2023
Readonly:                   false
Program Name:               rev50_linux64-bit
Language ID:                x86:LE:64:default (2.13)
Compiler ID:                gcc
Processor:                  x86
Endian:                     Little
Address Size:               64
Minimum Address:            00100000
Maximum Address:            _elfSectionHeaders::0000077f
# of Bytes:                 7805
# of Memory Blocks:         32
# of Instructions:          14
# of Defined Data:          114
# of Functions:             23
# of Symbols:               57
# of Data Types:            31
# of Data Type Categories:  2
Created With Ghidra Version:10.1.3
Date Created:               Wed Dec 13 08:44:08 EST 2023
ELF File Type:              shared object
ELF Original Image Base:    0x0
ELF Prelinked:              false
ELF Required Library [   0]:libc.so.6
ELF Source File [   0]:     crtstuff.c
ELF Source File [   1]:     rev_50.c
ELF Source File [   2]:     crtstuff.c
ELF Source File [   3]:
Executable Format:          Executable and Linking Format (ELF)
Executable Location:        /home/remnux/Downloads/rev50_linux64-bit
Executable MD5:             580bb78821c4b7cb20d060478771fd26
Executable SHA256:          bb28a152966bed0a369f30149a912982ea33b408794bfbd82e73c87ff4e184ff
FSRL:                       file:///home/remnux/Downloads/rev50_linux64-bit?MD5=580bb78821c4b7cb20d0604787
Relocatable:                true
```

Рисунок 2.15 – Технічна інформація про CrackMe файл

Після чого його можна відкрити та проаналізувати у браузері коду Ghidra. У браузері, як видно на рисунку 2.16, Ghidra провела дизасемблювання коду – код асемблера знаходиться у середньому вікні браузера. Спираючись на те що ця програма написана на мові C, слід зробити пошук головної функції main та обрати її у браузері коду. Після того як обрана функція main, можна також побачити декомпільований код цієї функції у правому вікну браузеру коду.

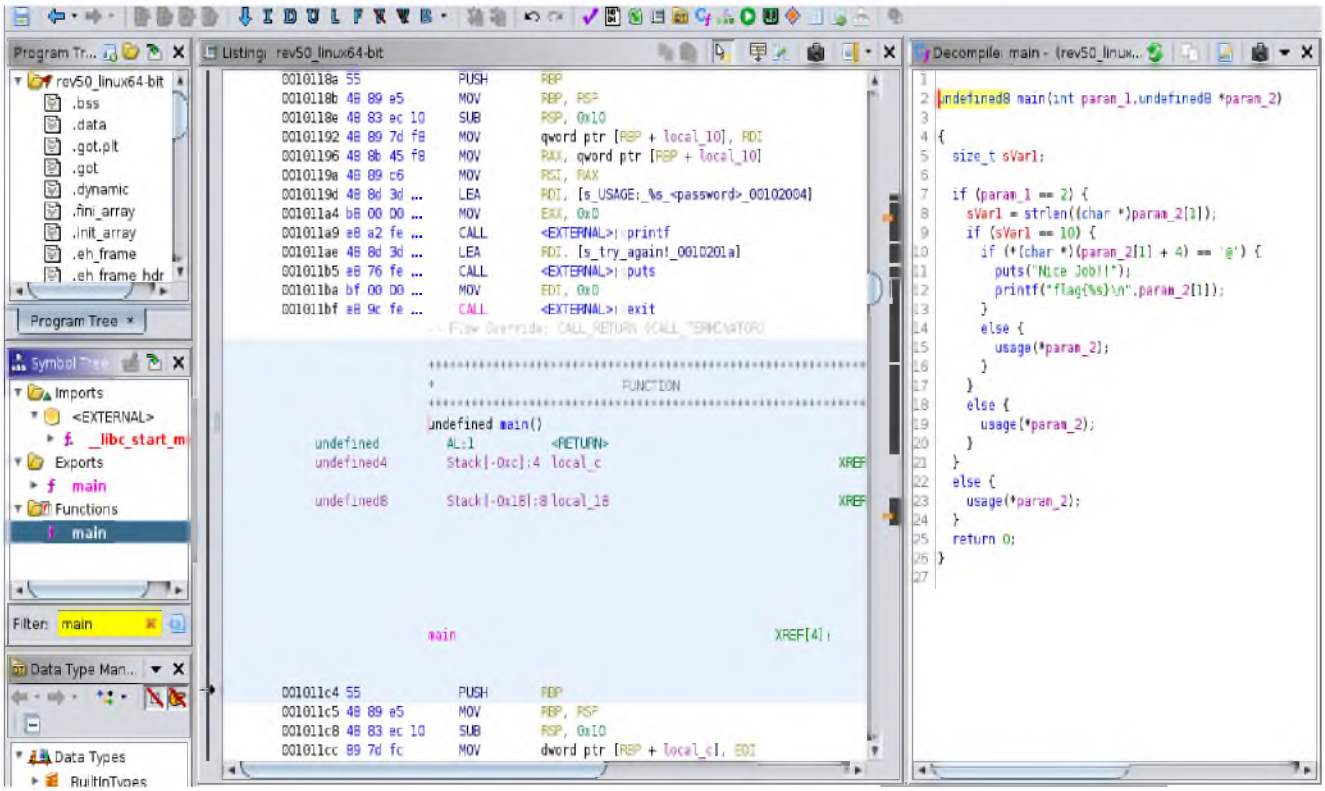


Рисунок 2.16 – Функція main файлу crackme у браузері коду Ghidra

Головна функція main представлена у невизначеному форматі: undefined8 main(int param_1,undefined8 *param_2). Але через те що відомо що файл написано на мові C, функція main має бути представлена у стандартизованому форматі: int main(int argc, char** argv), на що можна вказати Ghidra та замінити сигнатуру функції main, після чого декомпільований код стає набагато легше зрозуміти.

Як можна побачити на рисунку 2.17, після того як задано вірну сигнатуру функції main, можливо зробити аналіз коду та виявити що функція для роботи вимагає 1 аргумент довжиною 10 символів у якому п’ятим символом буде “@”. Інакше програма виконає функцію usage, яка просто повертає відповідь "try again!" та завершує програму кодом exit(0).

```
 2 int main(int argc,char **argv)
 3
 4 {
 5   size_t sVar1;
 6
 7   if (argc == 2) {
 8     sVar1 = strlen(argv[1]);
 9                     /* Перевірка на те що передано тільки 1
10                        аргумент */
11     if (sVar1 == 10) {
12                     /* Перевірка на те що довжина аргументу - 10
13                        символів */
14       if (argv[1][4] == '@') {
15                     /* Перевірка на те що п'ятий символ аргументу
16                        це @ */
17         puts("Nice Job!!");
18         printf("flag{%s}\n",argv[1]);
19       }
20       else {
21         usage(*argv);
22       }
23     }
24     else {
25       usage(*argv);
26     }
27   }
28   else {
29     usage(*argv);
30   }
31   return 0;
32 }
```

Рисунок 2.17 – Декомпільований код програми CrackMe з коментарями

Для перевірки вірності аналізу, можна запустити файл CrackMe та передати йому необхідний аргумент який було визначено тобто строку із 10 символів, у якої 5й символ це “@”. Як можна побачити на рисунку 2.18, аналіз було виконано вірно та при застосуванні аргументу “1234@546789” програма видає відповідь “Nice job” – гарна робота, а при застосуванні інших аргументів, у цьому прикладі це “test”, програма видає “try again” – спробуйте ще раз.

```
remnux@remnux:~/Downloads$ ./rev50_linux64-bit test
USAGE: ./rev50_linux64-bit <password>
try again!
remnux@remnux:~/Downloads$ ./rev50_linux64-bit 1234@67890
Nice Job!!
flag{1234@67890}
```

Рисунок 2.18 – Результат перевірки аналізу програми CrackMe

Таким чином можна займатися дизасемблюванням та декомпіляцією виконавчих файлів і подальшим їх статистичним аналізом, тобто вивченням їх

виконавчого коду. Але у реальних випадках код ЗПЗ може складатися з сотень файлів, частина з яких буде файлами приманками. А сам код програми буде дуже сильно обфусковано та дуже складно аналізувати, що ускладняється ще тим що сучасне ЗПЗ зазвичай розробляється на мові програмування Golang, у якому усі необхідні бібліотеки та залежності які необхідні програмі компілюються у єдиний файл, що робить його аналіз ще складнішим через величезну кількість бібліотек всередині файлу. [39]

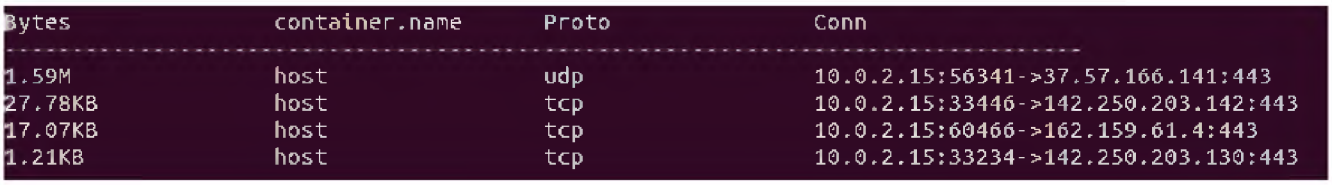
2.4.2 Динамічний аналіз зловмисного програмного забезпечення: вивчення поведінки вірусу в реальному часі.

Для динамічного аналізу системи буде використовуватися ПЗ sysdig. Sysdig дозволяє інструментувати фізичні та віртуальні машини на рівні ОС, працюючи з ядром Linux та фіксуючи системні виклики та інші події ОС. ПЗ також дає змогу створювати файли трасування для активності системи, подібно до того як це дозволяють робити мережеві утиліти tcpdump та Wireshark. Що дозволяє проаналізувати проблеми пізніше, не втрачаючи важливої інформації. Розширений стан системи зберігається у файлах трасування, щоб зафіксовано діяльність можна було ввести в повний контекст. [40]

Sysdig – це інструмент для системного аналізу та відлагодження у середовищі Linux. ПЗ надає можливості аналізу системних ресурсів, мережевої активності та інших аспектів системи, що може бути дуже зручним для аналізу поведінки ЗПЗ. Після інсталяції sysdig можна використовувати одразу у сирому форматі викликавши його в терміналі, але це призведе до того що у термінал будуть виводитися усі системні події і виклики, що трапляється з неймовірної швидкістю і неможливо для сприйняття людиною.

Для того щоб отримати можливий для розуміння людиною вивід у реальному часі, необхідно дуже конкретно задати фільтр виводу що дозволить отримувати конкретну інформацію необхідну для аналізу стану системи.

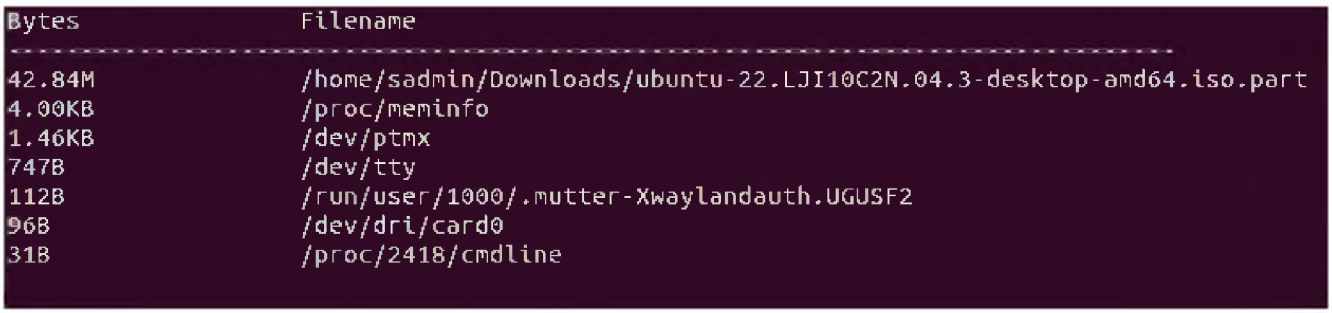
Такі фільтри можуть включати у себе моніторинг мережевого трафіку з сортуванням за максимальним об'ємом завдяки sysdig -c topprocs_net, як на рисунку 2.19.



Bytes	container.name	Proto	Conn
1.59M	host	udp	10.0.2.15:56341->37.57.166.141:443
27.78KB	host	tcp	10.0.2.15:33446->142.250.203.142:443
17.07KB	host	tcp	10.0.2.15:60466->162.159.61.4:443
1.21KB	host	tcp	10.0.2.15:33234->142.250.203.130:443

Рисунок 2.19 – Вивід мережевого трафіку за допомогою sysdig

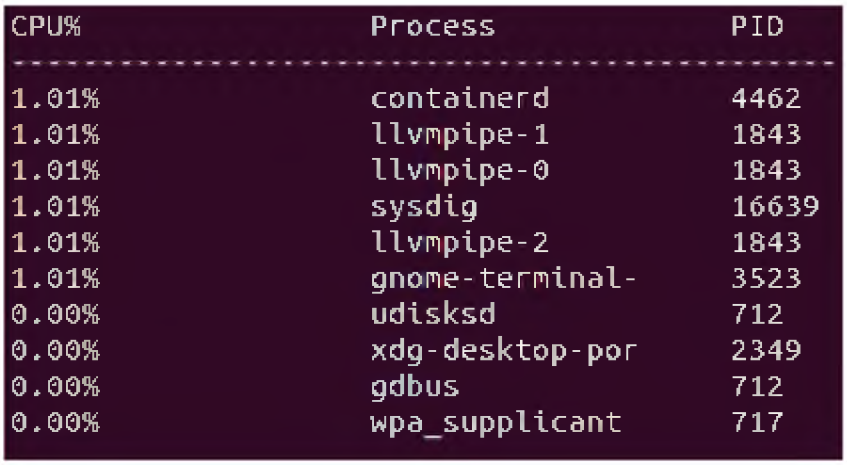
Або вивід процесів з самим великим навантаженням запису та зчитування завдяки sudo sysdig -pc -c topfiles_bytes, як на рисунку 2.20.



Bytes	Filename
42.84M	/home/sadmin/Downloads/ubuntu-22.LJI10C2N.04.3-desktop-amd64.iso.part
4.00KB	/proc/meminfo
1.46KB	/dev/ptmx
747B	/dev/tty
112B	/run/user/1000/.mutter-Xwaylandauth.UGUSF2
96B	/dev/dri/card0
31B	/proc/2418/cmdline

Рисунок 2.20 – Сортування процесів по об'єму запису та зчитування у sysdig

Чи відображення процесів з самим великим навантаженням на процесор як на рисунку 2.21.



CPU%	Process	PID
1.01%	containerd	4462
1.01%	llvmpipe-1	1843
1.01%	llvmpipe-0	1843
1.01%	sysdig	16639
1.01%	llvmpipe-2	1843
1.01%	gnome-terminal-	3523
0.00%	udisksd	712
0.00%	xdg-desktop-por	2349
0.00%	gdbus	712
0.00%	wpa_supplicant	717

Рисунок 2.21 – Сортуванням процесів по навантаженню на процесор у sysdig

Sysdig дозволяє фільтрувати свій вивід за процесом, аргументами що було передано до процесу при запуску, директорію, користувачем, вхідною та вихідною адресами, портом, мережевим протоколом, навантаженням на систему та будь якому іншому параметру.

Також ще одною дуже важливою функцією sysdig для аналізу ЗПЗ є можливість записувати усі ці події у файл формату scap, який можливо після цього аналізувати поза межами системи та ретроспективно. Наприклад, як на рисунку 2.22, із такого файлу за допомогою фільтру user.name=root -c spy_users було вилучено усі команди які запускалися від імені кореневого користувача root у системі. Що у цьому випадку дає можливість побачити що кореневим користувачем було створено директорії та змінено права доступу до файлів.

```
sadmin@UbuntuClient: $ sudo sysdig -r /home/sadmin/sysdig/dig2.scap user.name=root -c spy_users
5161 22:22:19 root) -bash
   5161 22:22:19 root) groups
   5161 22:22:19 root) /usr/bin/locale-check C.UTF-8
5161 22:22:19 root) sh -c cat /usr/etc/debuginfod/*.urls 2>/dev/null
   5161 22:22:19 root) cat /usr/etc/debuginfod/*.urls
5161 22:22:19 root) tr \n
   5161 22:22:19 root) /bin/sh /usr/bin/lesspipe
      5161 22:22:19 root) basename /usr/bin/lesspipe
5161 22:22:19 root) dirname /usr/bin/lesspipe
   5161 22:22:19 root) dircolors -b
   5161 22:22:19 root) mesg n
   5161 22:22:25 root) ls --color=auto -la
   5161 22:22:34 root) mkdir testo
   5161 22:22:49 root) sudo chmod ugo+rwx testo/
5161 22:22:49 root) chmod ugo+rwx testo/
   5161 22:22:57 root) mc
      5161 22:22:57 root) cons.saver /dev/pts/3
      5161 22:22:57 root) bash -rcfile .bashrc
         5161 22:22:57 root) /bin/sh /usr/bin/lesspipe
            5161 22:22:57 root) basename /usr/bin/lesspipe
5161 22:22:57 root) dirname /usr/bin/lesspipe
         5161 22:22:57 root) dircolors -b
            5161 22:23:02 root) cd /
```

Рисунок 2.22 – Відображення усіх команд від ім’я кореневого користувача за допомогою ПЗ sysdig

2.5 Вибір інструментів та платформ для практичної реалізації методів виявлення та аналізу зловмисного програмного забезпечення.

У другому розділі роботи було розглянуто найпопулярніші рішення для виявлення та аналізу вірусного програмного забезпечення, які є досі актуальним та активно підтримуються розробниками та спільнотою. У разі рішень для виявлення ЗПЗ на робочих станціях з метою захисту цих робочих станції, ці рішення можна використовувати у комерційному середовищі безкоштовно, але їх налаштування, моніторинг та підтримка буде вимагатиме багато часу зі сторони відповідального співробітника компанії або у разі великої кількісті робочих станції та особливо у випадку коли вони розподілені по декільком офісам – команди відповідальних

співробітників, скоріш за все системних адміністраторів. Це може працювати для невеличких компанії де знаходиться менш ніж 100-200 робочих станції, але для більш великих компанії не тільки більш зручно для компанії і системного адміністратора який буде керувати та налаштовувати систему, а також й більш фінансово вигідно буде придбати готове рішення-швейцарський ніж, яке буде включати у себе одразу усі необхідні інструменти – брандмауер, виявлення ЗПЗ завдяки як сигнатурам так і аномалійному виявленню, захист від спаму і фішингу та інші. Такі рішення, як буде видно із їх вартості, обійдуться дешевше аніж заробітна плата додатковим фахівцям для налаштування та підтримки саморобного набору окремих безкоштовних рішень, та скоріш за все буде набагато більш ефективним через те що над підтримкою цих рішень працює ціла окрема команда фахівців з кібербезпеки таких компанії як Microsoft або Kaspersky. І у додаток до цього ці рішення дозволяють керувати ними з хмарної консолі. Але тим не менш, варіант зі стеком безкоштовного ПЗ з відкритим вихідним кодом, теж буде розглянуто. Він все ще може бути актуальним для маленьких компаній. Якщо розглядати ПЗ для аналізу ЗПЗ, то у цьому випадку безкоштовне ПЗ, розглянуте у роботі, цілком можливо використовувати та активно використовується у комерційному середовищі.

2.5.1 Саморобна система на базі ПЗ з відкритим кодом

Для виявлення ЗПЗ та захисту робочих станцій:

- ClamAV – сигнатурне виявлення та виявлення за правилами YARA

- Suricata – мережева система IDS/IPS завдяки сигнатурам та також за правилами YARA, з можливості зробити один централізований сервер;

- Scirius – веб інтерфейс для керування правилами та наборами правил Suricata;

- Evebox – веб інтерфейс для перегляду подій та оповіщень Suricata

- Logstash – аналіз файлів журналювання;

- Kibana – для візуалізації інформації події Suricata, файлів журналювання або іншої;

- Elasticsearch – пошуковий рушій, який дозволяє більш ефективно працювати з Logstash та Kibana;

- UFW – брандмауер з використання вбудованої iptables;

- Webmin – безкоштовний веб інтерфейс для адміністрування робочих станції з можливістю централізації.

Для аналізу ЗПЗ:

6. ClamAV – для пошуку файлів у базі сигнатур та перевірки за YARA правилами;

7. Suricata – у цьому випадку для аналізу мережевих події викликаних ЗПЗ;

8. Scirius – веб інтерфейс для керування правилами та наборами правил Suricata;

9. Evebox – веб інтерфейс для перегляду подій та оповіщень Suricata

10. Ghidra або Radare2 – для дизасемблювання, декомпіляції та аналізу виконавчих файлів;

11. Sysdig – для глибокого аналізу усіх події у системі та системних викликів на рівні ядра.

Таким чином можна створити безкоштовну систему виявлення, протидії та глибокого аналізу ЗПЗ на робочих станціях та серверних машинах, використовуючи виключно рішення з відкритим вихідним кодом. Але яке, у випадку з виявленням та протидією ЗПЗ, буде вимагати набагато більш праці та часу у порівнянні з готовими рішеннями.

2.5.2 Готові рішення

Для виявлення ЗПЗ та захисту робочих станцій:

Microsoft Defender For Endpoints – рішення для захисту від Microsoft яке входить у підпис Microsoft 365 E5 разом з пакетом усіх інших програмних рішень Microsoft. Підтримує робочі станції під керуванням операційних систем Windows, MacOS та Linux. Включає у себе брандмауер, як сигнатурне так і аномалійне виявлення ЗПЗ, раннє виявлення та захист від атак програм-вимагачів, автоматичне виявлення пристроїв у мережі, автоматичний пошук вразливостей, набір утиліт для

емуляції атаки, глибокий аналіз інцидентів, централізоване управління через хмарну консоль або консоль локального серверу та інше. [41]

Ціна - ₴2100 за користувача за місяць або ₴7,560,00 за рік для компанії з 300 користувачами. Якщо відняти ціну базового пакету Microsoft 365 E3, який відрізняється саме відсутністю покращеного захисту, то ціна самого пакету захисту буде складати ₴800 за користувача за місяць або ₴2,880,000 за рік для компанії з 300 користувачами.

Це рішення скоріш за все буде найбільш привабливим для більшості компанії через те що окрім утиліт безпеки воно також включає у себе увесь набір ПЗ Microsoft – Office365, Teams, Access, Onedrive та інші, якими і так користується більшість, якщо не усі, комерційні компанії. А також дозволяє одночасно працювати з робочими станціями та серверами під керуванням ОС Windows, MacOS та Linux. У додаток до цього база ЗПЗ Microsoft вважається найкращою за кількістю відомих зразків ЗПЗ, що обумовлено тим що 70% усіх персональних комп'ютерів працюють під керуванням ОС Windows де за стандартом присутній Windows Defender який і збирає інформацію про ЗПЗ.

Kaspersky Endpoint Security – універсальне все в одному рішення від Kaspersky. Включає у себе виявлення ЗПЗ як сигнатурне так і поведінкове з автоматичним застосуванням заходів по нейтралізації цього ЗПЗ, аналіз мережевого трафіку, брандмауер, шифрування пристроїв, засоби керування робочою станцією – оновлення, застосування масових змін у налаштуваннях системи, керування ліцензіями ПЗ на системах, резервне копіювання, перегляд та керування пристроями та подіями з хмарної консолі або з застосуванням локального серверу. Працює виключно для систем під керуванням ОС Linux – для інших ОС необхідно буде придбати додаткове рішення. [42]

Ціна - ₴1450 за робочу станцію за рік або ₴450,000 за рік для компанії з 300 робочими станціями.

Як можна побачити, друге рішення є значно дешевшим, навіть з урахуванням вартості базового пакету програм від Microsoft, але рішення від Kaspersky не надає глибокого аналізу інцидентів та для застосування рішення для офісу з набором

різних ОС, необхідно буде придбати окремий додатковий продукт для Windows, а підтримки MacOS взагалі немає.

Для аналізу ЗПЗ:

У випадку з аналізом ЗПЗ усі безкоштовні рішення також можна ефективно використовувати у комерційному середовищі, але у випадку з Suricata та його доповненнями – Evebox, Scirius та стаком ELK – elasticsearch, logstash та kibana, набагато більш зручним та ефективним за часом вибором буде придбання готового та простого в налаштуванні рішення як IDSTower або SELKS.

IDSTower це веб інтерфейс для Suricata з набагато зручнішими інсталяцією та налаштуванням аніж ручні. Надає усі можливості що й набір утиліт Suricata який було розглянуто у роботі – керування правилами і подіями та додатково інтеграція з 14 безкоштовними та комерційними каналами Threat Intelligence, які охоплюють як правила IDS, так і індикатори компрометації (IOC), та комерційні набори правил виявлення ЗПЗ для Suricata. Додаткові можливості IDSTower будуть дуже зручними для виявлення та аналізу ЗПЗ і пошуку його у відомих сигнатурах та правилах. А можливість автоматичного швидкого та зручного розгортання та налаштування дозволяє ефективно використовувати його як IDS/IPS рішення у комерційному середовищі.[43]

Ціна ₴18,500 за сервер за рік з необмеженою кількістю пристроїв та кластерів.

Stamus Security Platform це комерційний варіант Selks – рішення яке є буквально абревіатурою розглянутих у роботі рішень – Suricata, Scirius, EveBox з додаванням вже налаштованого набору ELK та платформи Stamus. Це рішення також дозволяє автоматичне швидке розгортання, виявлення загроз по сигнатурам та правилам, інтеграцію з каналами Threat Intelligence, комерційний набір правил для Suricata та додатково автоматичний аналіз загроз з додаванням машинного навчання і власний рушій Stamus для виявлення та аналізу загроз.

Ціна ₴2800 на місяць для трафіку 1Гбіт/с або ₴33,000 на місяць.

Для виявлення сигнатур виконавчих файлів вибором все ще буде ClamAV.

Для дизасемблювання та декомпіляції і аналізу виконавчого коду також все ще можна використовувати відкрите безкоштовне рішення Ghidra, але у випадку з ЗПЗ для Linux кращим рішенням скоріш за все буду IDA Pro через вбудовану підтримку мови Golang на якій скоріш за все буде написано сучасне шкідливе ПЗ, так як Ghidra більш орієнтована на аналіз ЗПЗ написаного на мовах сімейства C, що є більш актуальним для аналізу ЗПЗ для Windows.

Ціна - ₴13500 за рік.

Для глибокого аналізу подій у системі найкращим відбором все ще лишається відкритий та безкоштовний Sysdig.

Та серед готових комерційних рішень для аналізу поведінки ЗПЗ найзручнішим та легшим у використанні буде хмарні платформи типу Joe's Sanbox, які дозволяють загрузити виконавчий файл у хмару постачальника сервісу, який запустить ЗПЗ у себе на платформі та автоматично надасть повний глибокий аналіз файлу, дії шо він виконує, процесів які він створює та змін у системі або налаштуваннях які він виконує. Підтримує системи під керуванням ОС Windows, MacOS та Linux, виконує аналіз та сканування як виконавчих файлів, так і інших типів документів, наприклад Excel документів з вбудованим запуском шкідливого коду, дозволяє взаємодіяти з інфікованою машиною у реальному часі, можливість перенаправити увесь трафік через обрану країну, аналіз завдяки аналогам розглянутих IDS/IPS рішень у роботі, надає детальні графіки і таблиці для усієї інформації та дії файлу, інтеграцію з каналами Threat Intelligence та ПЗ IDA Pro та інше. [44]

Ціна - ₴270,000 на рік з можливістю перевірки 100 файлів на місяць розміру до 750мб.

Підсумкова сума для набору готових рішень для середньої компанії з 300 робочими станціями на рік.

Для виявлення та захисту робочих станції:

- ₴0 – у разі використання рішень розглянутих у роботі з ручними інсталяцією та налаштуванням, але що передбачає використання додаткових фахівців для розгортання та підтримки;

- ₴450,000 у разі використання рішення від Kasperky для підприємств з виключно машинами під керуванням Linux;

- ₴7,560,000 у разі використання рішення від Microsoft для підприємств з різними ОС на робочих станціях та бажанням використовувати ліцензійний набор продуктів Microsoft.

Для аналізу ЗПЗ:

- ₴0 – у разі використання рішень розглянутих у роботі з ручними інсталяцією та налаштуванням, але що передбачає використання додаткових фахівців для розгортання та підтримки;

- ₴32,000 у разі використання безкоштовних рішень з додаванням IDSTower та IDA PRO;

- ₴270,000 у разі використання Joe's Sandbox без створення локальної машини для аналізу ЗПЗ;

- ₴302,000 у разі використання як локальної машини для аналізу ЗПЗ, так і хмарного рішення Joe's Sandbox.

2.6 Висновок до практичного розділу.

Роботою було практично проаналізовано та продемонстровано як працюють різні види виявлення та аналізу шкідливого, підозрілого або навіть звичайного програмного забезпечення для ОС Linux завдяки доступним програмним рішенням з відкритим кодом, які тим не менше активно використовуються у комерційному середовищі, та у випадку з таким ПЗ як ClamAV та Ghidra не тільки використовуються, а й розробляються та підтримуються провідними фахівцями з кібербезпеки такими як Cisco Talos Intelligence Group у випадку ClamAV та Агенством національної безпеки США у випадку Ghidra. Що дозволяє, використовуючи ПЗ та методи розглянуті у роботі, виявляти та аналізувати ЗПЗ не тільки великим комерційним компаніям а й невеликим підприємствам або студентам та ентузіастам. В подальшому було розглянуто варіанти та ціни комерційних платних рішень як для виявлення ЗПЗ і захисту від нього робочих станцій або серверних машин, так і для глибокого аналізу поведінки шкідливого

або потенційно шкідливого програмного забезпечення. Тема роботи є актуальною через повсюдне використання серверів під керуванням ОС Linux та зростаючу кількість робочих та персональних машин які також використовують системи на базі Linux, що обумовлено як безкоштовністю та зростаючою доступністю ОС Linux для рядових користувачів у зрівнянні з минулими роками, так і активною розробкою програмних слоїв типу Proton та Crossover які дозволяють запускати на ОС Linux програмне забезпечення розроблено виключно для ОС Windows. А актуальність загроз зі сторони ЗПЗ можна побачити завдяки розглянутим прикладам зразків ЗПЗ, атаки якими досі активно проводяться і жертвами яких є як великі компанії так і звичайні користувачі. Використовуючи методи та програмне забезпечення розглянуті у роботі як комерційні так і звичайні не комерційні користувачі можуть підвисити безпеку своєї робочої станції, а при наявності обережності та уважності зі сторони користувача, майже її гарантувати.

3 ЕКОНОМІЧНИЙ РОЗДІЛ

3.1 Постанова задачі

У економічному розділу необхідно буде визначити техніко-економічне обґрунтування (ТЕО) - економічну ефективність впровадження описаних у другому розділі заходів інформаційної безпеки.

Мета написання техніко-економічного обґрунтування це зробити фінансову оцінку передбачуваних економічних витрат на впровадження методів інформаційної безпеки, корисний результат завдяки їх впровадженню та їх співвідношення.

ТЕО має містити:

- стислий опис і значення проблеми, яка розглядається у кваліфікаційній роботі;

- обґрунтування необхідності та актуальності вирішення проблеми;

- аналіз очікуваних результати що впровадження програмних І технічних засобів забезпечення інформаційної безпеки;

- сутність запропонованого у кваліфікаційній роботі методу вирішення даної проблеми.

Для проведення техніко-технічного обґрунтування необхідно:

1) Розрахувати капітальні витрати на придбання та налагодження ПЗ та апаратних складових системи інформаційної безпеки які необхідні для впровадження методів описаних у другій частині;

2) Розрахувати річні експлуатаційні витрати на утримання і обслуговування ПЗ та апаратних складових;

3) Визначити річний економічний ефект від впровадження обраних заходів;

4) Визначити та проаналізувати показники економічної ефективності впроваджених заходів;

5) Зробити висновок про економічну доцільність впроваджених заходів.

3.2 Виконання розрахунків

Виконання розрахунків буде виконано для впровадження системи виявлення та протидії зловмисному програмному забезпеченню на прикладі коллцентра

банківської установи де працює близько 600 операторів та знаходиться 300 робочих станцій та 1 серверна машина. Оператори цього коллцентра виконують не тільки дзвінки, як у звичайному коллцентрі, я й займаються перевіркою документів клієнтів та їх транзакцій. Оператор працює з персональними даними клієнтів банку, роблячи у процесі знімки екрану (для уточнення питання або для завантаження у банківські комплекси для звітності). Тобто предметом інтересу зловмисника будуть персональні дані клієнтів цього банку і розрахунок можливих втрат буде виходити саме через потенційний виток цих даних через атаку зловмисного програмного забезпечення.

3.2.1 Розрахунок капітальних витрат

Капітальні інвестиції – це кошти, призначені для створення і придбання основних фондів і нематеріальних активів, що підлягають амортизації.

До капітальних інвестицій відносяться витрати на впровадження обраних заходів збереження безпеки інформації при роботі оператора на робочій станції. Це визначається виходячи з трудомісткості впровадження обраних заходів.

Трудомісткість реалізації впровадження заходів визначається тривалістю кожної робочої операції, починаючи з складання технічного завдання і закінчуючи оформленням документації (за умови роботи одного з спеціалістів інформаційної безпеки).

$$t = tmз + tв + ta + tозб + tр + tд \quad (3.1)$$

де ttз - тривалість складання технічного завдання на впровадження систем виявлення та протидії зловмисному програмному забезпеченню;

tв - тривалість аналізу ТЗ, вивчення інформації щодо можливих вразливостей безпеки інформації та їх пошуку;

ta – тривалість аналізу присутніх у поточній конфігурації вразливостей та їх реалізації;

tвз - тривалість визначення вимог до заходів, методів та засобів захисту;

tозб - тривалість вибору основних рішень з забезпечення безпеки інформації;

tвпр - тривалість впровадження обраних заходів;

tд - тривалість документального оформлення політики безпеки.

Для обраних заходів наведені величини становлять: ttз = 24 години, tв = 72 години, ta = 72 годин, tвз = 40 години, tозб = 40 години, tвпр = 80 годин, tд = 40 години.

Відповідно,

$$t = 24+72+72+40+40+80+40 = 368 \text{ годин}$$

Розрахунок витрат на розробку та впровадження заходів.

Витрати на розробку політики безпеки інформації Крп складаються з витрат на заробітну плату спеціаліста з інформаційної безпеки Ззп і вартості витрат машинного часу, що необхідний для розробки політики безпеки інформації Змч:

$$\text{Крп} = \text{Ззп} + \text{Змч} \quad (3.2)$$

Заробітна плата виконавця/виконавців враховує основну і додаткову заробітну плату, а також відрахування на соціальні потреби (пенсійне страхування, страхування на випадок безробіття, тощо) і визначається за формулою:

$$\text{Ззп} = t*\text{Зпр} = 368*218.75 = 80500 \text{ грн} \quad (3.3)$$

де t – загальна тривалість розробки політики безпеки, годин;

Зпр – середньогодинна заробітна плата інженера комп'ютерних систем з нарахуваннями, грн/година.

Вартість машинного часу для розробки політики безпеки інформації на ПК визначається за формулою:

$$\text{Змч} = t \cdot \text{Смч} \text{ , грн,} \quad (3.4)$$

де t - трудомісткість розробки політики безпеки інформації на ПК, годин;

Смч - вартість 1 години машинного часу ПК, грн./година.

Вартість 1 години машинного часу ПК визначається за формулою:

$$\text{Смч} = \text{Р} * t\text{нал} * Ce + \frac{\text{Фзал} + \text{На}}{F} + \frac{\text{Клпз} + \text{Напз}}{F}, \text{грн,} \quad (3.5)$$

де Р - встановлена потужність ПК, кВт;

Се - тариф на електричну енергію, грн/кВт·година;

ФЗал - залишкова вартість ПК на поточний рік, грн.;

На - річна норма амортизації на ПК, частки одиниці;

Нат - річна норма амортизації на ліцензійне програмне забезпечення, частки одиниц1;

Клт - вартість ліцензійного програмного забезпечення, грн.;

Fp - річний фонд робочого часу (за 40-годинного робочого тижня Fp =1920).

У цьому випадку:

P = 0.055 кВт;

Се = 2.64 грн/кВт*година

ФЗал = 12300 грн

На = 0.125

Клт = 0 (ПЗ не потребує покупки для впровадження, але вимагає щорічної підписки)

Fp = 1920

$$\text{Смч} = 0.055 * 368 * 2{,}64 + \frac{12300 * 0.125}{1920} = 535.14 \text{ грн}$$

$$\text{Крп} = 80{,}500 + 535.14 = 81035.14 \text{ грн}$$

Капітальні (фіксовані) витрати на проектування та впровадження заходів для усунення вразливостей інформаційної безпеки складають:

$$К = К_{пр} + К_{зпз} + К_{пз} + К_{аз} + К_{навч} + К_{н} \quad (3.6)$$

де Кпр - вартість розробки проекту інформаційної безпеки та залучення для цього зовнішніх консультантів, тис. грн;

Кзпз - вартість закупівлі ліцензійного основного й додаткового програмного забезпечення (ПЗ), тис. грн;

Крп - вартість розробки політики безпеки інформації, тис. грн;

Каз - вартість закупівлі апаратного забезпечення та допоміжних матеріалів, тис. грн;

Кнавч - витрати на навчання технічних фахівців і обслуговуючого персоналу, тис.грн;

Кн - витрати на встановлення обладнання та налагодження системи інформаційної безпеки, тис. грн.

Кпр = 100 тис. грн (тестування на проникнення)

Кзпз = 0 (усе необхідне ліцензійне ПЗ вже присутнє)

Каз = 0 (усе необхідне обладнання вже присутнє)

Кнавч = 120 тис. грн (витрати на навчання та підвищення кваліфікації інженера комп'ютерних систем стосовно роботи з централізованою системою виявлення і протидії ЗПЗ та перевірки мережі на вразлиовсті)

Кн = 0 (витрати на впровадження заходів вже входять у розраховане Крп)

$$К = 100 + 120 + 81 = 301 \text{ тис. грн}$$

3.2.2 Розрахунок поточних (експлуатаційних) витрат

Експлуатаційні витрати - це поточні витрати на експлуатацію та обслуговування об'єкта проектування за визначений період (наприклад, рік), що виражені у грошовій формі.

До поточних (експлуатаційних) варто відносити наступі витрати:

- вартість Upgrade-відновлення й модернізації системи (Св);

- витрати на керування системою в цілому (Ск);

- вартість підписки ПЗ (Спз)

- витрати, викликані активністю користувачів системи інформаційної безпеки (Сак - "активність користувача").

Під "витратами на керування системою" маються на увазі витрати, пов'язані з керуванням і адмініструванням серверів та інших компонентів системи інформаційної безпеки. До цієї статті витрат можна віднести наступні витрати:

- навчання адміністративного персоналу й кінцевих користувачів;

- амортизаційні відрахування від вартості обладнання та ПЗ;

- заробітна плата обслуговуючого персоналу;

- аутсорсинг (тобто залучення сторонніх організацій для виконання деяких видів обслуговування);

- навчальні курси й сертифікація обслуговуючого персоналу;

- технічне й організаційне адміністрування й сервіс.

Для цієї ситуації витрати за рік будуть складати:

Св = 12000 грн (витрати на змінні частини та обслуговування серверного обладнання, згідно з останнім роком)

Ск буде складатися з заробітної плати інженера, його навчання

Сзп = 218.35 грн/год * 1994 год = 435389.9 грн - заробітної плати інженера

Снав = 120000 грн – щорічний бюджет на навчання інженера

Ск = 555389.9 грн

Спз = 450000 грн

Сак часткова входить до Ск, так як прямою допомогою та додатковим налаштуванням також займається інженер комп'ютерних систем, тож Сак буде складатися з витрат на навчання операторів, тобто заробітної плати тренеру, що проводить навчання

Сак = 170.75 грн/год * 1994 год = 340475.5 грн

Отже, річні поточні (експлуатаційні) витрати на функціонування системи інформаційної безпеки складають:

С = Св + Ск + Спз + Сак (3.7)

С = 12000 + 555389.9 + 450000 + 340475.5 = 1357865.4 грн

3.3 Визначення річного економічного ефекту:

Загальний ефект від впровадження системи інформаційної безпеки визначається з урахуванням ризиків порушення інформаційної безпеки становить:

$$E = B \cdot R - C \quad (3.8)$$

де В - загальний збиток від атаки на вузол або сегмент корпоративної мережі, тис. грн;

R - очікувана імовірність атаки на вузол або сегмент корпоративної мережі, частки одиниці;

С - щорічні витрати на експлуатац1ю системи інформаційної безпеки, тис грн.

У даному випадку атака на робочу станцію не перешкоджає роботі оператора, не приводить до простою чи втраті інформації, та взагалі може довгий час пройти непоміченою. Збиток буде представляти з себе адміністративну відповідальність у виді штрафу від 500 до 1000 неоподатковуваних мінімумів доходів громадян [12].

Для розрахунків візьмемо 2 суми штрафу через те що втеча даних може відбуватись більш одного разу на рік.

$$В = 2 * 1342 \text{ грн} * 1000 = 2684000 \text{ грн}$$

$$Е = 2684000 * 0.8 – 1357865 = 789335 \text{ грн}$$

3.4 Визначення та аналіз показників економічної ефективності

Коефіцієнт повернення інвестицій ROSI показує, скільки гривень додаткового прибутку приносить одна гривня капітальних інвестицій на впровадження системи:

$$ROSI = \frac{E}{K}, \text{частки одиниці} \quad (3.9)$$

де Е – загальний ефект від реалізації захисних методів, грн;

К – капітальні інвестиції за варіантами, що забезпечили цей ефект, грн.

Коефіцієнт повернення інвестицій ROSI дорівнює:

$$ROSI = \frac{789{,}335}{1357865} = 0.58$$

Проект визнається економічно доцільним, якщо розрахункове значення коефіцієнта повернення інвестицій перевищує величину річної депозитної ставки з урахуванням інфляції:

$$ROSI > (Nрдс – Nінф) / 100 \quad (3.10)$$

Де Nрдс – річна депозитна ставка (16.5%);

Nінф – річний рівень інфляції (14,8%)

$0.58 > (16.5 – 14.8)/100 \sim 2.19 > 0.017$

Термін окупності капітальних інвестицій То показує, за скільки років капітальні інвестиції окупляться за рахунок загального ефекту від реалізації впровадження обраних заходів безпеки інформації:

$$То = \frac{К}{Е} = \frac{1}{ROSI}, \text{ років} \quad (3.11)$$

$$То = \frac{1}{0.58} = 1.72 \text{ року} = 368 \text{ днів}$$

3.5 Висновок економічного розділу

При проведені техніко-технічного обґрунтування було визначено, що розмір капітальних витрат на впровадження заходів виявлення та протидії атакам шкідливого програмного забезпечення на робочу станцію оператора коллцентра банківської установи складає 301 тис грн, річні поточні витрати на функціонування системи скла-дають 1 млн 357 тис грн.

Впровадження обраних заходів є економічно доцільним, оскільки коефіцієнт повернення інвестицій складає 0.58 грн до грн, що означає отримання 0.58 грн економічного ефекту на кожну гривню капітальних вкладень задля усунення описаних вразливостей, при цьому термін окупності буде складати 1.72 року ~ 628 днів.

ВИСНОВОК

У роботі було проаналізовано методи виявлення, протидії та аналізу шкідливого програмного забезпечення націленого на ОС Linux. Питання є актуальним через те що більш ніж 90% серверних машин у сіті використовує ОС Linux, а перехід робочих станції до цієї ОС є привабливим як для комерційного середовища, так і для звичайних користувачів персонального комп'ютера. А питання саме атак з використанням ЗПЗ є актуальним через тільки зростаючу з кожним роком кількість атак з його використанням, особливо для ОС Linux.

Було проаналізовано можливі вектори атак з використанням шкідливого програмного забезпечення на ОС Linux, та методи виявлення і аналізу ЗПЗ для систем під керуванням ОС Linux. Методи виявлення та аналізу було практично розглянуто з використанням програмного забезпечення яке активно використовується та у деяких випадках розробляється та підтримується провідними фахівцями з кібербезпеки такими як Cisco Talos та АНБ США.

Також були розглянуті готові комерційні рішення для виявлення та протидії шкідливому програмному забезпеченню, а у економічному розділі роботи була доведена доцільність впровадження та використання цих рішень на прикладі коллцентра банківської установи.

ПЕРЕЛІК ПОСИЛАНЬ

1. Adavelli M. 19 Surprising Linux Statistics Not Everyone Knew [Електронний ресурс] / Muninder Adavelli – Режим доступу до ресурсу: https://techjury.net/blog/linux-statistics/.

2. Petrosyan A. Annual number of ransomware attempts worldwide from 2017 to 2022 [Електронний ресурс] / Ani Petrosyan – Режим доступу до ресурсу: https://www.statista.com/statistics/494947/ransomware-attempts-per-year-worldwide.

3. Miller J. Linux Ransomware Poses Significant Threat to Critical Infrastructure [Електронний ресурс] / Jon Miller – Режим доступу до ресурсу: https://www.darkreading.com/vulnerabilities-threats/linux-ransomware-poses-significant-threat-to-critical-infrastructure.

4. Aver H. Attacks on virtualization systems and Linux servers [Електронний ресурс] / Hugh Aver – Режим доступу до ресурсу: https://www.kaspersky.com/blog/linux-vmware-esxi-ransomware-attacks/47988/.

5. James P. Tsunami Backdoor Can Be Used for Denial of Service Attacks [Електронний ресурс] / Peter James – Режим доступу до ресурсу: https://www.intego.com/mac-security-blog/tsunami-backdoor-can-be-used-for-denial-of-service-attacks/.

6. Pellitteri A. Threat news: Tsunami malware mutated. Now targeting Jenkins and Weblogic services [Електронний ресурс] / Alberto Pellitteri – Режим доступу до ресурсу: https://sysdig.com/blog/tsunami-malware-jenkins-weblogic/.

7. Tsunami DDoS Malware Distributed to Linux SSH Servers [Електронний ресурс] – Режим доступу до ресурсу: https://asec.ahnlab.com/en/54647/.

8. Antal G. FBI Closes Down Hive Ransomware Gang: What Does This Mean for the Security Landscape? [Електронний ресурс] / Gabriella Antal – Режим доступу до ресурсу: https://heimdalsecurity.com/blog/fbi-closes-down-hives-ransomware-gang/.

9. Biden says GOP 'threatening to destroy' economy; Elon Musk meets with lawmakers: Recap [Електронний ресурс] / [S. Elbeshbishi, D. Jackson, J. Garrison та ін.] – Режим доступу до ресурсу:

https://www.usatoday.com/story/news/politics/2023/01/26/live-politics-updates-trump-facebook-instagram-pelosi-video-rnc-chair/11123315002/.

10. Johnson K. FBI dismantles ransomware gang Hive's website; $130 million in ransom payments averted [Електронний ресурс] / Kevin Johnson – Режим доступу до ресурсу: https://www.usatoday.com/story/news/politics/2023/01/26/fbi-cyber-gang-hive-website/11126101002/.

11. What Is Cobalt Strike and How Does It Work? [Електронний ресурс] – Режим доступу до ресурсу: https://www.cynet.com/network-attacks/cobalt-strike-white-hat-hacker-powerhouse-in-the-wrong-hands/.

12. Кібератака на державні організації України з використанням шкідливої програми Cobalt Strike Beacon (CERT-UA#4227) [Електронний ресурс] – Режим доступу до ресурсу: https://cert.gov.ua/article/38155.

13. Hello Kitty Ransomware: In-Depth Analysis, Detection, and Mitigation – Sentinel One https://www.sentinelone.com/anthology/hello-kitty/

14. Hello Kitty Ransomware: In-Depth Analysis, Detection, and Mitigation [Електронний ресурс] – Режим доступу до ресурсу: https://www.sentinelone.com/anthology/hello-kitty/.

15. Abrams L. HelloKitty ransomware source code leaked on hacking forum [Електронний ресурс] / Lawrence Abrams – Режим доступу до ресурсу: https://www.bleepingcomputer.com/news/security/hellokitty-ransomware-source-code-leaked-on-hacking-forum/.

16. eCh0raix Ransomware Found Targeting QNAP Network-Attached Storage Devices [Електронний ресурс] – Режим доступу до ресурсу: https://www.trendmicro.com/vinfo/br/security/news/cybercrime-and-digital-threats/ech0raix-ransomware-found-targeting-qnap-network-attached-storage-devices.

17. Paganini P. Experts warn of a new ech0raix ransomware campaign targeting QNAP Network Attached Storage (NAS) devices [Електронний ресурс] / Pierluigi Paganini – Режим доступу до ресурсу: https://securityaffairs.com/132410/cyber-crime/ech0raix-ransomware-attacks.html.

18. Toulas B. - Winnti hackers split Cobalt Strike into 154 pieces to evade detection [Електронний ресурс] / Bill Toulas – Режим доступу до ресурсу: https://www.bleepingcomputer.com/news/security/winnti-hackers-split-cobalt-strike-into-154-pieces-to-evade-detection/.

19. Haruyama T. Tracking the entire iceberg - long-term APT malware C2 protocol emulation and scanning [Електронний ресурс] / Takahiro Haruyama – Режим доступу до ресурсу: https://www.virusbulletin.com/conference/vb2022/abstracts/tracking-entire-iceberg-long-term-apt-malware-c2-protocol-emulation-and-scanning/.

20. XORDDOS Malware Information [Електронний ресурс] – Режим доступу до ресурсу: https://success.trendmicro.com/dcx/s/solution/000278087.

21. Blocking Dedicated Attacking Hosts Is Not Enough: In-Depth Analysis of a Worldwide Linux XorDDoS Campaign [Електронний ресурс] / [Z. Chen, C. Lei, F. Liu та ін.] – Режим доступу до ресурсу: https://unit42.paloaltonetworks.com/new-linux-xorddos-trojan-campaign-delivers-malware/.

22. GAFGYT Overwiew [Електронний ресурс] – Режим доступу до ресурсу: https://www.trendmicro.com/vinfo/us/threat-encyclopedia/malware/GAFGYT/.

23. Toulas B. - Gafgyt malware exploits five-years-old flaw in EoL Zyxel router [Електронний ресурс] / Bill Toulas – Режим доступу до ресурсу: https://www.bleepingcomputer.com/news/security/gafgyt-malware-exploits-five-years-old-flaw-in-eol-zyxel-router/.

24. Joseph C. Paras Jha: How a Brilliant Mind Unleashed a Computer Monster [Електронний ресурс] / Charles Joseph – Режим доступу до ресурсу: https://threatpicture.com/people/paras-jha/.

25. What is the Mirai Botnet? [Електронний ресурс] // Cloudflare – Режим доступу до ресурсу: https://www.cloudflare.com/learning/ddos/glossary/mirai-botnet/.

26. Pandasecurity Sodinokibi Malware report [Електронний ресурс] / [J. Menes, P. Marqués, A. Sales та ін.] – Режим доступу до ресурсу: https://www.pandasecurity.com/en/mediacenter/sodinikibi-ransomware-report/.

27. Robinson P. What is REvil/Sodinokibi Ransomware [Електронний ресурс] / Philip Robinson – Режим доступу до ресурсу: https://www.lepide.com/blog/what-is-revil-sodinokibi-ransomware/.

28. Fakterman T. REvil / Sodinokibi: The Crown Prince of Ransomware [Електронний ресурс] / Tom Fakterman – Режим доступу до ресурсу: https://www.cybereason.com/blog/research/the-sodinokibi-ransomware-attack.

29. Alzahrani S. An Analysis of Conti Ransomware Leaked Source Codes [Електронний ресурс] / S. Alzahrani, Y. Xiao, W. Sun – Режим доступу до ресурсу: https://ieeexplore.ieee.org/document/9895237.

30. Elias M. Conti Group Targets ESXi Hypervisors With its Linux Variant [Електронний ресурс] / M. Elias, J. Tologonov, A. Mundo – Режим доступу до ресурсу: https://www.trellix.com/about/newsroom/stories/research/conti-group-targets-esxi-hypervisors-with-its-linux-variant/.

31. Weizman I. Conti Unpacked: Understanding ransomware development as a response to detection – a detailed technical analysis [Електронний ресурс] / I. Weizman, A. Pirozzi – Режим доступу до ресурсу: https://assets.sentinelone.com/sentinellabs/conti-ransomware-unpacked.

32. ClamAV documentation [Електронний ресурс] – Режим доступу до ресурсу: https://docs.clamav.net.

33. Ledesma J. - IDS vs. IPS: What Organizations Need to Know [Електронний ресурс] / Josue Ledesma – Режим доступу до ресурсу: https://www.varonis.com/blog/ids-vs-ips.

34. EveBox documentation [Електронний ресурс] – Режим доступу до ресурсу: https://evebox.org.

35. Sengupta S. Reverse Engineering Malware: Techniques And Tools For Analyzing And Dissecting Malicious Software [Електронний ресурс] / Sudip Sengupta – Режим доступу до ресурсу: https://sudip-says-hi.medium.com/reverse-engineering-malware-techniques-and-tools-for-analyzing-and-dissecting-malicious-software-4dd5949135f0.

36. Ghidra documentation [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/NationalSecurityAgency/ghidra.

37. Das S. - Reverse Engineering a Crackme (Level: Easy) [Електронний ресурс] / Samrat Das – Режим доступу до ресурсу: https://www.linkedin.com/pulse/reverse-engineering-crackme-level-easy-samraat-das/.

38. cbm-hackers's easy_reverse [Електронний ресурс] – Режим доступу до ресурсу: https://crackmes.one/crackme/5b8a37a433c5d45fc286ad83.

39. Dev T. GoLang the new Malware Language? [Електронний ресурс] / Tapendra Dev – Режим доступу до ресурсу: https://tapendradev.medium.com/golang-the-new-malware-language-94097baae223.

40. SysDig documentation [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/draios/sysdig.

41. Microsoft Defender for Endpoint [Електронний ресурс] – Режим доступу до ресурсу: https://www.microsoft.com/en-us/security/business/endpoint-security/microsoft-defender-endpoint.

42. Kaspersky Endpoint Security for Linux [Електронний ресурс] – Режим доступу до ресурсу: https://support.kaspersky.com/kes-for-linux/11.4.0.

43. IDSTower Documentation [Електронний ресурс] – Режим доступу до ресурсу: https://idstower.com.

44. Joe Sandbox Cloud overwiev [Електронний ресурс] – Режим доступу до ресурсу: https://www.joesecurity.org/joe-sandbox-cloud.

45. Кваліфікаційна робота магістра. Методичні рекомендації до виконання для студентів спеціальності 125 «Кібербезпека» (освітньо-професійна програма «Кібербезпека») / Упоряд.: О.Ю.Гусєв, В.І.Корнієнко, В.І.Магро, Д.С. Тимофєєв; М-во освіти і науки України, Нац. техн. ун-т «Дніпровська політехніка». – Д.: НТУ «ДП», 2022. – 34 с.

46. Кваліфікаційна робота магістра. Методичні рекомендації до виконання для студентів спеціальності 125 «Кібербезпека» (освітньо-професійна програма «Кібербезпека») / Упоряд.: О.Ю.Гусєв, В.І.Корнієнко, В.І.Магро, Д.С. Тимофєєв;

М-во освіти і науки України, Нац. техн. ун-т «Дніпровська політехніка». – Д.: НТУ «ДП», 2022. – 34 с.

ДОДАТОК А. Відомість матеріалів кваліфікаційної роботи

№	Формат	Найменування	Кількість листів	Примітки
Документація				
1	А4	Реферат	1	
2	А4	Список умовних скорочень	1	
3	А4	Зміст	3	
4	А4	Вступ	1	
5	А4	Стан питання. Постановка задачі.	21	
6	А4	Спеціальна частина.	42	
7	А4	Економічний розділ	7	
8	А4	Висновки	1	
9	А4	Перелік посилань	6	
10	А4	Додаток А	1	
11	А4	Додаток Б	1	
12	А4	Додаток В	1	
13	А4	Додаток Г	1	

ДОДАТОК Б. Перелік документів на оптичному носії

Залевский_МВ_125_22_1_ПЗ.docx

Залевский_МВ_125_22_1_ПЗ.pdf

Залевский_МВ_125_22_1_ДМ.pptx

Залевский_МВ_125_22_1_ПЗ.pdf.p7s

ДОДАТОК В. Відгук керівника кваліфікаційної роботи

В І Д Г У К

на кваліфікаційну роботу магістра студента групи 125-22-1

Залевського Максима Владиславовича

на тему: “ Методи виявлення та аналізу зловмисного програмного забезпечення в ОС Linux”

Пояснювальна записка складається зі вступу, трьох розділів і висновків, викладених на 92 сторінках.

Метою кваліфікаційної роботи є вивченя методів виявлення, аналізу та протидії шкідливому програмному забезпеченюю для машин під керуванням ОС Linux.

Тема кваліфікаційної роботи безпосередньо пов’язана з об’єктом діяльності магістра спеціальності 125 «Кібербезпека», а зміст та структура проекту дозволяють розкрити поставлену тему повністю.

Для досягнення поставленої мети у кваліфікаційної роботі вирішуються задачі виявлення шкідливого програмного забезпечення для ОС Linux, його аналізу та протидії йому на прикладках ПЗ що використовується провідними фахівцями з кібербезпеки.

Практична цінність полягає у розробці рекомендацій щодо виявлення, аналізу та протидії зловмисному програмному забезпеченню націленому на машини які використовують ОС Linux.

Оформлення пояснювальної записки до кваліфікаційної роботи виконано без відхилень від стандартів.

В ході виконання кваліфікаційної роботи студент Залевський М.В. проявив самостійність та показав добрий рівень володіння теоретичними положеннями з обраної теми. Автор уміє формулювати наукові та практичні завдання і знаходить адекватні засоби для їх вирішення та заслуговує присвоєння кваліфікації магістра за спеціальністю 125 Кібербезпека, освітньо-професійна програма «Кібербезпека».

Рівень запозичень у кваліфікаційній роботі не перевищує вимог “Положення про систему виявлення та запобігання плагіату”.

Кваліфікаційна робота заслуговує оцінки «85/Добре».

Керівник кваліфікаційної роботи
проф. каф. БІТ Є.В. Котух

Керівник спец. розділу
асистент кафедри БІТ Ю.А. Мілінчук

ДОДАТОК Г. Відгук керівника кваліфікаційної роботи

В І Д Г У К

Економічний розділ виконаний відповідно до вимог, які ставляться до кваліфікаційних робіт, та заслуговує на оцінку 92 б. («відмінно»).

Керівник розділу _____ доц. Пілова Д.П.

(підпис) (ініціали, прізвище)