Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації

Комплексна бакалаврська дипломна робота на тему:

«Кіберполігон для дослідження подій інформаційної безпеки. Частина 3.

Обробка на основі Elasticsearch»

Виконав студент 4 курсу групи 1БС-19б
спеціальності 125 «Кібербезпека»

Олександр ЯКІМОВ

Керівник: к. т. н., доцент кафедри ЗІ

Олеся ВОЙТОВИЧ

«16» червня 2023 р.

Рецензент: к. т. н., доцент кафедри ПЗ

Олена КОВАЛЕНКО

«16» червня 2023 р.

Допущено до захисту

Завідувач кафедри ЗІ

д. т. н, проф.

Володимир ЛУЖЕЦЬКИЙ

«16» червня 2023 р.

Вінниця ВНТУ – 2023 року

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації Рівень вищої освіти І (бакалаврський)
Галузь знань – 12 Інформаційні технології
Спеціальність – 125 Кібербезпека
Освітньо-професійна програма – Безпека інформаційних і комунікаційних систем

ЗАТВЕРДЖУЮ
Зав. кафедри ЗІ,
д. т. н., проф.
Володимир ЛУЖЕЦЬКИЙ
«___» _____ 2023 року

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

Якімову Олександру Павловичу

1. Тема роботи: «Кіберполігон для дослідження подій інформаційної безпеки. Частина 3. Обробка на основі Elasticsearch»,
керівник роботи: Войтович Олеся Петрівна, к. т. н., доцент кафедри ЗІ,
затверджені наказом ректора ВНТУ від 20 березня 2023 року №67

2. Строк подання студентом роботи 16 червня 2023 року.

3. Вихідні дані до роботи:
– аналіз відомих методів побудови кіберполігонів;
– створення кіберполігону на базі кафедри захисту інформації;
– налаштування систем менеджменту подій та інцидентів;
– експериментальне дослідження запропонованих рішень.

4. Зміст текстової частини: Вступ. 1. Аналіз предметної області. 2. Обґрунтування обраних рішень. 3. Тестування обраних рішень. Висновки. Список використаних джерел. Додатки.

5. Перелік ілюстративного матеріалу: Результати порівняння локальних та віддалених кіберполігонів (плакат А4). Постановка завдання (плакат А4). Структурна схема кіберполігону (плакат А4). Візуалізація атак (плакат А4). Результати роботи кіберполігону (плакат А4). Результати налаштування NetFlow (плакат А4). Структура логів (плакат А4).

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Войтович О. П., к.т.н., доц. каф.ЗІ	20.03.2023	16.06.2023
2	Войтович О. П., к.т.н., доц. каф.ЗІ	20.03.2023	16.06.2023
3	Войтович О. П., к.т.н., доц. каф.ЗІ	20.03.2023	16.06.2023

7. Дата видачі завдання 20 березня 2023 року.

КАЛЕНДАРНИЙ ПЛАН

ч	Назва етапів комплексної бакалаврської дипломної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз завдання. Вступ	20.03.23 – 26.03.23	
2	Аналіз джерел за напрямком комплексної бакалаврської дипломної роботи	27.03.23 – 09.04.23	
3	Розробка архітектури та алгоритму засобу	10.04.23 – 23.04.23	
4	Програмна реалізація та тестування	24.04.23 – 21.05.23	
5	Аналіз результатів тестування, висновки	22.05.23 – 24.05.23	
6	Оформлення пояснювальної записки	25.05.23 – 31.05.23	
7	Попередній захист та доопрацювання БДР	01.06.23 – 15.06.23	
8	Представлення БДР на захист	16.06.23 – 19.06.23	
9	Захист БДР	20.06.23 – 23.06.23	

Студент _____ Олександр ЯКІМО

Керівник роботи _____ Олеся ВОЙТОВИ

АНОТАЦІЯ

Бакалаврська дипломна робота складається з 55 сторінок формату А4, на яких є 26 рисунків, 3 таблиці, список використаних джерел містить 35 найменувань.

Бакалаврська робота присвячена впровадженню та налаштуванню Elasticsearch, третьої частини комплексної дипломної роботи по розробці кіберполігону. В результаті аналізу існуючих кіберполігонів було обрано розгортання локальної версії полігону. Наведено, які засоби можуть використовуватись для створення таких середовищ та обрано оптимальний набір інструментів для подальшої розробки. Після визначення потенційних атак, які можна впровадити у кіберполігоні та окреслення слідів, які вони можуть залишити, було налаштовано логування та за допомогою мови запитів Elasticsearch проаналізовано повідомлення на прикладі спроектованої атаки на систему. У результаті було отримано працездатну частину комплексної дипломної роботи, яка відповідає за збереження та обробку логів в системі.

Ключові слова: кіберполігон, Elasticsearch, логи, обробка логів, пошук, аналіз, інформація, пошуковий двигун.

ABSTRACT

The bachelor thesis consists of 55 pages of A4 format, on which there are 26 figures, 3 tables, the list of used sources contains 35 items.

The bachelor's thesis is devoted to the implementation and configuration of Elasticsearch, the third part of the complex thesis on the development of a cyber polygon. As a result of the analysis of the existing cyber polygons, the deployment of the local version of the polygon was chosen. The tools that can be used to create such environments are given, and the optimal set of tools for further development is selected. After identifying the potential attacks that can be implemented in the cyber polygon and delineating the traces that they can leave, logging was configured and messages were analyzed using the Elasticsearch query language for an example of a designed attack on the system. As a result, a workable part of the complex thesis was obtained, which is responsible for saving and processing logs in the system.

Keywords: cyber polygon, Elasticsearch, logs, log processing, search, analysis, information, search engine.

ЗМІСТ

ВСТУП

У сучасному світі, де інформаційні технології стають все більш поширеними, кібербезпека стає надзвичайно важливою для багатьох сфер життя. В той момент коли кіберзлочинці постійно намагаються використати вразливості систем, створюючи кібератаки, які без належного реагування можуть спричинити серйозні проблеми не тільки для звичних користувачів, а й для цілої держави, замахуючись навіть знищенням цілого державного блоку, потрібно віднайти спосіб зберегти системи від загроз. З плином часу такі атаки стають все більш складними та небезпечними, оскільки злочинці постійно вдосконалюють свої методи та технології, тому потрібно бути на крок попереду та запобігти великій катастрофі [1].

Кіберополігони – важливі інструменти у процесі забезпечення захисту, вони дозволяють фахівцям з кібербезпеки на практиці тренувати та тестувати свої кіберзаходи, розробляти стратегії в разі кібератак, створювати плани захисту та перевіряти їх ефективність використовуючи наявні технології.

Такі системи надають можливість виявляти потенційні уразливості в системі та вдосконалювати їх без ризику для реальних даних. Вивчення кіберополігонів дозволяє фахівцям з кібербезпеки бути краще підготовленими до можливих кібератак та вдосконалювати свої заходи захисту для зменшення ризику їхнього успіху.

Предметом дослідження у бакалаврській роботі є кіберполігон для дослідження подій кібербезпеки основі Elasticsearch.

Об’єктом дослідження є моніторинг та обробка логів в системах управління подіями інформаційної безпеки основі Elasticsearch.

Мета роботи полягає в покращенні тренування та обізнаності за рахунок впровадження кіберполігону, зокрема реалізації обробки логів, на основі зберігання та пошуку даних, спеціалізованим на аналізі та обробці великого обсягу структурованих та неструктурованих даних у реальному часі засобом Elasticsearch.

Задля досягнення мети необхідно виконати ряд задач, а саме:

1) здійснити аналіз наявних методів, які вже використовуються у кіберполігонах;
2) обрати оптимальне рішення на основі якого буде створено полігон;
3) здійснити тестування на предмет ефективності використання створеного методу.

Проміжні результати дослідження були представлені на LII Науково-технічній конференції факультету інформаційних технологій та комп'ютерної інженерії (2023) [1].

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Проблематика теми

Одними з головних проблем вирішення питань кібербезпеки та створення необхідних засобів для забезпечення захисту є стрімке зростання кібератак за складністю та небезпекою, які вони несуть за собою [1]. Наша країна стала мішенню численних кібератак з боку інших країн та кіберзлочинців, що, при неналежній увазі, може призвести до катастрофічних наслідків для урядових установ, компаній та громадян [2]. Однак, для запобіганню подібних атак потрібно отримати досвід та розуміння, як вони відбуваються. На допомогу в цьому приходять кіберполігони. Кіберполігон – це комплекс програмно-апаратних засобів для обробки інформації та забезпечення кібербезпеки, призначений для ознайомлення та покращення навичок на прикладі тестових стендів наявних систем [3].

Вивчення теми "кіберополігонів" для студентів, які вивчають кібербезпеку є надзвичайно важливим завданням з точки зору підвищення рівня їх обізнаності. Студенти повинні вміти використовувати кіберополігони для аналізу і тестування інформаційних систем з метою забезпечення їх надійності та захисту від кібератак. Враховуючи стрімке збільшення кібератак на інфраструктуру та підприємства, розвиток національної кібербезпеки в Україні потребує дослідження та вдосконалення, а впровадження кіберополігонів допоможе отримати необхідні навички для створення належного захисту від потенційних загроз [4].

Отже, висновок, що можна зробити з вищевказаного, полягає у необхідності активного розвитку використання таких кіберполігонів, оскільки вони виявляються незамінними інструментами для навчання студентів як теоретичних знань, так і практичних навичок впровадження та використання кіберполігонів, що може бути в нагоді студентам у майбутній професійній діяльності, а знання про кіберополігони та вміння їх використовувати можуть стати важливими перевагами при пошуку роботи у цій галузі.

1.2 Аналіз наявних рішень

1.2.1 Локальні кіберполігони

Локальний кіберполігон є спеціально створеним тестовим середовищем в межах власної організації, яке моделює реальну інфраструктуру та мережу. Він призначений для проведення тестів на кібербезпеку, аналізу вразливостей, виявлення атак та тренування заходів забезпечення безпеки [5].

Один із відомих навчально-тренувальних комплексів з кібербезпеки (кіберполігонів) в Україні має Військовий інститут телекомунікацій та інформатизації (він і використовується ЗСУ). Створений цей полігон у 2021 році та отримав назву «VITIsecurity». Цей комплекс призначений для виявлення, реагування, протидії та попередження кіберзагроз, аналізу та розслідування кіберінцидентів та підвищення якості освітнього процесу [6].

Комплекс складається з програмно-технічних засобів, ситуаційного центру, комплексу технічних засобів, який забезпечує стале функціонування програмно-апаратного ядра, надання можливості створення, модифікації та реалізації сценаріїв та контролю проведення навчальних занять, тренувань з кібербезпеки. Кіберполігон включає в себе 80 автоматизованих робочих місць для курсантів та слухачів інституту, що навчаються за спеціальністю «Кібербезпека» [6].

На базі серверного кластеру розгорнуто віртуальну інфраструктуру. Вона забезпечує надання базових сервісів, використання системи дистанційного навчання, проведення наукових досліджень, проведення змагань по типу «Захват прапору» та навчань із забезпечення захисту інфраструктури в кіберпросторі [6, 7].

Оглянувши наявні фото цієї системи було зроблено висновок, що за основу була взята SIEM система, яка дозволяє отримувати інформацію в режимі «реального часу» і демонструвати її на екрані у вигляді графіків та статистик. Моделювання атак відбувається як вручну, так і за допомогою скриптів, що

допомагає відпрацювати, перевірити та засвоїти матеріал по багатьом способам атак.

У Національному університеті біоресурсів і природокористування України було створено ще один полігон. "Навчальний кіберполігон" знаходиться в лабораторії кафедри комп'ютерних систем і мереж. Його головна мета – надання можливості студентам і викладачам отримати практичні навички у сфері кібербезпеки та розширити свої знання про захист комп'ютерних систем. Така лабораторія обладнана сучасними засобами виробництва FortiNet, Cisco, Mikrotik та D-Link, використання яких надає студентам реальний практичний досвід, адже вони мають доступ до високоякісного обладнання, яке використовується в справжніх комп'ютерних мережах та системах забезпечення безпеки [8].

У цій лабораторії студенти можуть відпрацьовувати побудову систем виявлення та попередження вторгнень у локальні мережі. Вони мають можливість створювати захищені мережі для передачі інформації та навчатися створювати захищені канали зв'язку між різними сегментами мережі. Такі навички є надзвичайно важливими у сучасному цифровому світі, де кіберзагрози стають все більшими та складнішими. Для навчання в лабораторії "Навчальний кіберполігон" використовується специфічне програмне забезпечення та відповідні операційні системи, що дозволяють студентам тестувати мережі та системи на проникнення, відслідковувати дії та вплив різноманітних шкідливих вірусів, та здобувати знання [8].

Проаналізувавши вигляд та інструментарій кіберполігону, можна зробити висновок, що система в цьому університеті базується на SIEM технологіях, що включає моніторинг трафіку з відображенням на дашбордах. Така система виглядає простіше у порівнянні з системою VITIsecurity та спрямована на швидке засвоєння необхідних навичок студентами та викладачами.

1.2.2 Віддалені кіберполігони

Віддалений кіберполігон – це віртуальне або фізичне середовище, яке знаходиться на віддаленому сервері або хмарній інфраструктурі. Такий полігон призначений для проведення тренувань, симуляцій та тестування кібербезпеки безпосередньо з комп'ютера або мережі користувача, де можна реалістично моделювати сценарії атак та вразливостей [5].

Приклад такого полігону, Hack The Box – онлайн-платформа, яка має чималу кількість віртуальних машин з різними рівнями складності і була створена для тестування знань на практиці [9]. Машини, які пропонує використовувати цей сервіс, містять уразливості, що можуть бути використані для отримання доступу до системи. З кожним разом, проходячи запропоновані сервісом машини, користувач отримує практичний досвід використання команд, програм, систем і знаходження вразливостей включаючи сканери портів, експлоіти, фішинг техніки та інші.

Користувачі можуть використовувати ці інструменти для аналізу та виявлення вразливостей у системах, а також для розробки власних методів захисту від кібератак, що допоможе відшліфувати знання та натренувати навички в як в області наступальної (Offensive Security), так і оборонної безпеки (Defensive Security).

Головна перевага використання Hack The Box – можливість вивчення кіберполігонів в режимі реального часу, використовуючи для цього комфортні умови навчання з будь-якої точки світу. Платформа надає доступ до віртуальних машин, що можуть бути використані для тестування різних інформаційних систем, включаючи мережі, веб-сайти та інші. Також Hack The Box надає можливість взаємодії з іншими користувачами платформи за допомогою форуму, де можна обговорювати проблеми безпеки, обмінюватись досвідом та порадами з іншими професіоналами у галузі кібербезпеки [9].

Його аналогом є сервіс під назвою TryHackMe [10]. Цей сервіс надає ті ж послуги, що й його попередник, але має деякі суттєві відмінності, одна з яких – підхід до навчання. TryHackMe надає користувачам більш організоване та

структуроване середовище навчання, на платформі доступні різні курси та вразливі машини з різними рівнями складності, які можуть бути виконані поетапно. TryHackMe також надає можливість користувачам спілкуватися з іншими учасниками та тренерами на форумі та чаті, що допомагає оперативно працювати та знаходити необхідні підказки, розширювати коло колег та просто провести весело час [10].

Порівнюючи ці сервіси, можна виявити ряд переваг та недоліків, наприклад, Hack The Box, в порівнянні з його аналогом, надає більш вільне середовище для тестування та експериментів, хоча на основній платформі все заточено саме на отримання практичних навичок. Всі матеріали для вивчення виділені в академію, тому спочатку користувачі мають можливість підготуватися, а далі потрапляють в «бойові реалії», де можуть самостійно виконувати завдання та досліджувати уразливості віртуальних машин.

Варто зазначити, що Hack The Box має більш складні завдання та вимагає від користувачів глибшого розуміння сфери кібербезпеки. Іншою відмінністю між цими платформами є їхні підходи до доступності та вартості. TryHackMe надає користувачам безкоштовний доступ до обмеженої кількості вразливих машин та курсів, але платна підписка надає доступ до більшої кількості ресурсів. З іншого боку, Hack The Box є платною платформою, але вона надає великий вибір віртуальних машин та завдань, включаючи безкоштовні опції [9,10].

Аналогів цих сервісів за останні роки збільшилось в кілька разів, але зазвичай звертаються саме до цих сервісів, які вже призвичаїлися і є провідними в цій ланці.

Проведемо порівняння локальних та віддалених кіберполігонів (табл. 1.1).

Таблиця 1.1 – Результати порівняння локальних та віддалених кіберполігонів

Аспекти порівняння	**Локальний кіберполігон**	**Віддалений кіберполігон**
Доступність	Обмежена (локальна)	Зручний доступ онлайн
Характеристики обладнання	Фізичне обладнання, розміщене на місці	Віртуальна інфраструктура, розміщена в хмарі
Практичні навички	Запрограмовані конкретні сценарії та налаштування; можливість створення сценаріїв (потребує глибших знань)	Гнучкість в створенні різних сценаріїв і налаштувань
Вартість	Середня	Знижена вартість у порівнянні з фізичним обладнанням
Масштабованість	Обмежена	Легко масштабується для великої кількості користувачів
Фізична безпека	Висока	Залежить від рівня безпеки віддаленого середовища
Керування	Локальний контроль над інфраструктурою	Залежить від провайдера хмарної інфраструктури
Місцезнаходження даних	Локальне зберігання даних	Віддалене зберігання даних в хмарі
Пропускна здатність мережі	Залежить від локального мережевого обладнання та інфраструктури	Залежить від якості Інтернет-з'єднання
Фізичний простір	Вимагає виділеного простору для лабораторії	Не потребує фізичного простору, можливість використання вже існуючих приміщень
Супроводження	Потребує власного технічного персоналу для обслуговування обладнання та інфраструктури	Технічне супроводження та підтримка надаються провайдером хмарної інфраструктури
Час виконання	Можлива затримка в налагодженні та у вирішенні проблем	Зручний доступ до віддаленої інфраструктури, що може пришвидшити час виконання
Системна безпека	Залежить від рівня фізичної та логічної захищеності локальної мережі	Залежить від надійності та безпеки хмарної інфраструктури та доступу до неї
Доступ до ресурсів	Обмежений доступ до локальних ресурсів	Зручний доступ до глобальних ресурсів та інших хмарних послуг

Робимо висновок, що кожен з видів має як свої переваги, так і свої недоліки та може бути використаний базуючись на відповідних потребах. Локальний кіберполігон характеризується обмеженою доступністю, оскільки вимагає фізичного обладнання, розміщеного на місці, і виділеного простору для розгортання. З його допомогою можна отримати запрограмовані сценарії та налаштування, але масштабованість обмежена і вартість залишаються на високому рівні. З поміж ряду особливостей локального полігону варто виділити те, що він залежить від локальної мережевої інфраструктури та потребує власного технічного персоналу для супроводження та обслуговування.

З іншого боку, віддалений кіберполігон має зручний доступ онлайн і залежить від віртуальної інфраструктури, розміщеної в хмарі. Він пропонує гнучкість у створенні різних сценаріїв та налаштувань, надає можливість швидкого доступу до глобальних ресурсів та інших хмарних послуг. Вартість віддаленого кіберполігону є зниженою у порівнянні з фізичним обладнанням, і його масштабованість дозволяє легко розширювати інфраструктуру для великої кількості користувачів. Технічне супроводження та підтримка здійснюються провайдером хмарної інфраструктури, а доступ до ресурсів не обмежений локальними ресурсами.

Оскільки в подальшому розробка системи буде проводитись на кафедрі університету, вибір залишається на локальному кіберполігоні. Це означає, що розглядається можливість створення та використання кіберполігону на власному обладнанні кафедри. Такий локальний кіберполігон надає більшу гнучкість та контроль над інфраструктурою та даними, оскільки він буде розташований внутрішньо в університетській мережі.

Використання локального кіберполігону дозволить забезпечити більшу конфіденційність та безпеку, оскільки всі дані та процеси будуть зберігатись та контролюватись внутрішньо в університетській мережі. Крім того, локальний кіберполігон дозволить студентам та дослідникам мати прямий доступ до системи та проводити експерименти та дослідження без необхідності залучення зовнішніх ресурсів. Однак, варто враховувати, що підтримка та розширення

локального кіберполігону може вимагати значних зусиль і ресурсів з боку університету. Необхідно буде забезпечити належну інфраструктуру, безпеку та підтримку системи, а також залучити кваліфікованих фахівців для розробки та підтримки кіберполігону.

1.3 Огляд інструментарію

У процесі розгортання кіберполігонів використовуються різноманітні технології та інструменти для створення ефективних та безпечних середовищ тренування та аналізу кіберзаходів. Ці технології надають можливість моделювання реалістичних сценаріїв кібератак, проведення експериментів та вдосконалення заходів кібербезпеки.

1.3.1 Класичний (фізичний) підхід

Якщо розглядати класичний підхід – на допомогу приходить фізичне забезпечення, за допомогою якого можна відтворити будь-яку мережу та цілу систему. З іншого боку, така масштабованість потребує значних витрат на обладнання, що створює обмеження у виборі необхідних пристроїв. Однак, використання фізичного забезпечення – не єдиний спосіб на сьогодення, та одним із масштабованих і, відносно, не дорогих способів є використання віртуальних машин.

1.3.2 Віртуалізація та віртуальні машини

Віртуалізація дозволяє створювати віртуальні середовища, які емулюють реальні мережі та системи використовуючи для цього віртуальні машини (Virtual Machines) та/або контейнери, які забезпечують ізоляцію та реплікацію окремих компонентів інфраструктури. Віртуалізація забезпечує гнучкість та масштабованість управління ресурсами та дозволяє створювати багатошарові архітектури для моделювання складних сценаріїв кібератак [11].

У сфері віртуалізації існує кілька відомих програм, що надають можливості створення віртуальних середовищ та ізоляції компонентів інфраструктури. Розглянемо деякі з них.

- VMware vSphere є одною з провідних платформ в галузі віртуалізації. Вона надає можливості створення та управління віртуальними машинами (VM), що дозволяє ізолювати та відтворювати окремі компоненти інфраструктури. VMware vSphere забезпечує широкі можливості для масштабування, управління ресурсами та забезпечення високої доступності віртуальних середовищ [12].
- Microsoft Hyper-V є платформою, яка входить до складу операційних систем Microsoft Windows Server. Вона надає можливості для створення та управління віртуальними машинами, дозволяючи ефективно використовувати ресурси фізичних серверів. Microsoft Hyper-V володіє рядом інтегрованих функцій, таких як забезпечення високої доступності та резервного копіювання [13].
- Oracle VM VirtualBox є безкоштовною платформою для віртуалізації, яка дозволяє створювати та запускати віртуальні машини на різних операційних системах, включаючи Windows, macOS та Linux. Вона надає гнучкі можливості налаштування віртуальних середовищ, таких як мережева конфігурація, обсяги дискового простору та доступ до периферійних пристроїв [14].

1.3.3 Системи генерації та управління трафіком

Для моделювання реальних загроз та кібератак широко використовуються системи генерації та управління трафіком, які дозволяють створювати реалістичний мережевий трафік, що імітує різні типи атак та навантаження. Вони дозволяють перевірити реакцію системи на атаки, виявити слабкі місця та оцінити ефективність захисних заходів. На сучасному ринку є чимало застосунків, які реалізують таку функцію, але було виділено чотири приклади систем генерації та управління трафіком, які широко використовуються для моделювання реальних загроз та кібератак:

- D-ITG є потужним інструментом для генерації реалістичного мережевого трафіку. Він надає можливість створювати трафік з різними типами пакетів, включаючи TCP, UDP та ICMP. D-ITG

використовується для моделювання різних типів атак та навантажень на мережу. Він дозволяє налаштовувати параметри трафіку, такі як розмір пакетів, інтервали між пакетами та багато іншого. Також цей засіб є відкритим програмним забезпеченням і має активну спільноту користувачів [15].

- Ixia BreakingPoint є розширеною системою для генерації та управління мережевим трафіком з акцентом на моделювання реалістичних загроз та кібератак. Ця система надає широкий спектр можливостей для створення різних видів трафіку, включаючи симуляцію атак, навантажень та вразливостей. Ixia BreakingPoint включає бібліотеку тестових векторів, яка містить сценарії тестування, що моделюють реальні загрози. Користувачі можуть створювати власні тестові вектори або використовувати готові розширення для моделювання різних типів атак. Ixia BreakingPoint також надає аналіз результатів тестування, включаючи виявлення вразливостей та проблем у системі [16].
- Spirent CyberFlood є комплексною системою для тестування кібербезпеки та моделювання кібератак. Вона надає засоби для генерації реалістичного мережевого трафіку з різними типами атак, включаючи DDoS, мережеві вразливості та шкідливі програми. Spirent CyberFlood дозволяє створювати складні сценарії тестування, де можна комбінувати різні типи атак та навантажень. Вона також надає засоби для аналізу результатів тестування та виявлення вразливостей у системі. Spirent CyberFlood підтримує широкий спектр протоколів та додаткові функціональні можливості для точного моделювання різних видів кібератак [17].
- tcpreplay є інструментом для відтворення зареєстрованого мережевого трафіку з метою тестування мережевих пристроїв та додатків. Він дозволяє відтворити реальний трафік, записаний з реальних мереж або згенерований штучно. tcpreplay зберігає весь

мережевий трафік у файлах PCAP, які потім можна використовувати для відтворення. Він підтримує різні опції, які дозволяють контролювати швидкість передачі пакетів, розподіл трафіку та інші параметри. tcpreplay є популярним інструментом серед мережних адміністраторів та фахівців у сфері кібербезпеки для тестування та аналізу мережевого трафіку [18].

Такі системи генерування та управління трафіком можуть використовуватись для створення реалістичного середовища, яке може точно відтворювати різні види кібератак. Вони дозволяють імітувати різноманітні загрози, такі як DDoS-атаки, мережеві вразливості, шкідливі програми та інші, що можуть стати загрозою для мережевої інфраструктури та систем безпеки, що дозволить тренувати та підготовити фахівців у сфері кібербезпеки на різних рівнях складності та різних типах загроз.

1.3.4 Системи пошуку збирання логів

Системи пошуку та збирання логів також грають важливу роль у розгортанні кіберполігонів, так як такі системи забезпечують ефективний збір, аналіз та візуалізацію даних, пов'язаних з активністю мережі та системи. Вони дозволяють збирати та аналізувати великі обсяги логів, моніторити різноманітні події та активність в мережі, а також виявляти підозрілі активності та атаки. Завдяки цим системам можна швидко виявляти вразливості та потенційні загрози, а також проводити подальший аналіз та розслідування інцидентів.

На сучасному ринку наявні чимало рішень, але з поміж них було обрано низку систем, які є провідними в цій галузі:

- Elasticsearch є потужною та розширеною системою пошуку та аналізу даних. Вона базується на Apache Lucene і забезпечує швидкий пошук, індексацію та аналіз великого обсягу даних. Elasticsearch використовує JSON-подібну структуру даних і має вбудовану підтримку розподіленого зберігання, автоматичну реплікацію та масштабованість. Вона підтримує розширені можливості пошуку, фільтрації, агрегації та візуалізації даних.

Elasticsearch також має широку спільноту користувачів та документацію [19].

- Apache Solr є високопродуктивною системою пошуку, індексування та аналізу даних. Вона також базується на Apache Lucene і надає розширені можливості пошуку, текстового аналізу, фасетного пошуку, геолокації та іншого. Solr підтримує горизонтальне масштабування, автоматичну реплікацію та розподілені запити. Вона також має багатий набір розширень та плагінів для розширення функціональності. Apache Solr також має активну спільноту та документацію [20].
- Microsoft Azure Search є хмарною платформою пошуку та аналізу даних, яка надає кероване рішення для пошуку в хмарі. Вона забезпечує легке використання, масштабованість та високу доступність. Azure Search підтримує індексацію та пошук різноманітних типів даних, включаючи текст, числа, геолокацію та багато іншого. Вона також надає розширені можливості фільтрації, сортування, агрегації та візуалізації даних. Azure Search інтегрується з іншими сервісами Microsoft Azure, що дозволяє легко розширювати його функціональність [21].

Проаналізувавши наявні рішення було проведено порівняння за рядом критеріїв, де було наведено наявність тих чи інших характеристик. За результатами цієї таблиці буде обрано засіб, що буде використовуватись в подальшій розробці кіберполігону.

У таблиці 1.2 представлено порівняння Elasticsearch та аналогів по різним характеристикам.

Таблиця 1.2 – Порівняння Elasticsearch та аналогів

<table>
<tr><th>Характеристика</th><th>Elasticsearch</th><th>Apache Solr</th><th>Microsoft Azure Search</th></tr>
<tr><td>Тип системи</td><td colspan="2">Розподілений</td><td>Хмарний</td></tr>
<tr><td>Повнотекстовий пошук</td><td colspan="3">Так</td></tr>
<tr><td>Масштабованість</td><td colspan="2" rowspan="2">Висока</td><td rowspan="2">Середня</td></tr>
<tr><td>Продуктивність</td></tr>
<tr><td>Легкість використання</td><td>Висока</td><td>Середня</td><td>Висока</td></tr>
<tr><td>Мови програмування</td><td colspan="2">Java, Python, PHP, і більше</td><td>.NET, Java, Python, і більше</td></tr>
<tr><td>Розширені можливості аналізу</td><td>Так</td><td>Так</td><td>Обмежені</td></tr>
<tr><td>Підтримка географічних даних</td><td colspan="3" rowspan="3">Так</td></tr>
<tr><td>Реплікація даних</td></tr>
<tr><td>Підтримка високої доступності</td></tr>
<tr><td>Ліцензія</td><td colspan="2">Apache License 2.0</td><td>Платна або безкоштовна</td></tr>
<tr><td>Екосистема</td><td colspan="3">Широка, активна спільнота</td></tr>
</table>

У порівнянні з аналогами, такими як Apache Solr та Microsoft Azure Search, Elasticsearch має декілька переваг. Він має більш простий у використанні та зрозумілий інтерфейс, що дозволяє розробникам швидше розгортати та налаштовувати систему, також відомий своєю високою продуктивністю та масштабованістю, що робить його популярним в областях, де необхідно обробляти великі обсяги даних (розподілений пошук, моніторинг систем та аналітика логів).

Головною особливістю цієї системи є те, що окрім пошуку та аналізу, вона виступає ще сховищем даних (документоорієнтована база даних). Основними одиницями збереження в Elasticsearch є індекси, типи та документи. Індекс – це контейнер, який містить однорідні документи з схожою структурою. Цей контейнер слугує для організації даних та надає можливість ефективного пошуку і аналізу. Кожен індекс може мати декілька типів, які використовуються для групування документів з подібною структурою [19].

Кожен документ у Elasticsearch представляє собою структуровану одиницю інформації, яка міститься в індексі. Документи визначаються у форматі

JSON (JavaScript Object Notation) та складаються з пар «ключ-значення». Вони можуть містити різні типи даних, такі як текст, числа, дати, масиви тощо. Кожен документ має унікальний ідентифікатор, що дозволяє однозначно ідентифікувати його в межах індексу. На рисунку 1.1 показано структуру даних в Elasticsearch [22].

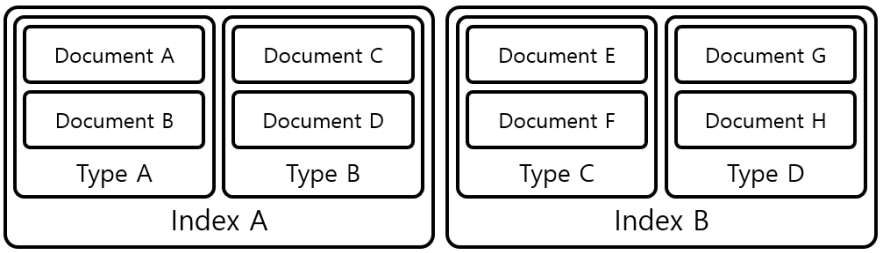


Рисунок 1.1 – Структура даних в Elasticsearch

У кожному документі поля відповідають різним аспектам даних. Наприклад, поле "title" може містити назву документа, а поле "content" – його вміст. Поля можуть мати різні типи даних, такі як текстові, числові, дати, географічні тощо. Elasticsearch автоматично індексує дані в кожному полі, що дозволяє ефективно шукати та аналізувати їх.

Elasticsearch розроблений на основі Apache Lucene, який є потужною, відкритою бібліотекою для повнотекстового пошуку. Розподілена архітектура є ключовою особливістю Elasticsearch, так як вона надає змогу горизонтально масштабувати систему, дозволяючи обробляти великі обсяги даних шляхом розподілу їх на кілька вузлів (нод). Кожен вузол в системі зберігає та обробляє частину даних, а Elasticsearch розподіляє завдання пошуку та аналізу між ними для досягнення оптимальної продуктивності [19].

Також варто розглянути, як працює розподілення даних в Elasticsearch та навести основні одиниці, які використовуються в ньому:

– Кластер (Cluster) Elasticsearch складається з однієї або більше фізичних або віртуальних машин, які об'єднуються для спільної роботи. Кожен кластер має унікальне ім'я, щоб його можна було ідентифікувати у мережі. Кластер забезпечує розподілене збереження та обробку даних.

- Нода (Node) є окремим екземпляром Elasticsearch, який працює в межах кластера. Кожна нода є самостійним сервером, який виконує обробку запитів, зберігання даних та інші функції Elasticsearch. Кластер може містити багато нод, що дозволяє розподілено обробляти навантаження та забезпечувати високу доступність даних.
- Шард (Shard) є базовим блоком розподіленого зберігання даних в Elasticsearch. Кожний індекс у кластері може бути розділений на декілька шардів, а кожен шард може бути розміщений на окремій ноді. Шарди дозволяють розподілити великі обсяги даних на декілька фізичних або віртуальних серверів, що поліпшує продуктивність та масштабованість Elasticsearch. Шарди також забезпечують відновлюваність даних в разі відмови ноди, оскільки кожний шард може мати репліки.
- Репліка (Replica) є копією шарду, яка забезпечує резервну копію даних та підвищує доступність Elasticsearch. Кожний шард може мати одну або більше репліку. Репліки автоматично синхронізуються з основним шардом і дозволяють розподілено обробляти запити і покращувати завантаження системи.

Ці компоненти – кластери, ноди та шарди – відіграють важливу роль у розподіленому збереженні даних та розподіленому обчисленні в Elasticsearch, так як за вони дозволяють забезпечити високу доступність, масштабованість та надійність системи, а також покращити продуктивність обробки даних [19].

Для кращого уявлення про те, як зберігаються дані в Elasticsearch, на рисунку 1.2 ілюстративно зображено архітектуру бази даних в Elasticsearch [23].

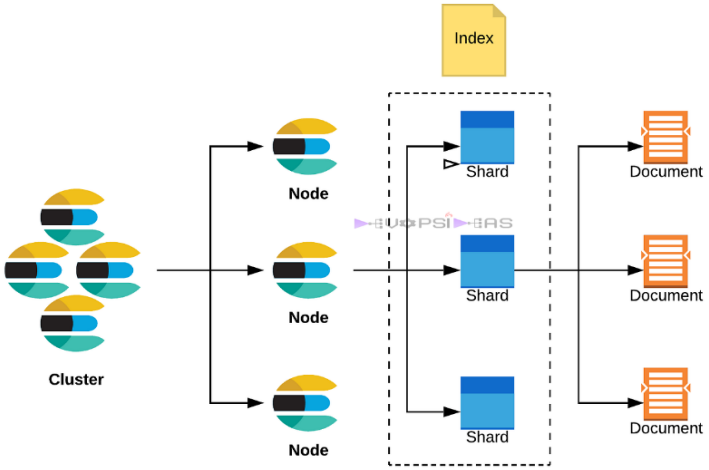


Рисунок 1.2 – Архітектура Elasticsearch

Враховуючи зазначені фактори, Elasticsearch може бути відмінним вибором для розробників та організацій, які потребують потужної та ефективної системи пошуку та аналізу даних. В подальшому за основу обробки логів в кіберполігоні буде обрано саме Elasticsearch в силу його можливостей та легкості використання.

Розглянуті інструменти надають широкий спектр функціональності для моделювання реальних загроз та кібератак, а також для тестування ефективності заходів захисту. Такі інструменти дозволяють створювати реалістичне середовище, що імітує різні види атак та навантажень, та надають можливості для аналізу результатів тестування та виявлення вразливостей.

Застосування цих інструментів при розробці системи на кафедрі університету дозволить студентам отримати практичний досвід у сфері кібербезпеки, створюючи можливість проведення експериментів з різними типами атак, параметрами трафіку та аналізувати результати, що допоможе розширити їхні знання та навички. Крім того, ці інструменти можуть бути використані для проведення лабораторних практикумів, де студенти будуть мати можливість розібратися з проблемами кібербезпеки на практиці та зробити поглиблений аналіз вразливостей та захисних заходів.

Отже, у цьому розділі було оглянуто основні наявні інструменти для впровадження кіберполігону, які в подальшому будуть взяті до уваги при розробці системи на кафедрі університету (або лабораторного практикуму).

1.4 Постановка завдання

Проаналізувавши наявні рішення і вже впровадженні кіберполігони, для подальшого виконання дипломної роботи було обрано рішення впровадження локального полігону. Такий варіант допоможе студентам власноруч створити та протестувати необхідний функціонал використовуючи як фізичні, так і віртуальні машини, а окрім цього, впровадження локального полігону дозволить студентам використовувати такий полігон в зручний для них час, без обмежень на кількість користувачів або ресурсів.

В рамках локального кіберполігону, який буде впроваджений для виконання дипломної роботи, будуть використовуватись дві сторони: атакуюча та захищаюча. Атакуюча сторона буде відповідальна за розробку та виконання різних видів атак на систему, що тестується. Захищаюча сторона буде відповідальна за захист системи та реагування на атаки з боку атакуючої сторони. Для реалізації цих сторін будуть використовуватись або фізичні, або віртуальні машини, на яких будуть запущені потрібні сервіси та програми.

Розробка та виконання атак буде відбуватись на окремій фізичній/віртуальній машині, яка буде виділена для атакуючої сторони. Захищаюча сторона буде захищати систему, яка запущена на іншій віртуальній або фізичній машині, та виконувати необхідні дії для запобігання атакам.

Отже, в результаті впровадження локального кіберполігону студенти зможуть власноруч створити та протестувати різні атаки та методи захисту в реальному часі, що дозволить отримати більш глибоке розуміння у використанні таких систем, на практиці отримати досвід та як захищати свої системи. Такий полігон може бути використаний як засіб підготовки до кіберспортивних змагань, де команди змагаються у здатності захистити свої системи від атак та проникнень і стати потужним інструментом для навчання та розвитку навичок в галузі кібербезпеки.

2 ОБГРУНТУВАННЯ ОБРАНИХ РІШЕНЬ

2.1 Схема кіберполігону

Кіберполігон складається з низки різних компонентів, які спільно створюють контрольоване середовище для моделювання реальних сценаріїв кібератак та виконання експериментів у сфері кібербезпеки. Основними компонентами кіберполігону виступають машини, які виконують роль атакуючих агентів та захищених систем.

Команда атакуючих, які можуть бути представлені фізичними або віртуальними машинами та/або контейнерами, виконують різноманітні атаки з метою тестування стійкості захищених систем. Атакуючі машини виконують роль зловмисників або хакерів, які мають зацікавленість у проникненні в мережеву систему та здійсненні різноманітних атак в арсеналі широкого кола підготовлених атак, використання різних методів, включаючи перехоплення пакетів, використання вразливостей або введення шкідливих програм, з метою отримання несанкціонованого доступу до ресурсів системи [24].

Захищені системи, в свою чергу, можуть бути представлені окремими фізичними або віртуальними машинами та/або контейнерами, які виконують роль об'єктів, що піддаються атакам та мають захисні механізми для запобігання або виявлення загроз та виконують роль оборони та захисту мережі від потенційних атак. Основною метою цих систем є забезпечення безпеки та недоступності для зловмисників. Для досягнення цієї мети, машини захисту використовують різноманітні захисні механізми, такі як брандмауери, системи виявлення вторгнень та системи захисту від вторгнень. Такі машини також можуть виявляти підозрілу активність за допомогою аналізу і моніторингу мережевого трафіку та системних журналів в пошуку незвичних дій або активності, що може свідчити про потенційні атаки. Це може включати спостереження за надмірними спробами входу, незвичайними запитами або незвичайною використанням ресурсів [24].

Обидві машини, як атакуюча, так і захищаюча, функціонуватимуть в локальній мережі, яка може бути побудована з використанням різних технологій, наприклад Ethernet або Wi-Fi, та мати різні фізичні топології, такі як зірка або шина. Ця локальна мережа створює необхідні засоби комунікації між машинами, що дозволяє їм обмінюватись даними під час проведення атак та захисту.

Комутатор, що поєднує машини в одну мережу є одним із ключових структури кіберполігону та виконує роль центрального мережевого пристрою, який забезпечує комутацію даних між машинами. Комутатор отримує мережеві пакети від однієї машини та пересилає їх до відповідних машин, враховуючи їх мережеві адреси. Цей процес забезпечує з'єднання та комунікацію між атакуючою та захищаючою машинами під час проведення атак та захисту (рис. 2.1).

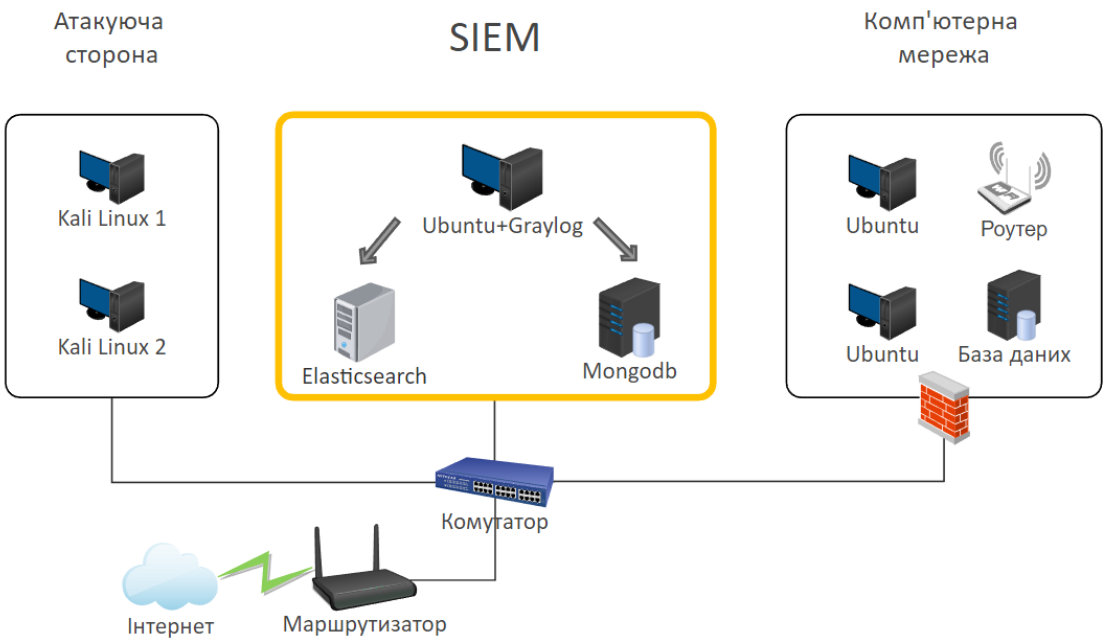


Рисунок 2.1 – Топологія майбутньої мережі

Отже, після визначення необхідних пристроїв і компонентів мережі можемо отримати ілюстративний вигляд кіберполігону. Цей вигляд включає детальне відображення фізичних пристроїв, які будуть використовуватись та впроваджуватись на практиці, а саме: сервери, маршрутизатори, комутатори, а також комп’ютери, що діють у мережі. Також було відображено логічні зв'язки, залежності між цими пристроями та компонентами мережі (див. рис. 2.1).

2.2 Потенційні атаки та сліди

Наступним важливим етапом забезпечення безпеки мережі полягає в розумінні і визначенні потенційних загроз і атак, з якими може зіткнутись система. Для цього необхідно провести аналіз поширених мережевих атак, описати ці атаки та визначити сліди, які вони залишають в логах. Такий підхід дозволяє підготуватись до можливих загроз, розробити ефективні методи розпізнавання цих атак і вчасно реагувати на них для забезпечення надійності та безпеки мережевої інфраструктури.

У таблиці 2.1 описано поширені мережеві атаки, їх опис та сліди, які вони залишають за собою в логах [25].

Таблиця 2.1 – Описи поширених мережевих атак та сліди, які вони залишають

Назва атаки	**Опис**	**Сліди атаки в логах**
DoS (Denial of Service) / DDoS (Distributed Denial of Service)	Атака, спрямована на перевантаження ресурсів системи, щоб унеможливити доступ для легітимних користувачів.	1. Зниження продуктивності мережевих ресурсів та сервісів. 2. Збільшена кількість запитів або спроб доступу до системи в логах. 3. Моніторинг рівня навантаження, кількості запитів та спроб доступу до системи.
Phishing	Атака, що маскується під достовірний джерело або ситуацію з метою обману користувачів і отримання їхніх особистих або фінансових даних.	1. Відмова в обслуговуванні широкого кола систем або ресурсів одночасно. 2. Значний потік запитів або незвичайний трафік в логах. 3. Аналіз джерела запитів, розподілений характер запитів
SQL Injection	Використання зловмисником неочищених або некоректно оброблених вхідних даних для виконання зловмисного SQL-коду на веб-сайті або базі даних.	1. Несанкціоноване внесення змін або витягування даних з бази даних. 2. Незвичайні або шкідливі запити SQL в логах. 3. Виявлення підозрілих символів або шаблонів, що вказують на SQL-ін’єкцію.
Man-in-the-Middle (MitM)	Зловмисник проникає між двома комунікуючими сторонами та перехоплює або змінює передачу даних між ними без їхнього відома.	1. Перехоплення та перенаправлення комунікації між двома сторонами. 2. Незвичайна поведінка проксі-серверів або зміна маршруту комунікації. 3. Аналіз записів комунікації, виявлення несподіваного або підозрілого середовища.

Продовження табл. 2.1

Назва атаки	Опис	Сліди атаки в логах
Cross-Site Scripting (XSS)	Впровадження зловмисного коду в веб-сторінки, який виконується на браузері користувача, з метою отримання доступу до конфіденційних даних або керування сесією.	1. Вбудовування шкідливого скрипту на веб-сторінку та його виконання в браузері користувача. 2. Наявність незвичайного або підозрілого коду в логах. 3. Аналіз вхідних даних та перевірка на наявність HTML/JavaScript-тегів.
Cross-Site Request Forgery (CSRF)	Атака, при якій зловмиснику вдається змусити аутентифікованого користувача здійснити небажані дії без його належного відома або згоди.	1. Наявність незвичайних або підозрілих запитів до веб-додатків. 2. Перевірка наявності неочікуваних або некоректних параметрів запиту. 3. Аналіз відповідей сервера, виявлення незвичайних дій, відправлених без згоди користувача.
DNS Spoofing	Атака, при якій зловмисник перехоплює DNS-запити із метою надати некоректні відповіді, спрямовуючи користувачів на шкідливі веб-сайти або фальшиві сервери.	1. Несподівана або неправильна адреса IP для доменного імені. 2. Моніторинг запитів DNS та перевірка відповідності адреси IP запитуваного домену.
Remote Code Execution (RCE)	Атака, при якій зловмисник виконує зловмисний код на вразливій системі з віддаленої локації, отримуючи повний контроль над нею.	1. Незвичайна активність або спроби виконання коду на вразливих системах. 2. Аналіз логів системи на наявність неавторизованого виконання команд або неочікуваної активності.
Brute Force Attack	Атака, при якій зловмиснику спроби вгадати або перебрати всі можливі комбінації паролів для злому доступу до системи або облікового запису.	1. Багато невдалих спроб авторизації з різних користувацьких акаунтів. 2. Моніторинг активності авторизації, виявлення підозрілих IP-адрес та незвичайних спроб доступу.
Social Engineering	Атака, що використовує маніпулювання людьми для отримання конфіденційної інформації або надання несанкціонованого доступу до системи.	1. Незвичайні або підозрілі взаємодії користувачів зі співробітниками або системою. 2. Аналіз журналів користувацької активності, виявлення нестандартних або підозрілих дій.
Zero-day Exploit	Атака, що використовує вразливості в програмному забезпеченні, які ще не відомі розробникам або не мають виправлень безпеки.	1. Незвичайна або підозріла активність на вразливих системах. 2. Аналіз журналів системних подій, виявлення незвичайних процесів, змін в системних ресурсах.

Кожна атака має свою унікальну характеристику і залишає певні сліди, які можна виявити шляхом моніторингу і збереження логів, які в подальшому можна проаналізувати, обробити або зробити відповідне правило.

Деякі атаки, такі як Denial of Service (DoS) та Distributed Denial of Service (DDoS), можна розпізнати за збільшеною кількістю запитів або незвичайним трафіком в логах. Тоді як інші атаки, наприклад SQL Injection або Cross-Site Scripting (XSS), можуть бути виявлені за незвичайними запитами або наявністю шкідливого коду в логах.

Після окреслення можливих слідів, які залишають після себе різні види атак стає значно простіше виявити та розпізнати ці атаки у мережевому середовищі. Такі сліди можуть бути знайдені за допомогою аналізу системних логів, виявлення аномальних дій, незвичайного трафіку, а також проведення перевірок на наявність підозрілих параметрів або даних. Розпізнавання атак забезпечує можливість прийняти відповідні заходи безпеки, такі як блокування шкідливого трафіку, відновлення системи до безпечного стану або запуск процедур розслідування. Такий підхід дозволяє забезпечити ефективний рівень безпеки мережі та зменшити можливість негативного впливу атак на функціонування і цілісність системи.

2.3 Обробка логів

Після отримання в попередньому розділі слідів, які можуть бути залишеними після атак, можна переходити до самого збору логів, для подальшого їх аналізу і реалізації ефективного механізму обробки цих логів. Цей процес включає такі етапи, як парсинг лог-файлів для отримання структурованої інформації, вилучення необхідних даних, виявлення патернів та аномалій, а також виконання різноманітних аналітичних операцій з метою виявлення цікавих аспектів та забезпечення високої якості даних для подальшого використання. На рисунку 2.3 структурно показано шлях, який проходять логи від джерела до їх використання [26].

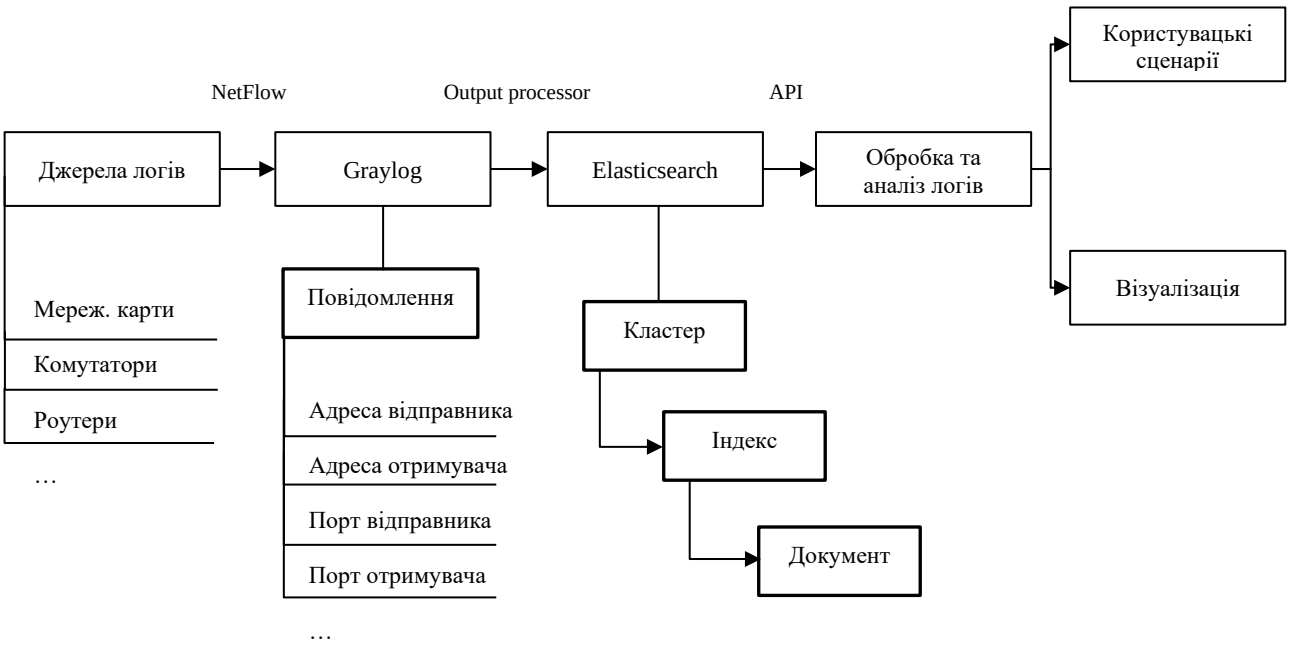


Рисунок 2.3 – Схема шляху, який проходять логи

Розглянемо кроки на прикладі атаки DoS/DDoS. Атака проходить через машину, через що залишає по собі слід у вигляді мережевих аномалій/сповільненої роботи мережі/логів. Джерелами логів можуть бути різні пристрої, які проводять моніторинг мережевого трафіку (мережеві екрани, системи виявлення вторгнень, системи логування або інші). Після отримання цієї інформації, логи потрапляють на сервер Graylog, далі, за допомогою вбудованих утиліт виводу логів, передається в Elasticsearch, де в подальшому може оброблятися за допомогою фільтрів та правил. Після отримання структурованої інформації, її можна буде зручно використати для виконання користувацьких сценаріїв або засобів візуалізації (наприклад Kibana), а додаткові налаштування та інші скрипти можуть бути використані для розширення можливостей аналізу.

2.4 Отримання логів

Як було й описано в попередньому розділі, джерелами логів можуть бути різні пристрої, які проводять моніторинг мережевого трафіку. Таким пристроєм може виступити як комутатор, так і мережева карта комп’ютера, яка буде слідкувати за трафіком, що буде передаватися саме через цей пристрій. І постає питання, як краще їх збирати. Хоча методів є чимало.

Перший з них – маршрутизаторо-орієнтований, другий – не орієнтований на маршрутизатори. Функціональність моніторингу, який вбудований у самі маршрутизатори і не вимагає додаткової установки програмного чи апаратного забезпечення, називають методами, що базуються на маршрутизаторі. Не засновані на маршрутизаторах методи вимагають встановлення апаратного та програмного забезпечення та надають більшу гнучкість [27].

Перш за все, розглянемо які є способи збору мережевого трафіку за допомогою маршрутизатора.

2.4.1 Маршрутизаторо-орієнтовані методи

Один з найпоширеніших методів моніторингу, а також збору логів, який використовується на маршрутизаторах – це протокол простого мережевого моніторингу (SNMP). Це протокол прикладного рівня, який є частиною протоколу TCP/IP, щодозволяє адміністраторам керувати продуктивністю мережі, знаходити та усувати мережеві проблеми, планувати зростання мережі. Він збирає статистику з трафіку до кінцевого хосту через пасивні датчики, які реалізуються разом із маршрутизатором.

Для протоколу SNMP притаманні три ключові компоненти: керовані пристрої (Managed Devices), агенти (Agents) та системи управління мережею (Network Management Systems – NMSs) [28]. На рисунку 2.4 показано основні компоненти SNMP і зв’язок між ними [28].

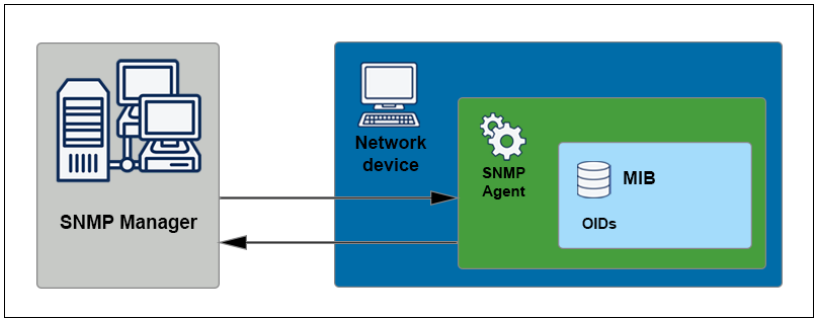


Рисунок 2.4 – Зв’язок між компонентами SNMP

Управляючі пристрої включають в себе агентів, які здійснюють збір статистики про трафік та інші параметри [28].

Для збору логів за допомогою SNMP, SNMP-агент на кожному пристрої повинен бути налаштований на відправку лог-повідомлень SNMP-менеджеру. Лог-повідомлення можуть містити інформацію про помилки, попередження, події та інші важливі події, що сталися на пристрої.

SNMP-менеджер отримує ці лог-повідомлення і може виконувати різні операції з ними, такі як архівування, візуалізація, аналіз, генерація сповіщень тощо. Залежно від можливостей SNMP-менеджера, можна виконати широкий спектр дій зі збраними лог-даними.

Важливо зазначити, що SNMP зазвичай використовується для збору простої моніторингової інформації, а не для повноцінного збору та аналізу логів. Тому, цей варіант не повністю підійде для подальшої розробки кіберполігону, розглянемо інший метод моніторингу.

Віддалений моніторинг (RMON) розширює можливості протоколу простого мережевого моніторингу (SNMP) шляхом включення різноманітних мережевих моніторів та консольних систем. RMON дозволяє змінювати та аналізувати дані, отримані під час моніторингу мережі.

Існує дві специфікації віддаленого моніторингу мережі – RMON1 і RMON2 (розширення до RMON1). Він додає ще дев’ять груп даних, які відносяться до рівня мережі (рівень 3) і рівня додатків (рівень 7) моделі OSI [29].

RMON1 Management Information Base (MIB) – ієрархічна база даних, яка визначає інформацію, яку консоль RMON може запитувати в агента за допомогою SNMP, і надає статистику трафіку на контролі доступу до середовища (MAC) і фізичних рівнях [29]. Групи, які використовує RMON1, зображені на рисунку 2.5 [29].

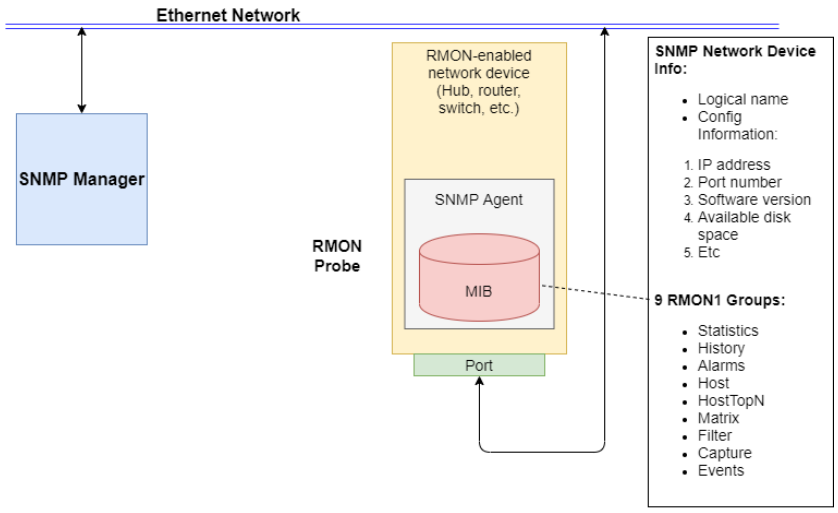


Рисунок 2.5 – Групи RMON1

RMON1 (Remote Monitoring) є протоколом, який використовує дев'ять різних груп моніторингу для отримання різних видів інформації про мережу. Ці групи включають Statistics (Статистика), яка збирає статистичні дані про кожен інтерфейс моніторингу пристрою. Це дозволяє отримати інформацію про пропускну здатність, кількість пакетів, помилки та інші показники, які можуть бути корисними для аналізу мережі.

Ще одна група - History (Історія), яка записує періодичні статистичні вибірки з мережі для подальшого аналізу. Це дає можливість вивчати тренди використання мережевих ресурсів та виявляти аномалії або проблеми, які можуть виникнути з часом.

Alarm (Сповіщення) група порівнює статистичні дані з заданими пороговими значеннями та генерує сповіщення про виникнення подій. Це дозволяє операторам мережі вчасно реагувати на проблеми або ненормальну активність, таку як перевищення пропускної здатності або висока кількість помилок.

Інші групи, такі як Host (Хост), HostTopN (Вершина хостів), Filters (Фільтри), Packet capture (Захоплення пакетів), Events (Події) та Token ring (Кільцеві лексеми), надають різноманітну інформацію про стан хостів у мережі, аналізують трафік на основі певних критеріїв, захоплюють пакети для

подальшого аналізу, контролюють генерацію та реєстрацію подій та забезпечують моніторинг мережі на основі кільцевих лексем.

RMON2 MIB (Management Information Base) розширює можливості RMON1, надаючи більш детальну інформацію про статистику трафіку. Він включає такі групи, як Protocol Directory (Каталог протоколів), Protocol Distribution (Розповсюдження протоколу), Network Layer Host (Хост мережевого рівня), Network Layer Matrix (Матриця мережевого рівня), Application Layer Host (Хост прикладного рівня), Application Layer Matrix (Матриця прикладного рівня), User History (Історія користувача), Probe Configuration (Конфігурація датчика) та RMON Conformance (Відповідність RMON).

Ці групи дозволяють отримати додаткову інформацію про протоколи, що використовуються в мережі, розподіл протоколів, обсяг трафіку за протоколами, історію використання користувачами та встановлення параметрів конфігурації датчика RMON2. Вони є корисними для адміністраторів мережі для аналізу трафіку, виявлення проблем та планування ресурсів мережі [29].

RMON має свої переваги та недоліки. Одна з переваг полягає у можливості отримання розширеної моніторингової інформації. В порівнянні зі SNMP, RMON надає більш детальні дані про мережевий трафік, включаючи розмір пакетів, час відправки та отримання, заголовки й інші параметри, що дозволяють проводити глибокий аналіз.

Ще одна перевага RMON полягає в його незалежності від виробника обладнання. Цей стандарт підтримується більшістю виробників, що дає можливість збирати дані з різних пристроїв і проводити централізований моніторинг та аналіз на спеціальному сервері.

Проте, RMON має й свої недоліки. Використання RMON вимагає більшої складності в налаштуванні та розгортанні порівняно з SNMP. Для ефективного аналізу та інтерпретації отриманих даних може знадобитися додатковий рівень знань та досвіду.

Крім того, збір детальних даних за допомогою RMON може створювати велике навантаження на мережу, особливо при великому обсязі трафіку. Це може

вплинути на продуктивність мережі та спричинити перевантаження мережевих з'єднань. Тому, при використанні RMON, слід уважно враховувати цей фактор і вживати заходів для зменшення навантаження на мережу.

Розглянемо ще один метод моніторингу, заснований на маршрутизаторах – NetFlow, протокол мережевого моніторингу, який використовується для збору та аналізу даних про мережевий трафік. Цей протокол дозволяє отримати інформацію про споживання пропускної здатності мережі, джерела та призначення пакетів, а також інші важливі метрики.

Його інфраструктура складається з трьох компонентів: FlowCaching, FlowCollector і Data Analyzer [30]. Структуру та приклад використання зображено на рисунку 2.6 [30].

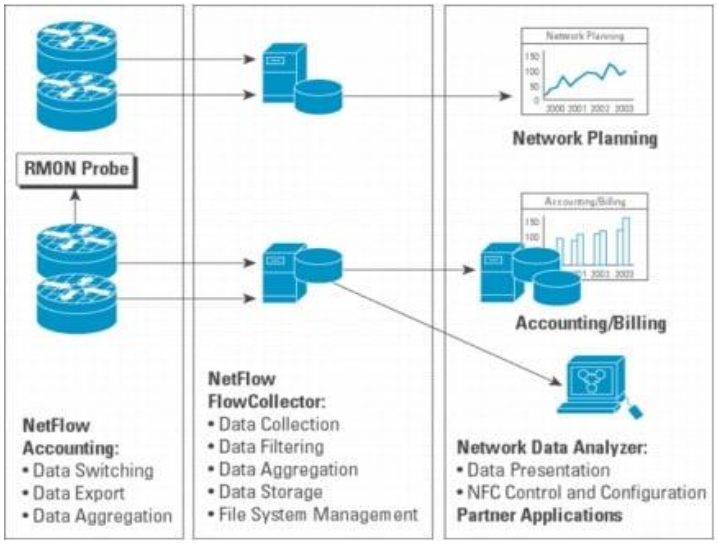


Рисунок 2.6 – Структура NetFlow

FlowCaching відповідає за аналіз та збір даних про IP-потоки, які проходять через інтерфейс. Він створює кеш-записи для активних потоків, що містять інформацію про кількість пакетів та байтів у кожному потоці. FlowCollector відповідає за збір, фільтрацію та збереження даних. Він збирає історичну інформацію про потоки, що пройшли через інтерфейс. FlowCollector також може застосовувати фільтри та агрегувати дані, щоб зменшити обсяг і зберігати лише потрібну інформацію. Data Analyzer використовується для подання та аналізу даних. Зібрані дані можуть бути використані для різних цілей, таких як моніторинг мережі, планування, облік та побудова мережі. Інструменти, такі як

NetFlow Analyzer [31], допомагають адміністраторам аналізувати дані NetFlow та відображати їх у зручному для користувача вигляді, наприклад, у вигляді діаграм та графіків.

Розглянемо кілька переваг та недоліків протоколу NetFlow.

NetFlow дозволяє збирати докладну інформацію про трафік, включаючи джерело, призначення, порти, протоколи, обсяги трафіку тощо, що дає змогу аналізувати мережеву активність з високою роздільною здатністю і зробити докладніші висновки про роботу мережі. Також, використовуючи NetFlow навантаженість на мережеве обладнання менша, в порівнянні з попередніми протоколами. SNMP і RMON можуть вимагати великого обсягу трафіку, оскільки вони надсилають запити на збір даних, в той час як NetFlow працює на базі потоків, що дозволяє значно зменшити навантаження на мережеве обладнання. Останньою перевагою, яка є суттєвою для створення кіберполігону, це аналіз мережевого трафіку в реальному часі, якраз NetFlow надає таку можливіст. Використовуючи цей протокол є можливість оперативно виявляти аномальну активність, проблеми з безпекою або загрози, що може зберегти час і зусилля при вирішенні проблем мережі. Хоча, і цей протокол не обійшовся без недоліків, для використання протоколу NetFlow необхідна підтримка з боку мережевого обладнання. Не всі маршрутизатори та комутатори підтримують цей протокол, тому може знадобитися оновлення обладнання або використання спеціальних модулів. NetFlow може збирати великі обсяги даних про трафік, що вимагає значних ресурсів для зберігання та обробки цих даних. При неправильному налаштуванні може виникнути проблема з обробкою інформації та зайнятістю ресурсів.

2.4.2 Не зв’язані з маршрутизатором методи

Активний моніторинг використовує пробні вимірювання між двома кінцевими точками для виявлення проблем у мережі. Цей підхід включає вимірювання метрик, таких як корисність, маршрутизатори/маршрути, затримка пакетів, повтор пакетів, втрати пакетів та нестійка синхронізація прибуття [27].

Прикладом активних інструментів вимірювання – команда ping, яка відправляє ICMP-пакети до точки призначення та вимірює затримку та втрати пакетів. Також інструмент traceroute використовується для визначення топології мережі шляхом відстеження шляху, який пакети проходять від джерела до призначення. Рисунок 2.7 зображує приклад активного моніторингу за допомогою EXO-запитів [27].

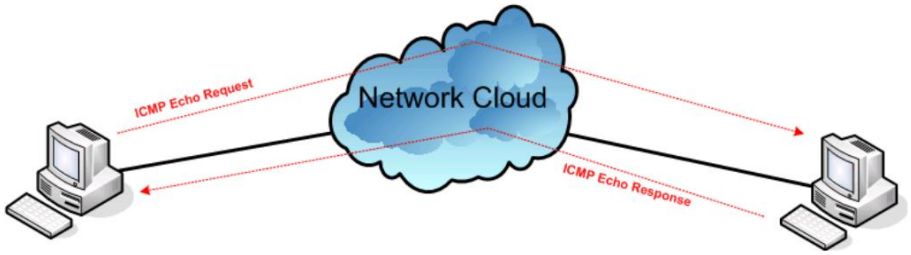


Рисунок 2.7 – Приклад активного моніторингу

Варто зазначити, що активний моніторинг може впливати на нормальний трафік мережі, оскільки пробні пакети можуть втручатися в роботу мережі. Це може ставити під сумнів достовірність отриманих даних. Загалом активний моніторинг є менш поширеним методом порівняно з пасивним моніторингом, оскільки він вимагає додаткових ресурсів мережі та може впливати на її нормальну роботу.

В контексті збирання логів для кіберполігону, активний моніторинг може використовуватися для збору і аналізу інформації про стан мережі та виявлення потенційних кіберзагроз. Пробні вимірювання, такі як використання команди ping або інструменту traceroute, можуть допомогти виявити аномалії, затримки або втрати пакетів, що можуть свідчити про атаки або несправності в мережевій інфраструктурі. Проте, через втручання в трафік може збільшити кількість та розкинути справжні запити, які залишають після себе мережеві пристрої, що в подальшому може викликати чимало проблем з обробкою та аналізом цих даних.

З іншого боку, пасивний моніторинг може бути використаний для збору інформації про трафік у мережі без втручання у нього. Використання програм, що витягують пакети, дозволяє отримувати дані про трафік, суміш протоколів, бітрейт, синхронізацію пакетів та інші характеристики. Ці дані також можуть

бути записані у вигляді логів для подальшого аналізу та виявлення аномалій, атак або незвичних активностей в мережі.

Також на відміну від активного моніторингу, пасивний збирає інформацію лише про одну точку в мережі. Вимірювання відбуваються набагато краще, ніж між двома точками при активному моніторингу. На рисунку 2.8 зображено установку системи пасивного моніторингу, де монітор розміщений на одиничному каналі між двома кінцевими точками та спостерігає трафік коли той проходить каналом [27].

Рисунок 2.8 – Встановлення пасивного моніторингу

Хоча пасивний моніторинг немає витрат, які має активний моніторинг, він має свої недоліки. З пасивним моніторингом, вимірювання можуть бути проаналізовані лише оф-лайн і вони не представляють колекції. Це створює проблему, пов'язану з обробкою великих наборів даних, зібраних під час вимірювання.

Також існує комбінація цих двох методів моніторингу (комбінований метод), який дозволяє поєднати переваги активного та пасивного моніторингу. Активний моніторинг може використовуватися для проведення цілеспрямованих вимірювань та діагностики проблем, тоді як пасивний моніторинг забезпечує неперервний збір інформації про трафік. Комбінація цих двох підходів дозволяє отримати більш повну та об'єктивну картину мережевої активності, що допомагає виявляти аномалії, атаки або незвичайні події в мережі.

У результаті комбінований метод моніторингу забезпечує більш широкий охоплюючий підхід до збору логів у кіберполігоні, що сприяє виявленню та реагуванню на кіберінциденти швидше та ефективніше, що допоможе реалізувати моніторинг логів для подальшої розробки кіберполігону [27].

Проаналізувавши наявні методи та виокремивши найкращий спосіб щоб повністю реалізувати отримання мережевого трафіку, обрано використання програм, які працюватимуть у фоні – демонів (daemons), та проводять збір за протоколом NetFlow (комбінований метод).

2.5 Розробка і аналіз фільтрів та агрегацій

Наступним кроком потрібно провести аналіз та встановити, яка саме інформація буде піддаватися моніторингу або вилученню з журналів (події, певні умови або фільтри, що будуть використовуватися для відбору необхідних даних) [32].

Для кращого уявлення про процес обробки логів у системі Elasticsearch, на рисунку 2.9 наведено структурну схему на якій зображено компоненти та послідовності дій, що відбуваються під час обробки логів.

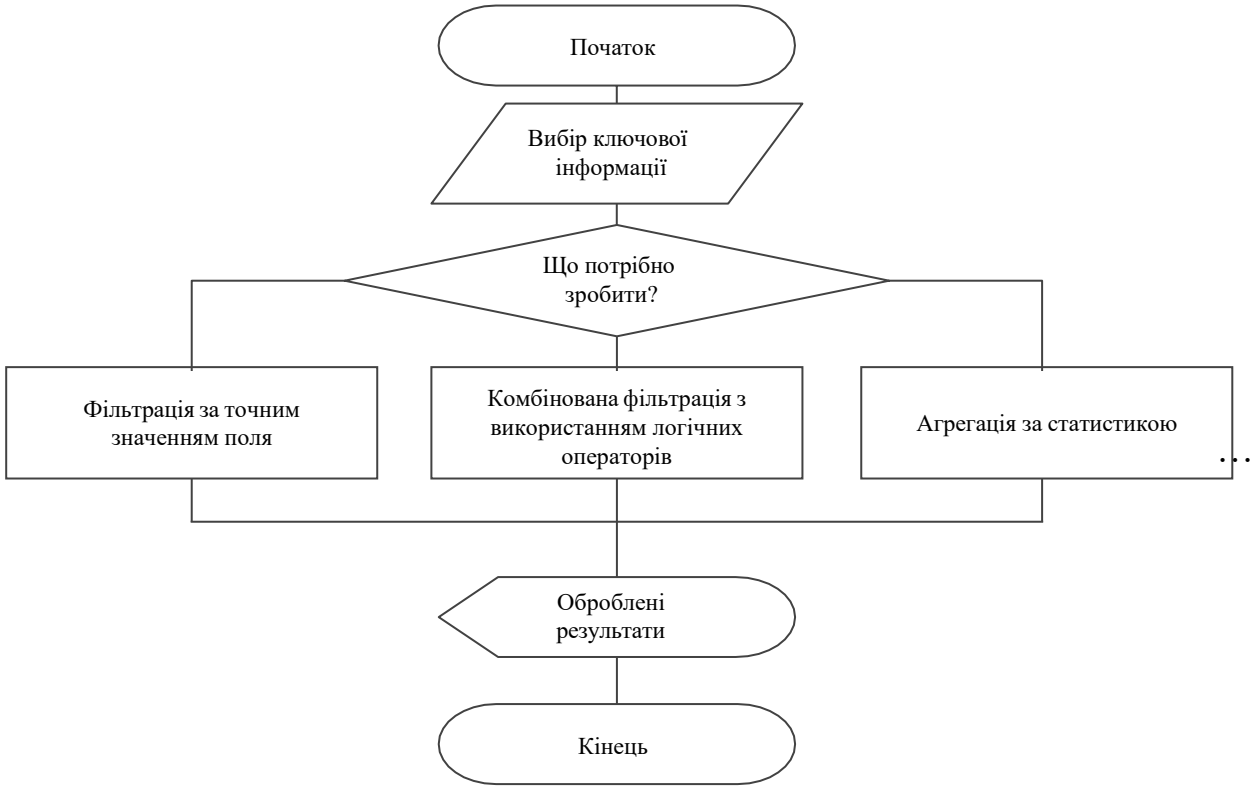


Рисунок 2.9 – Структурна схема обробки логів

Перш за все, потрібно обрати ключову інформацію, наприклад, потрібно отримати всі записи за певною IP-адресою. Отже ключовою інформацією в

цьому випадку буде конкретна адреса, яка в подальшому буде оброблятись за допомогою фільтрів та агрегацій. Що потрібно зробити? Знайти всі записи. І на основі цього вже будуть обиратись фільтри та агрегації для кращого опрацювання та виводу результатів. Структурна схема не дає можливості показати всі можливі агрегації та фільтри в Elasticsearch, але низку із них буде описано далі, засновучись на офіційній документації [32].

Фільтри використовуються засновуючись на ключовій інформації, та яку дію потрібно виконати. Наприклад, якщо шукати точну назву пристрою чи адресу, на яку проводились атаки, потрібно використати фільтр за точним значенням. Такий фільтр дозволяє швидко вибрати документи, які мають конкретне значення у певному полі і оперативно отримати бажаний результат. За допомогою нього можна шукати атаки, що спрямовані на певний пристрій за його точною назвою або IP-адресою.

У фільтрі за точним значенням поля використовується запит типу *term*. Такий фільтр дозволяє вибрати документи з певним точним значенням у вказаному полі. Для прикладу, з нахилом в кіберполігон, можна реалізувати запит для вибору атак на пристрій з точною назвою "router":

```
{ "query": { "term": { "device_name": "router" } } }
```

У випадку коли потрібно отримати дані за діапазоном значень, на допомогу прийде наявна фільтрація за діапазоном, такий фільтр допомагає обмежити результати запиту до документів, які мають значення у певному діапазоні. У фільтрі за діапазоном значень поля використовується запит типу *range*. Для цього потрібно вказати поле, де буде встановлений діапазон, а також значення *gte* (більше або рівне) і *lte* (менше або рівне). Приклад запиту для вибору атак, які сталися між 1-го січня 2023 року і 31-го грудня 2023 року:

```
{ "query": { "range": { "attack_date": { "gte": "2023-01-01",
"lte": "2023-12-31" } } } }
```

Фільтрація за частковим збігом дозволяє шукати документи, в яких певне текстове поле містить певну частину слова, буде в нагоді, коли буде важко написати всю назву пристрою. Для цього фільтру використовується запит типу

match. Наприклад, якщо нам потрібно знайти пристрої, чия назва містить підрядок, "fire" варто скористатись таким запитом:

```
{ "query": { "match": { "device_name": "fire" } } }
```

Також фільтрація за наявністю поля використовується для вибору даних, в яких певне поле присутнє або відсутнє. У фільтрі за наявністю поля використовується запит типу *exists*. Щоб перевірити наявність поля – потрібно вказати його ім’я. Наприклад, щоб вибрати логи, які мають поле "system_log", буде використаний такий запит:

```
{ "query": { "exists": { "field": "system_log" } } }
```

Комбінована фільтрація з використанням логічних операторів дозволяє створювати складніші фільтри, використовуючи логічні оператори AND, OR і NOT. Це дозволяє комбінувати різні умови фільтрації для точнішого вибору документів, які задовольняють певним критеріям. Комбінована фільтрація використовує запит типу *bool*, де можна використовувати логічні оператори (*must, must_not, should*) для поєднання інших фільтрів разом. З таким фільтром можна вказати різні умови для різних полів. Наприклад, щоб вибрати атаки, які сталися на певний пристрій у певний день тижня, наступний запит буде у нагоді:

```
{ "query": { "bool": { "must": [ { "term": { "device_name": "router" } }, { "term": { "day_of_week": "Monday" } } ] } } }
```

Також, для кращого аналізу варто використовувати агрегації, що використовуються для отримання агрегованих результатів з великого обсягу даних та дозволяють здійснювати різноманітні операції, такі як підрахунок кількості документів, обчислення середнього значення, максимального або мінімального значення, розподілу даних, групування за певними критеріями тощо.

Для використання агрегацій в Elasticsearch потрібно скласти запит, який включає спеціальні об'єкти агрегацій. Ці об'єкти визначають тип агрегації (наприклад, термін, діапазон, середнє значення) і поле, за яким потрібно провести аналіз. Запит може містити одну або кілька агрегацій, що дозволяє отримувати комплексні аналітичні дані. Після виконання запиту Elasticsearch виконує агрегації на основі даних у вказаному індексі і повертає результати.

Для кращого розуміння, далі будуть наведені кілька прикладів агрегацій та їх використання [33]. Агрегація термінів дозволяє згрупувати дані за певним полем і підрахувати кількість входжень кожного терміну, що може бути корисно для створення розподілу даних, знаходження найпопулярніших значень та групування даних за певною характеристикою. Припустимо, що ми хочемо згрупувати дані за полем "source_ip" і підрахувати кількість входжень кожного терміну (IP-адреса). Запит для цієї агрегації в може мати такий вигляд:

```
{ "size": 0, "aggs": { "source_ip_agg": { "terms": { "field": "source_ip", "size": 10 } } } }
```

Щоб розбити дані на певні інтервали значень поля і підрахувати кількість документів, що потрапляють в кожен діапазон, можна використати агрегацію діапазону. Агрегація діапазону може бути використана для аналізу числових даних, наприклад, розміру пакетів або тривалості з'єднання. Вона дозволяє розбити дані на інтервали (наприклад, 0-100, 100-500, 500+) і визначити кількість випадків у кожному діапазоні варто скористатись наступним запитом:

```
{ "aggs": { "packet_size_ranges": { "range": { "field": "packet_size", "ranges": [{ "to": 1000 }, { "from": 1000, "to": 5000 }, { "from": 5000 } ] } } } }
```

Для того щоб обчислити середнє значення числового поля варто використати агрегацію середнього значення. Для кіберполігону це може бути в нагоді для визначення середнього часу проведених атак, середнього розміру пакету тощо. Для прикладу, щоб обчислити середній час проведених атак потрібно використати наступний запит:

```
{ "aggs": { "avg_attack_duration": { "avg": { "field": "attack_duration" } } } }
```

Агрегації максимуму і мінімуму дозволяють знайти найбільше і найменше значення числового поля, наприклад, максимальну швидкість передачі даних або найменший час відновлення системи після інциденту. Це допомагає ідентифікувати екстремальні значення та визначити межі нормальної роботи системи, а наступний запит допоможе отримати значення максимуму та мінімуму для конкретних полів:

```
{ "aggs": { "max_data_transfer": { "max": { "field": "data_transfer" } }, "min_data_transfer": { "min": { "field": "data_transfer" } } } }
```

Агрегація кількості документів дозволяє підрахувати загальну кількість документів, що задовольняють певний критерій. Наприклад, можна підрахувати кількість інцидентів за типом або кількість вразливостей, що були виявлені за певний період. Це дозволяє отримати загальну статистику та визначити активність та розподіл даних. Приклад запиту для підрахунку кількості інцидентів за типом:

```
{ "aggs": { "incident_count": { "value_count": { "field": "incident_type " } } } }
```

Наведені приклади були наведені для пояснення синтаксису на основі прикладів і будуть змінюватися відповідно до конкретної схеми даних та вимог пошуку. Отже, на основі наявних фільтрів, та можливості розширити їх за допомогою комбінацій, логічних операторів і агрегацій, буде здійснюватися обробка необхідних даних при аналізі інформації про певні події, а далі на їх основі отримувати структуровані частини для подальшого ретельного опрацювання.

2.6 Висновок до розділу

Отже, в ході опрацювання даного розділу розглянуто майбутню структуру кіберполігону, який складається з частини атакуючих та захищаючих, наведено топологію мережі та описано, які компоненти використовуватимуться при розгортанні. Оскільки в подальшому будемо працювати з різними атаками, наведено таблицю з потенційними атаками та слідами, які вони залишають після себе. Розглянуто шлях, який проходять логи, від джерел до їх використання. Наступним кроком наведено кілька способів збирання логів, що складалися з методами, які засновані на маршрутизаторах та без них, та обрано оптимальний варіант для подальшого впровадження. Також розглянуто приклади фільтрів та агрегацій, що будуть в нагоді при створенні запитів для отримання інформації з логів.

3 ЕКСПЕРЕМЕНТАЛЬНІ ДОСЛІДЖЕННЯ

3.1 Впровадження та налаштування Elasticsearch

Для того, щоб встановити та налаштувати необхідне середовище, обрано дистрибутив Ubuntu 22.04, оновимо необхідні компоненти та проведемо інсталяцію потрібних застосунків. Виконаємо оновлення списку пакетів і повне оновлення всіх встановлених пакетів в системі Ubuntu до найновішої версії та перезавантажимо систему.

```
sudo apt update && sudo apt -y full-upgrade
[ -f /var/run/reboot-required ] && sudo reboot -f
```

Встановимо необхідні пакети для функціонування Elasticsearch, а саме Java Development Kit (JDK) версії 11 без графічного інтерфейсу, яка необхідна для запуску Elasticsearch. Використаємо наступний ряд команд:

```
sudo apt update
sudo apt install openjdk-11-jre-headless
java -version
```

В результаті буде перевірено встановлену версію Java, щоб впевнитись, що все встановилось без проблем (рис. 3.1).

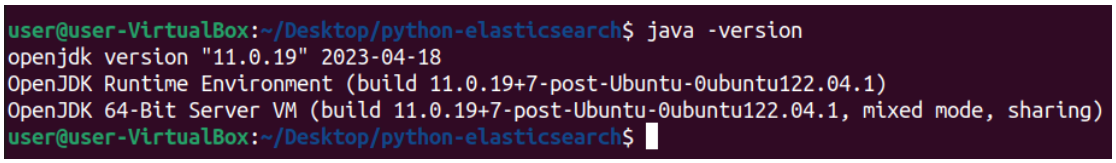


Рисунок 3.1 – Перевірка успішного встановлення Java

Компоненти Elasticsearch недоступні в стандартних репозиторіях пакетів Ubuntu. Однак їх можна встановити за допомогою APT після додавання списку джерел пакетів Elastic. Виконаємо наступні команди для встановлення.

```
apt install curl
curl -fsSL https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo gpg --dearmor -o /usr/share/keyrings/elastic.gpg
echo "deb [signed-by=/usr/share/keyrings/elastic.gpg] https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-7.x.list
```

```
sudo apt update
sudo apt install elasticsearch
```

Далі проведемо налаштування конфігураційних файлів. Налаштування Elasticsearch знаходяться у файлі /etc/elasticsearch/elasticsearch.yml і так як в подальшому ми будемо працювати з системою Graylog – пропишемо це в конфігураційному файлі. За замовчуванням Elasticsearch прослуховує все наявні мережеві інтерфейси.

```
sudo nano /etc/elasticsearch/elasticsearch.yml
```

Налаштування конфігураційного файлу Elasticsearch проілюстровано на рисунку 3.2.

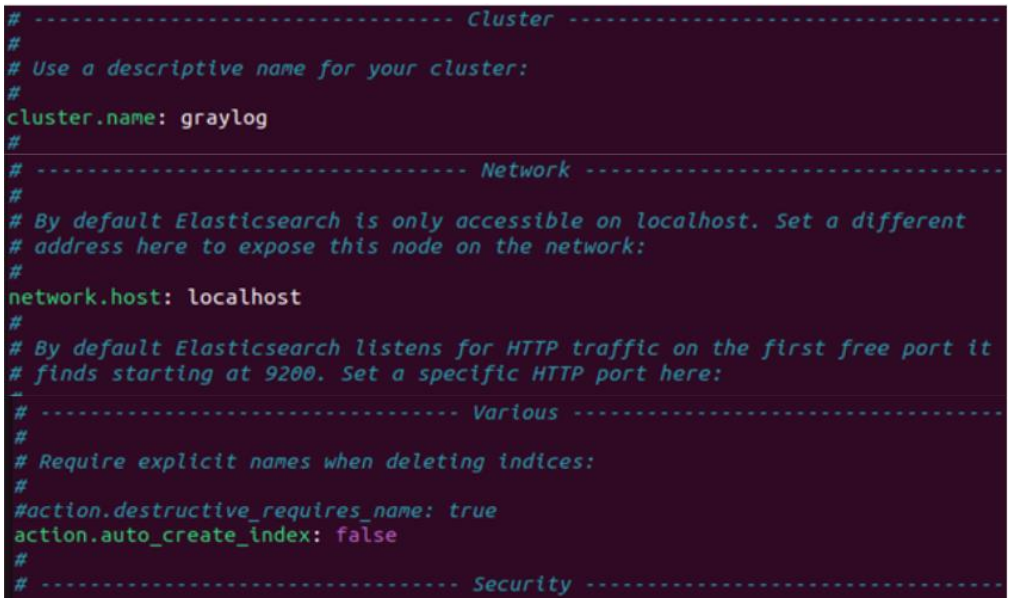
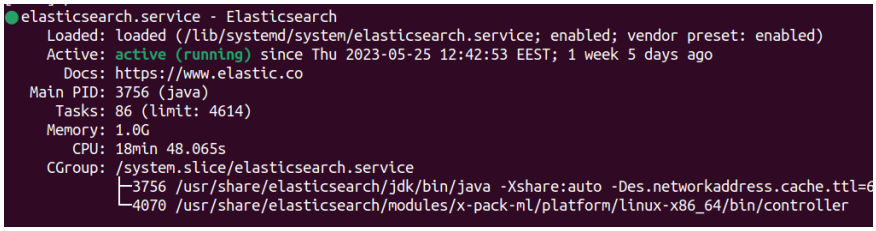


Рисунок 3.2 – Налаштування в конфігураційному файлі Elasticsearch

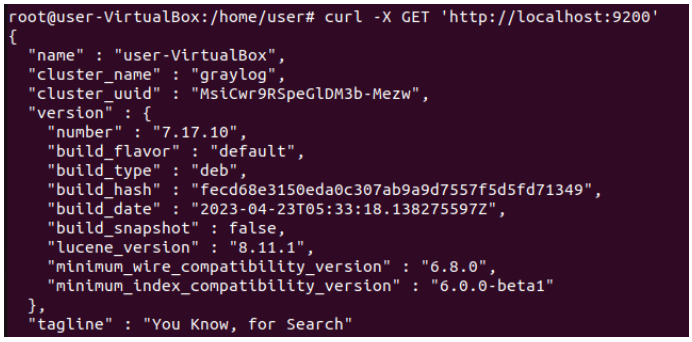
Перевіримо роботу та статус сервісу Elasticsearch за допомогою наступних команд:

```
sudo systemctl start elasticsearch
sudo systemctl status elasticsearch
sudo systemctl enable elasticsearch
curl -X GET 'http://localhost:9200'
```

Перевіримо працездатність Elasticsearch за допомогою команди, що управляє процесами “systemctl” (рис.3.3 (а)) та за допомогою API запиту до Elasticsearch (рис. 3.3 (б)).



а)



б)

Рисунок 3.3 – Результати перевірки налаштувань та працездатності Elasticsearch (а – перевірка працездатності сервісу через systemctl, б – перевірка за допомогою API запиту)

Після виконання ряду команд для встановлення та проведення налаштувань, встановлення необхідних пакетів, налаштування конфігураційних файлів та впровадження взаємодії з іншими компонентами системи, отримано успішно налагоджено робочу систему Elasticsearch. Тепер, Elasticsearch готовий до приймання, індексації та аналізу великого обсягу даних, що надає можливості для виявлення цікавих тенденцій, залежностей та здійснення розширених пошукових запитів.

Таким чином, етап встановлення та налаштування Elasticsearch успішно завершений, застосунок працює належним чином і наступним кроком йде збір логів.

3.2 Налаштування та збір логів

Наступним кроком, потрібно реалізувати збір логів та використати для цього зручний метод, для використання як на фізичних, так і на віртуальних

машинах. В ході аналізу наведеного в попередніх розділах визначено, що для цього краще використати демони (daemons) та протокол NetFlow.

Для того, щоб виконати це завдання, в нагоді стане утиліта fprobe – це інструмент, призначений для перехоплення мережевого трафіку і генерації NetFlow-потоків, та використовується для некерованих мереж, де немає підтримки протоколу NetFlow на мережевих пристроях.

Робота Fprobe полягає в тому, що він прослуховує мережевий трафік і аналізує його, створюючи при цьому NetFlow-потоки, які містять інформацію про джерело, призначення, протоколи, порти та інші характеристики пакетів. Далі такі потоки можуть бути направлені на сервер, який підтримує аналіз NetFlow для подальшої обробки [34].

Встановимо та налаштуємо необхідні компоненти.

```
sudo apt-get update
sudo apt-get install fprobe
```

Наступне, що потрібно зробити – це встановити відповідні налаштування при встановленні, в подальшому можна змінити в конфігураційному файлі (/etc/default/fprobe). Для цього використаємо наступні команди:

```
ifconfig
nano /etc/default/fprobe
```

В результаті виконання команд буде перевірено наявні мережеві інтерфейси (рис.3.4) та налаштувань в конфігураційному файлі fprobe (рис.3.5).

```
user@user-VirtualBox:~/Desktop/python-elasticsearch$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.137.128  netmask 255.255.255.0  broadcast 192.168.137.255
        inet6 fe80::2827:f122:e092:1a9e  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:90:50:7e  txqueuelen 1000  (Ethernet)
        RX packets 473374  bytes 510961488 (510.9 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 191863  bytes 22421036 (22.4 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Рисунок 3.4 – Результати перевірки наявних мережевих інтерфейсів

```
#fprobe default configuration file

INTERFACE="ens33"
FLOW_COLLECTOR="localhost:2055"

#fprobe can't distinguish IP packet from other (e.g. ARP)
OTHER_ARGS="-fip"
```

Рисунок 3.5 – Перевірка налаштувань в конфігураційному файлі fprobe

Перезавантажимо утиліту та перевіримо її статус використовуючи наступні команди:

```
sudo service fprobe restart
sudo service fprobe status
```

В ході використання команд буде проведена перевірки налаштувань та працездатності fprobe (рис. 3.6).

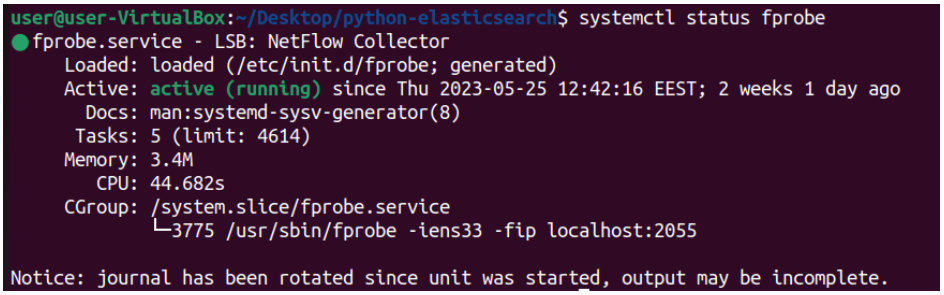


Рисунок 3.6 – Результати перевірки налаштувань та працездатності fprobe

Отже, після виконаних дій маємо робочий та налаштований колектор, який працює на порті 2055 та проводить прослуховування. Далі, щоб обробити логи, їх потрібно перенаправити в Graylog, де їх отримає Elasticsearch, а опісля з’явиться можливість їх аналізувати. Крок підключення колектора до Graylog буде описано та показано в другій частині бакалаврської роботи «Обробка подій на основі Graylog».

3.3 Реалізація фільтрів та агрегацій

Перевіримо наявність логів в Elasticsearch використовуючи API (та виведемо всю інформацію в кращому для читання вигляді за допомогою атрибуту «..?pretty»):

```
curl -XGET "localhost:9200/_search?pretty"
```

Результат виконання команди зображено на рисунку 3.7.

Рисунок 3.7 – Результат перевірки наявності логів

Наступним кроком, перевіримо наявні індекси у системі Graylog. Індекси в Graylog зазвичай мають префікс "graylog_".

```
curl -XGET http://localhost:9200/_cat/indices
```

В ході виконання команди було отримано список наявних індексів з їх станом, доступністю, розмірами та іншими параметрами (рис.3.8)

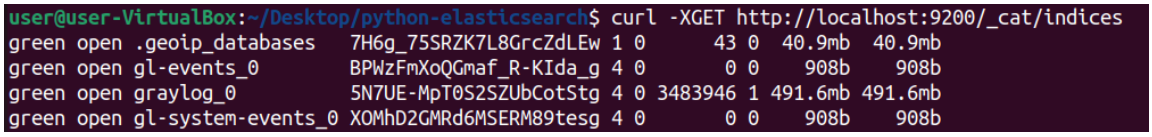


Рисунок 3.8 – Результат перевірки наявних індексів

За результатом перевірки було отримано інформацію, що в системі Graylog присутні такі індекси:

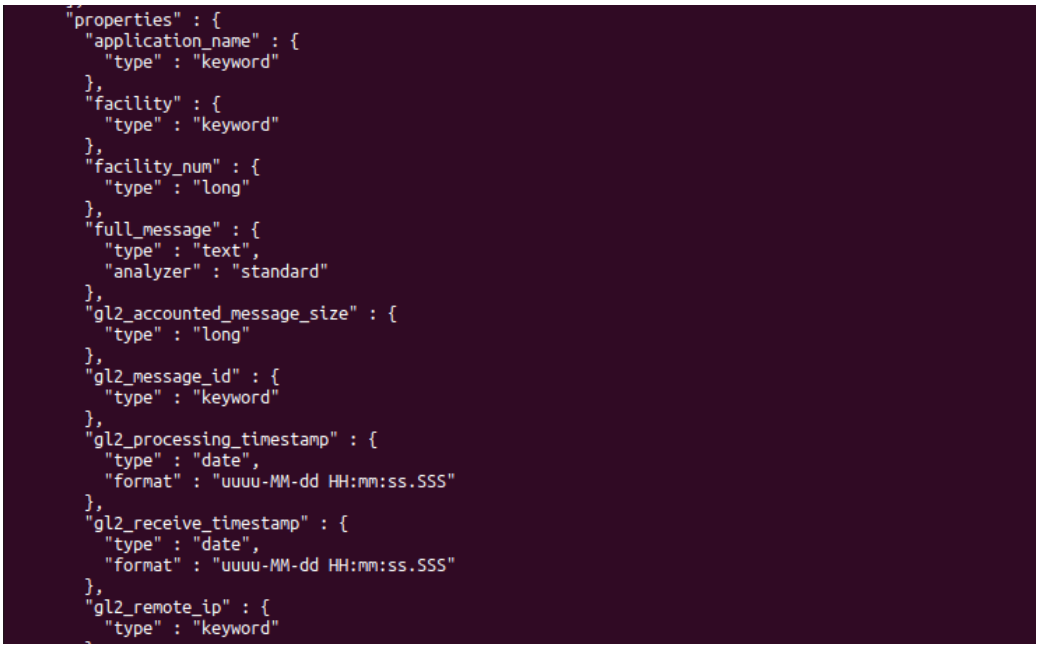
- .geoip_databases – індекс для географічних баз даних.
- gl-events_0 – індекс для подій, пов'язаних з Graylog.
- graylog_0 – основний індекс, де зберігаються логи та повідомлення.
- gl-system-events_0 – індекс для системних подій у Graylog.

Зелений статус (green) означає, що індекси готові до використання і здорові. Далі, будемо використовувати індекс "graylog_0" для пошуку потрібних

логів. Перш за все, потрібно отримати інформацію щодо структури цього індексу, щоб знати, з чим далі працювати. Для цього виконаємо наступну команду:

```
curl -XGET http://localhost:9200/graylog_0/_mapping
```

В результаті виконання цієї команди буде виведена структура індексу "graylog_0" в Elasticsearch. Також буде наведена інформація як організовані дані в індексі і який тип даних має кожне поле. Ця інформація дозволяє зрозуміти, як дані увійшли в Elasticsearch і як їх можна запитувати та аналізувати. Через те, що структура занадто велика, на рисунку 3.9 було наведено лише частину структури.



```
"properties" : {
  "application_name" : {
    "type" : "keyword"
  },
  "facility" : {
    "type" : "keyword"
  },
  "facility_num" : {
    "type" : "long"
  },
  "full_message" : {
    "type" : "text",
    "analyzer" : "standard"
  },
  "gl2_accounted_message_size" : {
    "type" : "long"
  },
  "gl2_message_id" : {
    "type" : "keyword"
  },
  "gl2_processing_timestamp" : {
    "type" : "date",
    "format" : "uuuu-MM-dd HH:mm:ss.SSS"
  },
  "gl2_receive_timestamp" : {
    "type" : "date",
    "format" : "uuuu-MM-dd HH:mm:ss.SSS"
  },
  "gl2_remote_ip" : {
    "type" : "keyword"
```


Рисунок 3.9 – Частина структури індексу "graylog_0"

Так як було виведено лише частину структури, її елементи та поля буде описано далі.

Поля загальної інформації:

- application_name (тип: keyword): Назва додатку або компоненту, якому належить лог.
- facility (тип: keyword): Категорія логу.
- facility_num (тип: long): Числове значення, що відповідає категорії логу.

Поля повідомлення:

- full_message (тип: text): Повне текстове повідомлення логу.

Поля, пов'язані з Graylog:

- gl2_accounted_message_size (тип: long): Розмір обробленого повідомлення в байтах.
- gl2_message_id (тип: keyword): Унікальний ідентифікатор повідомлення.
- gl2_processing_timestamp (тип: date): Час обробки повідомлення в форматі "yyyy-MM-dd HH:mm:ss.SSS".
- gl2_receive_timestamp (тип: date): Час отримання повідомлення в форматі "yyyy-MM-dd HH:mm:ss.SSS".
- gl2_remote_ip (тип: keyword): Віддалена IP-адреса, з якої було отримано повідомлення.
- gl2_remote_port (тип: long): Віддалений порт, з якого було отримано повідомлення.
- gl2_source_input (тип: keyword): Унікальний ідентифікатор вхідного потоку, з якого було отримано повідомлення.
- gl2_source_node (тип: keyword): Унікальний ідентифікатор вузла Graylog, на якому було отримано повідомлення.
- level (тип: long): Рівень логу.

Поля, пов'язані з логами NetFlow:

- nf_bytes (тип: long): Кількість байтів в NetFlow пакеті.
- nf_dst (тип: keyword): Адреса призначення у форматі IP або доменного імені.
- nf_dst_address (тип: keyword): Адреса призначення у форматі IP.
- nf_dst_as (тип: long): ASN (Autonomous System Number) призначення.
- nf_dst_mask (тип: long): Маска підмережі призначення.
- nf_dst_port (тип: long): Порт призначення.
- nf_flow_packet_id (тип: long): Унікальний ідентифікатор NetFlow пакета.
- nf_pkts (тип: long): Кількість пакетів в NetFlow потоці.

- nf_proto (тип: long): Числовий ідентифікатор протоколу в NetFlow.
- nf_proto_name (тип: keyword): Назва протоколу в NetFlow.
- nf_snmp_input (тип: long): Вхідний порт SNMP (Simple Network Management Protocol) в NetFlow.
- nf_snmp_output (тип: long): Вихідний порт SNMP в NetFlow.
- nf_src (тип: keyword): Адреса джерела у форматі IP або доменного імені.
- nf_src_address (тип: keyword): Адреса джерела у форматі IP.
- nf_src_as (тип: long): ASN джерела.
- nf_src_mask (тип: long): Маска підмережі джерела.
- nf_src_port (тип: long): Порт джерела.
- nf_start (тип: date): Час початку NetFlow потоку.
- nf_stop (тип: date): Час завершення NetFlow потоку.
- nf_tcp_flags (тип: long): Прапори TCP (Transmission Control Protocol) в NetFlow.
- nf_tos (тип: long): Значення TOS (Type of Service) в NetFlow.
- nf_version (тип: long): Версія NetFlow.

Інші поля:

- process_id (тип: keyword): Унікальний ідентифікатор процесу.
- source (тип: text): Джерело логу.
- streams (тип: keyword): Список потоків, до яких належить лог.
- timestamp (тип: date): Час запису логу.

Виходячи з результатів, надалі будемо працювати з полями, які пов'язані з логами NetFlow. Ці поля містять інформацію про різні аспекти NetFlow, такі як адреси джерела та призначення, порти, кількість пакетів, протоколи, часові мітки та інші характеристики NetFlow.

Знайдемо записи NetFlow за допомогою запиту який містить фільтр, що виводить всі логи з індексу:

```
curl -XGET "http://localhost:9200/graylog_0/_search?pretty" -H "Content-Type: application/json" -d '{ "query": { "match_all": {} } }'
```

Результатів вийшло багато, тому далі для прикладу буде наведено лише один лог-результат (рис. 3.9).


```
      {
        "_index" : "graylog_0",
        "_type" : "_doc",
        "_id" : "aa4a83b4-f193-11ed-b498-000c2990507e",
        "_score" : 1.0,
        "_source" : {
          "gl2_accounted_message_size" : 347,
          "level" : 4,
          "gl2_remote_ip" : "192.168.137.128",
          "gl2_remote_port" : 52118,
          "streams" : [
            "000000000000000000000001"
          ],
          "gl2_message_id" : "01H0ANE0J96FC2NP4Y1EJ28GEZ",
          "source" : "user-VirtualBox",
          "message" : "[ 5565.195394] IN=lo OUT= MAC=00:00:00:00:00:00:00:00
:00:00:00:00:08:00 SRC=192.168.137.128 DST=192.168.137.128 LEN=290 TOS=0x00
PREC=0x00 TTL=64 ID=8026 DF PROTO=UDP SPT=53829 DPT=5149 LEN=270",
          "gl2_source_input" : "645f885b76e67038084bb3b1",
          "application_name" : "kernel",
          "facility_num" : 0,
          "gl2_source_node" : "72db8cfe-a25e-41d8-ae67-f703c2ccfe64",
          "facility" : "kernel",
          "timestamp" : "2023-05-13 13:36:46.011"
        }
      }
    ]
  }
}
```


Рисунок 3.9 – Приклад логу, що було отримано

Використаємо агрегації та фільтри, щоб перевірити кількість логів по полю "gl2_remote_ip" за допомогою наступної команди:

```
curl -XGET "http://localhost:9200/_search?pretty" -H 'Content-Type: application/json' -d '{ "size": 0, "aggs": { "ip_count": { "terms": { "field": "gl2_remote_ip", "size": 10 } } } }'
```

Результат виконання команди наведено на рисунку 3.10.


```
"aggregations" : {
  "ip_count" : {
    "doc_count_error_upper_bound" : 0,
    "sum_other_doc_count" : 0,
    "buckets" : [
      {
        "key" : "192.168.137.128",
        "doc_count" : 3460208
      },
      {
        "key" : "127.0.0.1",
        "doc_count" : 24448
      }
    ]
  }
```


Рисунок 3.10 – Результат виконання команди

З результатів можна сказати, що кількість мережевого трафіку, яка була збережена у логах, є доволі великою і без додаткових фільтрів та агрегацій не обійтись. Для перевірки останніх логів за часом надходження – використаємо сортування у зворотному порядку за полем "timestamp" (варто взяти до уваги, це час коли лог записано до Elasticsearch, а актуальний час запису NetFlow-потоку варто шукати за допомогою поля "nf_start"). Ось приклад запиту для отримання останніх 10 логів з індексу "graylog_0":

```
curl -XGET "http://localhost:9200/_search?pretty" -H 'Content-Type: application/json' -d' { "query": { "match_all": {} }, "sort": [ { "timestamp": { "order": "desc" } } ], "size": 10 }'
```

Так як логів багато, було проілюстровано лише один результат отриманих логів, які зображено на рисунку 3.11.

```
{
  "_index" : "graylog_0",
  "_type" : "_doc",
  "_id" : "7a745440-077b-11ee-b064-000c2990507e",
  "_score" : null,
  "_source" : {
    "nf_dst" : "239.255.255.250:1900",
    "nf_stop" : "2023-06-10T10:41:11.891Z",
    "nf_version" : 5,
    "gl2_remote_ip" : "127.0.0.1",
    "gl2_remote_port" : 49033,
    "nf_pkts" : 4,
    "source" : "127.0.0.1",
    "gl2_source_input" : "646bae9bcf82341db70912e1",
    "nf_tcp_flags" : 0,
    "nf_dst_port" : 1900,
    "nf_bytes" : 792,
    "nf_src_as" : 0,
    "nf_proto" : 17,
    "nf_dst_address" : "239.255.255.250",
    "nf_src_port" : 61292,
    "gl2_source_node" : "72db8cfe-a25e-41d8-ae67-f703c2ccfe64",
    "timestamp" : "2023-06-10 10:42:20.000",
    "nf_src" : "192.168.137.1:61292",
    "nf_dst_mask" : 0,
    "gl2_accounted_message_size" : 543,
    "nf_flow_packet_id" : 22859,
    "streams" : [
      "000000000000000000000001"
    ],
    "gl2_message_id" : "01H2JECQW5M7BEG97401PCDPJG",
    "message" : "NetFlowV5 [192.168.137.1]:61292 <> [239.255.255.250]:1900 proto:17 pkts:4 bytes:792",
    "nf_dst_as" : 0,
    "nf_tos" : 0,
    "nf_src_address" : "192.168.137.1",
    "nf_proto_name" : "UDP",
    "nf_src_mask" : 0,
    "nf_snmp_input" : 0,
    "nf_snmp_output" : 0,
    "nf_start" : "2023-06-10T10:41:08.862Z"
  },
  "sort" : [
    1686393740000
  ]
 }
]
```


Рисунок 3.11 – Частковий результат отримання логів

Повністю інформацію навести одним знімком екрану буде проблематично, так як логів багато, та інформація, яку вони містять, дуже велика, тому було наведено лише останній лог, який записано в індекс.

Надалі будемо працювати лише з логами NetFlow, а так як зазвичай в повідомленнях ці логи мають префікс "NetFlowV5…", відфільтруємо та знайдемо відповідні логи. Скористаємося сортуванням з попереднього запиту та сформуємо новий:

```
curl -XGET "http://localhost:9200/graylog_0/_search?pretty" -H "Content-Type: application/json" -d '{ "query": { "match": { "message": "NetFlowV5" } }, "sort": [ { "timestamp": { "order": "desc" } } ], "size": 10  }'
```

Наприклад, на сервер була здійснена атака в певний період часу, пам’ятаємо, що Elasticsearch працює з всесвітнім координованим часом UTC (Київський час UTC+3) [35], зробимо запит за полем "nf_start" та перевіримо наявні логи.

```
curl -XGET "http://localhost:9200/graylog_0/_search?pretty" -H "Content-Type: application/json" -d '{ "query": { "bool": { "must": [ { "match": { "message": "NetFlowV5" } }, { "range": { "nf_start": { "gte": "2023-06-13T15:10:00.000Z", "lte": "2023-06-13T15:15:00.000Z" } } } ] } } }'
```

На рисунку 3.12 було зображено отримані результати за агрегацією по часу.

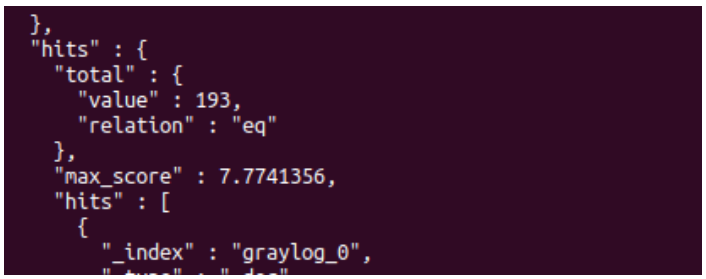
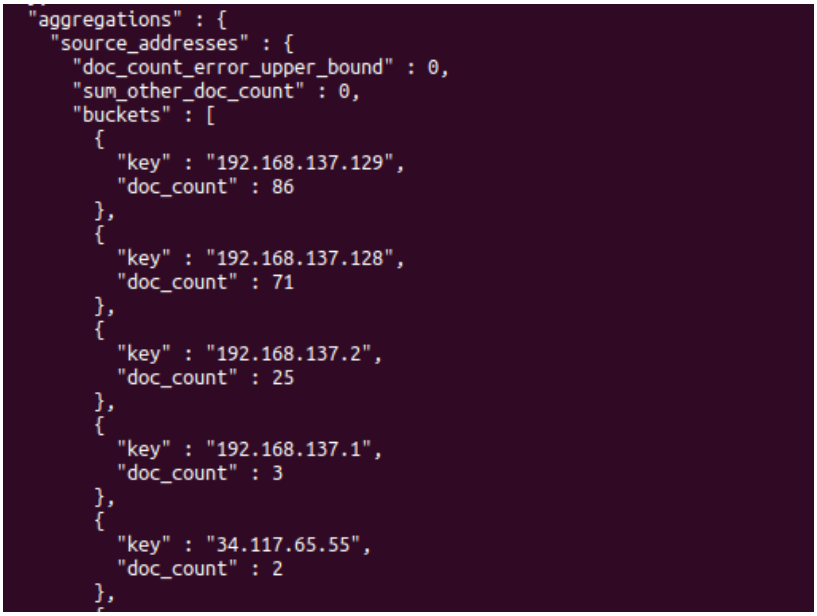


Рисунок 3.12 – Кількість логів, які увійшли в цей проміжок часу

Тепер, обмеживши часові рамки, та отримавши меншу кількість логів, можна проводити аналіз логів та продовжувати працювати з ними. Наступним кроком, потрібно визначити джерело атаки, тому перевіримо наявні адреси та кількість запитів від них. Використаємо агрегацію підрахунку кількості запитів за адресою. Оновимо запит, включаючи агрегацію за полем "nf_src_address" для підрахунку кількості запитів від кожної адреси:

```
curl -XGET "http://localhost:9200/graylog_0/_search?pretty" -H "Content-Type: application/json" -d '{ "query": { "bool": { "must": [ { "match": { "message": "NetFlowV5" } }, { "range": { "nf_start": { "gte": "2023-06-13T15:10:00.000Z", "lte": "2023-06-13T15:15:00.000Z" } } } ] } }, "aggs": { "source_addresses": { "terms": { "field": "nf_src_address", "size": 10 } } } }'
```

На рисунку 3.13 було зображено отримані результати за агрегацією конкретній адресі та обмеженням часу.



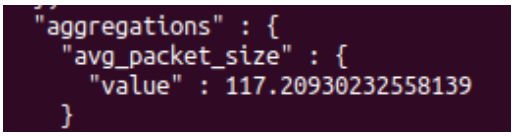
```
"aggregations" : {
  "source_addresses" : {
    "doc_count_error_upper_bound" : 0,
    "sum_other_doc_count" : 0,
    "buckets" : [
      {
        "key" : "192.168.137.129",
        "doc_count" : 86
      },
      {
        "key" : "192.168.137.128",
        "doc_count" : 71
      },
      {
        "key" : "192.168.137.2",
        "doc_count" : 25
      },
      {
        "key" : "192.168.137.1",
        "doc_count" : 3
      },
      {
        "key" : "34.117.65.55",
        "doc_count" : 2
      },
```


Рисунок 3.13 – Отримані результати агрегації

За результатами можемо сказати, що найбільше запитів було зроблено з адреси 192.168.137.129, звідки й здійснювалася атака (192.168.137.1/24 – локальна мережа, 192.168.137.128 машина з Elasticsearch). Далі можемо проаналізувати, який в середній розмір пакету надсилала ця машина. Наступний запит допоможе знайти середній розмір пакету за наявною адресою машини.

```
curl -XGET "http://localhost:9200/graylog_0/_search?pretty" -H "Content-Type: application/json" -d '{ "query": { "bool": { "must": [ { "match": { "message": "NetFlowV5" } }, { "term": { "nf_src_address":"192.168.137.129" } }, { "range": { "nf_start": { "gte": "2023-06-13T15:10:00.000Z", "lte": "2023-06-13T15:15:00.000Z" } } } ] } }, "aggs": { "avg_packet_size": { "avg": { "field": "nf_bytes" } } } }'
```

Рисунок 3.14 показує, який середній розмір пакету, що відправлено під час атаки.



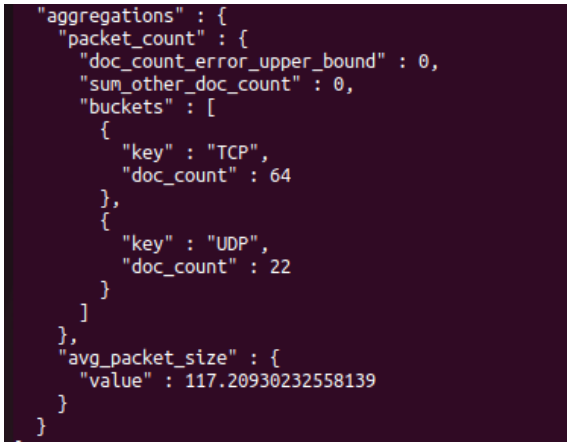
```
  "aggregations" : {
    "avg_packet_size" : {
      "value" : 117.20930232558139
    }
```


Рисунок 3.14 – Отримані результати агрегації

Такий запит можна розширити і дізнатись кількість запитів по різним протоколам. Додамо ще одну агрегацію до попереднього запиту.

```
curl -XGET "http://localhost:9200/graylog_0/_search?pretty" -H "Content-Type: application/json" -d ' { "query": { "bool": { "must": [ { "match": { "message": "NetFlowV5" } }, { "term": { "nf_src_address": "192.168.137.129" } }, { "range": { "nf_start": { "gte":"2023-06-13T15:10:00.000Z", "lte":"2023-06-13T15:15:00.000Z" } } } ] } }, "aggs": { "packet_count": { "terms" : { "field" : "nf_proto_name ", "size":10 } }, "avg_packet_size": { "avg": { "field": "nf_bytes" } } } } '
```

На рисунку 3.15 було зображено отримані результати комбінованої агрегації.



```
  "aggregations" : {
    "packet_count" : {
      "doc_count_error_upper_bound" : 0,
      "sum_other_doc_count" : 0,
      "buckets" : [
        {
          "key" : "TCP",
          "doc_count" : 64
        },
        {
          "key" : "UDP",
          "doc_count" : 22
        }
      ]
    },
    "avg_packet_size" : {
      "value" : 117.20930232558139
    }
  }
```


Рисунок 3.15 – Отримані результати комбінованої агрегації

У запиті, що наведено вище, використано Elasticsearch для пошуку логів з повідомленням "NetFlowV5" в певний часовий діапазон та з певною IP-адресою в полі "nf_src_address". В результаті ми отримали загальну кількість пакетів за значеннями поля "nf_proto_name" та середній розмір пакету.

Отже, за допомогою запитів, комбінацій фільтрів та агрегацій можна проводити ручний аналіз логів, перевіряти кількість та наявність тих чи інших полів, обчислювати значення та робити вибірку по відповідним критеріям. Попередній приклад показує, як можна використовувати Elasticsearch для аналізу логів та витягування корисної інформації. Знання про типи атак, їхні сліди та інші аспекти безпеки можуть допомогти розширити цей аналіз і адаптувати його до конкретних потреб та інших видів атак.

3.4 Висновок до розділу

В ході опрацювання цього розділу розглянуто розгортання та налаштування системи Elasticsearch на основі операційної системи Ubuntu. Проведено перевірку на працездатність системи після налаштувань та після отримання позитивного результату, йшов наступний крок, налаштування та збір логів. В цьому розділі описано що використовувалось для моніторингу та збору повідомлень, в результаті обрано утиліту fprobe, після чого проведено встановлення та налаштування на моніторинг мережевого трафіку, що в подальшому перенаправлятиметься в Elasticsearch для обробки та аналізу.

Наступним важливим етапом стало тестування системи пошуку та обробки логів на основі спроектованої атаки на систему. За допомогою запитів, заснованих на JSON структурах, використаних фільтрів та агрегацій, отримано необхідну інформацію про пакети та в ході опрацювання яких визначено потенційного атакуючого. Також наведено результати проведеної роботи у вигляді наявних запитів та знімків екрану з отриманою інформацією.

В результаті, задача аналізу логів за допомогою Elasticsearch показана на практиці, реалізована на можливий максимум і в подальшому може бути розширена при виконанні магістерської роботи.

ВИСНОВКИ

При виконанні комплексної дипломної роботи було опрацьовано інформацію про різні способи впровадження кіберполігонів та розглянуто основні засоби, які стануть у нагоді при їх розгортанні. Було проведено аналіз відомих пошукових систем та обрано одну, на якій далі буде заснована частина збору та обробки лог-даних.

Наступним кроком було наведено структуру даних та архітектуру обраного засобу. Після чого було описано наявні фільтри і агрегації, наведено структурну схему їх використання, які в подальшому будуть використовуватись для аналізу та обробки інформації з логів.

Також було встановлено та налаштовано систему для зберігання та обробки логів Elasticsearch. Після того, як було сконфігуновано та отримано працездатну частину Elasticsearch, було впроваджено генерування та збір NetFlow-трафіку. Цей трафік в подальшому буде використаний для вилучення необхідної інформації за допомогою наявних фільтрів та агрегацій, використовуючи мову API-запитів засновану на JSON-структурах.

Завершальним етапом стало тестування на прикладі спроектованої DDoS-атаки на систему, в ході якого було проведено обробку даних. За допомогою знімків екрану було показано, яку інформацію було отримано з запитів.

Отже, в результаті виконаної роботи було виконано поставлені завдання в повному обсязі та отримано робочу частину кіберполігону у вигляді системи збору та обробки лог-даних, яка доповнить наявний функціонал та завершить повну систему.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Якімов О. П. Elasticsearch як оптимальне рішення пошуку та аналізу подій кібербезпеки у режимі реального часу [Електронний ресурс] / О. П. Якімов, О. П. Войтович // Матеріали LII Науково-технічної конференції факультету інформаційних технологій та комп'ютерної інженерії, Вінниця, 22-23 червня 2023 р. – Електрон. текст. дані. – 2023. – Режим доступу: https://conferences.vntu.edu.ua/index.php/all-fitki/all-fitki-2023/paper/view/17609 (дата звернення: 26.05.2023).

2. Ukraine conflict: One year of cyberattacks and operations. *CyberPeace Institute*. URL: https://cyberpeaceinstitute.org/news/ukraine-conflict-one-year-anniversary/ (Accessed: 27.05.2023).

3. Досвід створення навчально-тренувального кіберполігону [Електронний ресурс] // Матеріали збірника тез доповідей Міжнародної науково-практичної конференції «Застосування інформаційних технологій у підготовці та діяльності сил охорони правопорядку», Харків, 15-16 березня 2017 р. – Електрон. текст. дані. – 2023. – Режим доступу: http://kinf.nangu.edu.ua/since_files/Doc/tezNPK_2017_1.pdf (дата звернення: 28.05.2023).

4. Д. Безуглий, "Інформаційна безпека України: огляд останніх тенденцій", Фізико-математична освіта, вип. 2(16), с. 13–17, 2018 (дата звернення: 28.05.2023).

5. Даник Ю. Г. Особливості створення кіберполігонів для дослідження комплексних кібердій та підготовки фахівців з кібербезпеки. *Протиборство у кібернетичному просторі*, м. Київ. Київ, 2019. С. 96. URL: https://doi.org/10.33099/2311-7249/2019-34-1-95-102 (дата звернення: 28.05.2023).

6. У ЗСУ функціонуватиме надсучасний кіберполігон VITIsecurity: на кафедрі кібербезпеки розгорнуто навчально-тренувальний комплекс. *Військовий інститут телекомунікацій та інформатизації*. URL: https://www.viti.edu.ua/news/3247 (дата звернення: 28.05.2023).

7. L. McDaniel, E. Talvi and B. Hay, "Capture the Flag as Cyber Security Introduction," 2016 49th Hawaii International Conference on System Sciences (HICSS), Koloa, HI, USA, 2016, pp. 5479-5486. URL: https://doi.org/10.1109/HICSS.2016.677 (Accessed: 29.05.2023).

8. Свої двері відкрив «Навчальний кіберполігон». *Національний університет біоресурсів і природокористування України*. URL: https://nubip.edu.ua/node/96492 (дата звернення: 29.05.2023).

9. Hacking Training For The Best. *Hack The Box*. URL: https://www.hackthebox.com (Accessed: 30.05.2023).

10. TryHackMe | Cyber Security Training. *TryHackMe*. URL: https://tryhackme.com (Accessed: 30.05.2023).

11. Яшанов С. М. Віртуальні машини в системі інформаційно-навчального середовища вищого закладу освіти. URL: https://enpuir.npu.edu.ua/handle/123456789/5888 (дата звернення: 01.06.2023).

12. Guthrie, Forbes, Scott Lowe, and Kendrick Coleman. *VMware vSphere design*. John Wiley & Sons, 2013 (Accessed: 01.06.2023).

13. Velte, Anthony, and Toby Velte. *Microsoft virtualization with Hyper-V*. McGraw-Hill, Inc., 2009 (Accessed: 01.06.2023).

14. Virtualbox, Oracle VM. "Oracle vm virtualbox." *Change* 107 (2011): 1-287 (Accessed: 01.06.2023).

15. Emma D., Pescape A., Ventre G. Analysis and experimentation of an open distributed platform for synthetic. Proceedings. 10th IEEE International Workshop on Future Trends of Distributed Computing Systems, 2004. FTDCS 2004., Suzhou, China. URL: https://doi.org/10.1109/ftdcs.2004.1316627 (Accessed: 02.06.2023).

16. Ixia BreakingPoint – 8.20. Ixia. URL: https://support.ixiacom.com/version/ixia-breakingpoint-820 (Accessed: 02.06.2023).

17. Spirent CyberFlood – Spirent. Automated Testing and Assurance Solutions – Spirent. URL: https://www.spirent.com/assets/u/spirent_cyberflood (Accessed: 02.06.2023).

18. Tcpreplay – Pcap editing and replaying utilities. Tcpreplay – Pcap editing and replaying utilities. URL: https://tcpreplay.appneta.com (Accessed: 02.06.2023).

19. Enterprise Search documentation [8.8] | Elastic. Elasticsearch Platform – Find real-time answers at scale | Elastic. URL: https://www.elastic.co/guide/en/enterprise-search/current/index.html (Accessed: 03.06.2023).

20. Welcome to Apache Solr. Welcome to Apache Solr – Apache Solr. URL: https://solr.apache.org (Accessed: 03.06.2023).

21. Azure Cognitive Search – Cloud Search Service | Microsoft Azure. Cloud Computing Services | Microsoft Azure. URL: https://azure.microsoft.com/en-us/products/search (Accessed: 03.06.2023).

22. [Elasticsearch Data Structure]. Ssup2 | Ssup2 Blog. URL: https://ssup2.github.io/images/theory_analysis/Elasticsearch_Data_Structure/Elasticsearch_Data_Structure.PNG (Accessed: 03.06.2023).

23. [Elasticsearch Architecture]. DevopsIdeas – Devops tutorials and cloud computing. URL: https://devopsideas.com/wp-content/uploads/2018/11/Elasticsearch_component_relation-1024x834.png (Accessed: 04.06.2023).

24. Cyber Range – Types and Use Cases. URL: https://www.purplesynapz.com/cyber-range-types-and-use-cases (Accessed: 04.06.2023).

25. Ooteghem K. V. Threats to network security and network attacks. Parallels Remote Application Server Blog – Application virtualization, mobility and VDI. URL: https://www.parallels.com/blogs/ras/network-attacks/ (Accessed: 04.06.2023).

26. Sending in log data. URL: https://go2docs.graylog.org/5-0/getting_in_log_data/getting_in_log_data.html (Accessed: 05.06.2023).

27. Гончарук В. В. Способи аналізу мережевого трафіку комп’ютерної мережі [Електронний ресурс] / В. В. Гончарук, М. А. Пєтух // Матеріали Міжнародної науково практичної Інтернетконференції, Вінниця, 24-25 жовтня

2016 р. – Електрон. текст. дані. – 2016. – Режим доступу: https://ir.lib.vntu.edu.ua/bitstream/handle/123456789/31451/391-394.pdf?sequence=1&isAllowed=y (дата звернення: 05.06.2023).

28. What is SNMP? {Versions, Monitoring, and Components}. Knowledge Base by phoenixNAP. URL: https://phoenixnap.com/kb/what-is-snmp (Accessed: 06.06.2023).

29. What is RMON?. DPS Telecom. URL: https://www.dpstele.com/blog/what-is-rmon.php (Accessed: 06.06.2023).

30. Cisco IOS Netflow Data Sheet. Cisco. URL: https://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/ios-netflow/product_data_sheet0900aecd80173f71.html (Accessed: 06.06.2023).

31. NetFlow Analyzers and NetFlow Tools. Kentipedia. URL: https://www.kentik.com/kentipedia/netflow-analyzers-and-netflow-tools/ (Accessed: 06.06.2023).

32. Filter search results | Elasticsearch Guide [8.8] | Elastic. Elasticsearch Platform – Find real-time answers at scale | Elastic. URL: https://www.elastic.co/guide/en/elasticsearch/reference/current/filter-search-results.html (Accessed: 07.06.2023).

33. Aggregations | Elasticsearch Guide [8.8] | Elastic. Elasticsearch Platform – Find real-time answers at scale | Elastic. URL: https://www.elastic.co/guide/en/elasticsearch/reference/current/search-aggregations.html (Accessed: 07.06.2023).

34. fprobe. SourceForge. URL: https://sourceforge.net/projects/fprobe/ (Accessed: 07.06.2023).

35. Date histogram aggregation | Elasticsearch Guide [8.8] | Elastic. Elasticsearch Platform – Find real-time answers at scale | Elastic. URL: https://www.elastic.co/guide/en/elasticsearch/reference/current/search-aggregations-bucket-datehistogram-aggregation.html#_time_zone_2 (Accessed: 07.06.2023).

ДОДАТКИ

Додаток А

ПРОТОКОЛ ПЕРЕВІРКИ БАКАЛАВРСЬКОЇ ДИПЛОМНОЇ РОБОТИ НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: Кіберполігон для дослідження подій інформаційної безпеки. Частина 3. Обробка на основі Elasticsearch

Автор роботи: Якімов Олександр Павлович

Тип роботи: бакалаврська дипломна робота

Підрозділ кафедра захисту інформації ФІТКІ
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність – 96.4% Схожість – 3.6%.

Аналіз звіту подібності (відмітити потрібне):

√ 1. Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

☐ 2. Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

☐ 3. Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку _____ Каплун В. А.
(підпис) (прізвище, ініціали)

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи _____ Якімов О. П.
(підпис) (прізвище, ініціали)

Керівник роботи _____ Войтович В. П.
(підпис) (прізвище, ініціали)

Додаток Б

Методичні вказівки

Мета роботи: Ознайомитися з процесом встановлення та налаштування Elasticsearch для зберігання та обробки лог-даних.

Порядок виконання:

Крок 1: Оновлення системи

Першим кроком перед встановленням Elasticsearch потрібно провести оновлення операційної системи. Для того, щоб виконати цей крок необхідно виконати наступні команди:

```
sudo apt update && sudo apt -y full-upgrade
[ -f /var/run/reboot-required ] && sudo reboot -f
```

Ці команди оновлять пакети в системі та, якщо необхідно, перезавантажать систему для застосування оновлень.

Крок 2: Встановлення необхідних застосунків

Для успішної установки та налаштування Elasticsearch потрібно встановити деякі необхідні залежності. Виконайте наступні команди для встановлення цих залежностей:

```
sudo apt update
sudo apt install vim apt-transport-https openjdk-11-jre-headless uuid-runtime pwgen curl dirmngr
```

Виконавши цей ряд команд, будуть встановлені необхідні пакети, включаючи Java Runtime Environment (JRE) і деякі інші утиліти, необхідні для роботи Elasticsearch.

Крок 3: Встановлення та налаштування Elasticsearch

Після встановлення необхідних залежностей можна переходити до встановлення самої Elasticsearch. Виконайте наступні команди для встановлення та налаштування Elasticsearch:

```
sudo apt install curl
curl -fsSL https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo gpg --dearmor -o /usr/share/keyrings/elastic.gpg
echo "deb [signed-by=/usr/share/keyrings/elastic.gpg] https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-7.x.list
sudo apt update
sudo apt install elasticsearch
sudo nano /etc/elasticsearch/elasticsearch.yml
```

У першій команді встановлюється утиліта curl, яка використовується для завантаження необхідних ключів та пакетів. Наступні команди завантажують ключі та встановлюють пакет Elasticsearch з офіційного репозиторію Elastic. Після цього відкриється конфігураційний файл Elasticsearch для налаштування.

У відкритому файлі /etc/elasticsearch/elasticsearch.yml знайдіть наступні рядки та внесіть зміни:

```
cluster.name = graylog
network.host = localhost
action.auto_create_index = false
```

Ці налаштування встановлюють назву кластера Elasticsearch, обмежують доступ до Elasticsearch лише з локальної машини та вимикають автоматичне створення індексів. Після збереження змін в файлі, закрийте його. В результаті налаштування у файлі мають виглядати як на рисунку 1.

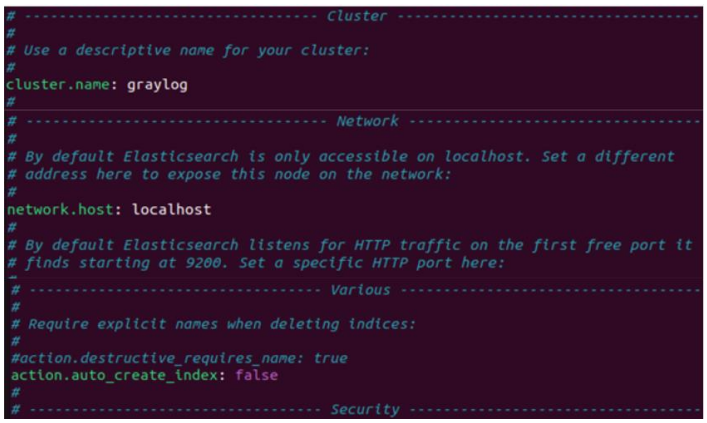


Рисунок 1 – Налаштування в конфігураційному файлі Elasticsearch

Крок 4: Запуск Elasticsearch

Для запуску Elasticsearch і налаштування його на автоматичний запуск при перезавантаженні системи. необхідно виконати наступну команду:

```
sudo systemctl start elasticsearch
sudo systemctl enable elasticsearch
```

Крок 5: Перевірка стану Elasticsearch

Щоб переконатися, що Elasticsearch успішно встановлено і працює, потрібно виконати наступну команду:

```
curl -X GET 'http://localhost:9200'
```

Використовуючи дану команду, буде отримано інформацію про Elasticsearch, включаючи його версію та стан, результат виконання представлено на рисунку 2.

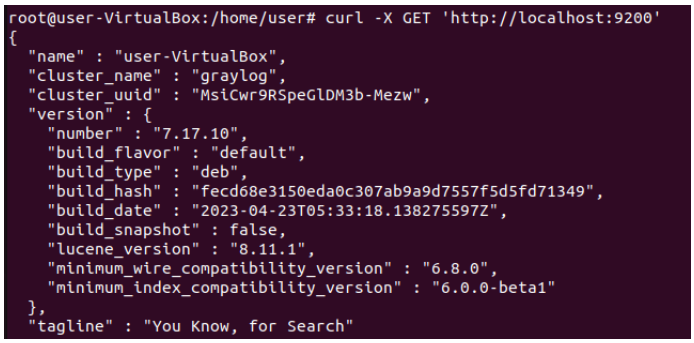


Рисунок 2 – Перевірка працездатності Elasticsearch

Отримавши результати, можна зробити висновок, що Elasticsearch встановлено правильно та система готова до використання.

Крок 6: Встановлення та налаштування fprobe

Робота Fprobe полягає в тому, що він прослуховує мережевий трафік і аналізує його, створюючи при цьому NetFlow-потоки, які містять інформацію про джерело, призначення, протоколи, порти та інші характеристики пакетів. Далі такі потоки можуть бути направлені на сервер, який підтримує аналіз NetFlow для подальшої обробки.

Встановимо та налаштуємо необхідні компоненти:

```
sudo apt-get update
sudo apt-get install fprobe
```

Наступне, що потрібно зробити – це встановити відповідні налаштування при встановленні (мережевий інтерфейс та порт 2055), в подальшому ці налаштування можна змінити в конфігураційному файлі (/etc/default/fprobe). Для цього використаємо наступні команди:

```
ifconfig
nano /etc/default/fprobe
```

В результаті виконання команд буде перевірено наявні мережеві інтерфейси, що зображено на рисунку 3, та налаштувань в конфігураційному файлі fprobe, які зображено на рисунку 4.

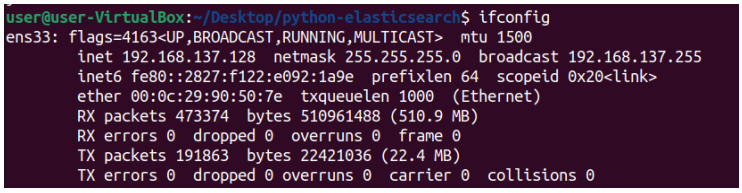


Рисунок 3 – Результати перевірки наявних мережевих інтерфейсів

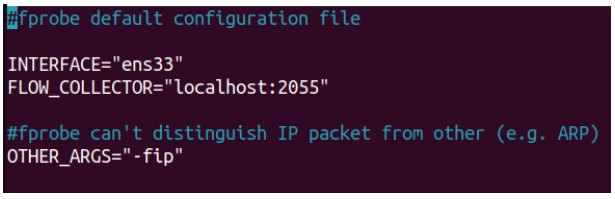


Рисунок 4 – Перевірка налаштувань в конфігураційному файлі fprobe

Перезавантажимо утиліту та перевіримо її статус використовуючи наступні команди:

```
sudo service fprobe restart
sudo service fprobe status
```

В ході використання команд буде проведена перевірки налаштувань та працездатності fprobe (рис. 5).

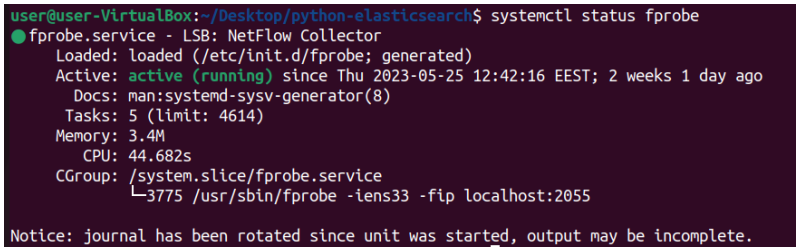


Рисунок 5 – Результати перевірки налаштувань та працездатності fprobe

В результаті пророблених дій буде отримано працездатну версію Elasticsearch та fprobe, що завершує процедуру встановлення та налаштування.

ІЛЮСТРАТИВНА ЧАСТИНА

Кіберполігон для дослідження подій інформаційної безпеки. Частина 3. Обробка на основі Elasticsearch

(Назва бакалаврської кваліфікаційної роботи)

Виконав: студент 4 курсу групи 1БС-19б
спеціальності 125 Кібербезпека
_____ Олександр ЯКІМОВ
_____ 2023 р.

Керівник: к. т. н., доцент каф. ЗІ
_____Олеся ВОЙТОВИЧ
_____ 2023 р.

РЕЗУЛЬТАТИ ПОРІВНЯННЯ ЛОКАЛЬНИХ ТА ВІДДАЛЕНИХ КІБЕРПОЛІГОНІВ

Аспекти порівняння	Локальний кіберполігон	Віддалений кіберполігон
Доступність	Обмежена (локальна)	Зручний доступ онлайн
Характеристики обладнання	Фізичне обладнання, розміщене на місці	Віртуальна інфраструктура, розміщена в хмарі
Практичні навички	Запрограмовані конкретні сценарії та налаштування; можливість створення сценаріїв (потребує глибших знань)	Гнучкість в створенні різних сценаріїв і налаштувань
Вартість	Середня	Знижена вартість у порівнянні з фізичним обладнанням
Масштабованість	Обмежена	Легко масштабується для великої кількості користувачів
Фізична безпека	Висока	Залежить від рівня безпеки віддаленого середовища
Керування	Локальний контроль над інфраструктурою	Залежить від провайдера хмарної інфраструктури
Місцезнаходження даних	Локальне зберігання даних	Віддалене зберігання даних в хмарі
Пропускна здатність мережі	Залежить від локального мережевого обладнання та інфраструктури	Залежить від якості Інтернет-з'єднання
Фізичний простір	Вимагає виділеного простору для лабораторії	Не потребує фізичного простору, можливість використання вже існуючих приміщень
Супроводження	Потребує власного технічного персоналу для обслуговування обладнання та інфраструктури	Технічне супроводження та підтримка надаються провайдером хмарної інфраструктури
Час виконання	Можлива затримка в налагодженні та у вирішенні проблем	Зручний доступ до віддаленої інфраструктури, що може пришвидшити час виконання
Системна безпека	Залежить від рівня фізичної та логічної захищеності локальної мережі	Залежить від надійності та безпеки хмарної інфраструктури та доступу до неї
Доступ до ресурсів	Обмежений доступ до локальних ресурсів	Зручний доступ до глобальних ресурсів та інших хмарних послуг

ТОПОЛОГІЯ МЕРЕЖІ

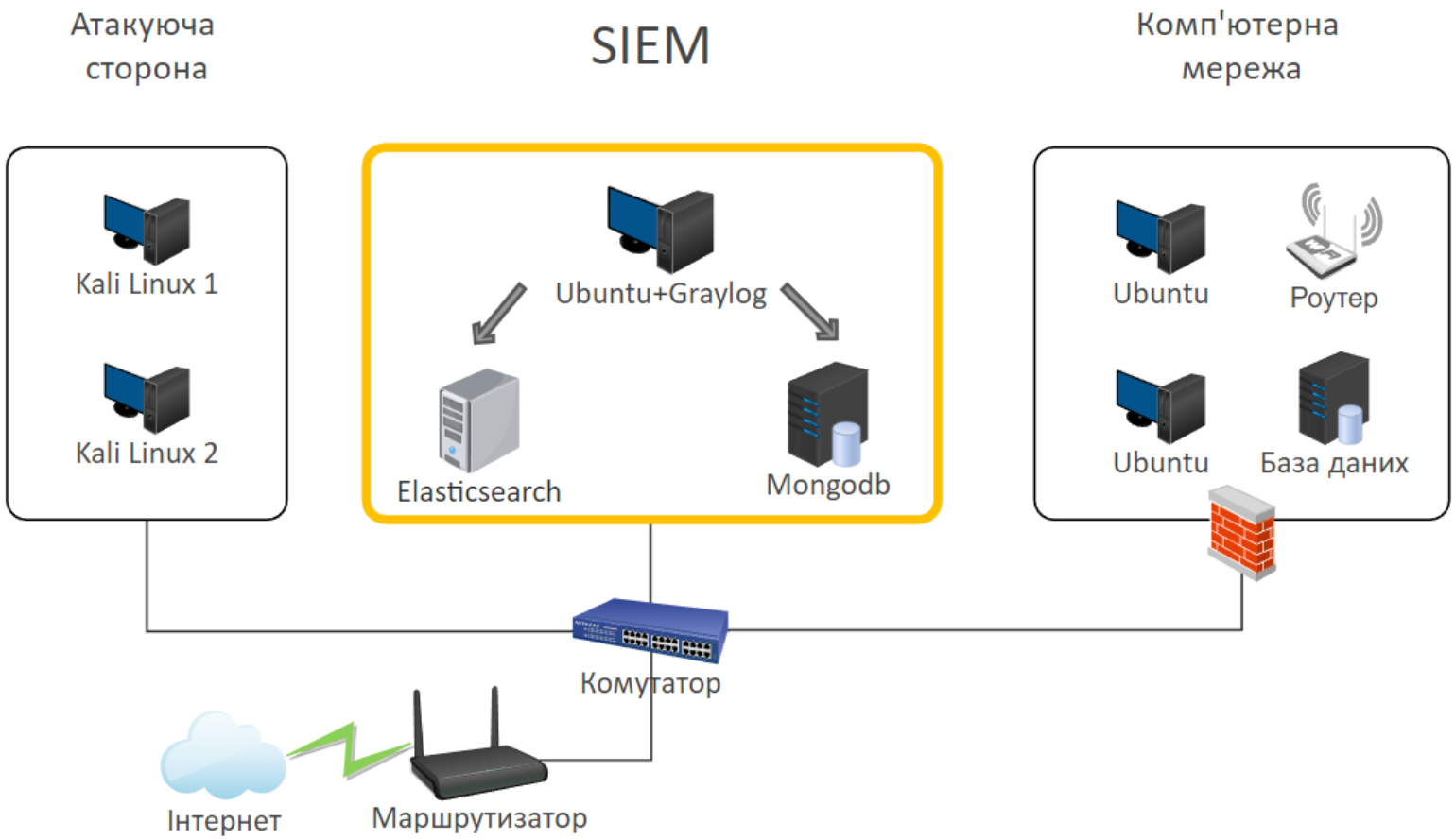


СХЕМА ОБРОБКИ ЛОГІВ

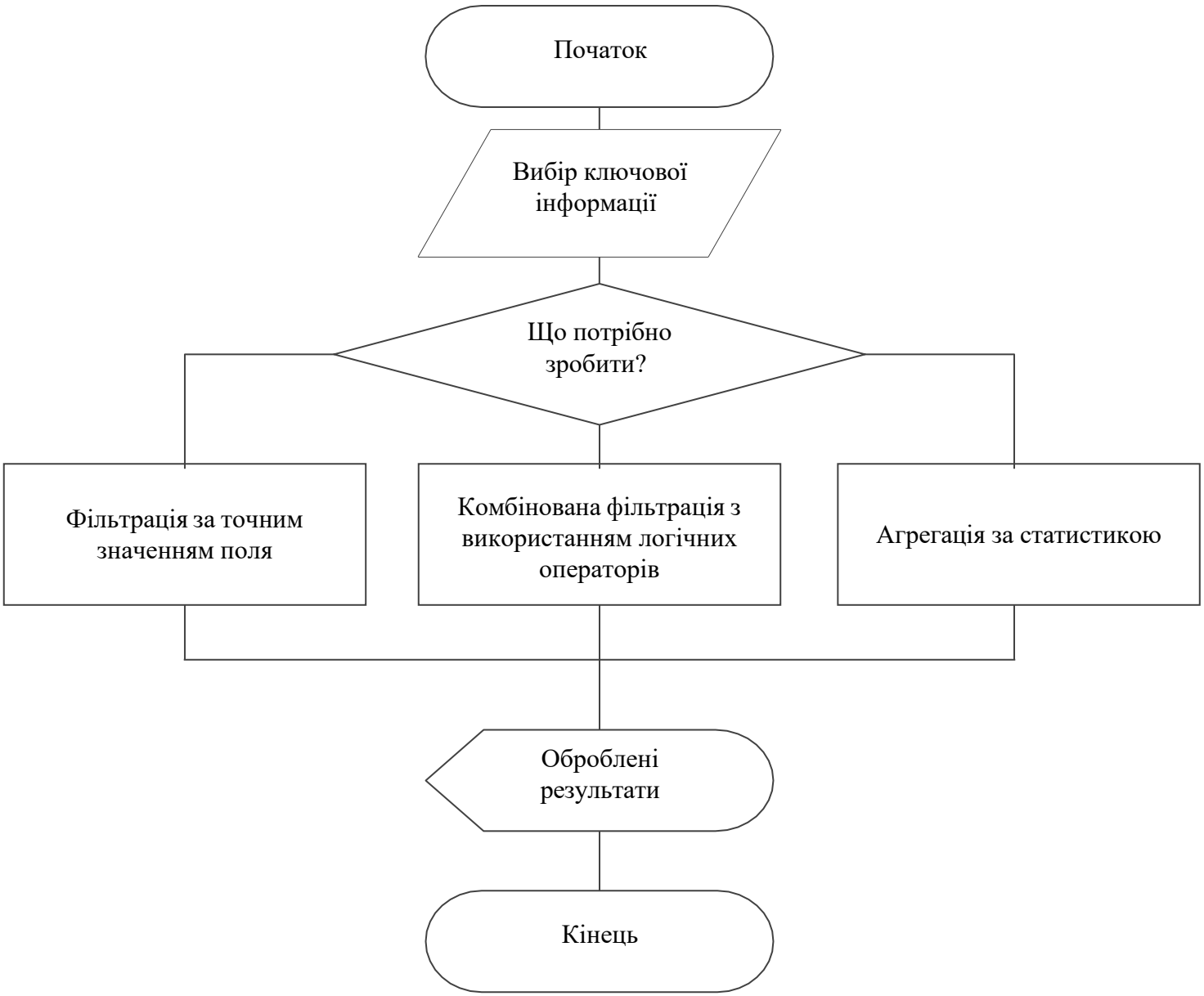
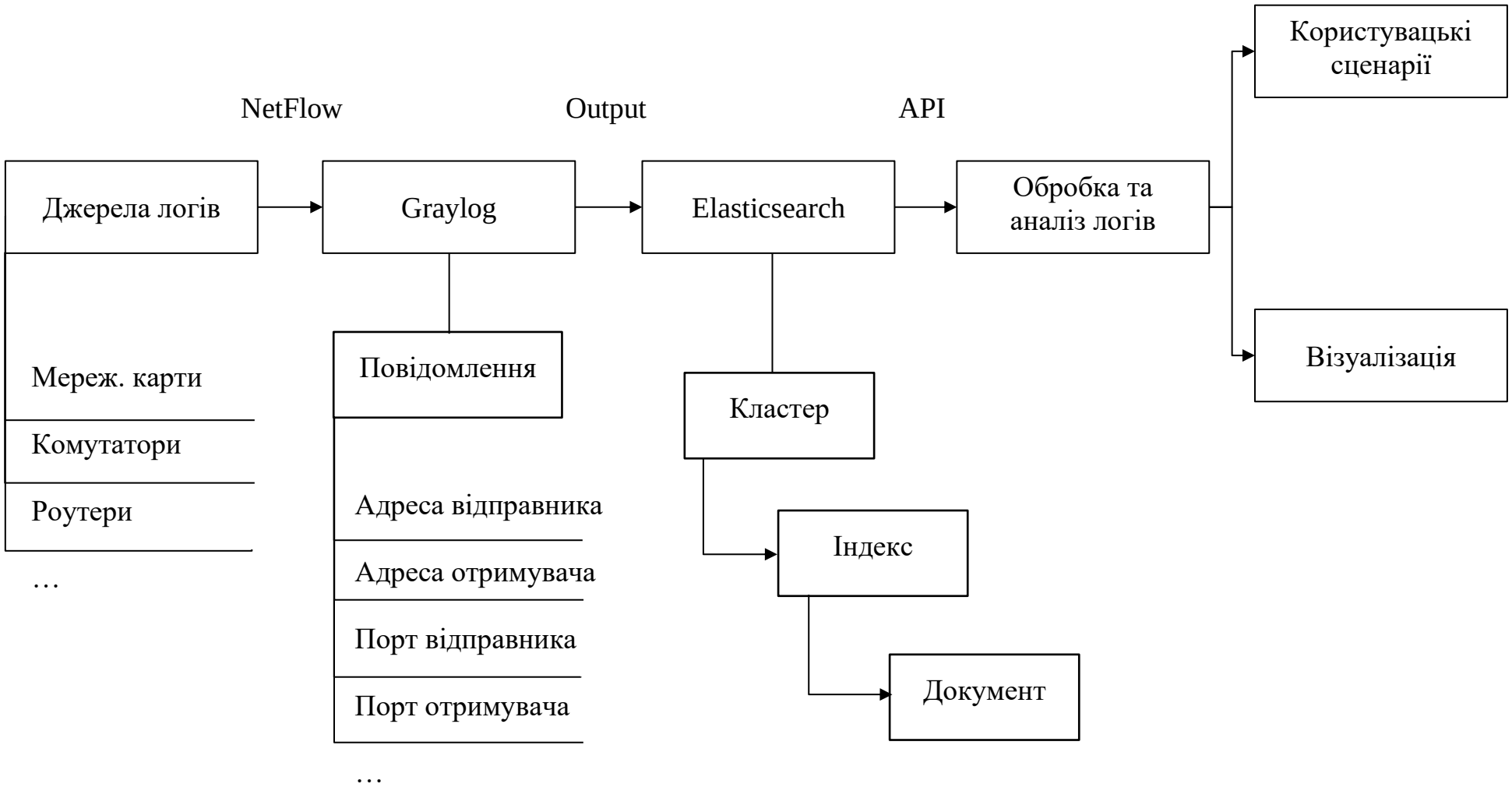


СХЕМА ШЛЯХУ, ЯКИЙ ПРОХОДЯТЬ ЛОГИ



ЧАСТКОВИЙ РЕЗУЛЬТАТ ОТРИМАННЯ ЛОГІВ


```
[
  {
    "_index" : "graylog_0",
    "_type" : "_doc",
    "_id" : "7a745440-077b-11ee-b064-000c2990507e",
    "_score" : null,
    "_source" : {
      "nf_dst" : "239.255.255.250:1900",
      "nf_stop" : "2023-06-10T10:41:11.891Z",
      "nf_version" : 5,
      "gl2_remote_ip" : "127.0.0.1",
      "gl2_remote_port" : 49033,
      "nf_pkts" : 4,
      "source" : "127.0.0.1",
      "gl2_source_input" : "646bae9bcf82341db70912e1",
      "nf_tcp_flags" : 0,
      "nf_dst_port" : 1900,
      "nf_bytes" : 792,
      "nf_src_as" : 0,
      "nf_proto" : 17,
      "nf_dst_address" : "239.255.255.250",
      "nf_src_port" : 61292,
      "gl2_source_node" : "72db8cfe-a25e-41d8-ae67-f703c2ccfe64",
      "timestamp" : "2023-06-10 10:42:20.000",
      "nf_src" : "192.168.137.1:61292",
      "nf_dst_mask" : 0,
      "gl2_accounted_message_size" : 543,
      "nf_flow_packet_id" : 22859,
      "streams" : [
        "000000000000000000000001"
      ],
      "gl2_message_id" : "01H2JECQW5M7BEG97401PCDPJG",
      "message" : "NetFlowV5 [192.168.137.1]:61292 <> [239.255.255.250]:1900 proto:17 pkts:4 bytes:792",
      "nf_dst_as" : 0,
      "nf_tos" : 0,
      "nf_src_address" : "192.168.137.1",
      "nf_proto_name" : "UDP",
      "nf_src_mask" : 0,
      "nf_snmp_input" : 0,
      "nf_snmp_output" : 0,
      "nf_start" : "2023-06-10T10:41:08.862Z"
    },
    "sort" : [
      1686393740000
    ]
  }
]
```

ПЕРЕВІРКА ПРАЦЕЗДАТНОСТІ ТА СТРУКТУРИ ЛОГІВ У ELASTICSEARCH

```
root@user-VirtualBox:/home/user# curl -XGET "localhost:9200/_search?pretty"
{
  "took" : 5,
  "timed_out" : false,
  "_shards" : {
    "total" : 12,
    "successful" : 12,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 10000,
      "relation" : "gte"
    },
    "max_score" : 1.0,
    "hits" : [
      {
        "_index" : "graylog_0",
        "_type" : "_doc",
        "_id" : "aa4a8390-f193-11ed-b498-000c2990507e",
        "_score" : 1.0,
        "_source" : {
          "gl2_accounted_message_size" : 347,
          "level" : 4,
          "gl2_remote_ip" : "192.168.137.128",
          "gl2_remote_port" : 52118,
          "streams" : [
            "000000000000000000000001"
          ],
          "gl2_message_id" : "01H0ANE0J9GJYV1C7X1PX77MVG",
          "source" : "user-VirtualBox",
          "message" : "[ 5565.195139] IN=lo OUT= MAC=00:00:00:00:00:00:00:00:00:00:0
          "gl2_source_input" : "645f885b76e67038084bb3b1",
          "application_name" : "kernel",
          "facility_num" : 0,
          "gl2_source_node" : "72db8cfe-a25e-41d8-ae67-f703c2ccfe64",
          "facility" : "kernel",
          "timestamp" : "2023-05-13 13:36:46.011"
        }
```

```
user@user-VirtualBox:~/Desktop/python-elasticsearch$ curl -XGET http://localhost:9200/_cat/indices
green open .geoip_databases   7H6g_75SRZK7L8GrcZdLEw 1 0      43 0  40.9mb  40.9mb
green open gl-events_0        BPWzFmXoQGmaf_R-KIda_g 4 0       0 0    908b    908b
green open graylog_0          5N7UE-MpT0S2SZUbCotStg 4 0 3483946 1 491.6mb 491.6mb
green open gl-system-events_0 XOMhD2GMRd6MSERM89tesg 4 0       0 0    908b    908b
```

```
      "properties" : {
        "application_name" : {
          "type" : "keyword"
        },
        "facility" : {
          "type" : "keyword"
        },
        "facility_num" : {
          "type" : "long"
        },
        "full_message" : {
          "type" : "text",
          "analyzer" : "standard"
        },
        "gl2_accounted_message_size" : {
          "type" : "long"
        },
        "gl2_message_id" : {
          "type" : "keyword"
        },
        "gl2_processing_timestamp" : {
          "type" : "date",
          "format" : "uuuu-MM-dd HH:mm:ss.SSS"
        },
        "gl2_receive_timestamp" : {
          "type" : "date",
          "format" : "uuuu-MM-dd HH:mm:ss.SSS"
        },
        "gl2_remote_ip" : {
          "type" : "keyword"
```

ВИКОРИСТАННЯ ФІЛЬТРІВ ТА АГРЕГАЦІЙ НА ПРИКЛАДІ СПРОЕКТОВАНОЇ АТАКИ


```
      {
        "_index" : "graylog_0",
        "_type" : "_doc",
        "_id" : "aa4a83b4-f193-11ed-b498-000c2990507e",
        "_score" : 1.0,
        "_source" : {
          "gl2_accounted_message_size" : 347,
          "level" : 4,
          "gl2_remote_ip" : "192.168.137.128",
          "gl2_remote_port" : 52118,
          "streams" : [
            "000000000000000000000001"
          ],
          "gl2_message_id" : "01H0ANE0J96FC2NP4Y1EJ28GEZ",
          "source" : "user-VirtualBox",
          "message" : "[ 5565.195394] IN=lo OUT= MAC=00:00:00:00:00:00:00:00
:00:00:00:00:08:00 SRC=192.168.137.128 DST=192.168.137.128 LEN=290 TOS=0x00
PREC=0x00 TTL=64 ID=8026 DF PROTO=UDP SPT=53829 DPT=5149 LEN=270",
          "gl2_source_input" : "645f885b76e67038084bb3b1",
          "application_name" : "kernel",
          "facility_num" : 0,
          "gl2_source_node" : "72db8cfe-a25e-41d8-ae67-f703c2ccfe64",
          "facility" : "kernel",
          "timestamp" : "2023-05-13 13:36:46.011"
        }
      }
    ]
  }
}
```

```
"aggregations" : {
  "source_addresses" : {
    "doc_count_error_upper_bound" : 0,
    "sum_other_doc_count" : 0,
    "buckets" : [
      {
        "key" : "192.168.137.129",
        "doc_count" : 86
      },
      {
        "key" : "192.168.137.128",
        "doc_count" : 71
      },
      {
        "key" : "192.168.137.2",
        "doc_count" : 25
      },
      {
        "key" : "192.168.137.1",
        "doc_count" : 3
      },
      {
        "key" : "34.117.65.55",
        "doc_count" : 2
      },
```

```
  "aggregations" : {
    "packet_count" : {
      "doc_count_error_upper_bound" : 0,
      "sum_other_doc_count" : 0,
      "buckets" : [
        {
          "key" : "TCP",
          "doc_count" : 64
        },
        {
          "key" : "UDP",
          "doc_count" : 22
        }
      ]
    },
    "avg_packet_size" : {
      "value" : 117.20930232558139
    }
  }
```