КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультету радіофізики, електроніки та комп'ютерних систем
Кафедра комп'ютерної інженерії

МОНІТОРИНГ ТРАФІКУ МЕРЕЖІ СТУДМІСТЕЧКА В РЕАЛЬНОМУ ЧАСІ

Кваліфікаційна робота бакалавра
студента 4 року навчання
спеціальність: 123 «Комп'ютерна інженерія»
Георгія ГЕОРГАДЗЕ

Науковий керівник:
канд. технічних наук Євген СЛЮСАР,
асистент кафедри
комп'ютерної інженерії,радіотехніки та
радіоелектронних систем

Рецензент:
канд. фіз.-мат. наук Іван КОЛОМІЄЦЬ
асистент кафедри електрофізики

До захисту допускаю:

Завідувач кафедрою
канд. фіз.-мат. наук Юрій БОЙКО
доцент кафедри комп'ютерної інженерії

Ухвалено на засіданні кафедри "____" _____2022 р., протокол № ____

Київ - 2022

РЕФЕРАТ

Кваліфікаційна робота містить 41 сторінку, 29 рисунків, 1 додаток, використано 9 інформаційних джерел.

ELK, IPFIX, СИСТЕМА МОНІТОРИНГУ, ELASTICSEARCH, LOGSTASH, KIBANA, FILEBEAT.

Об’єктом даної роботи є мережа студміста КНУ. Предметом роботи є трафік мережі студміста КНУ.

Метою роботи є створення системи моніторингу трафіку мережі студміста КНУ в реальному часі.

Інстументи реалізації — програмне забезпечення Logstash, Kibana, Filebeat, Elasticsearch, протокол IPFIX, операційна система Ubuntu та апаратні компоненти системи.

Результати роботи — проведено аналітичний огляд існуючої системи моніторингу, спроєктовано архітектуру, реалізовано прототип системи, протестовано прототип на дотримання вимог та можливості повноцінної роботи.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ І ПОЗНАЧЕНЬ

БД — база даних

SNMP — Simple Network Management Protocol

IP — Internet Protocol — інтернет протокол

ПЗ — програмне забезпечення

PDU — Protocol Data Unit

B-K-D — деревоподібна структура даних

VLANID — ідентифікаційний набір цифр для

AS — автономна система

ВСТУП

У наш час мережа Інтернет відіграє головну роль у комунікації та пошуку інформації, що особливо актуально для студентів вищих навчальних закладів. Зважаючи на різке збільшення кількості ресурсів та користувачів навантаження на мережеву інфраструктуру постійно зростає, що призводить до виникнення аварійних ситуації.

Для забезпечення безперебійної роботи інфрастуктури потрібно постійно контролювати навантаження та швидко реагувати на збої у роботі. Рішенням, що розглянуто в цій роботі є застосування системи моніторингу трафіку. Оскільки кількість пристроїв, що задіяні у мережевій інфраструктурі студентського міста КНУ, постійно зростає, було створено систему моніторингу, яка надає можливість загального огляду мережевої інфраструктури. Вона дає змогу значно зменшити час виявлення та усунення збоїв та аварій. Однак, через використання протоколу SNMP та ПЗ, що не можуть надати детальний зміст пакетів, адміністрування та контроль інфраструктури мережі студміста КНУ не є повноцінними, що збільшує ймовірність виникнення аварійної ситуації, яка не може бути виявлена за допомогою існуючої системи.

РОЗДІЛ 1. ОГЛЯД СИСТЕМ МОНІТОРИНГУ ТРАФІКУ

1.1 Застосування систем моніторингу

Система моніторингу — це набір програмного забезпечення, що дозволяє спостерігати та аналізувати роботу інфраструктури і сервісів в режимі реального часу[1].

Головною причиною використання систем моніторингу є впровадження постійного контролю над важливою частиною інфраструктури та забезпечення швидкого реагування на аварійні ситуації різного типу. Також, системи моніторингу допомагають зробити максимально точний аналіз причин виникнення проблем у роботі інфраструктури, за допомогою постійного спостереження та логування будь-яких параметрів, значення яких вийшло за рамки норми.

У залежності від обладнання та сфери застосування системи моніторингу поділяють на:

- моніторинг мережі — побудова топології мережі, збір та графічне відображення інформації про стан з’єднання та навантаженість пристроїв;
- моніторинг робочих станцій та серверів — збір та аналіз інформації про параметри продуктивності та завантаження пристроїв, стан операційної системи пристрою та апаратної частини;
- моніторинг сервісів — створення моделі типу “сервіс-ресурс”, збір та графічне відображення доступності сервісів.

1.2 Моніторинг трафіку мережі студентського містечка КНУ

Кількість пристроїв, що забезпечують безперебійну роботу мережевої інфраструктури студентського міста КНУ зростає кожного року. Зі збільшенням кількості обладнання, яке потребує адміністрування, збільшується також і складність швидкого реагування на проблеми, що виникають під час роботи інфраструктури. Крім цього виникає гостра потреба аналізу трафіку та моніторингу якості сервісу.

Зважаючи на всі вимоги, було створено систему моніторингу, загальну схему якої зображено на Рис. 1.

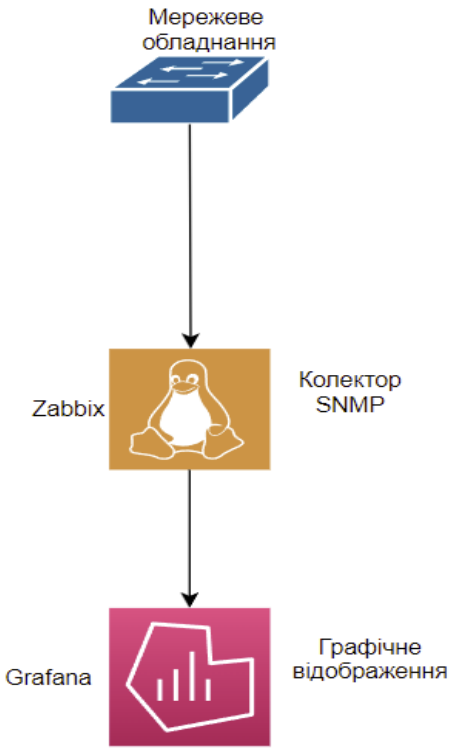

Рис. 1. Схема системи моніторингу

Система моніторингу складається з декількох елементів:

- мережеве обладнання, яке і є об’єктом моніторингу;
- колектор;
- системою графічного відображення.

Також, головною ланкою цієї системи є протокол SNMP що використовується для постійного опитування та отримання інформації з мережевого обладнання.

1.3 Аналіз чинної системи моніторингу трафіку

1.3.1 Структура та функціонал системи

Зважаючи на постійне збільшення кількості приладів, що забезпечують роботу мережевої інфраструктури, було створено систему моніторингу, яка складається з таких компонентів:

- колектор — система, що надсилає запити, отримує та обробляє інформацію з мережевого пристрою;
- система графічного відображення — інструмент, що візуалізує оброблені дані у реальному часі.
- протокол взаємодії — інтернет-протокол, за допомогою якого відбувається взаємодія між колектором та об’єктом моніторингу.

Роль колектора у даній системі виконує програмне забезпечення Zabbix разом з базою даних, що відправляє запити до об’єкта моніторингу, та обробляє дані, що надходять у відповідь.

Системою графічного відображення є платформа для візуалізації та аналізу даних Grafana. Інформація, яку візуалізує Grafana, зберігається у базі даних.

Протоколом взаємодії є стандартний інтернет-протокол SNMP, що використовується для опитування, збору інформації та управління пристроями в IP-мережах. Для отримання даних з пристроїв, використовуються snmp-агенти та snmp-traps, принцип роботи яких вказано на Рис. 2.

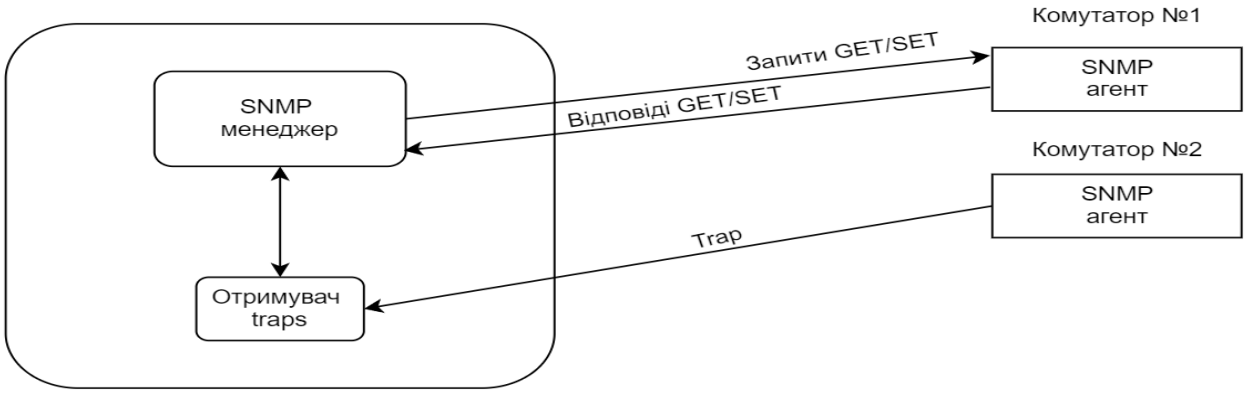

Рис. 2. Загальна схема опису роботи протоколу SNMP

SNMP-агенти - це ПЗ, яке розташоване на об’єкті моніторингу та використовується для отримання запитів від SNMP-менеджера та надавання очікуваної інформації про об’єкт.

SNMP-trap - це особливий вид PDU, який надсилає SNMP-агент незалежно від отримання GET/SET запитів. Головною метою використання SNMP-trap є невідкладне повідомлення SNMP-менеджера про неполадки у роботі пристрою.

За допомогою протоколу SNMP можна отримати загальну інформацію про пристрій та його стан. Варто звернути увагу, що протокол SNMP не може надати детальну інформацію про мережу, а саме перегляд мереж, віртуальних структур та іншої інформації, що міститься у фактичному трафіку. Крім цього, використання snmp-агентів потребує двосторонньої взаємодії, що ускладнює моніторинг інфраструктури при зростанні кількості об’єктів моніторингу. Це накладає велику кількість обмежень по видимості внутрішньої мережі, діагностуванню локальних збоїв у роботі мережі та неполадок, що неможливо виявити стандартними методами. Наприклад, ефективність балансування трансляції внутрішніх адрес та детектування спроб генерації автотрафіку. Також, при використанні лише SNMP швидко досягається ліміт функціонала моніторингу, так як SNMP не може аналізувати трафік чи надавати інформацію про його типізацію.

1.4 Постановка задачі

Завершуючи огляд чинної системи моніторингу, можна зробити висновки про неможливість контролю та глибокого аналізу трафіку. Інструменти, які наразі використовуються, не можуть надавати детальний зміст пакетів, що зменшує можливість виявлення мережевих атак, адміністрування трафіку та моніторингу якості сервісу.

Підводячи підсумок, можна сформувати наступні задачі для створення системи моніторингу мережі у студентському містечку КНУ:

- вибір протоколу, що дозволяє детально розглядати будь-який трафік з мережевого приладу;
- створення архітектури системи моніторингу, а саме клієнтської та серверної частин;
- розгортання системи на основі створеної архітектури;
- тестування системи.

РОЗДІЛ 2. ПРОЄКТУВАННЯ СИСТЕМИ МОНІТОРИНГУ ТРАФІКУ

2.1 Структура системи моніторингу трафіку та вимоги до неї

Система моніторингу мережі - це багатокомпонентна система програмного та апаратного забезпечення, що дозволяє спостерігати за мережевою інфраструктурою у реальному часі, та швидко реагувати на збої у роботі мережевої інфраструктури. Головною метою створення системи моніторингу мережі є забезпечення контролю над трафіком та якістю сервісу.[2]

Для можливості постійного моніторингу трафіку та логування даних система моніторингу мережі має відповідати наступним параметрам.

Відмовостійкість. Можливість повноцінного функціонування системи при виході з ладу певного компонента. Відмовостійкість забезпечується здатністю системи розгортатись на декількох вузлах системи та поєднуватись у кластер та частковою незалежністю компонентів між собою.

Детальне фільтрування. Система фільтрує вхідні дані для забезпечення швидкого пошуку інформації та аналізу по ключовому параметру.

Незалежність від вендора. Спроможність системи працювати з різноманітними вендорами.

Зважаючи на сформовані вимоги було складено перелік компонентів, що необхідні для функціонування системи:

- Колектор — система, що отримуватиме дані від об’єкта моніторингу.
- Сховище даних — програмне забезпечення, яке зберігатиме дані для візуалізації
- Фільтр даних — конвеєр для обробки даних
- Система графічного відображення — користувацький інтерфейс, що дозволяє візуалізувати дані.
- Протокол взаємодії — мережевий протокол, що дозволяє здійснювати детальний аналіз трафіку.

Взявши до уваги вимоги та перелік компонентів, було створено структуру

системи моніторингу трафіку, яка зображена на Рис. 3.

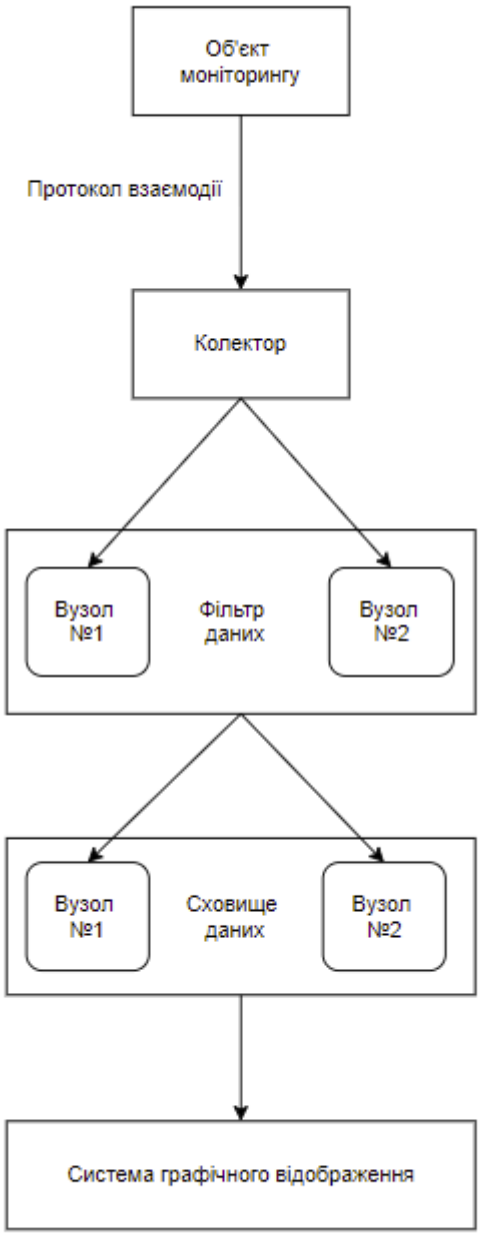

Рис. 3. Структура системи моніторингу мережі

Структура, що приведена на Рис. 3 містить колектор, два вузли, що забезпечують фільтрацію даних, два вузла сховища даних та систему графічного відображення інформації.

Потрібно зауважити, що використання декількох вузлів фільтрації даних, дозволяє виконати балансування навантаження, значно зменшуючи можливість

створення “вузького місця” між колектором та фільтром даних. У випадку, коли балансування навантаження не є доцільним, випадковим чином буде обрано лише один вузол, що буде виконувати фільтрацію даних. При недоступності обраного вузла, система автоматично вибиратиме наступний вузол.

Сховище даних, яке складається з декількох вузлів зменшує вірогідність пошкодження або недоступності даних та дозволяє розподіляти задачі.

2.2 Вибір інструментів та програмного забезпечення

2.2.1 Протокол взаємодії

Під час створення системи моніторингу трафіку виникає питання який протокол взаємодії обрати для найбільшої ефективності системи. Вибір протоколу впливає на всі компоненти, що будуть задіяні у даній схемі.

Спираючись на вищезгадані вимоги, а саме на незалежність від вендора та детальний аналіз трафіку, було обрано протокол IPFIX, який було створено на основі протоколу NetFlow, що розроблений компанією Cisco Systems.

IPFIX — це протокол для передачі інформації про IP-Flow в мережі.[3] Моніторинг IP-Flow дозволяє з’ясувати причину перевантаження мережі, оптимізувати мережеву інфраструктуру, аналізувати стан обраного об’єкта.

IP-Flow — це послідовність пакетів, що проходять через точку спостереження протягом певного інтервалу часу. Пакети, що належать до одного потоку, мають спільні властивості, такі як: однакові поля транспортного заголовка, однакові параметри самого пакета, однакові параметри тіла пакета.[4]

Пакет IPFIX складається із заголовка пакета та декількох наборів інформації, що поділяються на:

- Набір даних
- Набір шаблонів
- Набір шаблонів параметрів

Заголовок пакета містить у собі основну інформацію про пакет, а саме

версію IPFIX, довжину пакета, порядковий номер[5]. Формат заголовку пакета зображено на Рис.4.

Набір даних — запис, що містить значення параметрів, які описані у наборі шаблонів.

Набір шаблонів — запис, що визначає та характеризує структуру полів в наборі даних.

Набір шаблонів параметів — запис, що доповнює набір шаблонів, вказує на місце застосування набору даних.

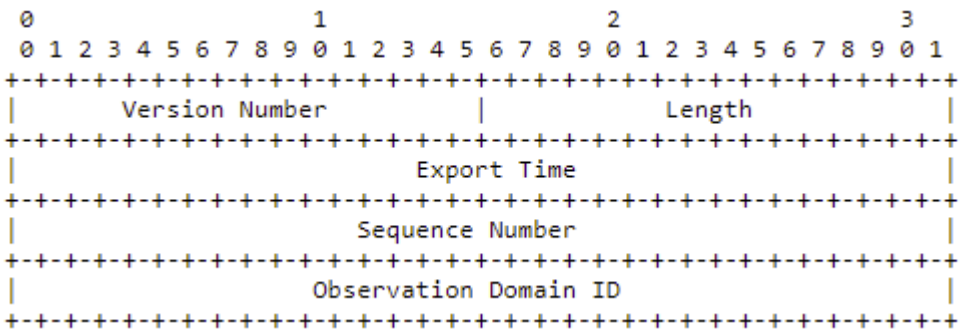

Рис.4 Формат заголовку пакета IPFIX

Заголовок пакета складається з полів:

Version Number — версія IPFIX, значення для поточної версії 0x000a.

Length — довжина пакета в октетах, включаючи заголовок і набори даних.

Export Time — час відправлення пакета з експортера.

Sequence Number — лічильник пакетів у конкретному потоці.

Observation Domain ID — 32-бітний ідентифікатор домена спостереження.

Головними перевагами протоколу IPFIX є тип “push” та модульний формат пакетів.

Тип “push” означає відсутність двосторонньої взаємодії між отримувачем та відправником. Кожний відправник періодично відправляє IPFIX-пакети до отримувача без взаємодії зі сторони отримувача. Також, протокол IPFIX підтримує можливість відправки пакетів до декількох колекторів, які виконують роль отримувача, що дозволяє зменшити можливість зупинення моніторингової системи через недоступність отримувача. Недоступність отримувача є точкою

відмови системи, тому можливість використання декількох отримувачів є вагомим кроком для створення відмовостійкої системи.

Модульний формат дозволяє видозмінювати структуру пакета, що дозволяє передавати різну інформацію в залежності від потреби.

2.2.2 Колектор

При виборі колектора, потрібно звернути увагу на можливість використання різних протоколів взаємодії, розгортання у будь-якому середовищі та регулювання швидкості передачі даних до фільтра.

Зважаючи на вищезгадані умови, було обрано ПЗ Filebeat, що має модульну структуру, та дозволяє взаємодіяти з більшістю загальновідомих форматів логування.

Загальний алгоритм роботи Filebeat зображено на Рис. 5

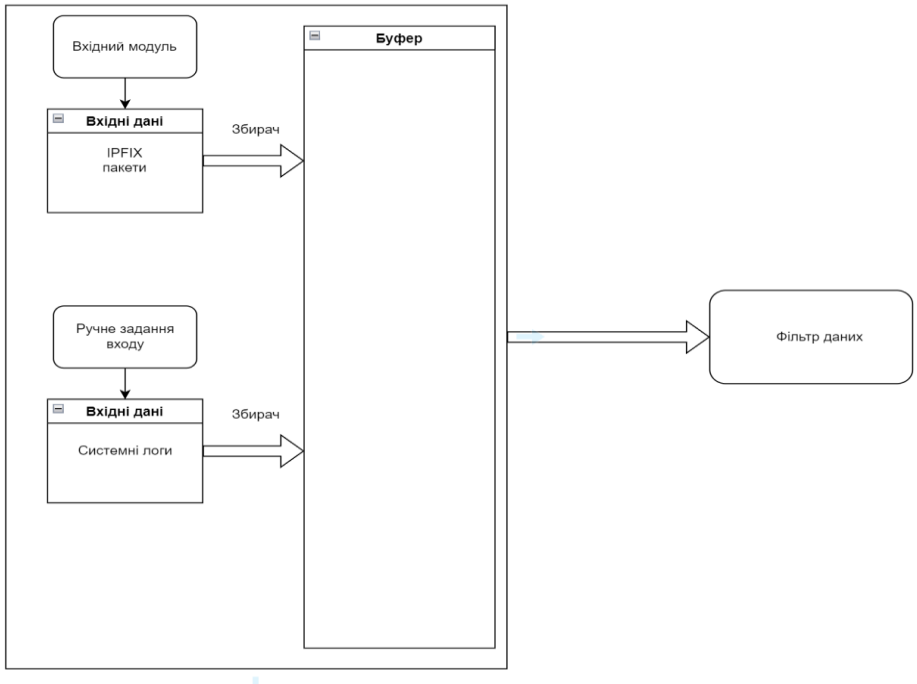

Рис. 5. Загальний алгоритм роботи Filebeat

Описати вхідні дані можливо за допомогою модулів та ручного налаштування.Модулі, що містяться у Filebeat дозволяють швидко почати роботу з Filebeat, оскільки містять конфігурації за замовчуванням для більшості

загальновідомих форматів логування. При потребі взаємодії з особливим форматом, який не підтримується модулями, використовується ручне налаштування.

Збирач відповідає за читання вхідних даних в одному файлі. Для кожного файлу запускається окремий збирач. Читання даних відбувається рядково, після закінчення читання рядка, збирач відправляє дані у буфер, з якого далі виконується відправка до фільтра даних. При недоступності фільтра, збирач логує інформацію про стан файлу та рядки, що були прочитані та надіслані, та припиняє зчитування файлу. Після відновлення доступу до фільтра, збирач продовжить зчитування файлу.[6]

Зважаючи на те, що файл може бути переміщений або перейменований, Filebeat присвоює унікальний ідентифікатор для кожного файла.Також, Filebeat гарантує однократну доставку до фільтра без втрати даних.

2.2.3 Фільтр даних

Фільтр даних є одним із головних компонентів системи моніторингу, завдання якого перетворення різноманітних даних у загальний формат для більш ефективного аналізу. Головні вимоги до фільтра даних полягають у можливості динамічно перетворювати формати даних незалежно від початкового формату і складності, виконувати детальну та гнучку фільтрацію для кожного типу даних.

Беручи до уваги вищезгадані вимоги, в якості фільтра даних було обрано Logstash.

Logstash — конвеєр для обробки даних у реальному часі. Завдяки широкому вибору вхідних, фільтраційних та вихідних плагінів, logstash має можливість отримувати та надсилати дані з різних джерел. Загальний алгоритм роботи Logstash зображено на Рис. 6.

Рис. 6. Загальний алгоритм роботи Logstash

Конвеєр logstash поділений на 3 частини:

- Вхід
- Фільтри
- Вихід

Вхід — перший етап в конвеєрі, який генерує події в залежності від отриманих даних.

Фільтри — проміжний етап, у якому комбінується фільтрування і накладення умов для аналізу та структурування даних.

Вихід — кінцевий етап конвеєра, що відправляє дані у сховище. Logstash має можливість відправляти відфільтровані та структуровані дані відразу до декількох отримувачів.

Крім цього, вхід та вихід підтримують застосування потокових фільтрів, для кодування та декодування даних.[7]

2.2.4 Сховище даних

Сховище даних — це програмне забезпечення, яке надає можливість зберігати та індексувати дані. Головними вимогами до сховища даних є можливість підтримувати швидкий пошук даних для графічного відображення у режимі реального часу та можливість об’єднання вузлів у кластер для запобігання втрати даних.

На роль сховища даних було обрано ElasticSearch — розподілену пошукову та аналітичну систему, яка забезпечує швидкий пошук за допомогою інвертного індексу. ElasticSearch є маштабованою системою, що дозволяє отримувати Продуктивність ElasticSearch не є статичною, вона залежить від складності типів даних. Оцінка продуктивності ElasticSeatch є загальнодоступною інформацією, та створюється за допомогою методології порівняльного аналізу. [8] Головною ідеєю такого оцінювання є надання допомоги адміністраторам виявляти зниження продуктивності системи та обирати характеристики пристроїв для повноцінної роботи.[9] Приклад оцінки продуктивності системи для суворо типізованих даних зображено на Рис. 7.

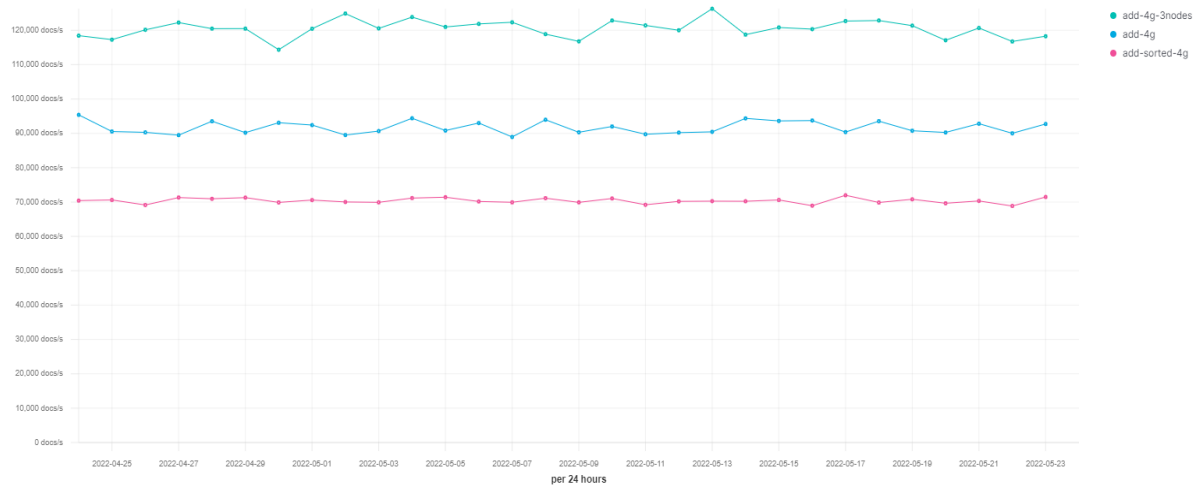

Рис. 7. Оцінка продуктивності ElasticSearch

Інвертний індекс — це структура даних, поля якої являють собою пари “ключ-значення.” За замовчуванням, ElasticSearch індексує всі поля вхідних даних, та визначає метод зберігання даних. Наприклад, текстові поля будуть зберігатись у інвертних індексах, а числові — у B-K-D дереві.

Роботу ElasticSearch можливо поділити на чотири головні функції:

1) Управління документами

2) Спів-ставлення

3) Аналіз

4) Методологія пошуку

Управління документами - створення індексу та типу для створення організації даних.

Спів-ставлення — процес визначення документу та його полів.

Аналіз — процес конвертації неструктурованих даних в структурований формат, що є оптимізованим для пошуку. Ключовою особливістю для пошукової системи elasticsearch є виконання аналізу даних під час індексації або пошуку полів.

Методологія пошуку — використання запитів, що адресуються до elasticsearch. Запити поділяються на:

1) Складені запити — охоплюють декілька запитів для з'єднання результату їх роботи.

2) Текстові запити — дають змогу знайти проаналізовані текстові поля, такі як тіло електронного листа. Запит обробляється за допомогою спеціального індексатора, що був застосований під час дії функції управління документами.

3) JOIN-запити — представляють виконання запитів у стилі SQL у розподіленій системі elasticsearch. Але через зменшення ефективності використання ресурсів системи стандартні JOIN-запити були поділені на вкладені(nested) запити та запити has_child/has_parent.

4) Запити на рівні термінів — знаходження документів за допомогою точних значень у структурований даних. Приклади структурованих даних містять діапазони дат, ip-адрес, ідентифікаторів товарів. При використанні даного типу запиту, алгоритм пошуку відрізняється від загального, виконується лише пошук по точним термінам, що зберігаються у полях документів.

2.2.5 Система графічного відображення

Система графічного відображення — інструмент для зручного відображення та перегляду даних. Головна ідея системи графічного відображення полягає у візуалізації великої кількості даних, аналіз яких у текстовому форматі займає багато часу.

Система графічного відображення має відповідати двом вимогам, а саме: можливість налаштування сповіщень, різноманіття візуалізацій даних.

Зважаючи на вимоги, на роль системи графічного відображення було обрано Kibana.

Kibana — користувацький інтерфейс на основі браузера для аналізу та візуалізації даних у вигляді графіків, діаграм та гістограм.

2.3 Архітектура системи

Обравши програмне забезпечення та протокол взаємодії для майбутньої системи моніторингу, потрібно створити детальну архітектурну схему системи для покращення швидкості розгортання та зменшення часу налагодження роботи системи.

Зважаючи на те, що система моніторингу трафіку залежить не лише від системи програмного забезпечення для аналізу та відображення трафіку, але і від мережевих приладів, які будуть надсилати дані для моніторингу, можна вважати, що архітектура системи моніторингу є різновидом архітектури “клієнт - сервер”[10], де клієнтом вважається система програмного забезпечення, а сервером — об’єкт моніторингу.

2.3.1 Архітектура серверної частини

Архітектура серверної частини змінюються в залежності від об’єкта моніторингу. Одним із найпоширеніших варіантів архітектури можна вважати мережевий прилад, який містить сенсор IPFIX за замовчуванням. У такому випадку, архітектура серверної частини повністю забезпечена виробником та не підлягає зміні.

Іншим варіантом є випадок у якому в якості об’єкта моніторингу використано unix-подібну систему. У такому разі, для отримання даних з пристрою, використовується програмне забезпечення, що емулює сенсор IPFIX. Прикладом такого ПЗ може слугувати ipfixprobe, загальна схема роботи якого зображена на Рис. 8.

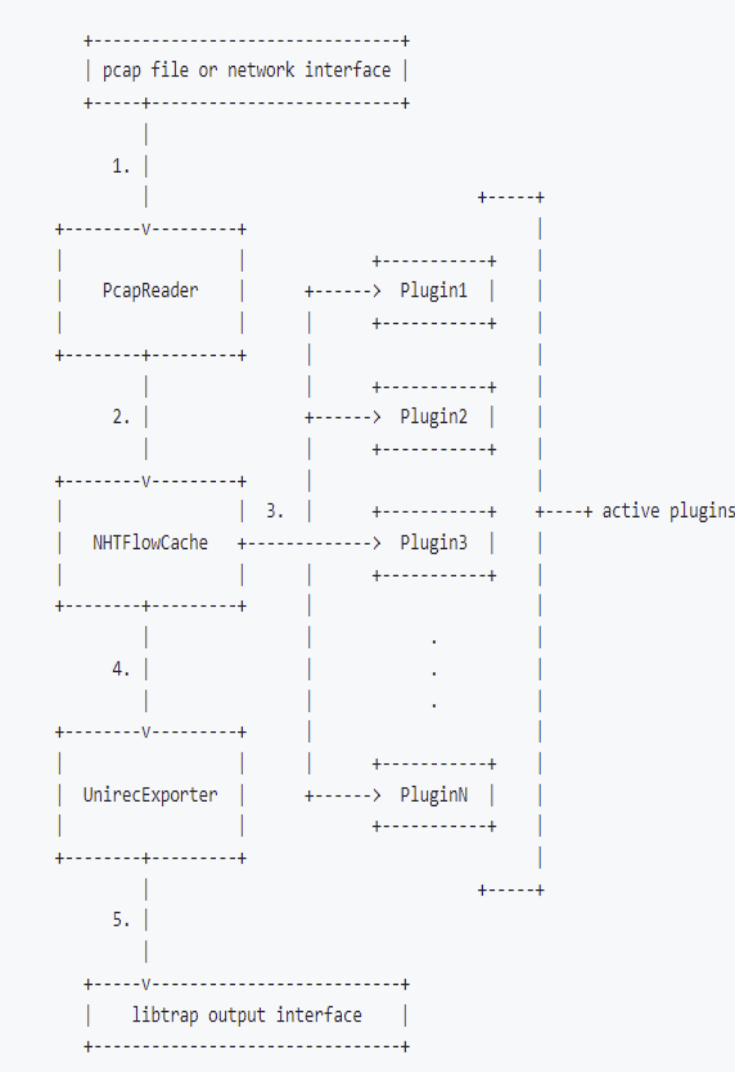

Рис. 8. Загальна схема роботи ПЗ ipfixprobe

2.3.2 Архітектура клієнтської частини

Архітектура клієнтської частини будується на основі структурної схеми системи моніторингу та обраного програмного забезпечення. Структурна схема системи зображена на Рис.3. Головними компонентами клієнтської частини системи є колектор, фільтр даних, сховище даних та система графічного відображення. Колектором слугує програмне забезпечення Filebeat, фільтром даних — Logstash, сховищем даних — ElasticSearch, та Kibana у якості системи графічного відображення. Вищезгадане програмне забезпечення формує ELK stack.

Взявши до уваги всі вимоги до системи моніторингу було створено архітектуру клієнтської частини, яку зображено на Рис.9

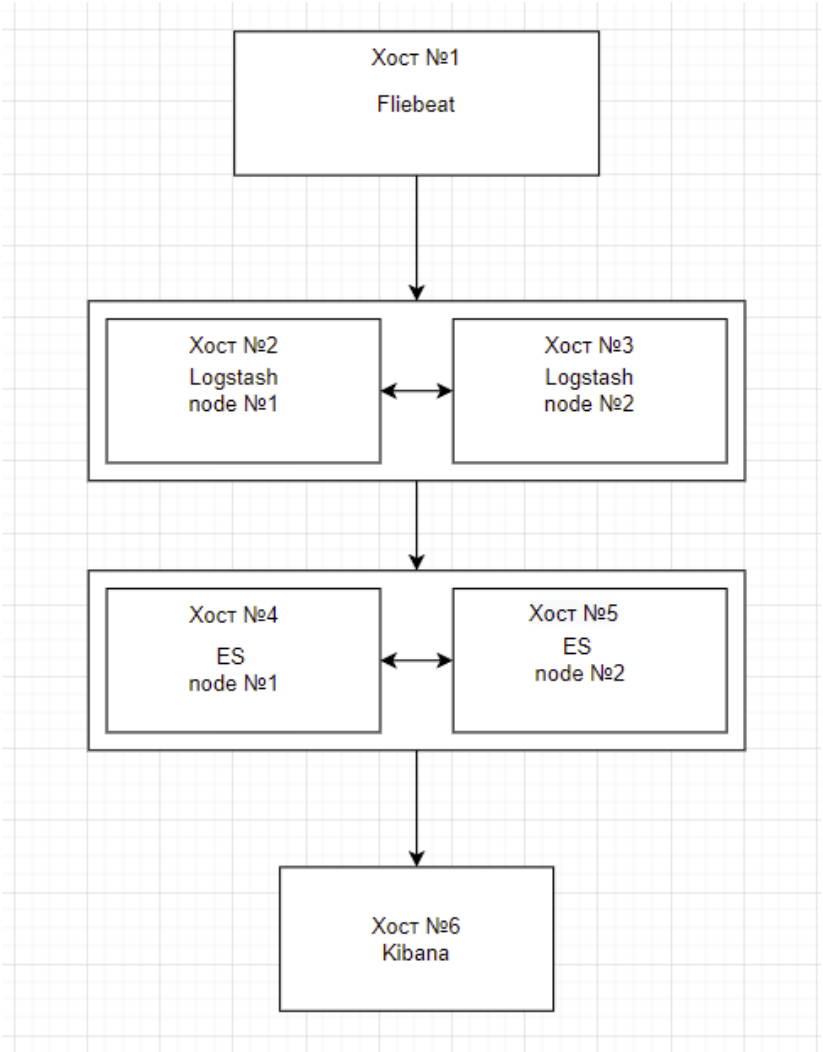

Рис. 9. Архітектура клієнтської частини

Виділення окремого хоста під кожний компонент гарантує уникнення єдиної точки відмови всієї системи. Використання декількох вузлів для фільтрації та зберігання надасть системі відмовостійкість. Крім того, є можливість, створення резервного колектора та системи графічного відображення для зменшення часу недоступності системи.

РОЗДІЛ 3. РОЗГОРТАННЯ СИСТЕМИ ТА ТЕСТУВАННЯ

3.1 Розгортання клієнтської частини

Для реалізації клієнтської частини системи було обрано віртуальну машину з наступними параметрами:

- Операційна система: Ubuntu 20.04.4 TLS
- Розмір оперативної пам’яті: 10 GB
- Кількість ядер процесора: 4

Вищеописана конфігурація дозволяє розгорнути Filebeat, один вузол Logstash, один вузол ElasticSearch, Kibana та ПЗ для емуляції сенсора IPFIX. Вибрана конфігурація дозволяє повноцінно протестувати систему. Архітектура системи для тестування зображена на Рис 10.

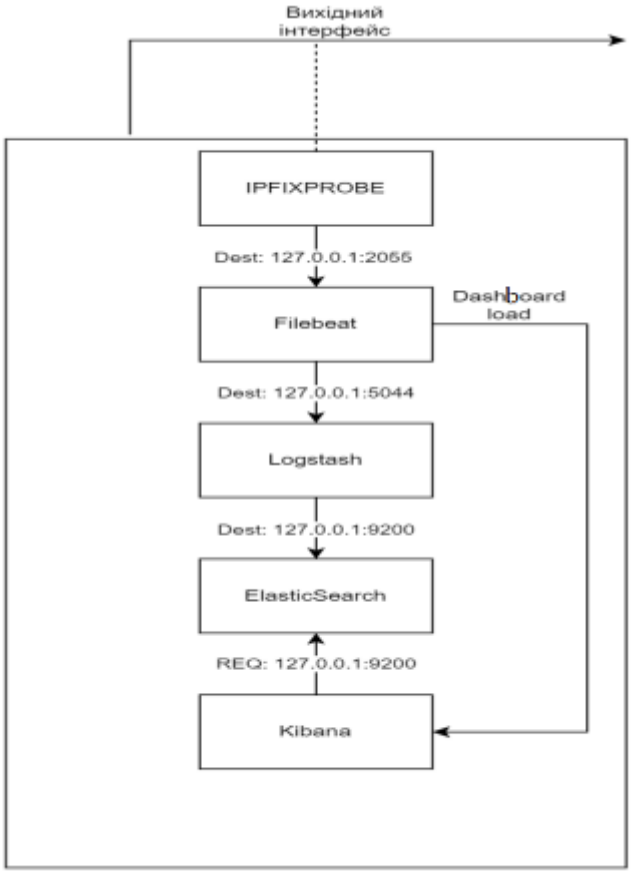

Рис. 10 Архітектура системи для тестування

3.1.1 Розгортання ES

Встановлення ElasticSearch на більшості платформ відбувається за допомогою готових пакетів. Для встановлення ElasticSearch потрібно здійснити наступні кроки:

1) Додати у систему публічний ключ репозиторію:

wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -

Додавання публічного ключа потрібно для підтвердження легальності та безпечності програмного забезпечення.

2) Додати репозиторій ElasticSearch у систему:

echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee /etc/apt/sources.list.d/elastic-7.x.list

Вказання репозиторію ElasticSearch потрібно для встановлення ПЗ за допомогою інструмента LAPT apt-get install. При виконанні команди apt-get install система управління пакетами шукає посилання на віддалений репозиторій у якому міститься потрібне ПЗ.

3) Виконати оновлення бази даних доступних пакетів та встановити ПЗ ElasticSeatch :

apt update && apt install elasticsearch

4) Запуск ПЗ та додавання його в автозапуск:

systemctl enable elasticsearch.service

systemctl start elasticsearch.service

Додавання ПЗ в автозапуск надає можливість автоматичного повторного запуску ElasticSearch при перезавантаженні операційної системи.

5) Перевірка коректного запуску та роботи ПЗ. Для перевірки коректного запуску та статусу ElasticSearch можна використати інструмент управління диспетчера системи — systemctl, що наведено на Рис. 11.

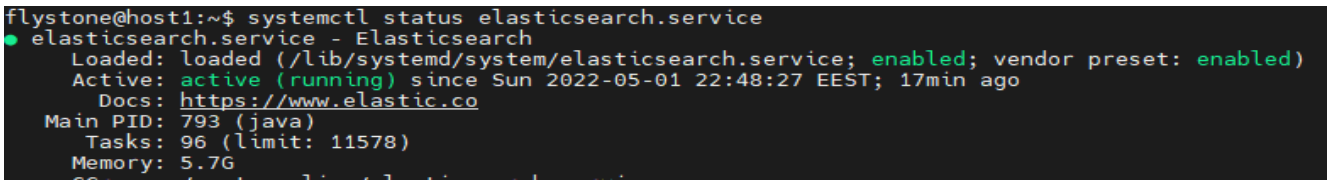

Рис. 11 Використання інструмента управління диспетчера системи

При коректному запуску ПЗ можливо перевірити його роботу за допомогою службової програми cURL, що показано на Рис. 12.

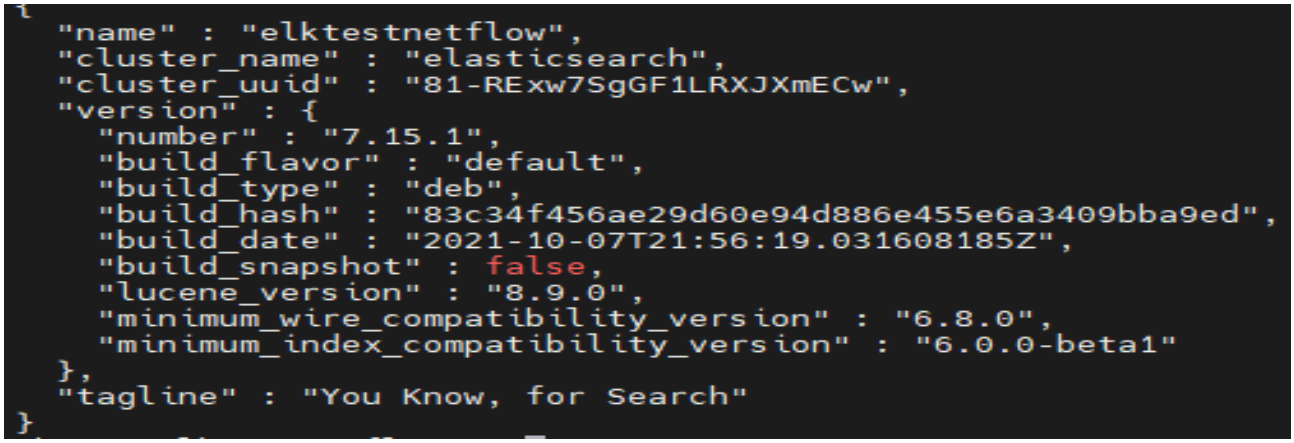

Рис.12. Перевірка роботи Elasticsearch

Після успішного запуску ElasticSearch потрібно налаштувати конфігурацію для тестування системи. Налаштування ElasticSearch розміщені у файлі elasticsearch.yml, що знаходиться у директорії /etc/elasticsearch. На даному етапі найважливішими параметрами є:

- network.host — вказання ip-адреси доступу до ElasticSearch
- discovery.seed_hosts — вказання ip-адрес вузлів ElasticSearch
- xpack.security.enable — ввімкнення авторизації

Зважаючи на локальне розміщення усіх компонентів для тестування, у параметрах network.host та discovery.seed_hosts потрібно вказати адресу 127.0.0.1. Параметр xpack.security.enabled потрібно встановити у режим true. Після застосування змін, потрібно перезавантажити ПЗ та перевірити працездатність.

Перевірка працездібності виконується за допомогою утиліти командного рядка netstat з використанням опцій t, u, l, n, p, що зображено на Рис. 13.

```
root@host1:~# netstat -tulnp | grep 9200
tcp6       0      0 127.0.0.1:9200          :::*                    LISTEN      793/java
```

Рис. 13. Перевірка працездібності ElasticSearch

Також, при ввімкненні авторизації необхідно згенерувати паролі, для кожного компонента, який взаємодіє з ElasticSearch. Генерація паролів виконується за допомогою спеціальної утиліти, яка встановлюється разом з ElasticSearch. Створенні паролі потрібно зберегти та додати у налаштування кожного компоненту. Приклад роботи утиліти генерації паролів зображено на Рис. 14.

```
root@kibanahost:~# /usr/share/elasticsearch/bin/elasticsearch-setup-passwords auto
Initiating the setup of passwords for reserved users elastic,apm_system,kibana,kibana_system,logstash_system,be
ats_system,remote_monitoring_user.
The passwords will be randomly generated and printed to the console.
Please confirm that you would like to continue [y/N]y


Changed password for user apm_system
PASSWORD apm_system = mYDKkayxYrKUdSPk6aKg

Changed password for user kibana_system
PASSWORD kibana_system = drgKsZGX2SzpbU1M8Jrw

Changed password for user kibana
PASSWORD kibana = drgKsZGX2SzpbU1M8Jrw

Changed password for user logstash_system
PASSWORD logstash_system = a6Bl0OU3qDKtzB1aUVRw

Changed password for user beats_system
PASSWORD beats_system = qPWkpzOuSug41kPT4G9G

Changed password for user remote_monitoring_user
PASSWORD remote_monitoring_user = Puai9Hmyuo62LGveG6u9

Changed password for user elastic
PASSWORD elastic = 7Vo3lMk3aEAFNesTAKIJ
```

Рис 14. Приклад роботи утиліти генерації паролів ElasticSearch

3.1.2 Розгортання Kibana

Встановлення ПЗ Kibana виконується аналогічним до встановлення ElasticSearch способом, а саме встановленням deb-пакета з репозиторію офіційного виробника. Після запуску програмного забезпечення та додавання в автозавантаження потрібно перевірити працездатність Kibana за допомогою утиліти netstat. За замовчуванням, Kibana прослуховує лише порт 5601, тому можливо відфільтрувати вивід утиліти netstat, як зображено на Рис. 14.

```
root@host1:~# netstat -tulnp | grep 5601
tcp        0      0 192.168.1.11:5601       0.0.0.0:*               LISTEN      797/node
```

Рис. 14. Перевірка працездатності Kibana

Переконавшись у працездатності ПЗ, потрібно виконати налаштування Kibana. Для початку роботи, потрібно звернути увагу на такі параметри:

- server.host — вказання ip-адреси хоста, що дозволяє отримати доступ до Kibana усій локальній мережі.
- server.publicBaseUrl — вказання доменного імені, при невикористанні такого, можливо вказати ip-адресу хоста.
- elasticsearch.username/password — задання даних, що були згенеровані утилітою ElasticSearch, для авторизації
- xpack.security.enabled — ввімкнення авторизації

Після закінчення конфігурації потрібно перезавантажити програмне забезпечення. При успішній конфігурації, можна перевірити роботу Kibana за допомогою web-інтерфейса, який доступний за сокетом: server.publicBaseUrl:5601. Первинний вигляд web-інтерфейса Kibana зображено на Рис. 15.

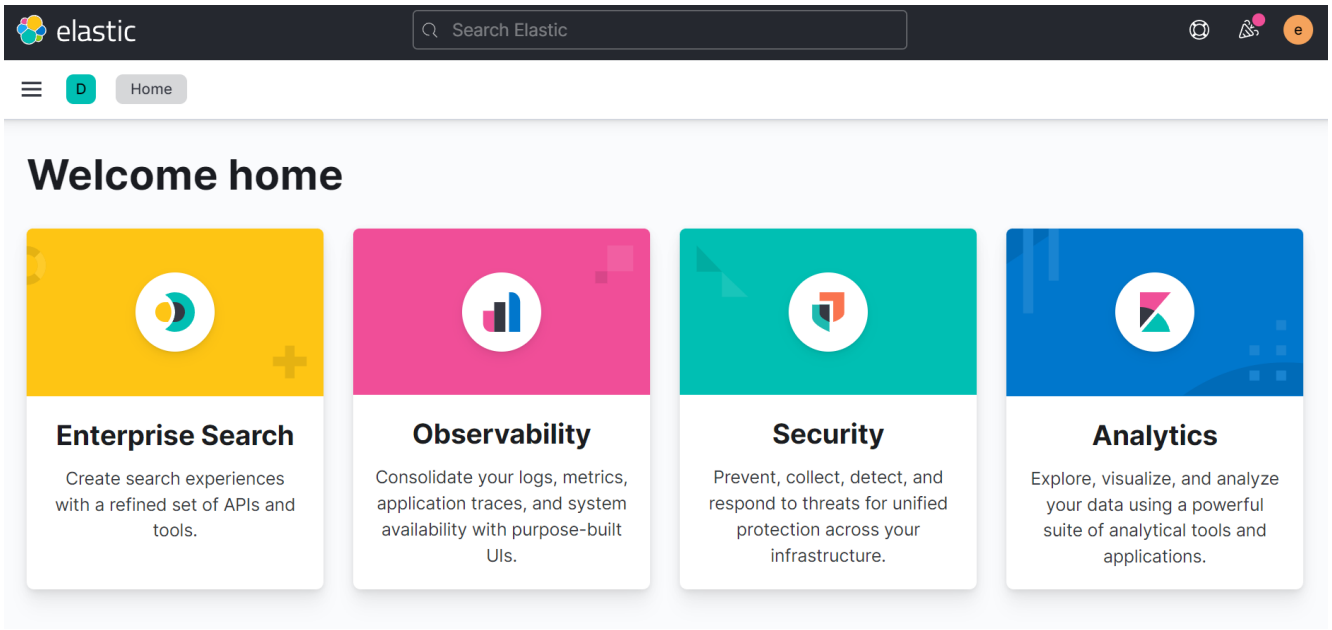

Рис. 15. Web-інтерфейс Kibana

3.1.3 Розгортання FileBeat

Встановлення ПЗ Filebeat виконується за допомогою інструмента LAPT apt-get install. Перед запуском програмного забезпечення потрібно виконати його первинну конфігурацію, а саме:

- Вказати адресу або доменне ім’я Kibana:

setup.kibana: host: "http://192.168.1.11:5601"

- Вказати тимчасову відправку напряму в ElasticSearch:

output.elasticsearch: hosts: ["localhost:9200"]

- Ввімкнути необхідний модуль Filebeat:

filebeat modules enable netflow

Після завершення первинної конфігурації, потрібно запустити Filebeat та перевірити його працездатність за допомогою інструмент управління диспетчера системи — systemctl, що зображено на Рис. 16.

```
root@host1:~# systemctl status filebeat
● filebeat.service - Filebeat sends log files to Logstash or directly to Elasticsearc
     Loaded: loaded (/lib/systemd/system/filebeat.service; enabled; vendor preset: en
     Active: active (running) since Mon 2022-05-02 22:14:07 EEST; 13h ago
       Docs: https://www.elastic.co/beats/filebeat
   Main PID: 753 (filebeat)
      Tasks: 11 (limit: 11578)
     Memory: 125.5M
```

Рис. 16. Перевірка працездатності Filebeat

Також, при вказанні адреси або доменного імені Kibana, Filebeat автоматично завантажує стандартні панелі усіх доступних модулів, які можна переглянути у web-інтерфейсі Kibana. Приклад доступних панелей зображено на Рис. 17.

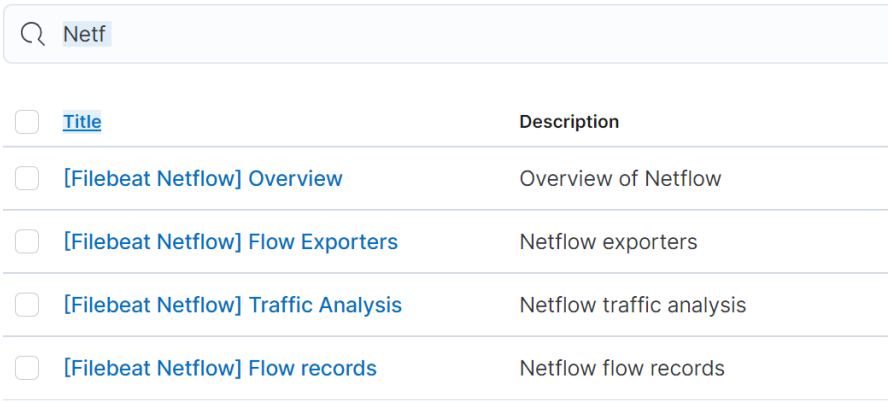

Рис. 17 Приклад доступних панелей

3.1.4 Розгортання Logstash

Встановлення ПЗ Logstash виконується за допомогою інструмента LAPT apt-get install. Перед запуском ПЗ потрібно додати його в автозавантаження за допомогою команди systemctl enable logstash.service. Для початку роботи з Logstash потрібно описати три конфігурації, а саме: конфігурацію “input”, яка відповідає за отримання даних з Filebeat, “filter” — опціональна конфігурація обробки даних, та “output”, яка описує відправку даних в ElasticSearch.

Конфігурація “input” має містити у собі вказання на джерело даних та порт, на який будуть надходити дані. Зважаючи на використання Filbeat з модулем netflow у ролі колектора, конфігурація “input” набуває такого вигляду:

```
input {
    beats {
        port => 5044 }}
```

Конфігурація “output” містить дані, які необхідні для подальшої відправки інформації. При використанні ElasticSearch у ролі сховища даних, потрібно вказати дані для авторизації, індекс даних, адресу вузла. Для тестової системи конфігурація буде мати такий вигляд:

```
output {
   elasticsearch {
       user => "logstash_netflow"
       password => "qwerty"
        hosts    => "127.0.0.1:9200"
       index  => "filebeat-7.17.2-2022.04.03-000001"
}}
```

3.2 Розгортання серверної частини

3.2.1 Розгортання у середовищі unix-подібних систем

Для можливості моніторингу пристроїв з unix-подібною операційною системою потрібно використати ПЗ, що емулює сенсор IPFIX. Стандартним представником такого ПЗ є ipfixprobe.

Встановлення ipfixprobe виконується за допомогою клонування git-репозиторію розробника та компіляції ПЗ на локальній машині.

Для успішної компіляції ПЗ потрібно завантажити додаткові бібліотеки, а саме: libatomic та libpcap.

Клонування репозиторію виконується за допомогою команди git clone. Після успішного клонування репозиторію, потрібно скомпілювати ПЗ, а саме:

1) Перейти у директорію репозиторію:

cd ipfixprobe

2) Побудувати конфігураційний скрипт за допомогою утиліти autoreconf:

autoreconf -I

3) Виконати файл configure:

./configure

4) Виконати компіляцію за допомогою утиліти make:

make

5) Скопіювати скомпільовані файли у місце призначення:

make install

Виконавши компіляцію, потрібно запустити ПЗ з обраними опціями, а саме:

1) --input — вказання інтерфейсу моніторингу

2) --storage — зміна параметрів передачі ip-потоків

3) --output — вказання протоколу передачі, адреси та порта отримувача.

У тестовій системі, об’єктом моніторингу вважається інтерфейс eth0, що з’єднаний за допомогою віртуального комутатора з мережею. Оскільки, колектор IPFIX встановлений локально, та за замовчуванням використовує протокол udp та

порт 2055. Повна команда запуску ПЗ зображена на рис 18.

```
root@host1:~/ipfixprobe# ipfixprobe -i 'raw;ifc=eth0' -s 'cache;split' -o 'ipfix;u;host=127.0.0.1;
port=2055'
```

Рис.18 Команда запуску ПЗ ipfixprobe

При вимиканні ПЗ надається стисла інформація про роботу ipfixprobe, що зображена на Рис. 19.

```
^CInput stats:
  #   packets    parsed          bytes   dropped     qtime status
  0        29        29           5291         0         0     ok
Output stats:
  #   biflows   packets          bytes   dropped status
  0         5        29           4885         0     ok
```

Рис.19. Інформація про роботу ipfixprobe

3.3 Тестування системи

Після завершення розгортання клієнтської та серверної частин системи можливо розпочати тестування працездатності системи при довільних умовах та при щільному потоку трафіку.

Первинне тестування виконується за допомогою перевірки отримання даних у сховище. За допомогою web-інтерфейса Kibana можливо переглянути будь-які дані, що надходили у ElasticSearch у різні проміжки часу. Для цього потрібно перейти у вкладку Discover, посилання на яку знаходиться у боковому меню Kibana. Приклад отримання даних зображено на Рис.20.

Рис. 20. Перегляд даних, що надійшли у сховище.

Для наступного тестування потрібно перейти у вкладку Dashboard та обрати графічне відображення отриманих даних. Оскільки у тестовій системі використовується лише протокол IPFIX, потрібно обрати будь-яку панель модуля netflow, перелік яких зображений на Рис. 21.

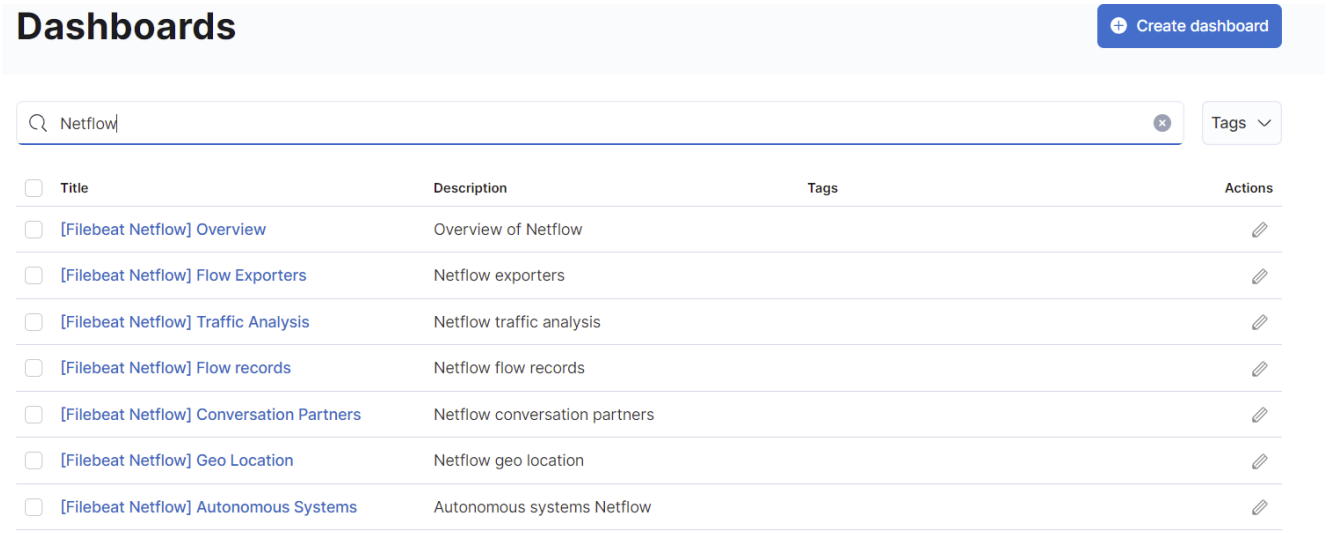

Рис. 21. Перелік панелей netflow

Обравши панель, потрібно звернути увагу на дані, що візуалізовані у різних діаграмах та гістограмах. Приклад функціонуючої панелі зображено на Рис. 22.

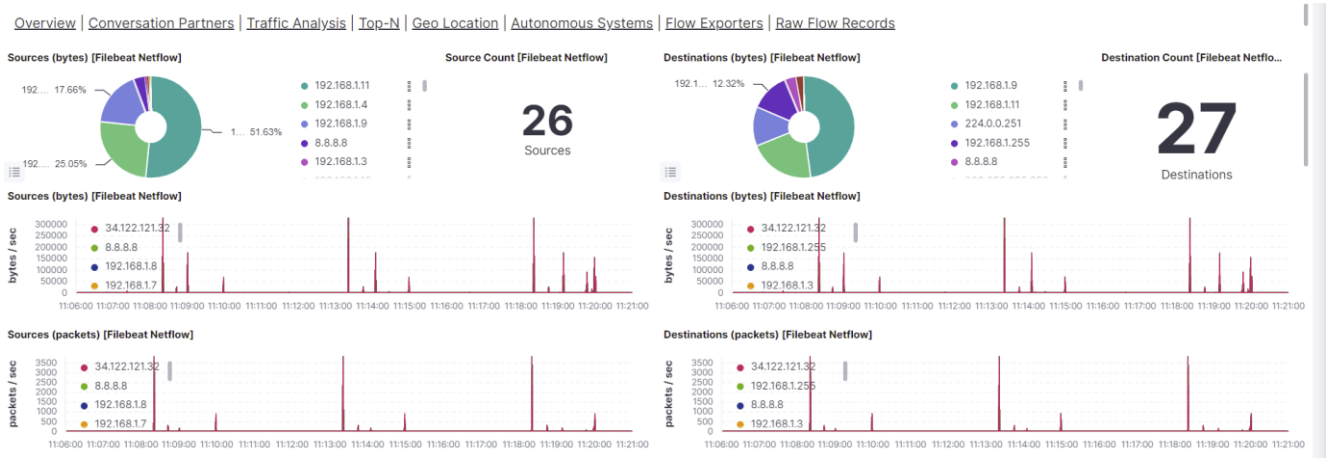

Рис. 22. Панель Traffic Analysis модуля netflow

При відсутності візуалізації даних, потрібно звернути увагу на індекси, які вказані для кожного джерела. Стандартні панелі, які імпортуються з filebeat, мають прив’язку до індексу filebeat-* та візуалізують лише дані, що позначені ним.

Для вирішення цієї проблеми, потрібно вказати необхідний індекс у конфігурації “output” ПЗ Logstash або змінити прив’язку панелей до індексу.

Після загального тестування системи потрібно перевірити можливість роботи системи при щільному потоку трафіку. Для генерування потоків трафіку можливо використати ПЗ trafgen.

Trafgen — відкрите програмне забезпечення, що генерує запити до випадкових адрес. Для тестування системи, було використано конфігурацію, що вказана у Додатку А. Приклад використання trafgen зображено на Рис. 23.

```
root@host1:~# trafgen --cpp --dev eth0 --conf udp.trafgen
     7 packets to schedule
   420 bytes in total
Running! Hang up with ^C!

^C
    45094059 packets outgoing
  2705643540 bytes outgoing
          65 sec, 970412 usec on CPU0 (6300180 packets)
          65 sec, 971146 usec on CPU1 (5994663 packets)
          65 sec, 978093 usec on CPU2 (6445348 packets)
          65 sec, 970428 usec on CPU3 (6931024 packets)
          65 sec, 979384 usec on CPU4 (6517642 packets)
          65 sec, 970383 usec on CPU5 (6317173 packets)
          65 sec, 967316 usec on CPU6 (6588029 packets)
```

Рис. 23 Використання trafgen

Для перевірки системи було виконано годинний тест за допомогою ПЗ trafgen. Визначення кількості трафіку та швидкості передачі було виконано за допомогою ПЗ slurm, приклад роботи якого, зображено на Рис. 24.

```
    Active Interface: eth0                    Interface Speed: unknown

    Current RX Speed: 0.53 KB/s             Current TX Speed: 37270.75 KB/s
  Graph Top RX Speed: 3.60 KB/s           Graph Top TX Speed: 49123.84 KB/s
Overall Top RX Speed: 9.58 KB/s         Overall Top TX Speed: 49123.84 KB/s
    Received Packets: 155272              Transmitted Packets: 1692217181
     GBytes Received: 0.058 GB              GBytes Transmitted: 94.573 GB
 Errors on Receiving: 0                 Errors on Transmission: 0
```

Рис. 24. Приклад роботи slurm під час тестування системи.

Перед початком тесту потрібно дізнатись початкове значення кількості та швидкості трафіку, що зображено на Рис.25.

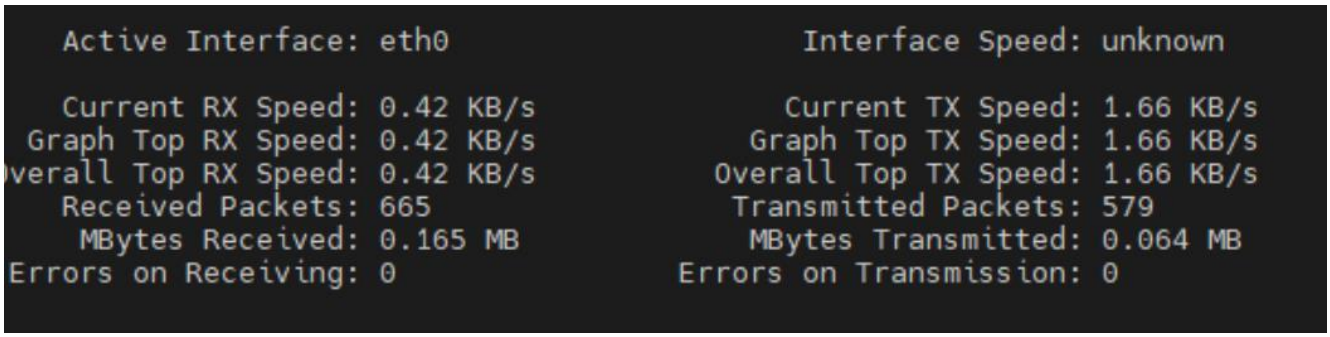

Рис. 25. Кількість та швидкість трафіку перед початком тесту

Оскільки відомі початкові значення та стан системи, можливо починати тестування, а саме:

1) Запустити програмне забезпечення ipfixprobe:

ipfixprobe -i 'raw;ifc=eth0' -s 'cache;split' -o 'ipfix;u;host=127.0.0.1;port=2055'

2) Запустити програмне забезпечення trafgen:

trafgen --cpp --dev eth0 --conf udp.trafgen

Після закінчення часу тестування, потрібно дізнатись значення кількості трафіку та середню швидкість передачі даних. Для виконання цього завдання потрібно скористатись ПЗ slurm, що надасть детальну інформацію про інтерфейс. На Рис. 26 зображена інформація про стан інтерфейсу під час тестування.

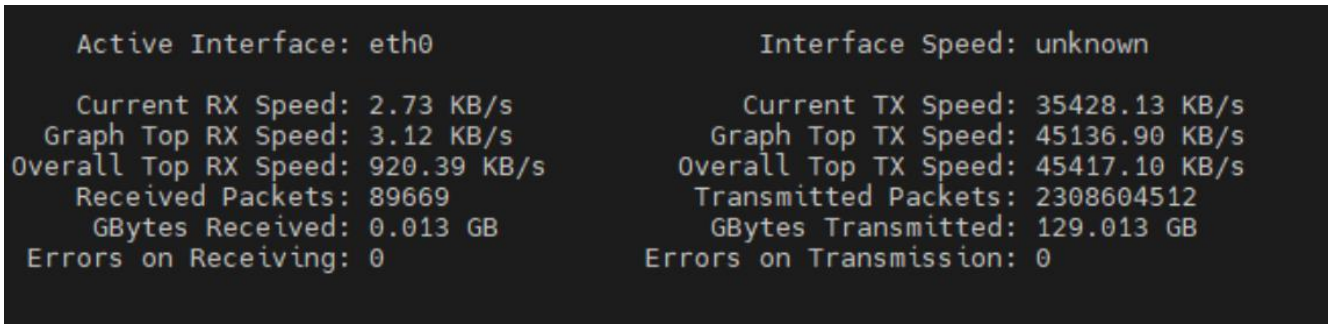

Рис. 26. Кількість та швидкість трафіку під час проведення тесту

Для перевірки результатів тестування, а саме можливі збої у роботі системи моніторингу потрібно скористатись web-інтерфейсом Kibana. У вкладці Discover

відображенні усі дані, що надходили у сховище та гістограма кількості пакетів у дискретний момент часу. На Рис. 27 зображена гістограма кількості пакетів, що були надіслані у сховище під час тестування.

Рис. 27. Гістограма кількості пакетів

За допомогою гістограми можливо виявити несправності системи, а саме різка зміна кількості пакетів сигналізує про збої у роботі колектора або виникнення вузького місця між колектором та фільтром даних. При виявленні різкої зміни кількості пакетів потрібно проаналізувати роботу колектора та фільтра даних за допомогою логів.

Зважаючи на результати тестування, можна виділити основні тези:

1) Система може повноцінно працювати незважаючи на щільність трафіку.

2) Під час тестування не вдалось досягнути вузького місця або спровокувати збої програмного забезпечення.

Також, протягом часу тестування було згенеровано достатню кількість трафіку для повноцінного огляду візуалізації та аналізу розміру даних, що надійшли у сховище.

Загальна інформація про трафік відображається на панелі NetFlow Overview, приклад якої зображено на Рис. 28. Зважаючи на розміщення тестової системи та використання гіпервізора, пакети IPFIX не містять певних даних, а саме : VLANID, AS, тип сервісу та географічне розташування об’єктів моніторингу.

Незважаючи на неповноцінність даних, панель NetFlow Overview надає детальну інформацію про відправників та отримувачів, а саме їх адреси.

Рис. 28. Панель NetFlow Overview

Детальну інформацію про трафік наведено у панелі NetFlow Traffic Analysis, яку зображено на Рис. 29. За допомогою даної панелі можливо детально проаналізувати трафік та кількість користувачів.

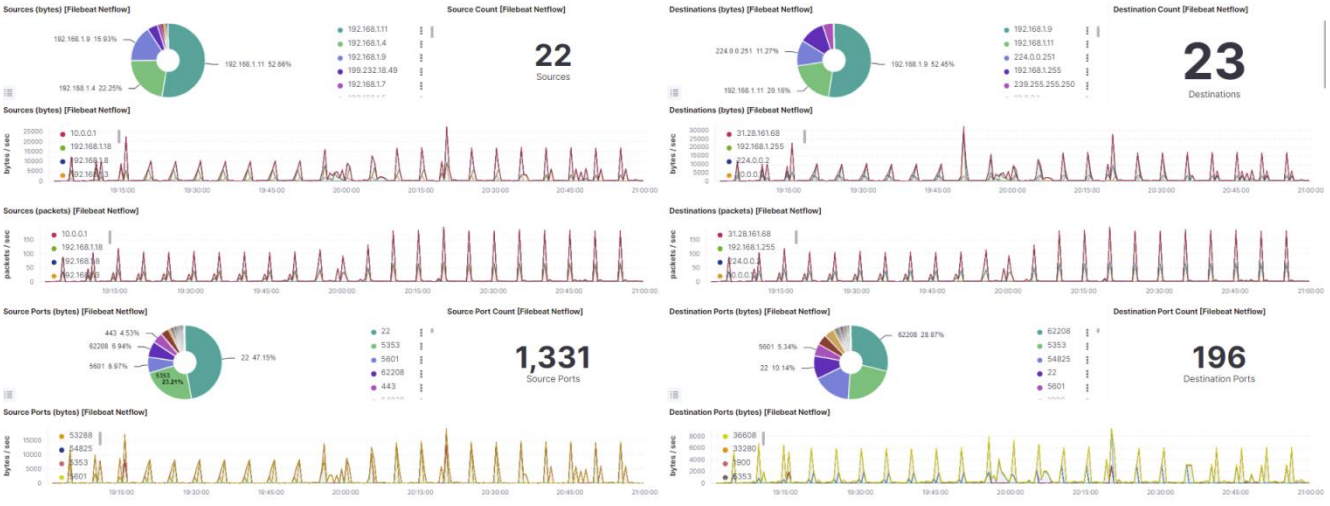

Рис. 29. Панель NetFlow Traffic Analysis

ВИСНОВКИ

У ході виконання кваліфікаційної роботи було проведено аналітичний огляд існуючої системи моніторингу трафіку студміста КНУ. Результат огляду показав неспроможність системи забезпечити контроль та глибокий аналіз трафіку, виявлення мережевих атак, адміністрування трафіку та моніторингу якості сервісу. Опираючись на результати аналізу та стандартизовані умови до системи моніторингу трафіку, було сформовано вимоги до архітектури програмного комплексу.

Згідно сформованих вимог було спроєктовано архітектуру системи, та виконано огляд інструментів і програмного забезпечення. Спроєктована архітектура є вискодоступною, дозволяє виконувати машабування системи, позбавленна єдиної точки відмови та залежності компонентів.

Зважаючи на розроблену архітектуру та обрані технології, було реалізовано прототип системи моніторингу, який продемонстрував можливості до глубокого аналізу трафіку в реальному часі. Завдяки використанню загальновідомих технологій, система може доповнюватись додатковими функціями, які розширять її функціонал.При необхідності, система здатна повноцінно функціонувати як єдина система моніторингу мережевої інфраструктури студміста КНУ. Також, було виконано тестування системи при різних умовах, результати якиз довели працездатність системи та можливість її використання для моніторингу мережі струдміста КНУ.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Splunk [Електронний ресурс] — Режим доступу: https://www.splunk.com/en_us/data-insider/what-is-it-monitoring.html

2. X. Wang, L. Wang, B. Yu, and G. Dong, “Studies on Network Management System framework of Campus Network,” presented at 2nd International Asia Conference on Informatics in Control, Automation and Robotics (CAR), 2010, Yantai, China.

3. RFC 5101 [Електронний ресурс] — Режим доступу: https://datatracker.ietf.org/doc/html/rfc5101

4. RFC 3917 [Електронний ресурс] — Режим доступу: https://datatracker.ietf.org/doc/html/rfc3917

5. RFC 5102 [Електронний ресурс] — Режим доступу: https://datatracker.ietf.org/doc/html/rfc5102

6. Elastic Guide Filebeat [Електронний ресурс] — Режим доступу: https://www.elastic.co/guide/en/beats/filebeat/current/filebeat-overview.html

7. James Hamilton, Brad Schofield, Manuel Gonzalez Berges, Jean-Charles Tournier, “SCADA STATISTICS MONITORING USING THE Elastic Stack” 2017, Barcelona, Spain.

8. Elastic Guide ES [Електронний ресурс] — Режим доступу: https://www.elastic.co/guide/en/elasticsearch/reference/current/elasticsearch-intro.html

9. Elastic BenchMarks [Електронний ресурс] — Режим доступу: https://elasticsearch-benchmarks.elastic.co/

10. Haroon Shakirat Oluwatosin, “Client-Server Model”