

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

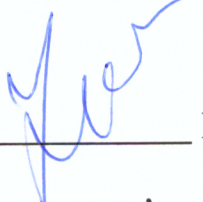
ПОЯСНЮВАЛЬНА ЗАПИСКА

до випускної кваліфікаційної роботи магістра
на тему: «Методи та технології збору, аналізу та
візуалізації даних на основі Elastic Stack в
корпоративній мережі»

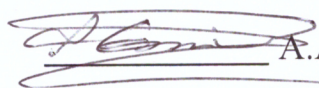
Виконав студент 2-го курсу групи КБм-23-1
спеціальності 125 «Кібербезпека та захист інформації»

 I.V. Кушнір

Керівник старший викладач кафедри комп'ютерної
інженерії та кібербезпеки

 I.G. Фальковський

Рецензент завідувач кафедри комп'ютерної інженерії та
кібербезпеки, кандидат технічних наук, доцент

 A.A. Єфіменко

Житомир – 2024

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

ЗАТВЕРДЖУЮ

Завідувач

кафедри комп'ютерної
інженерії та кібербезпеки

 Андрій Єфіменко
17 вересня 2024 р.

ЗАВДАННЯ

на випускню кваліфікаційну роботу магістра

Студент: Кушнір Іван Вікторович

Тема роботи: **«Методи та технології збору, аналізу та візуалізації даних на основі Elastic Stack в корпоративній мережі»**

затверджена Наказом університету від 17 вересня 2024 р. № 479/с

Термін здачі студентом закінченої роботи 18 грудня 2024 р.

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

1. Топологія локальної корпоративної мережі;
2. Проект локальної корпоративної мережі;
3. Використання рішення Elastic Stack.

Обладнання та програмне забезпечення:

1. ОС Windows Server 2019 Standard (v.1809);
2. ОС Windows 10 (v.1909);
3. ОС Ubuntu (v.22.04.3);
4. Програмне забезпечення Elastic Stack: Elasticsearch (v.8.15.3),

Kibana (v.8.15.3);

5. Мережевий емулятор GNS3 (v.2.2.49);
6. Гіпервізор Oracle VirtualBox (v.7.1.4).

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

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1	В.В. Воротніков	17.09.2024	17.09.2024
2	В.В. Воротніков	10.10.2024	10.10.2024
3	В.В. Воротніков	12.11.2024	12.11.2024

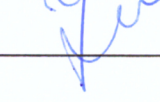
Студент  І.В. Кушнір

Керівник  І.Г. Фальковський

Календарний план

№ з/п	Назви етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка завдання	17.09.2024 – 23.09.2024	Виконано
2	Розробка плану роботи	24.09.2024 – 30.09.2024	Виконано
3	Аналіз теоретичних аспектів призначення лог-файлів	01.10.2024 – 07.10.2024	Виконано
4	Дослідження можливостей використання централізованих систем збору та аналізу лог-файлів	08.10.2024 – 14.10.2024	Виконано
5	Порівняння відомих безкоштовних рішень з відкритим вихідним кодом	15.10.2024 – 21.10.2024	Виконано
6	Розробка топології корпоративної мережі та схеми адресації	22.10.2024 – 28.10.2024	Виконано
7	Розробка системи збору та обробки логів в мережі	29.10.2024 – 04.11.2024	Виконано
8	Вибір обладнання	05.11.2024 – 11.11.2024	Виконано
9	Налаштування пристроїв мережі	12.11.2024 – 18.11.2024	Виконано
10	Налагодження роботи системи збору та аналізу лог-файлів	19.11.2024 – 25.11.2024	Виконано
12	Тестування функціонування мережі	26.11.2024 – 02.12.2024	Виконано
13	Написання пояснювальної записки	03.12.2024 – 13.12.2024	Виконано
14	Захист	18.12.2024	Виконано

Студент  І.В. Кушнір

Керівник  І.Г. Фальковський

АНОТАЦІЯ

Випускна кваліфікаційна робота магістра складається з пояснювальної записки, що містить вступ, три розділи, висновки, список використаної літератури та додатки. Текстова частина викладена на 81 сторінці друкованого тексту. Пояснювальна записка має 13 сторінок додатків. Список використаних джерел містить 21 найменування і займає 2 сторінки. В роботі наведено 54 рисунки та 3 таблиці. Загальний обсяг роботи – 94 сторінки.

Метою кваліфікаційної роботи є дослідження та практичне використання методів та технологій збору, аналізу та візуалізації даних на основі Elastic Stack в корпоративній мережі.

В рамках роботи було розглянуто теоретичні відомості стосовно лог файлів та роботи централізованих систем аналізу та збору. Окрему увагу приділено функціональним можливостям та архітектурі Elastic Stack. Виконано розробку топології локальної корпоративної мережі. Наведено сценарій налагодження пристроїв мережі та рішення Elastic Stack. Проведено тестування функціонування роботи мережі та методів збору та обробки логів в Elastic Stack.

КЛЮЧОВІ СЛОВА: ЛОГИ, БЕЗПЕКА, ELASTIC STACK, МЕРЕЖА, ПРИСТРОЇ, АГЕНТИ, МОНІТОРИНГ, МЕТРИКИ.

ANNOTATION

The final qualification work of the master consists of an explanatory note, including an introduction, three chapters, conclusions, a list of references, and appendices. The test part is set out on 81 pages of printed text. The explanatory note includes 13 pages of appendices. The list of references contains 22 sources and occupies 2 pages. The work contains 54 figures and 3 tables. The total volume of the work is 94 pages.

The purpose of the qualification thesis is to study and practically apply methods and technologies for collecting, analyzing, and visualizing data based on Elastic Stack in a corporate network.

The work examines theoretical information regarding log files and work of centralized analysis and collection systems. Particular attention is given to the functional capabilities and architecture of Elastic Stack. The development of a topology for a local corporate network was completed. A setup scenario for network devices and Elastic Stack solutions is provided. Testing of network functionality and methods for collecting and processing logs in Elastic Stack was conducted.

KEYWORDS: LOGS, SECURITY, ELASTIC STACK, NETWORK, DEVICES, AGENTS, MONITORING, METRICS.

ЗМІСТ

ВСТУП.....	8
РОЗДІЛ 1 ТЕОРЕТИЧНІ АСПЕКТИ ПОБУДОВИ ТА ФУНКЦІОНУВАННЯ ЦЕНТРАЛІЗОВАНИХ СИСТЕМ АНАЛІЗУ ЖУРНАЛІВ	10
1.1 Теоретичні відомості про лог файли.....	10
1.2 Використання централізованих систем аналізу журналів.....	16
1.3. Огляд безкоштовних рішень централізованого збору і обробки логів з відкритим вихідним кодом	23
1.4 Теоретичні відомості стосовно Elastic Stack.....	30
РОЗДІЛ 2 ПРОЄКТУВАННЯ СИСТЕМИ ЗБОРУ, АНАЛІЗУ ТА ВІЗУАЛІЗАЦІЇ ДАНИХ В КОРПОРАТИВНІЙ МЕРЕЖІ	37
2.1 Призначення корпоративних мереж	37
2.2 Проєктування топології мережі та схеми адресації	38
2.3 Розробка системи збору та обробки логів в Elastic Stack	44
2.4 Вибір обладнання та програмного забезпечення	47
РОЗДІЛ 3 ПОБУДОВА, НАЛАГОДЖЕННЯ ТА ТЕСТУВАННЯ РОБОТИ КОРПОРАТИВНОЇ МЕРЕЖІ	52
3.1 Проєктування і налаштування роботи проєкту в GNS3	52
3.2 Налаштування Elastic Stack і агентів	61
3.3 Тестування роботи мережі та Elastic Stack.....	69
ВИСНОВКИ.....	78
СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ	80
ДОДАТКИ.....	82

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

IoT – Internet of Thing

ОС – Операційна система

IDPS – Intrusion Detection and Prevention Systems

DLP – Data Leak Prevention

RADIUS – Remote Authentication Dial-In User Service

TACACS – Terminal Access Controller Access Control System

IP – Internet Protocol

JSON – JavaScript Object Notation

XML – EXtensible Markup Language

CSV – Comma-Separated Values)

ELK – Elasticsearch, Logstash и Kibana

ES|QL – Elasticsearch Query Language

AWS – Amazon Web Services

AD – Active Directory

LAN – Local Area Network

MAN – Metropolitan Area Network, MAN

WAN – Wide Area Network

NAT – Network Address Translation

PAT – Port Address Translation

ВСТУП

Актуальність. Сучасні корпоративні мережі стикаються з низкою викликів, які ускладнюють їхнє функціонування та управління. По мірі зростання кількості пристроїв у мережі, збільшується її складність: мережевий трафік, кількість IP-адрес та потреба в управлінні з'єднаннями між різними локаціями. Така інфраструктура ускладнює діагностику та вирішення проблем, що може займати значний час. Додатково, корпоративні мережі частою є цілями для кібератак, особливо з огляду на зростання віддаленої роботи та впровадження технологій Інтернету речей (IoT). Так згідно з даними IBM Cost of a Data Breach Report, середня глобальна вартість витоку даних у 2023 році становила 4,45 мільйона доларів. Без належного моніторингу й управління мережею організації ризикують не тільки продуктивністю, але й безпекою.

Для вирішення цієї проблеми організації вимушені вкладати кошти у надійні інструменти для моніторингу та залучати кваліфікованих фахівців для забезпечення ефективної роботи мережі. Однак, такі рішення та спеціалісти можуть бути доволі дорогими, що стає викликом для компаній з обмеженими ресурсами.

Ці виклики потребують застосування універсальних та безкоштовних рішень для збору, аналізу та візуалізації даних, що дозволяють ефективно контролювати роботу мережі та забезпечувати її захист. Одним з таких рішень є Elastic Stack — платформа з відкритим кодом, яка об'єднує набір інструментів з метою запровадження централізованого збору даних, їх аналізу і візуалізації в реальному часі.

Метою кваліфікаційної роботи є проєктування та налаштування системи збору, аналізу та візуалізації даних з використанням стеку програми Elastic в корпоративній мережі.

Мета обумовлює виконання наступних завдань:

- визначення призначення і джерел логування;
- аналіз сучасних методів збору, аналізу та візуалізації даних;

- огляд і порівняння безкоштовних рішень для централізованої організації системи збору;
- визначення функціональних можливостей Elastic Stack;
- розробка топології корпоративної мережі і схеми адресації;
- проектування системи збору, аналізу і візуалізації логів;
- вибір обладнання, налаштування та тестування роботи мережі та централізованої системи збору та аналізу логів з використанням Elastic Stack.

Об'єктом дослідження є корпоративна мережа та система централізованого збору, аналізу та візуалізації журналів з використанням Elastic Stack.

Предметом дослідження є процес аналізу і вибору централізованої системи збору логів, проектування корпоративної мережі, налаштування мережевих і кінцевих пристроїв, впровадження та налаштування системи збору і обробки журналів з використанням Elastic Stack в IT-інфраструктурі.

Результатом виконання кваліфікаційної роботи є налаштована централізована система збору і обробки даних з використанням Elastic Stack в корпоративній мережі організації. Працездатність проєкту і системи було перевірено та протестовано.

РОЗДІЛ 1 ТЕОРЕТИЧНІ АСПЕКТИ ПОБУДОВИ ТА ФУНКЦІОНУВАННЯ ЦЕНТРАЛІЗОВАНИХ СИСТЕМ АНАЛІЗУ ЖУРНАЛІВ

1.1 Теоретичні відомості про лог файли

Лог (лог-файл або журнал) – це текстовий файл, куди автоматично в хронологічному порядку записується важлива інформація про роботу системи чи програми [1]. Спочатку логи використовувалися в основному для виявлення і вирішення проблем на пристроях чи в програмах, але тепер вони виконують додаткові функції пов'язані з оптимізацією роботи системи та мережі, відслідковування і аналізу роботи пристроїв, розслідування зловмисних дій. Подібні файли можуть зберігатися на комп'ютері, на окремому сервері, в хмарному сховищі, на знімних носіях, тощо.

В лог-файли може записуватись різна інформація в залежності від завдань систем або програми [2]. До деяких популярних видів журналів можна віднести:

1) Доступу. Містить інформацію про події безпеки, як ось успішні та невдалі спроби автентифікації, доступу до файлів, змін політики безпеки, зміни облікового запису (створення та видалення облікового запису, призначення привілеїв облікового запису) і використання привілеїв.

2) Системні. Даний журнал містить інформацію про події, які були здійснені компонентами ОС, як завершення роботи системи або запуск певної служби. Як правило, лише невдалі події та найважливіші успішні події записуються в лог-файлі, але зазвичай можливості ОС дозволяють адміністраторам вказувати, які типи подій реєструватимуться. Дані про кожну подію зазвичай мають мітку часу, а інша допоміжна інформація може включати коди подій, статусів і помилок; назву служби; і обліковий запис користувача або системи, пов'язаний із подією.

3) Сервера. Даний файл містить список дій, які виконав сервер. До цього може відноситись детальна інформація про те коли та хто надсилав запит до сервера або програми на отримання доступу до певного файлу.

4) Подій. Цей журнал реєструє дані про використання і діяльність операційної системи, додатків або пристроїв.

5) Змін. Веде хронологічний запис виконаних змін в програмному забезпеченні. Для прикладу тут можуть містити інформація про відмінності між конфігураціями системи за певний проміжок часу або версії додатків.

6) Використання ресурсів. Міститься інформація стосовно значень використання оперативної пам'яті, процесорного часу, зайнятого обсягу на носії, швидкість передачі даних.

7) Безпеки. Даний тип журналу генерують пристрої і програми пов'язані із забезпеченням безпеки мережі. До них відносяться: система виявлення і запобігання вторгнень (IDPS – Intrusion Detection and Prevention Systems), система безпеки та керування подіями (SIEM – Security Information and Event Management), брандмауер, антивірус, веб-проксі, мережеві пристрої, система запобігання витоків інформації (DLP – Data Leak Prevention), сервери аутентифікації (Radius, TACACS), тощо.

Мета здійснення логування

Здійснюють регулярне зберігання та аналіз лог-файлів задля виконання наступних цілей [3]:

1) Моніторинг. Журнали дозволяють збирати дані про використання ресурсів, дії користувачів або виникнення подій у реальному часі. Це допомагає адміністраторам систем відслідковувати стан системи та вчасно реагувати на проблеми або події.

2) Аудит. Відслідковування дій користувачів у системі або збереження слідів змін важливих даних дозволяє встановлювати відповідальність за виникнення інцидентів безпеки і виявляти випадки несанкціонованого доступу або зловживання доступом до системи.

3) Розуміння поведінки користувачів. Збір та аналіз логів може надати цінну інформацію про те, як користувачі взаємодіють з системою, що може бути використано для покращення її функціональності або ефективності.

4) Аналіз даних. Інформація з журналів веб-серверів дозволяє отримувати дані про обсяг трафіку, час відвідування веб-сторінок, географію відвідувачів та іншу важливу інформацію, яка може бути використана для складання звітів.

5) Забезпечення безпеки. Інформація з журналів допомагають виявляти потенційні загрози безпеці, а також проводити розслідування випадків порушення безпеки. Вони можуть містити інформацію про спроби несанкціонованого доступу, аномальні дії, виявляти вразливості систем. Окремо слід згадати і про системи, які і націлені на протидію зловмисним діям (IDPS, антивіруси, фаєрволи і т.д.).

6) Юридичні вимоги. Низка законів, нормативних актів і стандартів додатково зобов'язують організації зберігати та переглядати певні журнали. В додатку А наведено перелік найбільш відомих нормативних актів і стандартів, які визначають зобов'язання організацій щодо зберігання і аналізу журналів.

Проблеми в управлінні журналами

Більшість організацій стикаються з труднощами, пов'язаними з зі збором і аналізом журналів, які зводяться до однієї основної проблеми: ефективне балансування обмеженої кількості ресурсів роботи з даними з постійно зростаючою кількістю самих журналів. Нижче наведено перелік основних проблем з якими стикаються під час роботи з логами:

– Велика кількість джерел інформації. Журнали розміщені на багатьох пристроях по всій мережі організації. Крім того, одне джерело може генерувати кілька видів журналів, наприклад, програма може зберігати спроби аутентифікації в одному файлі та мережеву активність в іншому.

– Розбіжність формату записів. Кожне джерело записує певну інформацію у своїх записах журналу, як-от IP-адреса та ім'я користувача. Для ефективності джерела журналів часто записують лише ті частини інформації, які вважають найважливішими, але це може ускладнити зв'язування подій, записаних різними програмами, оскільки вони можуть не мати спільних записаних значень (наприклад, перше джерело записує IP-адресу, але не ім'я користувача, а

джерело 2 записує ім'я користувача, але не IP-адресу). Кожен тип джерела журналу також може представляти значення по-різному. Подібні відмінності можуть бути незначними, наприклад одна дата буде записана у форматі «день-місяць-рік», а інша – у форматі «місяць-день-рік». Все це ускладнює процес зв'язування подій, записаних різними джерелами журналу.

– Різні формати зберігання подій. Більшість застосунків використовують різні формати для збереження логів, як текстові, JSON (JavaScript Object Notation), XML (EXtensible Markup Language), CSV (Comma-Separated Values) і бінарні файли. Деякі журнали призначені для читання людьми, а інші – ні; деякі журнали використовують стандартні формати, тоді як інші використовують власні формати. Деякі журнали створюються не для локального зберігання у файлі, а для передачі в іншу систему для обробки; поширеним прикладом цього є перехоплення SNMP. Для деяких форматів виводу, зокрема текстових файлів, є багато можливостей для послідовності значень у кожному записі журналу та роздільників між значеннями (наприклад, значення, розділені комами, значення, розділені табуляцією, XML).

Щоб полегшити аналіз журналів, організаціям часто доводиться впроваджувати автоматизовані методи перетворення журналів із різним вмістом і форматами в єдиний стандартний формат із узгодженим представленням полів даних. Неузгоджені формати журналів і представлення полів даних також створюють проблеми для людей, які переглядають журнали, яким необхідно розуміти значення різних полів даних у кожному журналі, щоб виконати ретельний перегляд. Більшість хостів в організації зазвичай реєструють інформацію пов'язану з безпекою комп'ютера, часто з кількома журналами на хост, через це сумарна кількість журналів, які відслідковуються може бути досить великою. Багато журналів записують великі обсяги даних щодня, тому загальний щоденний обсяг даних журналу в організації часто є величезним, а розподілений характер журналів, непослідовні формати журналів і обсяги журналів ускладнюють керування створенням і зберіганням журналів.

Проблема забезпечення захисту

Лог-файли часто містять записи про події безпеки систем та мережі. Подібна інформація може бути цінною, тому виникає питання забезпечення її конфіденційності та цілісності. Для прикладу, журнали можуть навмисно чи ненавмисно фіксувати приватну інформацію, таку як паролі користувачів і вміст електронних листів. Це викликає занепокоєння щодо безпеки та конфіденційності, як для спеціалістів, які аналізують журнали, так і для інших осіб, які можуть отримати доступ до журналів авторизованими чи неавторизованими засобами. Журнали, які недостатньо захищені під час зберігання або передачі, також можуть бути вразливими до навмисних та ненавмисних змін. Це може призвести до того, що зловмисна діяльність залишиться непоміченою через навмисне видалення відповідних лог-файлів. Наприклад, багато руткітів спеціально призначені для зміни журналів з метою видалення будь-яких доказів присутності шкідливої програми в системі.

Іншим важливим питанням є забезпечення доступності журналів. Багато журналів мають максимальний розмір для запису. Наприклад можуть зберігати лише 10 000 останніх подій або 100 мегабайт даних. Коли ліміт розміру досягнуто лог-файл може перезаписати старі дані новими даними або взагалі припинити журналювання, що призведе до втрати попередніх записів. Для вирішення цієї проблеми організаціям необхідно забезпечити зберігання архівних копій файлів журналів протягом тривалого періоду часу, що вимагає налаштування процесів архівування журналів. Але для зменшення розмірів журналів може бути доречним архівувати лише певні важливі записи.

Проблема аналізу журналів

В більшості організаціях системні адміністратори традиційно несуть відповідальність за виконання процесу аналізу журналу — перегляду записів для виявлення підозрілих або незвичних подій. Керівники організацій часто вважали це завданням із низьким пріоритетом, оскільки інші обов'язки адміністраторів, як-от вирішення неполадок із системою і усунення вразливостей безпеки

вимагають швидкого і пріоритетного реагування. Тому адміністратори, які відповідають за виконання аналізу подій лог-файлів, часто не проходять відповідного навчання і вимушені вчити все вже під час робочого процесу. Крім того, адміністратори часто не мають відповідних інструментів ефективних для автоматизації більшої частини процесу аналізу. Багато з цих інструментів є особливо корисні у пошуку закономірностей, які досить важко побачити, наприклад кореляції записів із кількох журналів, які стосуються однієї події.

Інша проблема полягає в тому, що багато адміністраторів вважають аналіз журналу нудним і неефективним процесом. Аналіз журналу часто сприймають, як інструмент для розслідування діяльності — щось, що робиться вже після виявлення проблеми через інші засоби, — а не превентивним, для ідентифікації поточної активності та пошуку можливих ознак майбутніх проблем. Традиційно більшість логів не аналізувалися в режимі реального або майже реального часу. Тому без запровадження надійних процесів аналізу журналів їх цінність значно знижується.

Шляхи вирішення

Незважаючи на згадані численні труднощі, з якими стикаються організації під час організації роботи з журналами, існують кілька ключових практик, які організація може використовувати, щоб уникнути та навіть вирішити безліч проблем:

1) Належним чином визначити пріоритети керування журналами в усій організації. Організація повинна визначити свої вимоги та цілі щодо ведення і моніторингу лог-файлів, щоб дотримуватись відповідних законів, нормативних актів та існуючі правила політик. І вже після цього організація може встановити пріоритети своїх цілей, дотримуючись балансу між зменшенням ризику та ресурсами необхідними для виконання функцій управління журналами.

2) Встановити політики та процедури для керування логами. Політики та процедури є корисними, оскільки вони забезпечують послідовний підхід у всій мережі, а також гарантують дотримання законів і нормативних вимог.

Періодичні аудити є одним із способів підтвердити, що всі закладені правила та політики виконуються.

3) Створити та підтримувати безпечну інфраструктуру для управління журналами. Для організації дуже корисно створити компоненти інфраструктури керування журналами та визначити, як ці компоненти взаємодіють. Це сприяє збереженню цілісності даних журналу від випадків випадкової або навмисної зміни чи видалення, а також збереженню конфіденційності даних в лог-файлах. Також важливо створити достатньо надійну інфраструктуру, щоб обробляти не лише очікувані обсяги журнальних даних, але й пікові обсяги під час екстремальних ситуацій (наприклад, інцидент пов'язаний з проникненням шкідливого програмного забезпечення, тестування на проникнення, сканування вразливостей, додання нових пристроїв).

4) Забезпечити достатню підтримку для всього персоналу, відповідального за управління журналами. Визначаючи схему керування логами, організації повинні забезпечити необхідний інструктаж відповідного персоналу щодо їхніх обов'язків, а також навчанню практичних вмінь необхідних для роботи з лог-файлами. Підтримка також включає надання відповідних інструментів, документації, консультацій з питань управління журналами та визначення відповідальних людей за управління журналами.

1.2 Використання централізованих систем аналізу журналів

Інструменти аналізу журналів (лог-файлів) відіграють важливу роль у безпеці, допомагаючи організаціям виявляти, досліджувати та реагувати на потенційні кіберзагрози та інциденти безпеки. Подібні інструменти можуть аналізувати дані журналів шукаючи шаблони та аномалії, які можуть вказувати на подію порушення безпеки [4]. Однак кількість додатків і обсяг даних, які генерують сучасні ІТ-системи, створюють велике навантаження на традиційні інструменти аналізу (notepad, grep, awk, тощо), що ускладнює процес і збільшує час обробки даних. Це зрештою викликає потребу в імплементації в ІТ-інфраструктурі централізованої системи збору і аналізу логів.

Система збору і аналізу логів – це інфраструктура або програмне забезпечення, яке призначене для централізованого збору, зберігання, обробки та аналізу логів (журналів), отриманих від пристроїв ІТ-інфраструктури, а також надає гнучкі інструменти для візуалізації накопичених даних, виявлення аномалій і аналізу подій [5]. Дана система може використовуватись, як в режимі реальному часі (моніторинг), так і для криміналістичного аналізу вже після того, як стався інцидент. Подібна система також може бути інтегрована з іншими інструментами безпеки такими, як система виявлення і запобігання вторгнень (IDPS), система безпеки та керування подіями (SIEM), брандмауер, антивірус. Подібні інтеграції дають змогу отримувати повне уявлення про стан безпеки мережі та оперативно реагувати на потенційні загрози.

Інфраструктура керування логами зазвичай виконує ряд наступних функцій, які сприяють зберігання та аналізу даних логів.

Загальні можливості

- Парсинг логів полягає у вилученні даних із логів, щоб отримані значення могли бути використані, як вхідні дані для іншого процесу журналювання. Простий приклад – це читання текстового файлу журналу, який містить 5 значень, розділених крапкою з комою, у кожному рядку і потрібно отримано ці 5 значень із кожного рядка.

- Фільтрація подій полягає у виключенні записів із логів з процесу аналізу, звітності або довготривалого зберігання, через причини пов'язані з відсутністю потреби зберігати ці дані. Наприклад, дублікати записів і стандартні інформаційні записи можуть бути відфільтровані, оскільки вони не надають корисної інформації аналітикам.

- Агрегація подій полягає в об'єднанні схожих записів у єдиний запис, що містить підрахунок кількості випадків події. Наприклад, маємо сотню записів, кожна з яких фіксує частину сканування, може бути об'єднана в один запис, що вказує, скільки хостів було проскановано в загалом. Агрегація часто здійснюється під час первинного генерування логів (генератор підраховує схожі

пов'язані події і періодично записує запис із зазначенням кількості), а також може здійснюватися в рамках процесів зменшення обсягу логів або кореляції подій.

Зберігання:

- Ротація логів полягає в закритті файлу журналу і відкритті нового файлу, коли перший файл вважається завершеним. Ротація логів зазвичай виконується відповідно до розкладу (наприклад, щогодини, щодня, щотижня) або коли файл журналу досягає певного розміру. Основні переваги ротації логів полягають у збереженні записів і підтримці розміру файлів у межах, зручних для управління. Після ротації файл журналу може бути стиснутий для економії місця. Крім того, під час ротації часто запускаються скрипти, що працюють із архівованим журналом. Багато генераторів логів мають вбудовані можливості для ротації логів, а також існують прості скрипти або сторонні утиліти, які можуть здійснювати ротацію логів, іноді пропонуючи додаткові функції, що не підтримуються самими генераторами.

- Архівування логів — це збереження журналів на тривалий час, зазвичай на змінних носіях, у мережевому сховищі (SAN) або на спеціалізованому пристрої чи сервері для архівування логів. Логи часто необхідно зберігати для дотримання юридичних або регуляторних вимог. Існують два типи архівування логів: збереження і утримання. Збереження логів – це регулярне архівування журналів як частина стандартних операційних процесів. Утримання логів – це збереження журналів, які зазвичай підлягали б видаленню, тому що вони містять записи про діяльність, що представляє особливий інтерес. Утримання логів зазвичай здійснюється для обробки інцидентів або розслідувань.

- Стиснення логів – це зберігання файлу журналу у спосіб, що зменшує обсяг простору для зберігання, необхідного для файлу без зміни змісту його даних. Стиснення логів часто виконується під час ротації або архівування журналів.

- Зменшення обсягу логів полягає у видаленні непотрібних записів із журналу для створення нового, меншого журналу. Зменшення обсягу логів і подій часто виконується разом із архівуванням логів, щоб у довготривалому зберіганні надходили тільки потрібні записи й поля даних.

- Конвертація логів – це процес парсингу журналу в одному форматі та збереження його записів у другому форматі. Наприклад, конвертація може полягати в тому, щоб узяти дані з журналу, що зберігається в базі даних, і зберегти їх у форматі XML у текстовому файлі. Багато генераторів логів можуть конвертувати свої власні журнали в інший формат; також існують сторонні утиліти для конвертації. Конвертація логів іноді включає дії, такі як фільтрація, агрегація та нормалізація.

- Нормалізація логів передбачає перетворення кожного поля даних журналу у певний формат і їх послідовну категоризацію. Одне з найпоширеніших застосувань нормалізації – це збереження дат і часу в одному форматі. Наприклад, один генератор логів може зберігати час події у форматі 12-годинного часу (6:21:00 PM EDT) і категоризувати його як Timestamp, тоді як інший генератор логів може зберігати його у форматі 24-годинного часу (18:21), а часовий пояс зберігати у іншому полі (наприклад, +1000), категоризуючи його як Time Zone. Нормалізація даних значно спрощує аналіз та звітність, коли використовується кілька форматів логів. Однак нормалізація може бути дуже ресурсомісткою, особливо для складних записів журналу (наприклад, типових журналів систем виявлення вторгнень).

- Перевірка цілісності файлів логів передбачає обчислення дайджесту повідомлення для кожного файлу та безпечно зберігання цього дайджесту, щоб забезпечити виявлення змін у архівованих логах. Дайджест повідомлення – це цифровий підпис, який унікально ідентифікує дані й має властивість, за якої зміна одного біта в даних спричиняє створення абсолютно іншого дайджесту. Найбільш поширені алгоритми дайджесту повідомлень – це MD5 і Secure Hash Algorithm 1 (SHA-1). Якщо файл журналу змінено і його дайджест повідомлення

перераховано, він не співпадає з оригінальним дайджестом, що свідчитиме про зміну файлу.

Аналізу:

- Кореляція подій полягає у виявленні взаємозв'язків між двома або більше записами в журналах. Найбільш поширена форма кореляції подій – це кореляція на основі правил, яка зіставляє кілька записів у журналах за певними критеріями, такими як IP-адреси та типи подій. Кореляція подій може також виконуватися іншими способами, наприклад, з використанням статистичних методів або інструментів візуалізації. Якщо кореляція виконується автоматизованими методами, зазвичай результат успішної кореляції – це новий запис у журналі, який об'єднує інформацію з кількох записів в одне ціле. Залежно від характеру цієї інформації, інфраструктура може також створити сповіщення, що вказує на необхідність подальшого розслідування ідентифікованої події.

- Перегляд логів – це відображення записів журналів у форматі, зрозумілому для людини. Більшість генераторів логів надають певні можливості перегляду журналів і також доступні сторонні утиліти.

- Звітність по логах полягає у відображенні результатів аналізу логів. Звіти зазвичай складаються для підсумовування значущої активності за певний період часу або для отримання детальної інформації, пов'язаної з конкретною подією або серією подій.

Компоненти інфраструктури керування журналами

Інфраструктура керування журналами складається з апаратного забезпечення, програмного забезпечення, мереж і носіїв, які використовуються для створення, передачі, зберігання, аналізу та видалення даних. Подібна система зазвичай складається з таких трьох рівнів:

- Генерація журналу. Перший рівень містить хости та додатки, які генерують лог-файли. Деякі хости запускають клієнтські додатки або служби журналу, які надають доступ до даних журналу через мережу для серверів журналу на другому рівні. Інші ж хости роблять свої логи доступними іншими

способами, наприклад дозволяючи серверам аутентифікувати їх і отримувати копії відповідних файлів.

- Аналіз та зберігання. Другий рівень складається з одного або кількох серверів журналу, які отримують дані журналу або копії даних журналу від хостів першого рівня. Інформація передається на сервери в режимі реального або майже реального часу або ж періодично пакетами на основі розкладу або кількості даних, які очікують на передачу. Сервери, які отримують ці лог-файли від кількох генераторів журналів, іноді називають колекторами або агрегаторами. Отримані дані можуть зберігатися на самих серверах або на окремих серверах баз даних.

- Моніторинг. Третій рівень містить інструменти, які можна використовувати для моніторингу та перегляду даних журналу, результатів автоматизованого аналізу, а також для створення звітів [6]. У деяких мережах на цьому рівні інструменти цього рівня можна використовувати для керування журнальними серверами та клієнтами. Крім того, привілеї користувача іноді можуть бути обмежені лише необхідними функціями. Другий рівень, аналіз і зберігання журналів, може сильно відрізнятися за складністю та структурою в кожній організації. Найпростішим є варіант, де єдиний сервер журналів виконує всі функції аналізу та зберігання (поєднує функції 2 і 3 рівнів).

Комунікації між компонентами інфраструктури керування логами зазвичай здійснюються через звичайні мережі організації, оскільки хости, що генерують журнали, можуть бути розташовані по всій мережі організації. Однак можна використовувати фізично або логічно окрему мережу для журналювання, особливо для отримання логів з ключових пристроїв (наприклад, міжмережевих екранів та систем виявлення вторгнень у мережу, які часто передають великі обсяги даних журналів), а також для передачі логів між серверами журналів. Під час масштабного інциденту зі шкідливим програмним забезпеченням або іншої мережевої атаки звичайні мережі можуть бути нестабільними або недоступними. Ще одним мотивом для використання окремої мережі для журналювання є

захист даних журналів у звичайних мережах організації від перехоплення. Якщо окрема мережа для журналювання не використовується, комунікації журналювання в звичайній мережі можуть бути захищені за допомогою додаткових засобів безпеки, таких як шифрування даних. Ще однією перевагою окремої мережі для журналювання є захист тих компонентів, які використовуються виключно для керування логами.

Організація може мати єдину інфраструктуру керування логами для всього підприємства, але частіше зустрічаються випадки наявності кількох окремих інфраструктур, які не обов'язково взаємодіють між собою. Наявність єдиної інфраструктури керування логами може забезпечити єдину точку для перегляду всіх відповідних журналів організації, але для більших організацій розмір такої інфраструктури та обсяг даних, які їй потрібно буде обробляти і зберігати, зазвичай роблять її непрактичною. Великі організації зазвичай мають кілька інфраструктур керування логами, іноді десятки або сотні. Обсяг кожної інфраструктури може визначатися багатьма факторами, включаючи внутрішню структуру організації, типи систем (наприклад, окрема інфраструктура для систем безпеки підприємства), типи журналів (наприклад, окрема інфраструктура для журналів аудиту застосунків) і місцезнаходження об'єктів.

Рекомендації щодо впровадження інфраструктури керування логами

Нижче наведено основні рекомендації щодо впровадження інфраструктури керування логами [7].

- Централізація журналювання – агрегація логів з різних джерел у єдиному місці для зручного аналізу та безпечного зберігання.
- Перелік подій для моніторингу – визначення лише важливих подій, які необхідно логувати.
- Виключення зайвої інформації – уникання збереження чутливої та зайвої інформації для економії ресурсів і підвищення безпеки.
- Активний моніторинг – оперативне відстеження логів і налаштування сповіщень для швидкої реакції на інциденти.

- Структуроване журналювання – зберігання даних у форматі, що полегшує їхній аналіз і використання для діагностики.
- Індксація логів – застосування індексування для швидкого пошуку та аналізу великих обсягів даних.
- Масштабоване зберігання – використання хмарних рішень для надійного зберігання великих обсягів логів.
- Оптимізація політики зберігання – встановлення різних термінів зберігання для логів з урахуванням нормативних вимог і аналітичних цілей.
- Збір з різних джерел – збір логів з мережевих пристроїв, систем безпеки, серверів, додатків тощо для всебічного моніторингу інфраструктури.

1.3. Огляд безкоштовних рішень централізованого збору і обробки логів з відкритим вихідним кодом

Інструменти для управління логами – це програми, які призначені для збирання, обробки та зберігання логових даних, що генеруються різними пристроями та додатками в ІТ-інфраструктурі організації. Ці інструменти використовуються для моніторингу та аналізу логів, щоб виявляти потенційні проблеми, відслідковувати продуктивність та отримувати дані щодо поведінки систем і пристроїв організації.

Вибір інструментів з відкритим вихідним кодом стає дедалі актуальнішим для організацій, що прагнуть підвищити ефективність та гнучкість своїх ІТ-рішень. Такі інструменти не лише дозволяють знизити витрати на ліцензії, але й надають можливість адаптувати програмне забезпечення під конкретні потреби компанії. Завдяки відкритому коду, організації отримують повний контроль над функціональністю інструментів, можливість інтеграції з іншими системами та підтримку активної спільноти розробників. У сучасних умовах, коли швидкість реагування на зміни в ІТ-інфраструктурі є критично важливою, інструменти з відкритим кодом стають важливим елементом для забезпечення конкурентоспроможності та технологічної незалежності.

Вибір же відповідного рішення для збору та управління логами вимагає ретельного аналізу кількох факторів, щоб переконатися, що воно відповідає конкретним потребам і цілям організації. Кожен інструмент має свої унікальні сильні сторони і підходить для різних потреб та середовищ. Слід пам'ятати, що найкраще програмне забезпечення обирається залежно від конкретних вимог та пріоритетів, чи то обробка великих обсягів даних, забезпечення високої швидкості роботи, чи зручний інтерфейс. Нижче наведено опис найпопулярніших інструментів з відкритим вихідним кодом [8]:

1) Elastic Stack (відомий також як ELK Stack), є популярним інструментом для збору, централізації, обробки та візуалізації логів. Він дозволяє збирати великі обсяги даних з різних серверів і централізувати їх в одному місці для подальшого аналізу. Стек Elastic складається з трьох основних компонентів:

Elasticsearch – це потужний, відкритий пошуковий та аналітичний механізм, побудований на основі Apache Lucene. Він здатен працювати з різними типами даних, як структурованими, так і неструктурованими, включаючи текстові, числові та геопросторові дані. Elasticsearch є основним компонентом стеку і відомий своєю швидкістю, масштабованістю та використанням REST API. Він зберігає, індексує та аналізує великі обсяги даних, що надходять з Logstash.

Logstash – це безкоштовний та відкритий інструмент для збору, обробки та перетворення даних. Він отримує дані від віддалених серверів за допомогою агентів під назвою «beats», які передають великі обсяги системних метрик і логів. Після обробки даних Logstash передає їх до Elasticsearch.

Kibana – це веб-платформа для візуалізації даних, яка працює разом з Elasticsearch. Вона дозволяє досліджувати та візуалізувати часові ряди даних і логи з Elasticsearch. Kibana надає інтуїтивні інформаційні панелі для візуалізації даних у вигляді діаграм, графіків, гістограм тощо.

Однією з найбільших переваг Elasticsearch є його масштабованість. Його можна легко розгортати на кількох серверах, що робить його ідеальним для

великомасштабних застосунків. Однак, налаштування та підтримка Elasticsearch можуть бути складними, особливо для користувачів з обмеженим досвідом роботи з розподіленими системами.

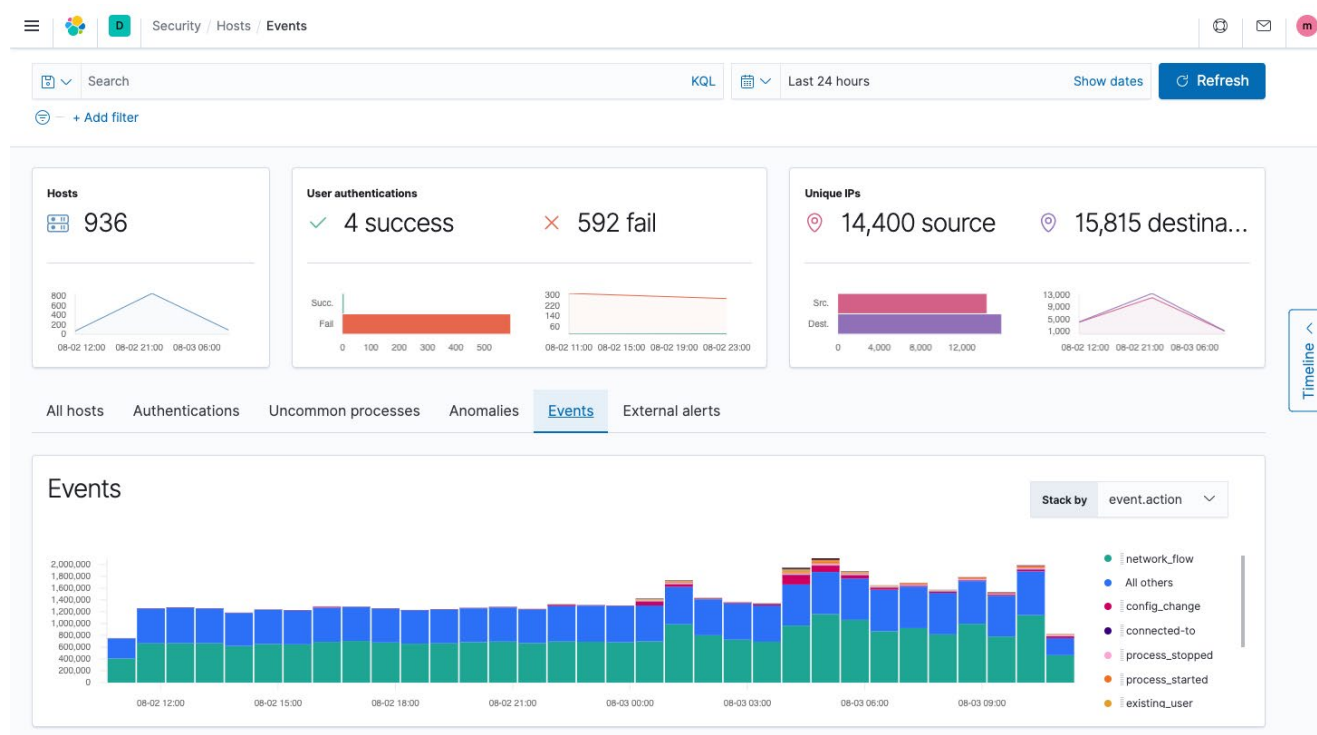


Рисунок 1.1 – Приклад інтерфейсу Elastic Stack

2) Graylog – це потужна платформа для керування логами з відкритим вихідним кодом, яка забезпечує централізоване зберігання та аналіз даних з різних джерел. Вона допомагає компаніям у реальному часі збирати, зберігати та аналізувати логи від різноманітних пристроїв та додатків. Використовуючи зібрані логи, Graylog надає цінні дані та підвищує видимість інфраструктури, додатків і систем безпеки підприємства. Крім того, цей інструмент управління логами пропонує такі функції, як пошук і сповіщення в реальному часі, інформаційні панелі та звіти, що робить його корисним для ІТ-операцій, команд безпеки та розробників. Graylog підтримує збір логів із різних джерел, зокрема серверів, додатків, мережевих пристроїв і хмарних платформ, дозволяє користувачам налаштовувати сповіщення на основі визначених умов і порогових значень, що забезпечує проактивний моніторинг та повідомлення про певні події.

Graylog відіграє важливу роль у прийнятті бізнес-рішень, пов'язаних із взаємодією користувачів з веб-додатками. Він збирає важливу аналітику про поведінку додатків і візуалізує ці дані на різних графіках, таких як стовпчасті діаграми, кругові діаграми та гістограми. Отримані дані можуть бути використані для ухвалення ключових бізнес-рішень і на відміну від Elastic Stack, Graylog пропонує єдине рішення для збору, обробки та візуалізації даних, усуваючи потребу в установці декількох компонентів.

Graylog підтримує широкий спектр джерел даних, включаючи syslog, GELF та різні API. Завдяки потужному механізму обробки потоків, Graylog дозволяє маршрутизувати повідомлення логів залежно від їхнього змісту, що дає змогу користувачам ефективніше організовувати свої дані. Проте, налаштування та конфігурація Graylog можуть виявитися складними, особливо для користувачів, які мають обмежений досвід роботи з розподіленими системами.

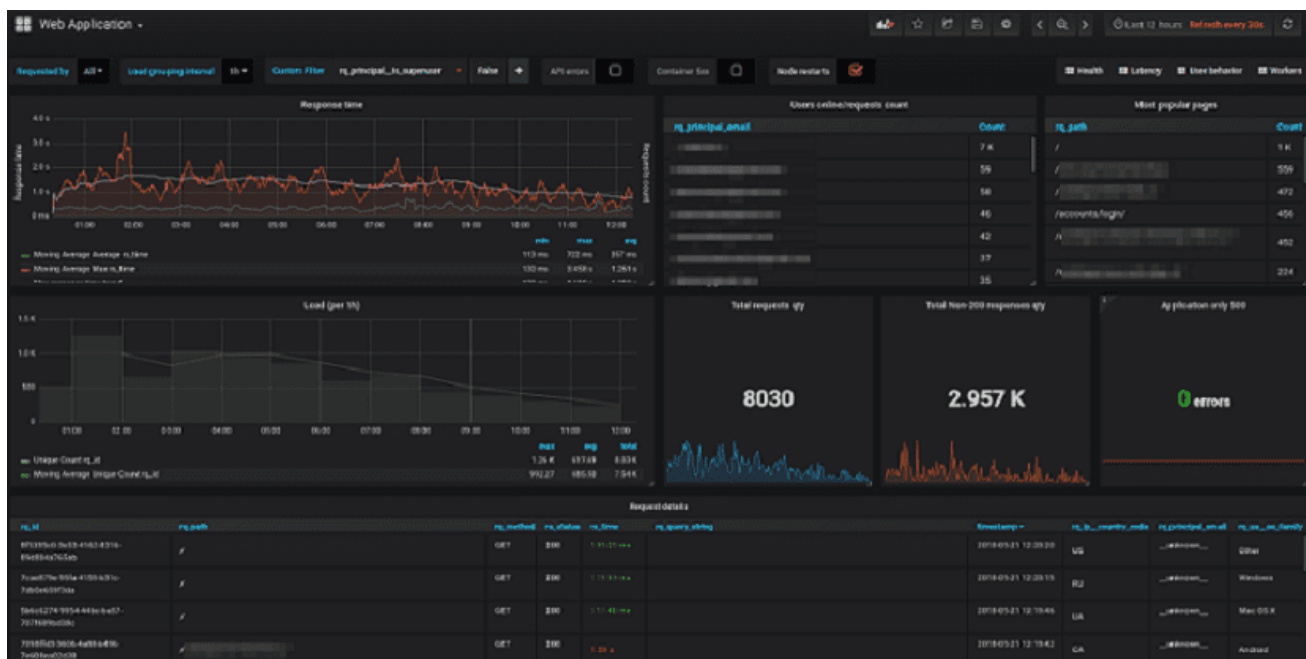


Рисунок 1.2 – Приклад інтерфейсу Graylog

3) SigNoz є відносно новою та універсальною платформою з відкритим вихідним кодом для управління та аналізу логів, яка забезпечує централізоване збирання, зберігання та обробку даних. Вона створена для того, щоб допомагати організаціям отримувати цінну інформацію щодо їхньої IT-інфраструктури,

застосунків та систем безпеки. SigNoz надає можливості реального часу для моніторингу, автоматизованого усунення неполадок та прогнозової аналітики.

Платформа підтримує збір логів з різноманітних джерел, таких як сервери, мережеві пристрої, додатки та хмарні сервіси. Для збору і обробки логових даних SigNoz використовує OpenTelemetry – стандарт інструментування для хмарних додатків, який стрімко став загальноновизнаним у світі.

SigNoz також оснащений графічним інтерфейсом користувача та інструментами візуалізації, розробленими спеціально для спостереження. Інтерфейс дозволяє швидко переходити між елементами та здійснювати детальний аналіз, що допомагає користувачам ефективно виявляти причини проблем.

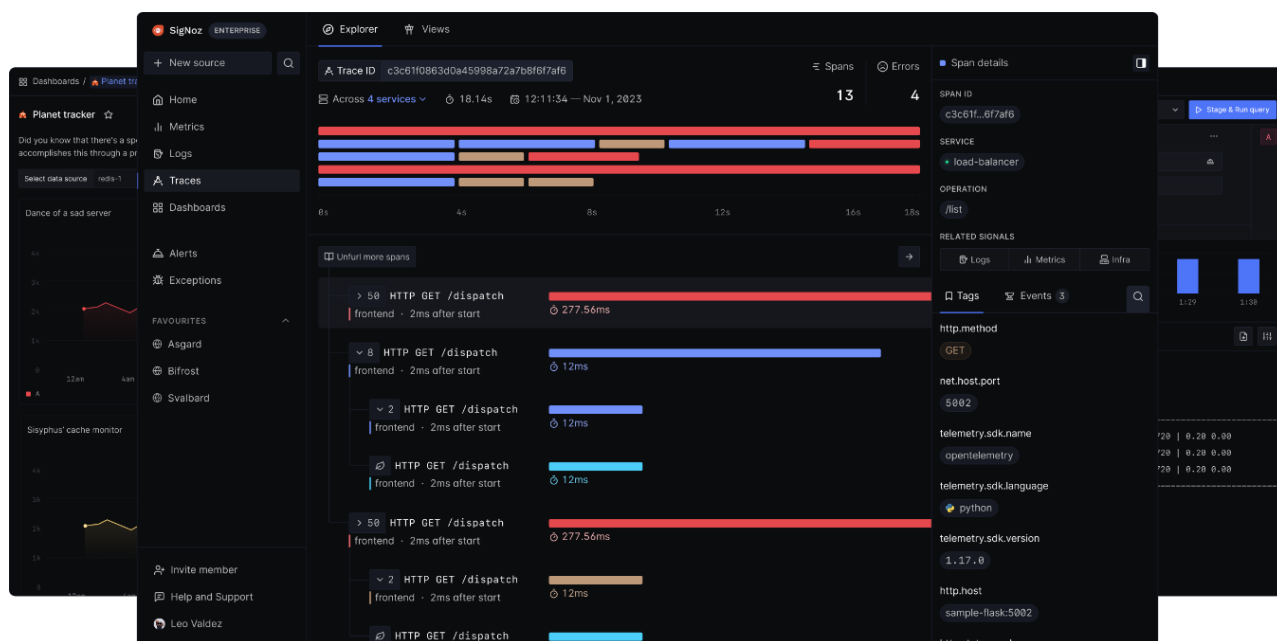


Рисунок 1.3 – Приклад інтерфейсу SigNoz

4) Grafana Loki – це система для управління логами, розроблена для інтеграції з Grafana. Вона забезпечує централізоване зберігання і обробку логів, використовуючи ефективний підхід до індексації метаданих, а не самих логів. Loki підтримує збір логів з різних джерел за допомогою агента Promtail і дозволяє користувачам здійснювати запити та візуалізувати дані через Grafana, що робить моніторинг і аналіз логів зручним і масштабованим.

Архітектура Loki було натхненна Prometheus – популярною системою моніторингу. Вона використовує подібний підхід до збору та зберігання логів, що робить її природним вибором для користувачів, вже знайомих з Prometheus. Архітектура Loki складається з кількох компонентів:

Loki server – сервер, який відповідає за зберігання і індексацію логів. Він отримує дані від агентів, таких як Promtail, які збирають логи з різних джерел і передають їх на сервер. Loki не індексує самі логи, а зберігає метадані, що робить його більш ефективним з точки зору витрат.

Promtail – це агент, який виконує функцію збору логів. Він працює з різними джерелами даних, такими як сервіси, контейнери та іншими компоненти інфраструктури і передає ці логи до сервера Loki. Promtail забезпечує передачу логів у форматі, який Loki може обробити і зберегти.

Grafana, популярний інструмент для візуалізації даних, інтегрується з Loki для створення зручного інтерфейсу для запитів і відображення логів. Завдяки такій інтеграції, користувачі можуть виконувати запити до логів і переглядати їх у вигляді графіків, діаграм і таблиць безпосередньо в Grafana. Це дозволяє легко поєднати моніторинг і аналіз логів, забезпечуючи глибоке розуміння роботи системи та швидке виявлення можливих проблем.

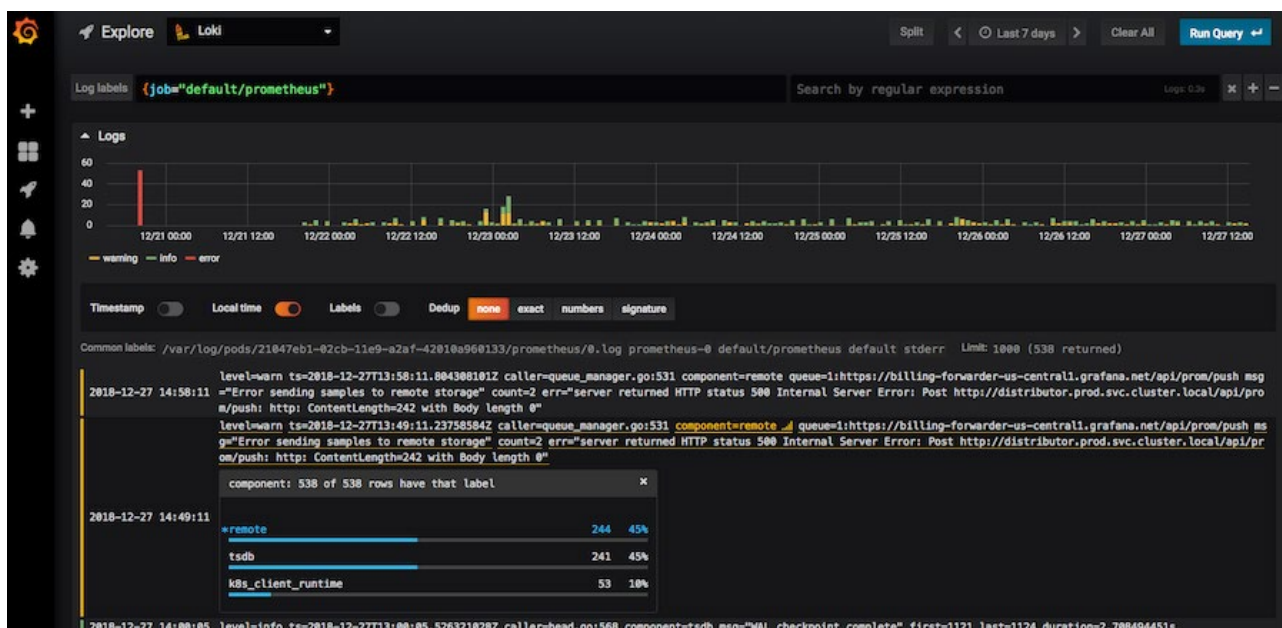


Рисунок 1.4 – Приклад інтерфейсу Grafana Loki

Підсумуємо переваги та недоліки кожного рішення в табл.1.1.

Таблиця 1.1

Порівняльна характеристика безкоштовних централізованих систем

Характеристика	Elastic Stack	Graylog	SigNoz	Grafana Loki
Тип ліцензії	Відкрита	Відкрита	Відкрита	Відкрита
Складові компоненти	Elasticsearch, Logstash, Kibana	Graylog Server, Elasticsearch, MongoDB	SigNoz UI, Collector, Tracing, Metrics	Promtail, Loki Server, Grafana
Збір даних	За допомогою Beats або агентів	Вбудовані інструменти збору даних	OpenTelemetry	Promtail
Масштабованість	Висока	Висока	Висока	Висока
Популярність	Висока	Висока, але менш популярна ніж Elastic Stack	Зростаюча популярність	Середня популярність
Індексація	Повна	Повна	Повна	Індексація лише метаданих
Візуалізація	Kibana	Вбудована візуалізація	Вбудована візуалізація	Grafana
Підтримка запитів	Через Kibana	Через веб-інтерфейс	Інтеграція з Prometheus для метрик і трасування	Через Grafana
Аналіз та моніторинг	Високий рівень аналітики та моніторингу	Високий рівень аналітики та моніторингу	Поглиблений аналіз із візуалізацією	Базовий аналіз, інтеграція з Grafana
Простота налаштування	Високий рівень	Середній рівень	Легка інтеграція	Середній рівень
Підтримка спільноти	Висока підтримка	Середня підтримка	Зростаюча спільнота	Висока підтримка
Документація	Повна	Середня	Середня	Середня
Інтеграції зі сторонніми інструментами	Широка підтримка	Підтримка інтеграцій з інструментами як Elasticsearch, Kafka, Grafana	Підтримка інтеграцій через OpenTelemetry та Prometheus	Широка підтримка інтеграцій з Grafana та іншими інструментами

Виходячи з наведеної таблиці та опису, SigNoz хоч і перспективним, але відносно новим рішенням, яке поки не має широкої підтримки з боку спільноти, через що можуть виникнути проблема з усуненням несправностей та підтримкою роботи системи.

Особливістю Grafana Loki є індексація лише метаданих логів. З одного боку це пришвидшує роботу системи та зменшує обсяг даних яких треба зберігати, а з іншого це зменшує можливість виконувати детальний пошук.

Elastic Stack і Graylog – обидва цікаві та популярні варіанти з широкою підтримкою спільноти, документацією та можливістю встановлення сторонніх інструментів. Проте Elastic Stack має дещо краще можливості додаткових налаштувань, що включає широкий список плагінів, можливість роботи майже з усіма відомими типами даних, різноманіття графічних інструментів для візуалізації даних через Kibana та в основі системи використовується Elasticsearch, який краще виконує пошук серед великого масиву даних.

1.4 Теоретичні відомості стосовно Elastic Stack

Elastic Stack (раніше відомий як ELK Stack) призначений для того, щоб допомогти користувачам імпортувати дані будь-якого формату та з будь-яких джерел, а також здійснювати пошук, аналіз та візуалізацію цих даних у реальному часі [9]. Основні компоненти стеку є [10]:

Elasticsearch – основний компонент, що забезпечує зберігання, пошук та аналіз даних. Він працює як розподілена пошукова система, яка ефективно масштабується.

Logstash – інструмент для збору, фільтрації та трансформації даних з різних джерел, після чого передає ці дані до Elasticsearch.

Kibana – інструмент для візуалізації даних, що дозволяє створювати графіки, панелі моніторингу та інші візуальні репрезентації даних, зібраних в Elasticsearch.

Beats – це агенти для збору даних, яка відправляє їх безпосередньо в Elasticsearch або Logstash.

Elastic Agent — це універсальний агент, який спрощує управління даними та інтеграцію з Elastic Stack. Він об'єднує функціональність декількох Beats у одному агенті, спрощуючи налаштування та управління

Application Performance Monitoring (APM) — це компонент Elastic Stack, який дозволяє відстежувати та аналізувати продуктивність додатків в реальному часі.

Детальніше розглянемо архітектуру роботи компонентів Elastic Stack:

Збір і обробка даних

1) Beats – це агенти передачі даних, які встановлюються на кінцевих пристроях і відправляють зібрані дані до Elasticsearch. Кожен Beat є окремим продуктом, який можна встановити окремо.

2) Elastic Agent – це універсальний агент для моніторингу логів, метрик та інших типів даних на хостах. Використання одного агента спрощує та прискорює розгортання моніторингу по всій інфраструктурі. Політики агентів визначають, які інтеграції ви хочете використовувати і на яких хостах їх запускати. Ви можете застосовувати політику Elastic Agent до кількох агентів одночасно, що спрощує управління конфігурацією на великій шкалі.

Elastic Agent можна розгорнути у двох режимах:

- Керування через Fleet. У цьому режимі політики Elastic Agent та їх життєвий цикл централізовано керуються через додаток Fleet у Kibana. Додаток також дозволяє централізовано додавати інтеграції з іншими популярними сервісами та системами. Це рекомендований варіант для більшості користувачів.
- Режим самостійного управління (standalone). У цьому випадку всі політики застосовуються до Elastic Agent вручну через файл YAML. Цей режим призначений для досвідченіших користувачів та кому не суттєва централізація.

Вибір методу залежить від ваших потреб, функціональних можливостей та бажання централізовано управляти вашими агентами.

Beats та Elastic Agent можуть надсилати дані безпосередньо до Elasticsearch або через Logstash, де дані можуть бути додатково оброблені і вдосконалені перед візуалізацією у Kibana.

3) APM – це система моніторингу продуктивності додатків, побудована на базі Elastic Stack. Вона дозволяє відстежувати роботу програмних сервісів і додатків в режимі реального часу, збираючи детальну інформацію про продуктивність, таку як час відгуку на вхідні запити, запити до бази даних, виклики кешів, зовнішні HTTP-запити та інші параметри. Завдяки цьому можна швидко виявляти та усувати проблеми з продуктивністю, забезпечуючи ефективне управління та оптимізацію роботи додатків.

4) Logstash – це інструмент для збору даних, який приймає вхідні неструктуровані дані з різних джерел і виводить вже оброблені дані у різні сховища, в тому числі Elasticsearch. Це універсальний конвеєр, який може інтегруватися з різними джерелами. Перед передачею даних в Elasticsearch, вони проходять очищення та трансформацію за допомогою фільтрів. Logstash має різні адаптери, які дозволяють обробляти дані з різних застосунків і нормалізувати ці дані для відправки в різні місця призначення. Обробка в Logstash організована у вигляді одного або кількох конвеєрів (pipelines). Logstash є важливою частиною процесу збирання та підготовки даних для подальшого аналізу.

5) Пайплайни прийому Elasticsearch дозволяють виконувати типові перетворення над даними перед тим, як індексувати їх у Elasticsearch. Пайплайн складається з низки налаштовуваних завдань, які називаються процесорами. Кожен процесор виконується послідовно, вносячи певні зміни до вхідних документів. Після виконання всіх процесорів Elasticsearch додає перетворені документи до потоку даних або індексу.

Зберігання

1) Elasticsearch (зберігання, пошук та аналіз) – це документно-орієнтована пошукова система, розроблена для зберігання, пошуку та управління даними

різних типів, включаючи структуровані, неструктуровані та частково структуровані дані. Вона базується на бібліотеці Apache Lucene і забезпечує потужні можливості для повнотекстового пошуку, аналітики та індексації великих обсягів інформації і є найбільш популярною пошуковою системою, яку використовуючи великі організації. Завдяки горизонтальній масштабованості та надійності, Elasticsearch є основним компонентом багатьох аналітичних рішень, що дозволяє ефективно працювати з даними різних форматів та розмірів. Цей інструмент широко застосовується в додатках, які значною мірою залежать від пошукової платформи для доступу, видалення та представлення даних. Зокрема, Elasticsearch є незамінним для веб-сайтів з великим обсягом інформації, де потрібно забезпечити швидкий і точний пошук.

Ще однією важливою сферою застосування Elasticsearch є аналітика. Він зазвичай використовується для прийому та аналізу даних лог-файлів практично в режимі реального часу з можливістю масштабування. Це дозволяє організаціям отримувати оперативну інформацію, необхідну для оптимізації роботи систем.

Elasticsearch є центральним компонентом Elastic Stack, який поєднує в собі швидкість і гнучкість повнотекстового пошуку з потужними аналітичними можливостями, використовуючи складну, але зручну для розробників мову запитів ES|QL (Elasticsearch Query Language). Цей інструмент підтримує роботу з різноманітними типами даних. Elasticsearch може бути інтегрований в існуючу систему для забезпечення ефективного зберігання, пошуку та аналізу даних.

Кластер Elasticsearch – це сукупність одного або більше вузлів (серверів), які разом зберігають всі дані і забезпечують можливості індексування та пошуку по всіх вузлах. Вузол – це окремий сервер, що входить до складу кластера, зберігає дані та бере участь в індексуванні та пошуку по кластеру. Індекс – це збірка документів (логів), які мають схожі характеристики. Документ є базовою одиницею інформації, яку можна індексувати.

Використання і аналіз даних

1) Kibana (візуалізація і аналіз) – це візуалізаційний інструмент для Elastic Stack, який надає користувачам можливість досліджувати та візуалізувати свої дані. Це відкрита платформа для створення візуалізацій, що дозволяє взаємодіяти з даними через потужні та привабливі графічні інтерфейси. Kibana оживляє дані, трансформуючи їх на різноманітні візуальні форми від гістограм до географічних карт. Різні візуалізації можна комбінувати в індивідуальні інформаційні панелі, що дозволяють ділитися отриманими даними з колегами або клієнтами. Kibana значно спрощує роботу для аналітиків даних і робить процес дослідження даних доступним та зрозумілим.

2) Клієнти Elasticsearch надають зручний механізм для управління API-запитами та відповідями до і від Elasticsearch використовуючи популярні мови програмування, таких як Java, Ruby, Go, Python та інших, що спрощує виконання запитів, індексацію документів, пошук та управління індексами безпосередньо з додатків. Доступні як офіційні, так і створені спільнотою клієнти.

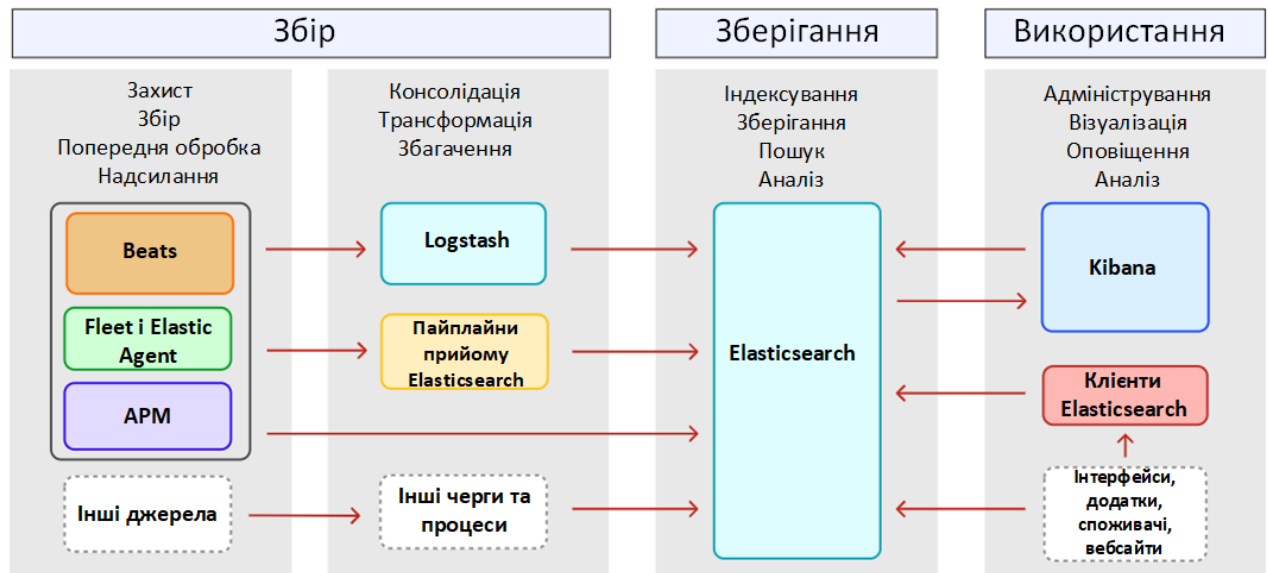


Рисунок 1.5 – Архітектура взаємодії компонентів Elastic Stack

Інтеграції Elastic Stack

Інтеграції Elastic Stack дозволяють легко збирати, зберігати, аналізувати та візуалізувати дані з різних джерел у межах єдиної платформи [11]. Це робить

Elastic Stack надзвичайно гнучким інструментом для моніторингу, аналітики, безпеки та управління даними.

Інтеграція Elastic – це набір компонентів, які визначають, як спостерігати за певним продуктом або послугою і включають наступне:

- Правила збору, зберігання та трансформації даних
- Варіанти конфігурації
- Готові кастомні панелі керування та візуалізації
- Документація
- Тести

Нижче перераховано деякі популярні доступні інтеграції в Elastic Stack, які спрощують збір та обробку даних:

- AWS (Amazon Web Services):
- CloudWatch: Збір логів та метрик з AWS-сервісів.
- S3: Збереження та аналіз журналів у S3.
- Elasticsearch Service: Хмарна версія Elasticsearch на AWS.
- Azure Monitor: Збір даних про стан ресурсів Azure.
- Azure Blob Storage: Для аналізу збережених даних.
- Google Stackdriver: Моніторинг та аналіз логів і метрик з GCP.
- Google Cloud Storage: Аналіз даних, збережених у хмарі.
- Docker: Збір логів та метрик з контейнерів.
- Kubernetes: Моніторинг кластерів Kubernetes та метрик з нодів.
- Microsoft SQL Server: Збір логів і метрик баз даних.
- MySQL та PostgreSQL: Моніторинг продуктивності баз даних.
- NGINX: Збір логів, моніторинг стану веб-серверів NGINX.
- Apache: Інтеграція для збору логів з веб-сервера Apache.
- Cisco: Збір логів з обладнання Cisco для аналізу трафіку та безпеки.
- Palo Alto Networks: Збір даних про безпеку з Palo Alto.
- Windows Event Logs: Збір подій Windows для моніторингу системної

безпеки та продуктивності.

- Active Directory (AD): Моніторинг подій у AD, включаючи зміни конфігурації, аутентифікацію та інші важливі події.
- Okta: Моніторинг подій аутентифікації та авторизації з Okta.
- І багато інших

Висновки до першого розділу

В даному розділі було розглянуто теоретичні відомості стосовно призначення лог файлів, інформацією яка в них записується та цілі з якої здійснюється збір, аналіз та подальше зберігання журналів подій.

Окрему увагу приділено використанню централізованих систем для аналізу логів, що дозволяють оптимізувати процес обробки великих обсягів даних та покращити управління мережею. Централізований підхід забезпечує швидший доступ до даних, полегшує моніторинг безпеки та дає можливість ефективніше реагувати на інциденти.

Було проведено огляд наявних безкоштовних рішень для централізованого збору та обробки даних з відкритим вихідним кодом. Ці рішення надають ефективні альтернативи комерційним продуктам і можуть бути впроваджені в організаціях з обмеженими ресурсами. В ході аналізу було досліджено такі продукти, як Elastic Stack, Graylog, SigNoz та Grafana Loki. Було наведено порівняльну таблицю ключових характеристик зазначених продуктів та визначено, що Elastic Stack краще з усіх зможе задовольнити всі потреби майбутнього проєкту корпоративної мережі.

Окрему увагу було приділено рішенню Elastic Stack. Було розглянуто його основні компоненти та принципи роботи. Визначено його архітектуру роботи та всі функціональні можливості, які він може надати для централізованого збору, обробки та візуалізації даних з пристроїв мережі. Визначено, що даний продукт є універсальним інструментом для використання у корпоративній мережі будь-якого розміру.

РОЗДІЛ 2 ПРОЄКТУВАННЯ СИСТЕМИ ЗБОРУ, АНАЛІЗУ ТА ВІЗУАЛІЗАЦІЇ ДАНИХ В КОРПОРАТИВНІЙ МЕРЕЖІ

2.1 Призначення корпоративних мереж

Корпоративна мережа – це інформаційна мережа, яка забезпечує зв'язок між різними відділами, підрозділами та офісами організації або компанії. Вона дозволяє співробітникам обмінюватися даними, спільно використовувати ресурси, такі як сервери, бази даних і принтери, а також забезпечує безпечний доступ до Інтернету та внутрішніх систем компанії.

Корпоративні мережі призначені для виконання таких завдань [12]:

- Забезпечення прикладних процесів передачею даних, управлінням та обміном інформаційними потоками;
- Організація взаємодії між комп'ютерами в межах організації;
- Розподілене або спільне використання обчислювальних ресурсів;
- Координація роботи різних систем і комплексів автоматизації;
- Створення транспортної інфраструктури для передачі інформаційних потоків у системі організаційно-економічного управління;
- Забезпечення зв'язку для інформаційного обміну між різними структурами організації.
- Оптимізація бізнес-процесів для покращення комунікації та взаємодії.
- Централізоване управління та моніторинг даних, ресурсів, системи безпеки.

Корпоративні мережі, по суті, розроблені для того, щоб сприяти взаємодії пристроїв та безперешкодній комунікації, оскільки вони надійно об'єднують всі системи. Це включає широкий спектр пристроїв, зокрема [13]:

- Сервери. Комп'ютери, що виконують функції зберігання та обробки даних, надання доступу до ресурсів та виконання різних мережевих служб (веб, файли, пошта, бази даних тощо).

- Клієнтські пристрої. Робочі станції або комп'ютери співробітників, ноутбуки, планшети та смартфони, які використовуються для доступу до мережі та виконання робочих завдань.
- Периферійне обладнання. Принтери, сканери, телевізори та інші подібні пристрої, які підключені до мережі для спільного використання.
- Маршрутизатори. Мережеві пристрої, що відповідають за зв'язок між різними мережами та передачу даних між внутрішніми сегментами мережі та зовнішніми мережами, зокрема Інтернетом.
- Комутатори. Мережеві пристрої, що забезпечують підключення та взаємодію різних комп'ютерів та інших пристроїв у локальній мережі (LAN).
- Міжмережеві екрани (Firewalls). Системи, що контролюють та захищають доступ до корпоративної мережі, запобігаючи несанкціонованому доступу та кіберзагрозам.
- Точки доступу. Пристрої, які забезпечують бездротовий доступ до мережі для мобільних пристроїв через Wi-Fi.
- Мережеві накопичувачі (NAS). Системи для централізованого зберігання та спільного доступу до файлів і даних.

2.2 Проєктування топології мережі та схеми адресації

Трирівнева (ієрархічна) топологія мережі була вперше запропонована компанією Cisco у 2002 році та стала загальноприйнятою практикою для розробки надійних, масштабованих і економічно вигідних мереж [14].

На початку мережі мали плоску архітектуру, яка дозволяла розширювати їх лише в одному напрямку через хаби та комутатори. Це ускладнювало фільтрацію небажаного трафіку та контроль трансляцій. Коли мережа зростала в розмірах, час відповіді погіршувався, що негативно позначалося на продуктивності. У зв'язку з цим виникла необхідність у новій мережевій архітектурі, що призвело до запровадження ієрархічного підходу.

Основні переваги ієрархічного дизайну мережі:

- Масштабованість. Трирівнева архітектура дозволяє легко додавати нові пристрої та мережеві сегменти без значних змін у вже існуючій структурі.
- Ефективність. Розділення мережі на рівні (ядра, розподілу, доступу) дозволяє оптимізувати маршрутизацію та управління трафіком, зменшуючи затримки та покращуючи загальну продуктивність.
- Контроль безпеки. Кожен рівень може мати свої політики безпеки, що дозволяє більш ефективно контролювати доступ до мережевих ресурсів.
- Легкість управління. Чітке розділення функцій між рівнями спрощує адміністрування та управління мережею, дозволяючи командам ІТ фокусуватися на специфічних аспектах мережевої інфраструктури.
- Зниження навантаження. Ієрархічний підхід дозволяє зменшити навантаження на мережеві пристрої, оскільки вони можуть бути спеціалізованими для виконання певних функцій.

Трирівнева топологія мережі — це архітектурна модель, що складається з трьох основних рівнів, кожен з яких виконує певні функції для забезпечення ефективності, масштабованості та управління мережевою інфраструктурою. Ця топологія часто використовується в корпоративних мережах. Нижче наведено опис і призначення кожного рівня:

1) Рівень ядра (Core Layer). Цей рівень відповідає за забезпечення високошвидкісної передачі даних між різними частинами мережі. Він забезпечує зв'язок між ярусом агрегування та іншими мережами, такими як WAN або Інтернет. Рівень ядра повинен бути надійним і ефективним для максимізації продуктивності, а також завжди доступним. Його слід проектувати з резервуванням, щоб уникнути єдиної точки відмови. Якщо виникне серйозна проблема, відновлення повинно бути швидким. Основні функції даного рівня:

- Високошвидкісна маршрутизація трафіку.
- Забезпечення надійності та доступності мережі.
- Підтримка резервування та відмовостійкості.

2) Рівень розподілу. (Distribution Layer). Цей рівень виконує функцію з'єднання між рівнем ядра та доступу. Він відповідає за маршрутизацію трафіку, обробку політик безпеки та управління трафіком. Цей рівень зазвичай має кілька комутаторів і маршрутизаторів (або комутаторів 3-го рівня, які виконують функції маршрутизації), при цьому кожен комутатор підключений до кількох пристроїв на рівні доступу. Основні функції даного рівня:

- Маршрутизація трафіку між різними сегментами мережі.
- Застосування політик безпеки та управління доступом.
- Агрегування трафіку з пристроїв рівня доступу.

3) Рівень доступу (Access Layer). Цей рівень відповідає за підключення кінцевих пристроїв. Він забезпечує локальний доступ до мережі та включає комутатори та точки доступу. Даний рівень вимагає високої кількості портів, щоб підтримувати велику кількість підключених пристроїв, але зазвичай не потребує високої пропускної здатності, оскільки кожен порт підключений лише до одного конкретного пристрою. З огляду на велику кількість користувачів та кінцевих пристроїв, цей рівень також вимагає забезпечення високого рівня безпеки, оскільки він є першою лінією між зовнішнім світом і користувачами. Основні функції даного рівня

- Підключення користувачів до мережі.
- Реалізація контролю доступу та безпеки.

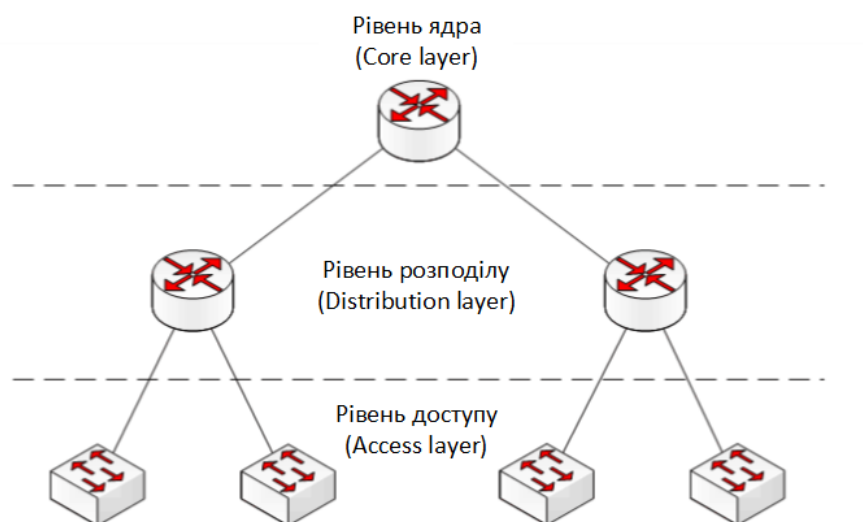


Рисунок 2.1 – Приклад трирівневої топології мережі

Зважаючи на всю вищезгадану інформацію було виконано проєктування логічної топології локальної корпоративної мережі невеликої організації, рис.2.2.

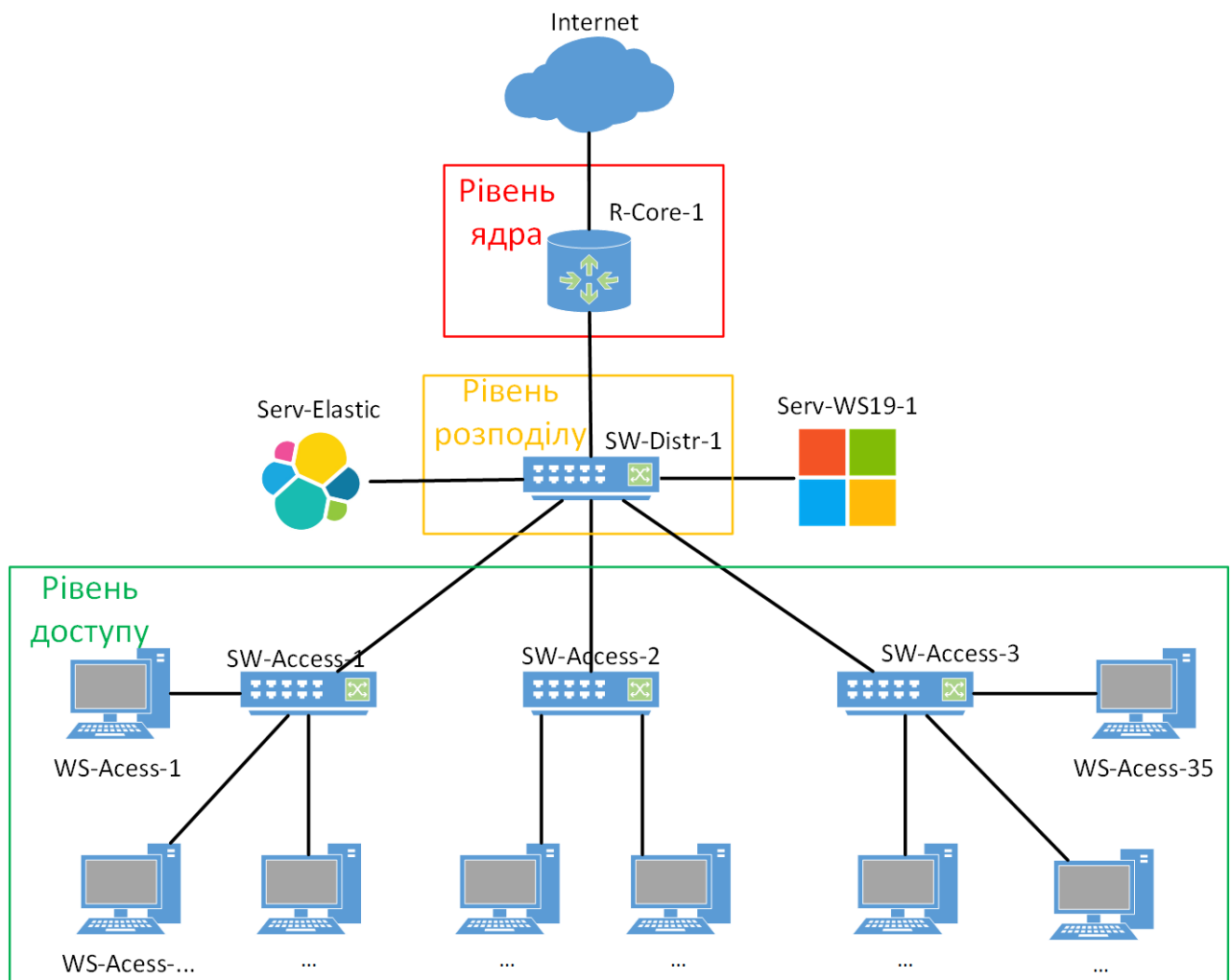


Рисунок 2.2 – Логічна топологія мережі

Детальніше розглянемо кожен рівень. На рівня ядра присутній маршрутизатор **R-Core-1**, який забезпечує доступ пристроїв корпоративної локальної мережі до Інтернету.

На рівні розподілу знаходиться комутатор **SW-Distr-1**, який забезпечує взаємодію між усіма іншими пристроями, а також зв'язок з маршрутизатором **R-Core-1**. До даного комутатора також підключене серверне обладнання, а саме робоча станція Ubuntu зі встановленим Elastic Stack та сервер на ОС Windows Server 2019 з налаштованою AD (Active Directory).

На рівні доступу розміщені комутатори SW-Acess-1, SW-Acess-2 та SW-Acess-3, до яких підключені клієнтські станції WS-Access-... через виту пару. Всі вищезгадані комутатори рівня доступу підключені до комутатора рівня розподілу SW-Dirst-1.

Використання трирівневої топології при розробці мережі забезпечує чітке розмежування функцій мережі: маршрутизатор на рівні ядра керує підключенням до Інтернету, комутатор на рівні розподілу забезпечує з'єднання між пристроями та управління трафіком, а до комутаторів на рівні доступу підключені робочі станції до загальної мережі. Така структура покращує ефективність мережі, полегшує адміністрування і забезпечує кращу масштабованість та надійність.

В загальному в мережі задіяно 1 маршрутизатор, 4 комутатори, 1 сервер з ОС Windows Server 2019, 1 станція Ubuntu зі встановленим Elastic Stack та 35 клієнтських станцій працівників.

Для забезпечення функціонування мережі потрібно використати дві мережі: одна між маршрутизатором та Інтернет-провайдером, а інша для локальної мережі. На маршрутизаторі можна налаштувати динамічне отримання адреси від провайдера, однак вже для локальної мережі необхідно зробити ретельний розподіл адрес для кожного пристрою. При чому мережеве обладнання та сервери повинні мати статичні адреси, а клієнтські комп'ютери – динамічні (від DHCP-сервера). В табл.2.1 наведено розрахунок кількості необхідних адрес між пристроями локальної мережі організації.

Таблиця 2.1

Розподіл адрес

Призначення адрес	IP-адреса
Адреса мережі	192.168.231.0
Маска мережі	255.255.255.0(/24)
Мінімальна адреса	192.168.231.1
Максимальна адреса	192.168.231.254
Шлюз за замовчуванням	192.168.231.254

Призначена Windows Server	192.168.231.240
Призначена станції з Elastic Stack	192.168.231.241
Призначені комутаторам	192.168. 231.250 – 192.168. 231.253
Діапазон адрес майбутніх пристроїв	192.168. 231.1 – 192.168. 231.35
Діапазон вільних адрес	192.168. 231.36 – 192.168. 231.239

На рис.2.3. наведено розподіл адрес між пристроями на топології мережі.

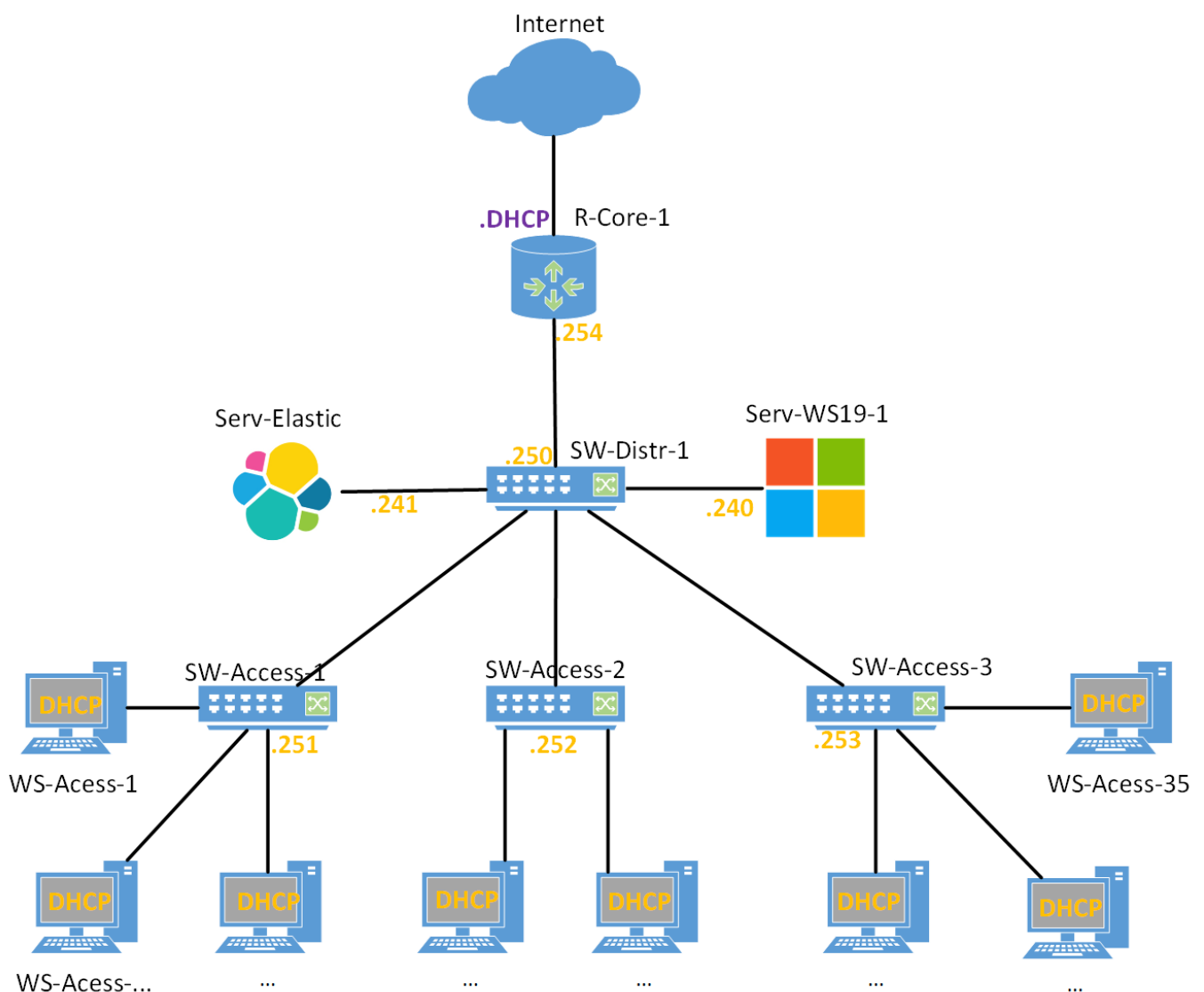


Рисунок 2.3 – Топологія мережі з адресацією

2.3 Розробка системи збору та обробки логів в Elastic Stack

Elastic пропонує два основні способи передачі даних до Elasticsearch [15]:

1) Beats — це легкі агенти для передачі операційних даних в Elasticsearch. Для різних типів даних, таких як журнали (логи), метрики та дані про доступність, існують окремі інструменти Beats. Залежно від того, які дані потрібно збирати, можливо, доведеться встановлювати кілька таких агентів на одному вузлі. До основних інструментів Beats належать Filebeat (логи і журнали програм), Metricbeat (метрики систем і служб), Packetbeat (мережевий трафік), Winlogbeat (журнали подій Windows), Auditbeat (дані аудиту Linux), Heartbeat (доступність системи). Кожен з них спеціалізується на зборі певного типу даних, що дозволяє ефективно моніторити і відправляти інформацію для аналізу в систему.

2) Elastic Agent — це універсальний агент, який дозволяє збирати і передавати дані про журнали, метрики, інформацію про безпеку та попередження про загрози в рамках єдиної системи. Elastic Agent спрощує управління збором даних, дозволяючи використовувати один застосунок на вузлі для одночасного збору кількох типів інформації, що значно зменшує потребу в установці окремих агентів для кожного виду даних.

З погляду на зручність та універсальність доцільно буде обрати метод збору даних з використанням Elastic Agent. Тим більше цими агентами можна централізовано і віддалено керувати з Elastic Stack і на власний розсуд обирати, які типи даних слід збирати, а які ні.

Для керування агентами використовується так званий Fleet Server – це компонент, що забезпечує підключення Elastic Agents до Fleet [16]. Він підтримує безліч підключень Elastic Agent і виступає як командний центр для оновлення політик агентів, збору інформації про їх стан та координації дій між агентами. Крім того, Fleet Server забезпечує масштабовану архітектуру: у міру зростання кількості агентів ви можете розгорнути додаткові Fleet Servers для управління збільшеним навантаженням. Крім цього Fleet Server дозволяє

встановити інтеграції – попередньо налаштовані модулі, які дозволяють збирати дані з різних джерел та служб без необхідності додаткових налаштувань. Ці інтеграції можуть обробляти логи, метрики та дані безпеки, забезпечуючи комплексне рішення для їх моніторингу та подальшої візуалізації.

На рис.2.4. наведено спосіб взаємодії компонентів Elastic Stack і Elastic Agents. Fleet Server управляє Elastic Agents, контролюючи їх підключення та поширюючи політики, які зберігаються в Elasticsearch. Elastic Agents реєструються через Fleet Server, отримують свої політики та використовують їх для збору й відправлення даних до Elasticsearch. Fleet Server також оновлює ці політики та координує роботу агентів, постійно підтримуючи зв'язок для передачі інформації про статус агентів і зміни політик. Elastic Agents, в свою чергу, можуть виконувати різноманітні функції: збирати логи, метрики, мережеву телеметрію, а також виконувати захист кінцевих точок від загроз (EDR). Все залежить від обраних інтеграцій.

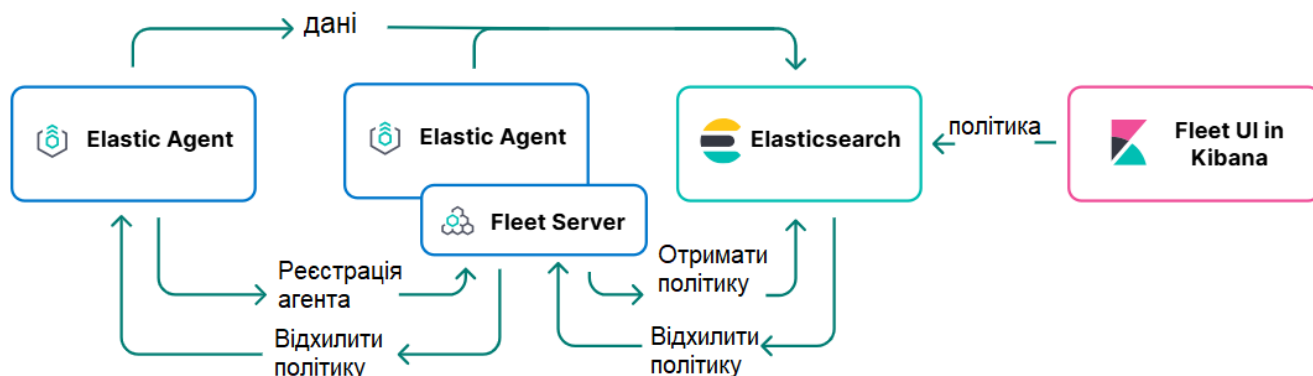


Рисунок 2.4 – Типова архітектура взаємодії агентів з Fleet Server

Тож зважаючи на згаданий спосіб з використанням агентів і буде базуватись система збору і візуалізації даних в проєкті мережі. На рис.2.5 умовними позначеннями показано процес передачі даних. На всіх кінцевих пристроях і Windows-сервері встановлено Elastic Agent, які будуть централізовано керуватись з Fleet Server, тобто з комп'ютера Serv-Elastic. Агенти з кінцевих пристроїв будуть передавати зібрану інформацію з пристроїв до Fleet Server, а сам Fleet Server буде займатись розсилкою політик і всіх необхідних

налаштувань до агентів. Таким чином на сервері Elastic буде здійснюватися централізоване керування, перегляд і обробка всіх даних, які будуть надходити від кінцевих пристроїв.

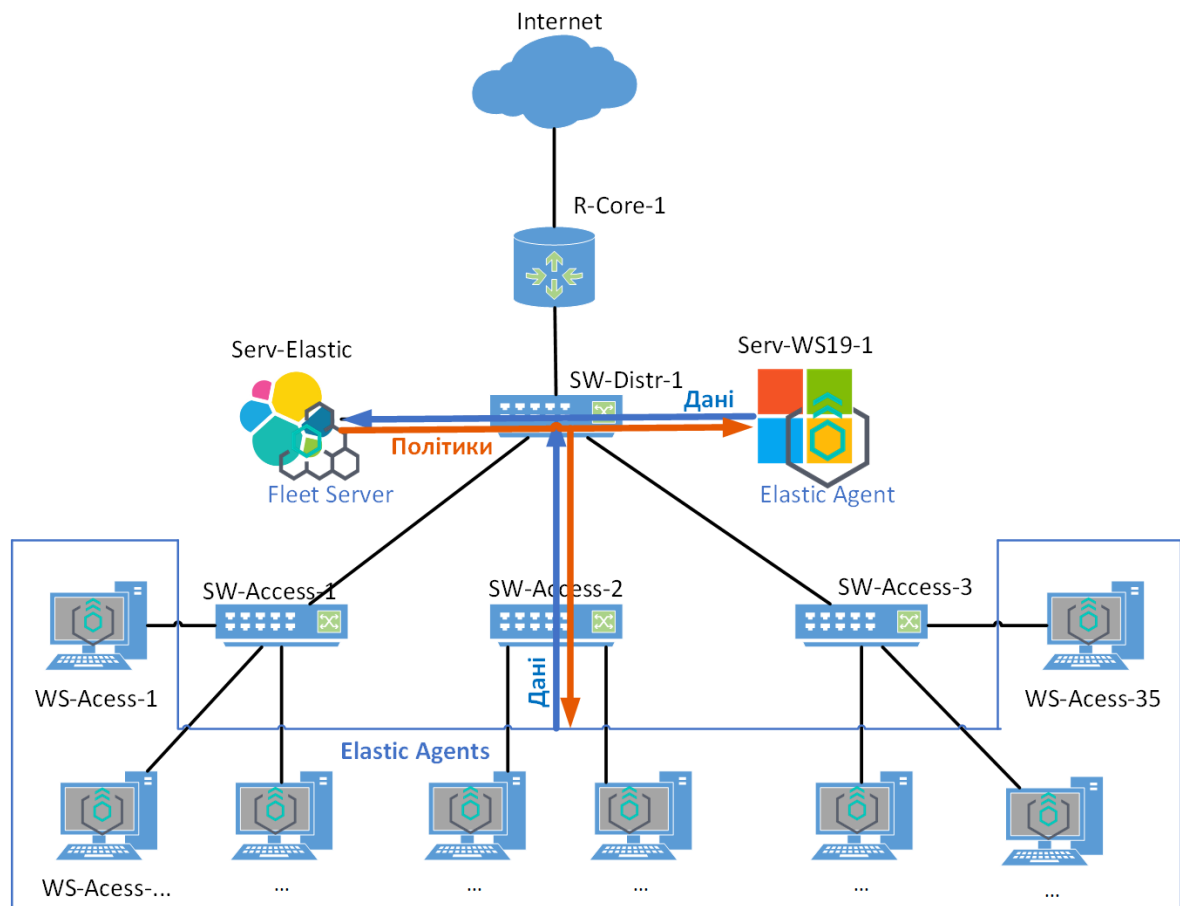


Рисунок 2.5 – Схема збору даних в мережі

Крім збору даних на сервері Elastic також здійснюватиметься їх аналіз і візуалізація за допомогою двох ключових компонентів: Elasticsearch та Kibana.

Elasticsearch — це розподілений пошуковий і аналітичний рушій та ядро всього Elastic Stack, який забезпечує швидкий і масштабований пошук та аналіз великих обсягів даних. Він підтримує індексацію як структурованих, так і неструктурованих даних, пропонує потужні можливості для їх аналізу та пошуку в реальному часі, а також дозволяє працювати через REST API.

Kibana — це інтерфейс для взаємодії з Elasticsearch, що надає інструменти для візуалізації, аналізу та моніторингу даних. Вона дозволяє створювати інтерактивні дашборди, візуалізувати інформацію у вигляді графіків та діаграм,

що значно спрощує аналіз даних. Kibana також забезпечує інструменти для моніторингу систем, управління індексами, налаштування сповіщень та безпеки.

2.4 Вибір обладнання та програмного забезпечення

При виборі маршрутизатора для корпоративної мережі невеликої організації варто зважати на пропускну здатність для необхідного обсягу трафіку, достатню кількість портів для підключень і безпеку. Важливо, щоб маршрутизатор мав підтримку VPN для захищеного віддаленого доступу, зручний інтерфейс для управління та можливість віддаленого адміністрування. Також маршрутизатор має бути масштабованим, тобто мати можливість додавати носії слоти для інтерфесів та модулі, щоб згодом можна було додати нові функції чи розширити його можливості.

Всі вищезгадані потреби може задовольнити маршрутизатор Cisco ISR4221/K9, рис.2.6.



Рисунок 2.6 – Маршрутизатор Cisco ISR4221/K9

Cisco ISR4221/K9 — це маршрутизатор із серії Cisco ISR 4000, розроблений для малого та середнього бізнесу [17]. Його конструкція поєднує високу продуктивність і багатофункціональність, що робить його надійним вибором для корпоративних мереж. Пристрій підтримує швидкість передачі даних до 35 Мбіт/с і забезпечує інтегровану безпеку, що включає функції міжмережевого екрану, VPN, контролю додатків, а також інших мережевих сервісів. Cisco ISR4221/K9 також дозволяє інтегрувати бездротові сервіси та забезпечує масштабування мережі при зростанні потреб компанії. Завдяки інтелектуальним функціям маршрутизації, він оптимізує роботу мережі, що

особливо корисно в умовах значного мережевого навантаження. Маршрутизатори даної серії підтримують роботу з протоколами NAT, IPSec VPN, RIP v1/v2, EIGRP, OSPF, BGP, PBR, PfR, а також адресацію IPv4 та IPv6. Маршрутизатори серії Cisco ISR4000 надають замовникам високоякісні користувацькі сервіси, високу масштабованість архітектури, та мінімальні витрати на встановлення.

При виборі комутатора для проекту мережі слід звернути увагу на достатню кількість портів для кінцевих пристроїв і достатню пропускну здатність. Окремо необхідно звернути увагу на безпеку. І важливо також, щоб комутатор був надійним і забезпечував хорошу технічну підтримку.

І одним з таких комутаторів є Cisco Catalyst WS-C2960X-24TS-L, рис.2.7, – керований комутатор, який призначений для використання в невеликих і середніх корпоративних мережах [18].



Рисунок 2.7 – Комутатор Cisco Catalyst WS-C2960X-24TS-L

Даний комутатор має 24 порти Ethernet 10/100/1000 Мбіт/с, що забезпечує високу швидкість передачі даних. Цей комутатор підтримує технології, такі як VLAN, що дозволяє сегментувати мережу для покращення управління трафіком. WS-C2960X-24TS-L оснащений функцією Power over Ethernet (PoE), що дозволяє живити підключені пристрої, наприклад IP-камери або телефони. Також підтримується QoS для управління пріоритетами трафіку, що важливо для забезпечення якості обслуговування, особливо в умовах високого навантаження. Комутатор має вбудовану функцію безпеки, зокрема аутентифікацію 802.1X, що забезпечує контроль доступу до мережі. Він також підтримує моніторинг та управління через SNMP, що спрощує адміністрування мережі. Cisco Catalyst WS-C2960X-24TS-L – це надійний вибір для середніх організацій, які потребують стабільного і гнучкого управління мережею.

Вибір комп'ютера з Windows Server 2019 для мережі обґрунтований його потужними можливостями управління, високим рівнем безпеки та підтримкою гібридних рішень, що дозволяє інтегрувати локальні ресурси з хмарними службами. Дана операційна система оптимізована для роботи з сучасним апаратним забезпеченням, що забезпечує підвищену продуктивність і ефективність використання ресурсів. Windows Server 2019 надає потужні можливості для управління Active Directory (AD), що є ключовим елементом для невеликих корпоративних мереж. Використання AD дозволяє централізовано управляти користувачами, комп'ютерами і політиками безпеки в організації. Це спрощує процес аутентифікації та авторизації, що підвищує безпеку та зручність для користувачів. Вдосконалені функції управління контейнерами спрощують розгортання нових додатків, роблячи Windows Server 2019 ідеальним вибором для невеликих організацій, які прагнуть до зростання та розвитку. Для комп'ютера, на якому буде встановлено Windows Server 2019, рекомендується процесор з мінімум двома ядрами, щонайменше 16 ГБ оперативної пам'яті (краще 32 ГБ або більше), а також SSD з обсягом не менше 250 ГБ для швидшого доступу до даних. Необхідно також мати Gigabit Ethernet картку для забезпечення швидкого з'єднання і надійний блок живлення.

Для комп'ютера, на якому планується розгортання Elastic Stack на ОС Ubuntu, рекомендуються дотримуватись наступних технічних вимог: чотириядерний процесор, не менше 8 ГБ оперативної пам'яті (бажано 16 ГБ і більше), а також SSD з обсягом не менше 256 ГБ для швидкого доступу до даних. Вибір Ubuntu для розгортання Elastic Stack замість Windows обґрунтований його відкритістю і гнучкістю, що дозволяє легше налаштовувати систему. Ubuntu має потужну спільноту підтримки і документації, що спрощує вирішення проблем. Крім того, Linux зазвичай вимагає менше ресурсів, ніж Windows, що підвищує ефективність використання апаратного забезпечення, що зрештою є досить актуальним в даному випадку.

Наостанок, кінцеві комп'ютери співробітників з ОС Windows 10 повинні мати такі параметри, що забезпечать стабільну продуктивність, безпеку та підтримку сучасного програмного забезпечення. Тому процесор має бути не нижче Intel Core i5 або його еквівалентом від AMD. Обсяг оперативної пам'яті має становити мінімум 8 ГБ, а оптимально – 16 ГБ. Об'єм SSD-диска рекомендовано не менше 256 ГБ для швидкого доступу до даних. Відеокарта інтегрованого типу підходить для офісних задач, але для роботи з мультимедіа чи графікою доцільно використовувати дискретні рішення. Обов'язковим є наявність мережевої карти Gigabit Ethernet. Також важливими є достатня кількість периферійних портів, таких як USB 3.0/3.1, HDMI чи DisplayPort для підключення додаткового обладнання. Усі ці параметри забезпечують відповідність сучасним вимогам, високу швидкість та надійність в корпоративному середовищі.

Висновки до другого розділу

В ході роботи над даним розділом було розглянуто цілі використання корпоративних мереж, їхні функції, переваги для бізнесу та пристрої які входять до її складу. Окремо було розглянуто трирівневу модель мережі (рівні доступу, розподілу і ядра), яка забезпечує зручність у масштабуванні, підвищену надійність та покращену пропускну здатність корпоративної інфраструктури.

Далі було виконано розробку топології мережі з дотриманням трирівневої топології мережі. В загальному в мережі задіяно 1 маршрутизатор, 4 комутатори, 1 сервер з ОС Windows Server 2019, 1 станція Ubuntu зі встановленим Elastic Stack та 35 клієнтських станцій працівників. І виходячи з необхідної кількості обладнання було виконано розподіл адрес між пристроями дотримуючи економії адресного простору.

Потім було виконано розробку системи збору та обробки даних в Elastic Stack в спроектованій мережі. Визначено, що Elastic Agents у поєднанні з пунктом керування Fleet Server на Elastic Stack зможуть найкраще забезпечити збір та передачу даних для подальшого їх опрацювання.

І на кінець було описано процес вибору оптимального обладнання та програмного забезпечення для корпоративної мережі, що включає маршрутизатор Cisco ISR4221/K9, комутатори Cisco Catalyst WS-C2960X-24TS-L, сервер з ОС Windows Server 2019, станцію з ОС Ubuntu для розгортання Elastic Stack та кінцеві пристрої з ОС Windows 10.

РОЗДІЛ 3 ПОБУДОВА, НАЛАГОДЖЕННЯ ТА ТЕСТУВАННЯ РОБОТИ КОРПОРАТИВНОЇ МЕРЕЖІ

3.1 Проєктування і налаштування роботи проєкту в GNS3

Демонстрація роботи проєкту мережі було виконано з використанням емулятора GNS3 і гіпервізора VirtualBox. GNS3 забезпечує гнучке моделювання мережевих топологій, а VirtualBox дозволяє легко розгортати віртуальні машини з різними ОС. Така комбінація дає можливість створити реалістичний сценарій для тестування проєкту. На табл.3.1 наведено параметри з'єднання пристроїв.

Таблиця 3.1

Параметри підключення пристроїв

Пристрій	Інтерфейс	Підключення до пристрою	Підключення до інтерфейсу
Internet	nat0	R-Core-1	Fa0/0
Маршрутизатор R-Core-1	Fa0/0	Internet	nat0
	Fa0/1	SW-Distr-1	Eth3/3
Комутатор SW-Distr-1	Eth0/0	Serv-Elastic	Eth0
	Eth0/1	Serv-WS19-1	Eth0
	Eth3/0	SW-Access-3	Eth3/3
	Eth3/1	SW-Access-2	Eth3/3
	Eth3/2	SW-Access-1	Eth3/3
	Eth3/3	R-Core-1	Fa0/1
Комутатор SW-Access-1	Eth0/0	WS-Acess-1	Eth0
	Eth0/1	WS-Acess-2	Eth0
	Eth3/3	SW-Distr-1	Eth3/2
Комутатор SW-Access-2	Eth0/0	WS-Acess-3	Eth0
	Eth0/1	WS-Acess-4	Eth0
	Eth3/3	SW-Distr-1	Eth3/1
Комутатор SW-Access-3	Eth0/0	WS-Acess-5	Eth0
	Eth0/1	WS-Acess-6	Eth0
	Eth3/3	SW-Distr-1	Eth3/0

На рис.3.1 наведено виконаний проєкт мережі в GNS3. Адресація пристроїв виконана згідно табл.2.1, а підключення згідно з табл.3.1.

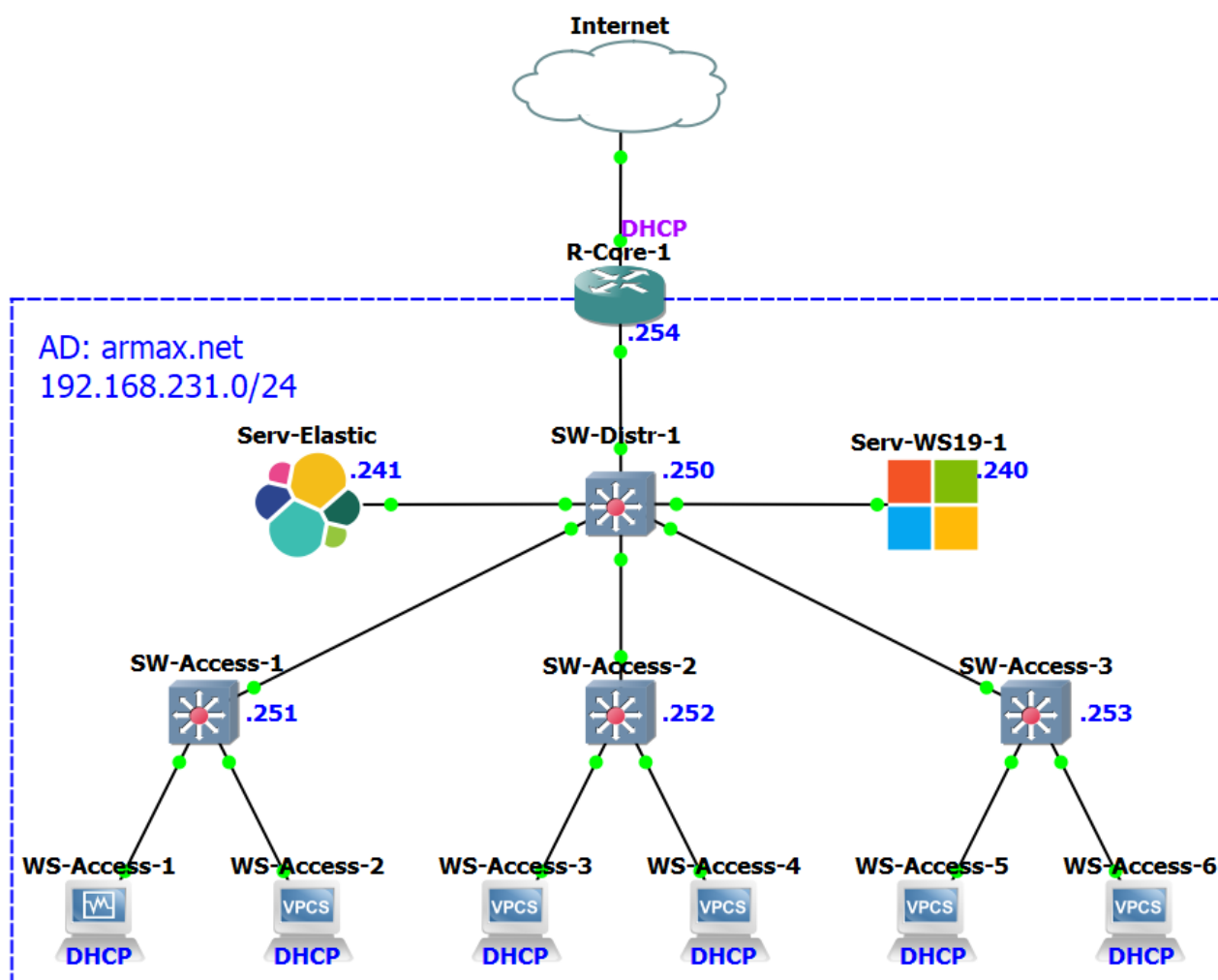


Рисунок 3.1 – Проєкт мережі в емуляторі GNS3

На єдиному маршрутизаторі R-Core-1 було налаштовано два інтерфейси для забезпечення підключення до Інтернету та внутрішньої мережі. Інтерфейс, який використовується для підключення до Інтернету отримує IP-адресу автоматично через DHCP, що спрощує налаштування та підключення. Інший інтерфейс має статичну IP-адресу для локальної мережі, що забезпечує надійну комунікацію з внутрішніми пристроями. Також було налаштовано назву домену (armax.net) та DNS-сервери (8.8.8.8 та 192.168.231.240) для підтримки функцій пошуку імен, що дозволяє маршрутизатору використовувати DNS для доступу до зовнішніх та внутрішніх ресурсів за їх іменами, рис.3.2.

```

R-Core-1#conf t
Enter configuration commands, one per line. End with CNTL/Z.
R-Core-1(config)#int fa0/0
R-Core-1(config-if)#description link-to-internet
R-Core-1(config-if)#ip address dhcp
R-Core-1(config-if)#no shutdown
R-Core-1(config-if)#exit
R-Core-1(config)#int fa0/1
R-Core-1(config-if)#description link-to-sw-distr-1
R-Core-1(config-if)#ip address 192.168.231.254 255.255.255.0
R-Core-1(config-if)#no shutdown
R-Core-1(config-if)#exit
R-Core-1(config)#ip domain-name armax.net
R-Core-1(config)#ip domain-lookup
R-Core-1(config)#ip dns server
R-Core-1(config)#ip name-server 8.8.8.8
R-Core-1(config)#ip name-server 192.168.231.240

```

Рисунок 3.2 – Налашдування адресації на маршрутизаторі R-Core-1

Далі на маршрутизаторі було налаштовано NAT (Network Address Translation), а саме у вигляді PAT (Port Address Translation) для забезпечення виходу внутрішніх IP-адрес у зовнішню мережу. Інтерфейс, підключений до Інтернету, позначено як nat outside, а інтерфейс внутрішньої мережі — як nat inside. Для внутрішньої мережі створено ACL (Access Control List), що дозволяє трафік з IP-адрес у діапазоні 192.168.231.0/24 для PAT. Далі було налаштовано NAT з перевантаженням також відомим як PAT, що дозволяє використовувати одну зовнішню IP-адресу для доступу декількох внутрішніх пристроїв до Інтернету, зберігаючи обмежений пул зовнішніх адрес і забезпечуючи доступ усіх пристроїв до пристроїв Інтернету.

```

R-Core-1(config)#int fa0/0
R-Core-1(config-if)#ip nat outside
R-Core-1(config-if)#exit
R-Core-1(config)#int fa0/1
R-Core-1(config-if)#ip nat inside
R-Core-1(config-if)#exit
R-Core-1(config)#access-list 1 permit 192.168.231.0 0.0.0.255
R-Core-1(config)#ip nat inside source list 1 interface fa0/0 overload

```

Рисунок 3.3 – Налашдування PAT на маршрутизаторі R-Core-1

Тепер переходимо до налаштувань комутаторів на прикладі SW-Distr-1. Тут було налаштовано базові параметри адресації. Зокрема, йому призначено IP-

адресу для забезпечення управління, задано шлюз за замовчуванням для зв'язку із зовнішніми мережами. Також було налаштовано доменне ім'я, включено функцію пошуку DNS і додано адреси DNS-серверів для забезпечення розпізнавання імен в мережі. Це дозволяє комутатору ефективно взаємодіяти з іншими мережевими пристроями та ресурсами.

```
SW-Distr-1(config)#int vlan 1
SW-Distr-1(config-if)#ip address 192.168.231.250 255.255.255.0
SW-Distr-1(config-if)#no shutdown
SW-Distr-1(config-if)#exit
SW-Distr-1(config)#ip default-gateway 192.168.231.254
SW-Distr-1(config)#no ip routing
SW-Distr-1(config)#ip domain-name armax.net
SW-Distr-1(config)#ip domain-lookup
SW-Distr-1(config)#ip name-server 192.168.231.254
SW-Distr-1(config)#ip name-server 192.168.231.240
```

Рисунок 3.4 – Налаштування адресації на комутаторі SW-Distr-1

Переходимо до налаштувань основного контролера домену, тобто до сервера Serv-WS19-1. Перш за все було виконано налаштування адресації даного пристрою, рис.3.5.

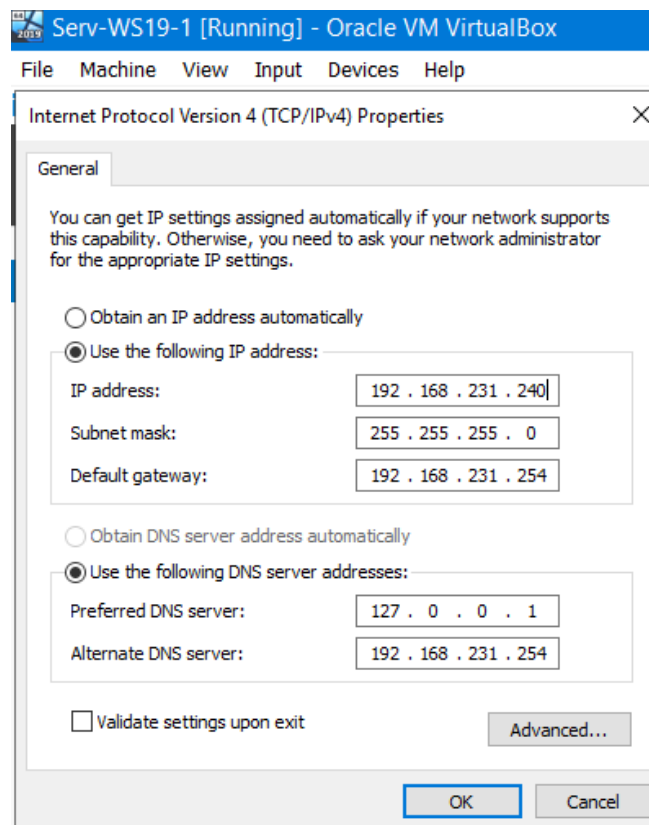


Рисунок 3.5 – Налаштування адресації на Serv-WS19-1

Далі йде створення домену armax.net, рис.3.6.

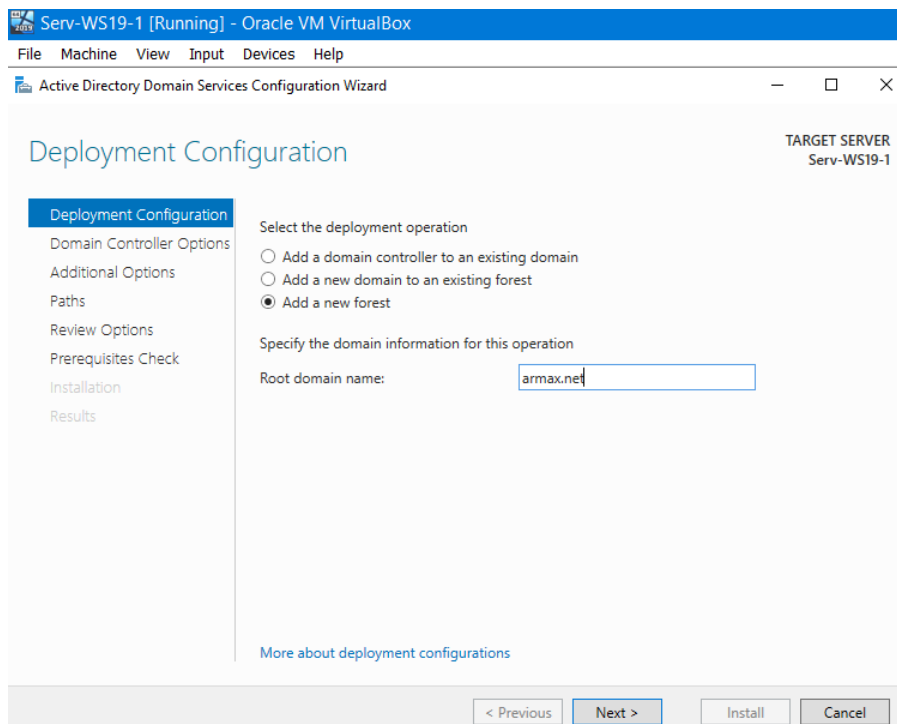


Рисунок 3.6 – Створення домену armax.net на Serv-WS19-1

І налаштування роботи DHCP-серверу. При цьому статичні адреси виключаються з області пулу, рис.3.7–3.11.

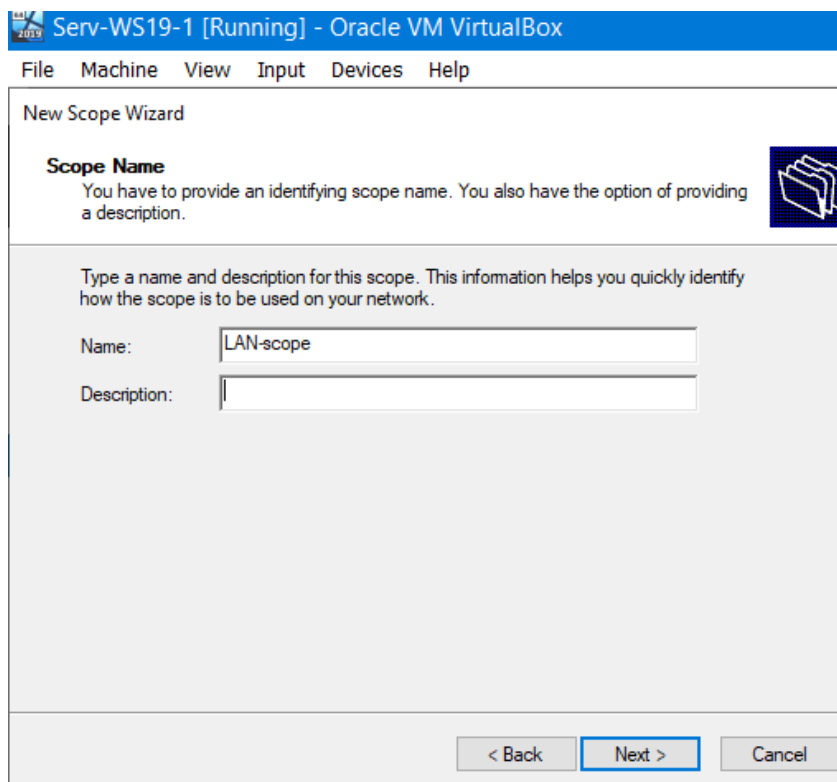


Рисунок 3.7 – Створення області LAN-scope

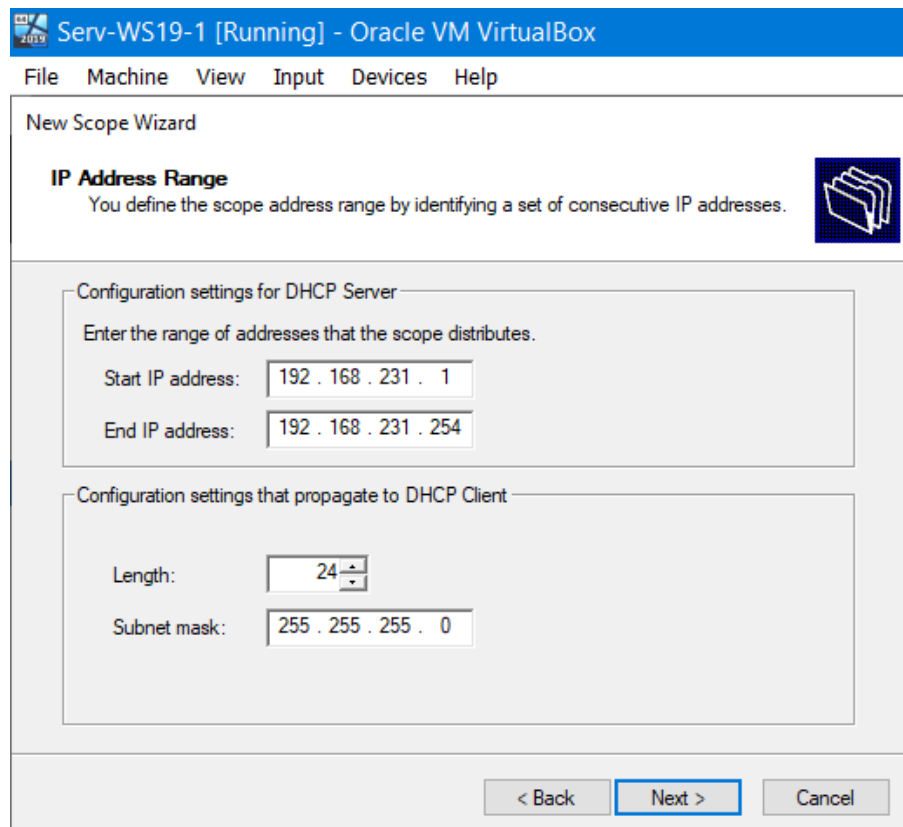


Рисунок 3.8 – Визначення діапазону роздачі адрес

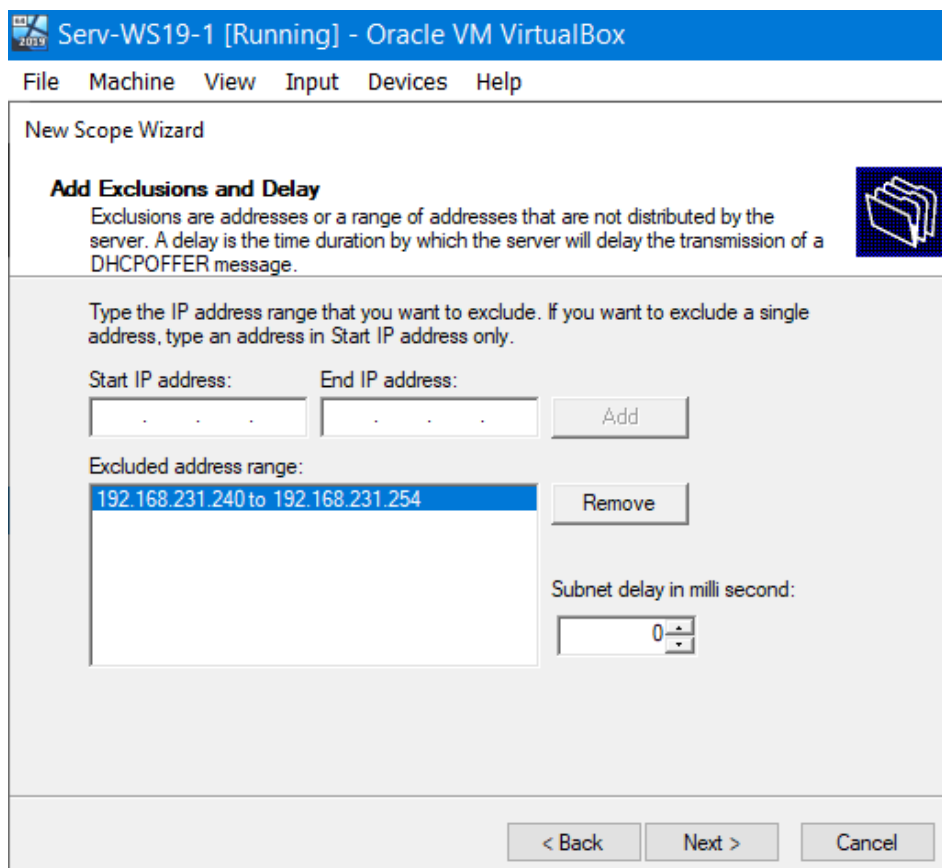


Рисунок 3.9 – Виключення статичних адрес з пулу

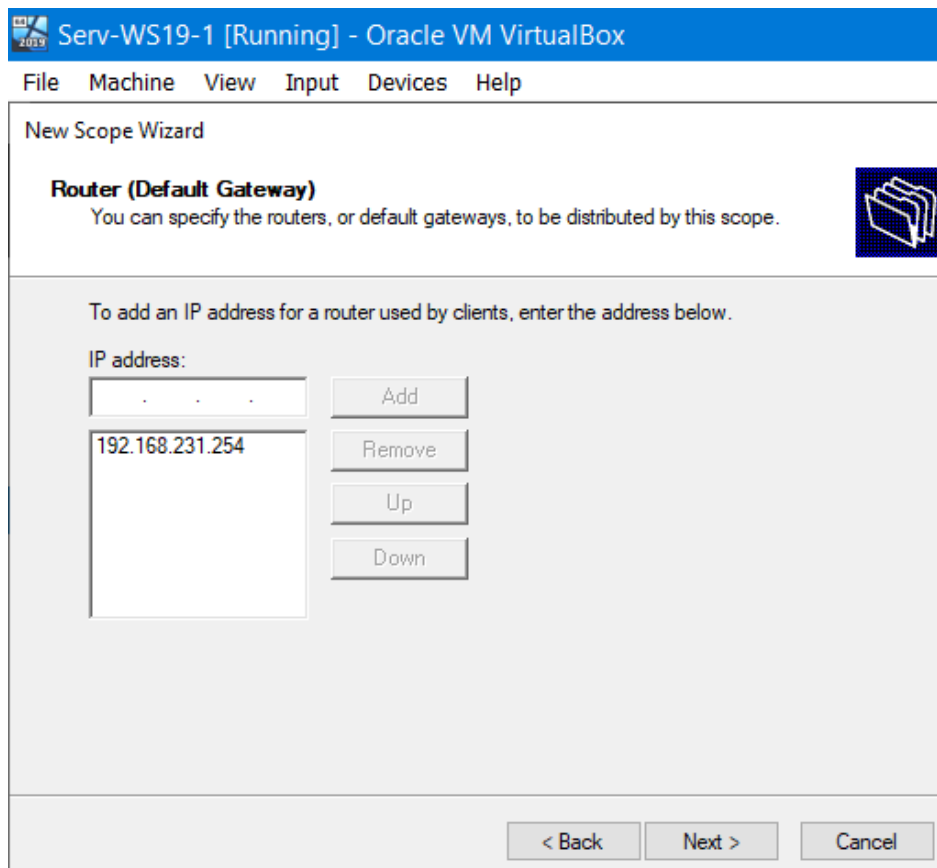


Рисунок 3.10 – Налаштування шлюзу за замовчуванням

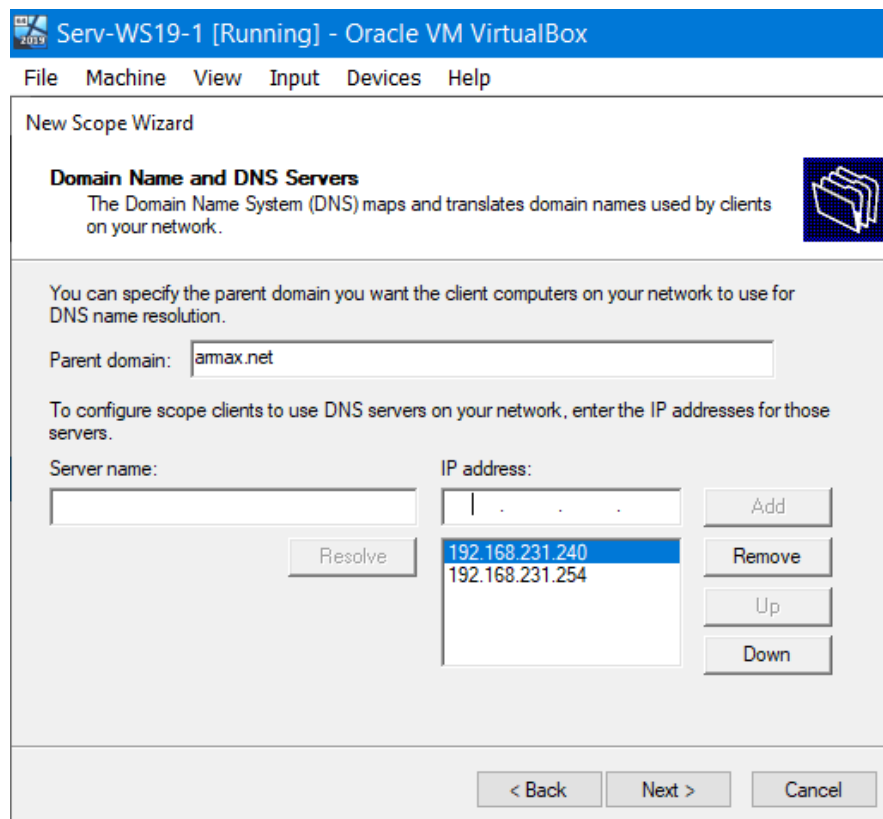


Рисунок 3.11 – Налаштування адрес DNS-серверів

Тепер на цьому ж сервері можна організувати роботу DNS-серверу і додати відповідні записи для мережевих пристроїв, аби до них можна було зручно звертатися за їх доменними іменами, рис.3.12. Записи ж для кінцевих комп'ютерів з'являться автоматично водночас з їх приєднанням до домену.

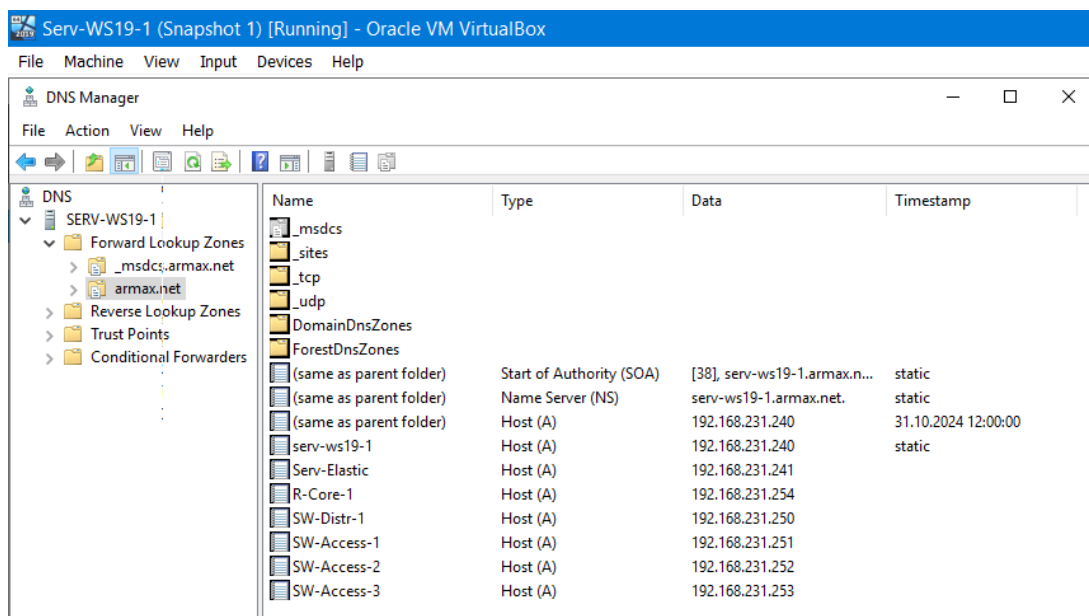


Рисунок 3.12 – Додання DNS записів типу А для мережевих пристроїв

І останнім лишається створити користувачів, які будуть використовуватися для входу на робочі станції, рис.3.13

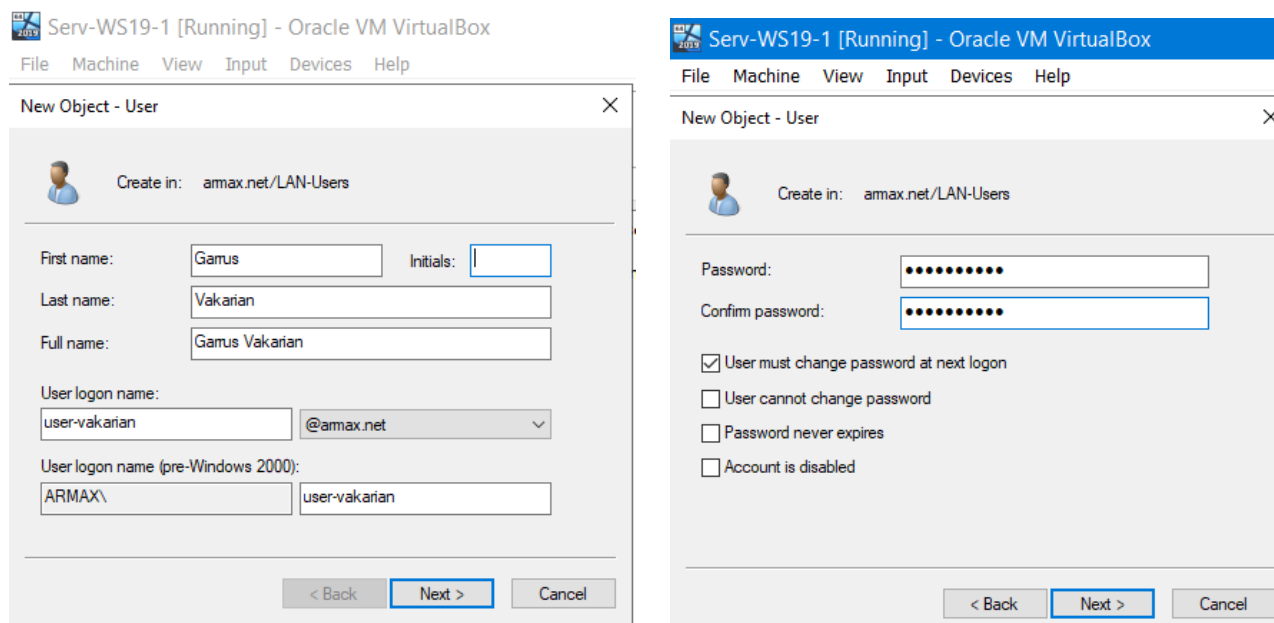


Рисунок 3.13 – Приклад створення користувача

Після того як було виконано основні налаштування на сервері можна перейти до кінцевих робочих станцій Windows. На прикладі WS-Access-1 налаштовуємо динамічне отримання адреси та виконуємо приєднання даного пристрою до домену armax.net, рис.3.14-3.15.

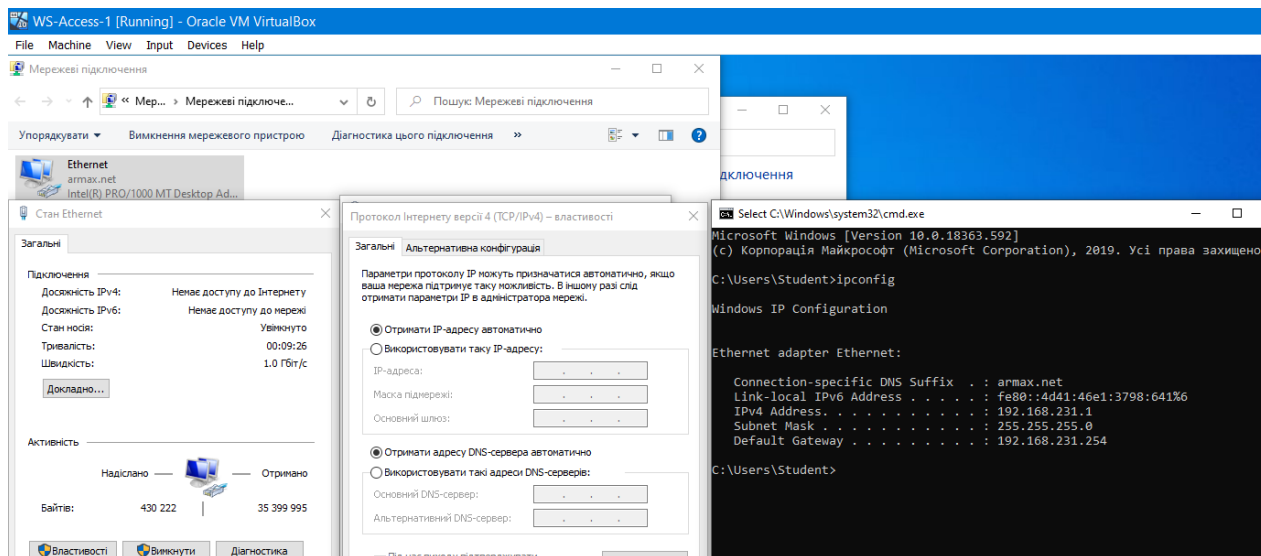


Рисунок 3.14 – Перевірка отримання динамічної адреси на робочій станції

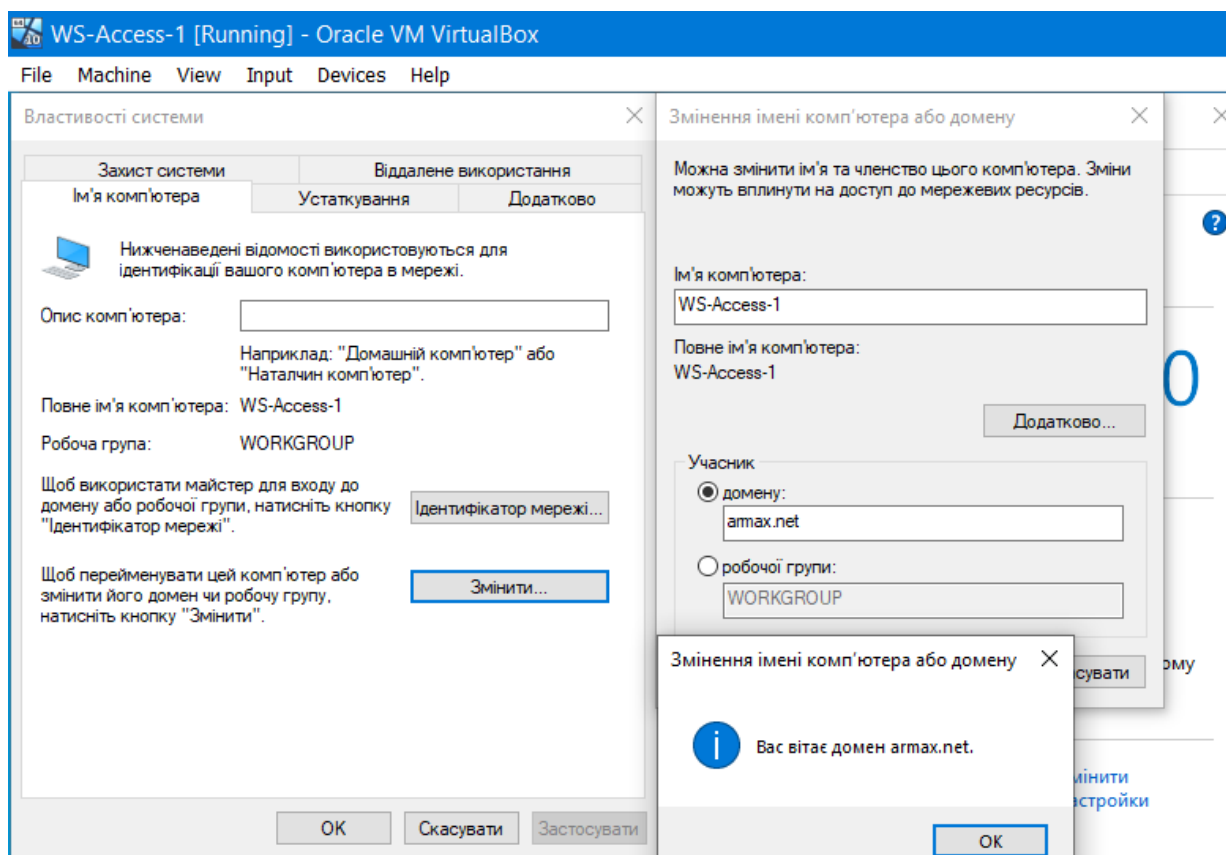


Рисунок 3.15 – Приєднання комп'ютера до домену armax.net

3.2 Налаштування Elastic Stack і агентів

Переходимо до розгортання Elastic Stack, тобто до станції Serv-Elastic з ОС Ubuntu. Перш за все тут налаштовується статична адресація, рис.3.16.

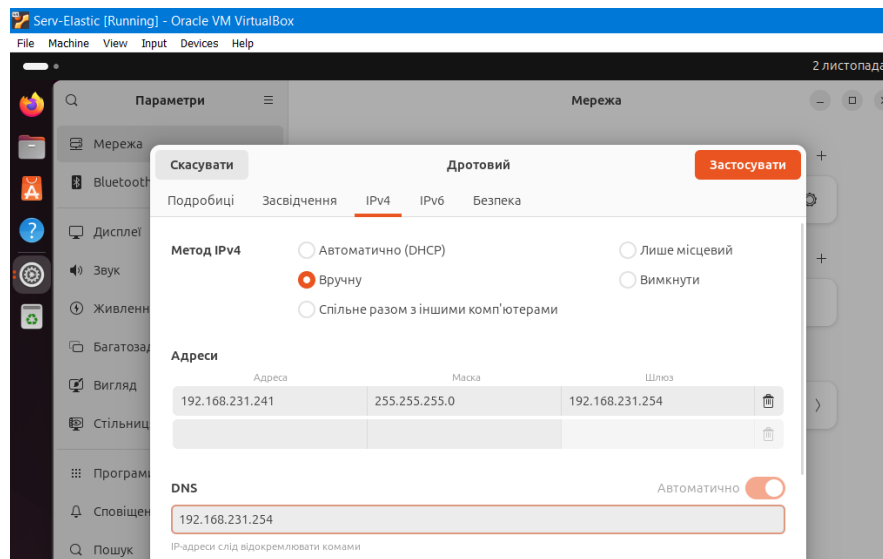


Рисунок 3.16 – Налаштування адресації

Розгортання Elastic Stack

Тепер переходимо до встановлення компоненту Elasticsearch відповідно до інструкції представленій на офіційному сайті [19], рис.3.17.

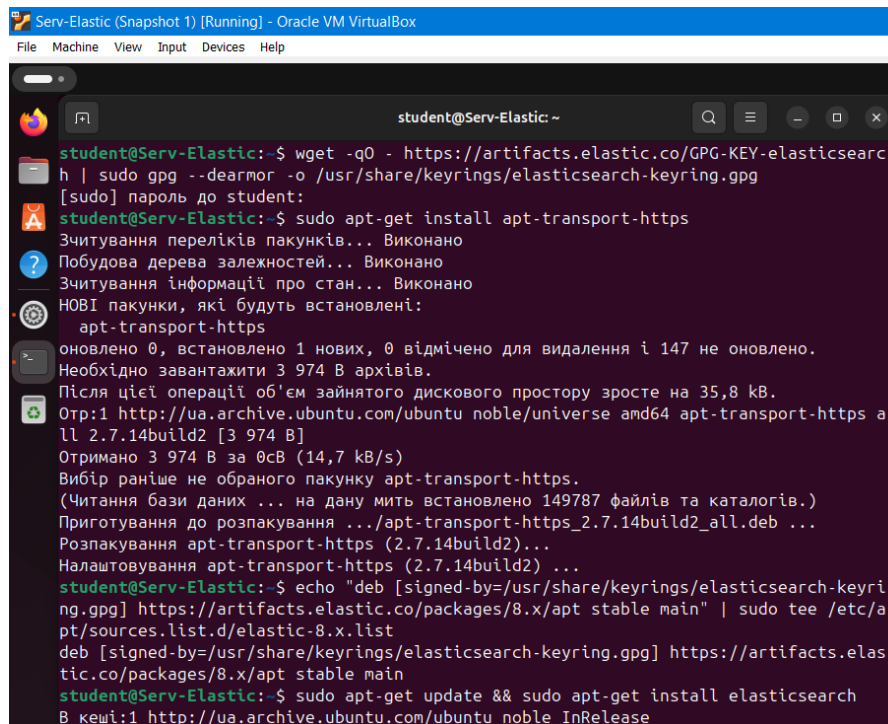
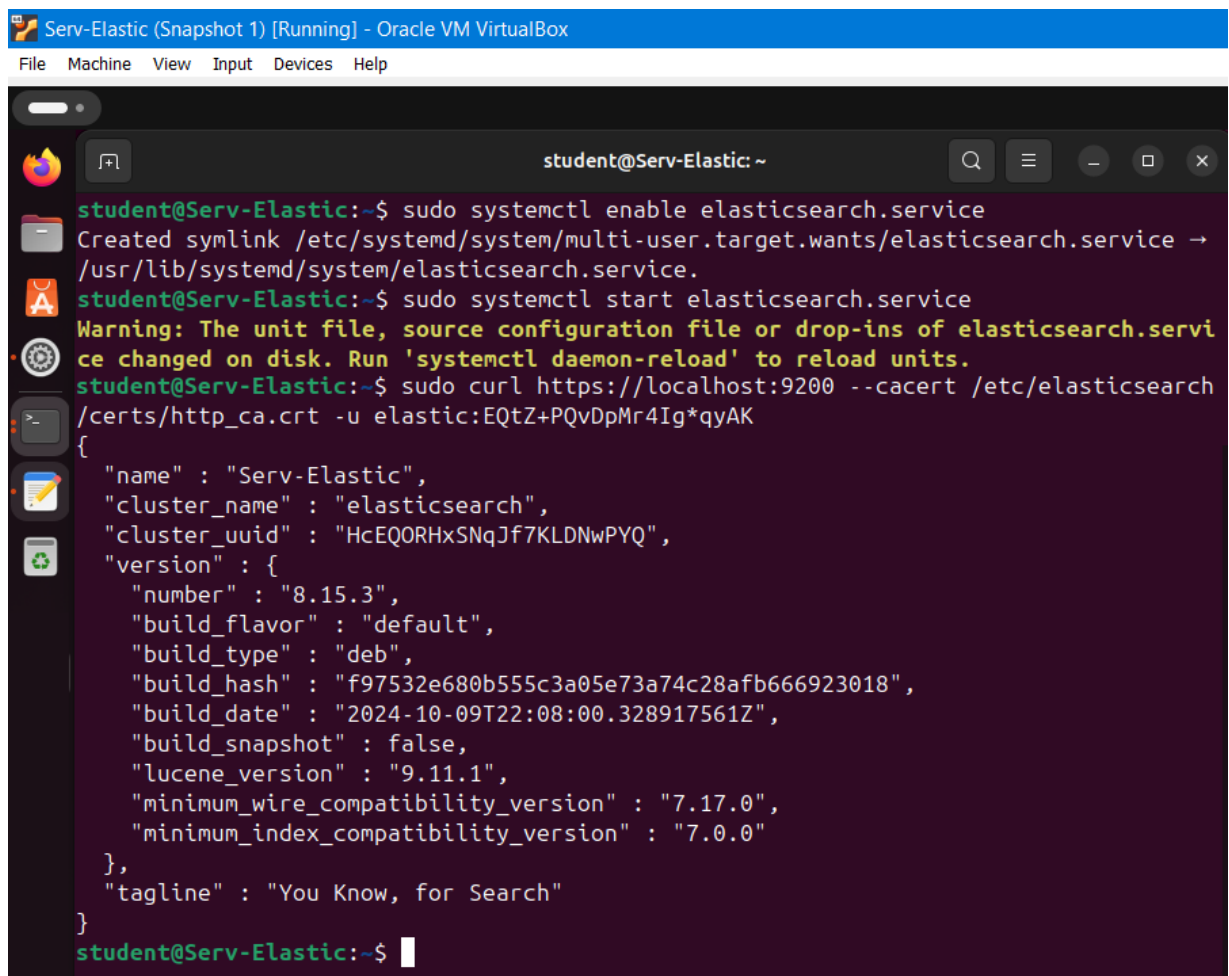


Рисунок 3.17 – Встановлення Elasticsearch

Використовуючи «systemctl enable» вказуємо, щоб дана служба автоматично запускалась при включенні комп'ютеру і безпосередньо включаємо роботи командою «systemctl start». З використанням команди «curl» отримуємо вміст веб-сторінки, що свідчить про правильне встановлення Elasticsearch, рис.3.18. Пароль для входу в систему було отримано під час встановлення, однак рекомендовано буде його змінити. Замінити адресу, хостнейм або здійснити інші додаткові налаштування Elasticsearch можна за допомогою файлу конфігурацій elasticsearch.yml.



```
Serv-Elastic (Snapshot 1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help

student@Serv-Elastic: ~
student@Serv-Elastic:~$ sudo systemctl enable elasticsearch.service
Created symlink /etc/systemd/system/multi-user.target.wants/elasticsearch.service → /usr/lib/systemd/system/elasticsearch.service.
student@Serv-Elastic:~$ sudo systemctl start elasticsearch.service
Warning: The unit file, source configuration file or drop-ins of elasticsearch.service changed on disk. Run 'systemctl daemon-reload' to reload units.
student@Serv-Elastic:~$ sudo curl https://localhost:9200 --cacert /etc/elasticsearch/certs/http_ca.crt -u elastic:EQtZ+PQvDpMr4Ig*qyAK
{
  "name" : "Serv-Elastic",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "HcEQORHxSNqJf7KLDNwPYQ",
  "version" : {
    "number" : "8.15.3",
    "build_flavor" : "default",
    "build_type" : "deb",
    "build_hash" : "f97532e680b555c3a05e73a74c28afb666923018",
    "build_date" : "2024-10-09T22:08:00.328917561Z",
    "build_snapshot" : false,
    "lucene_version" : "9.11.1",
    "minimum_wire_compatibility_version" : "7.17.0",
    "minimum_index_compatibility_version" : "7.0.0"
  },
  "tagline" : "You Know, for Search"
}
student@Serv-Elastic:~$
```

Рисунок 3.18 – Запуск служби Elasticsearch і перевірка роботи

Залишається виконати встановлення Kibana, яке теж здійснюється згідно настанов на офіційному сайті. Тут слід відмітити, що утилітою «elasticsearch-create-enrollment-token» генерується спеціальний токен для першого входу в Kibana, а інструментом «kibana-encryption-keys» створюються спеціальні ключи

шифрування, які слід додати у файл «kibana.yml». Далі аналогічно до Elasticsearch викоується запуск служби Kibana, рис.3.19. На цьому налаштування в консолі можна вважати завершеними. Надалі всі налагодження виконуватиметься переважно через веб-інтерфейс Kibana.

```

student@Serv-Elastic: ~
student@Serv-Elastic:~$ sudo apt install kibana
Зчитування переліків пакунків... Виконано
Побудова дерева залежностей... Виконано
Зчитування інформації про стан... Виконано
НОВІ пакунки, які будуть встановлені:
  kibana
оновлено 0, встановлено 1 нових, 0 відмічено для видалення і 145 не оновлено.
Необхідно завантажити 341 MB архівів.
Після цієї операції об'єм зайнятого дискового простору зросте на 1 016 MB.
Отр:1 https://artifacts.elastic.co/packages/8.x/apt/stable/main/amd64/kibana/amd64/8.15.3 [341 MB]
Отримано 341 MB за 1хв 51сВ (3 078 kB/s)
Вибір раніше не обраного пакету kibana.
(Читання бази даних ... на дану мить встановлено 151198 файлів та каталогів.)
Приготування до розпакування .../kibana_8.15.3_amd64.deb ...
Розпакування kibana (8.15.3)...
Налаштовування kibana (8.15.3) ...
Creating kibana group... OK
Creating kibana user... OK
Kibana is currently running with legacy OpenSSL providers enabled! For details and instructions on how to disable see https://www.elastic.co/guide/en/kibana/8.15/production.html#openssl-legacy-provider
Created Kibana keystore in /etc/kibana/kibana.keystore
student@Serv-Elastic:~$ sudo /usr/share/elasticsearch/bin/elasticsearch-create-enrollment-token -s kibana
eyJ2ZXIiOiI0LjE0LjAiLCJhZHIiOiIsImTAuMC4yLjE0IjkyMDAixSwiZmdyIjoiYzVhOTc1MTE0NThjZDQzYjIwMzRiNWVkMDkwZGE2ZDFLZmU3NDc0ZTYwOWM1YzE3MmY0NmU5MmM2NWQyNjU0NCIsImtleSI6IkQNGUtsKlCczQyVVhUSmswOFBoOmkwVWZMNHlfukJtRjJiOXBwdUIzbVEifQ==
student@Serv-Elastic:~$ sudo /usr/share/kibana/bin/kibana-encryption-keys generate
## Kibana Encryption Key Generation Utility

Settings:
xpack.encryptedSavedObjects.encryptionKey: 18e07e51152d3199fa9500e6c4c64b7d
xpack.reporting.encryptionKey: ee7de89f6a7380615fde09a8f7b4d5e7
xpack.security.encryptionKey: 1361d5c49fe3e188b2579edad0c8eaae

```

Рисунок 3.19 – Встановлення Kibana

Після цього переходимо за посиланням <http://serv-elastic:5601/>. У вікні вводимо токен який щойно був згенерований та логін з паролем для elastic. Якщо все було виконано правильно, то отримуємо наступне вікно, рис.3.20. Початкове налаштування Elastic Stack на цьому можна вважати закінченим,

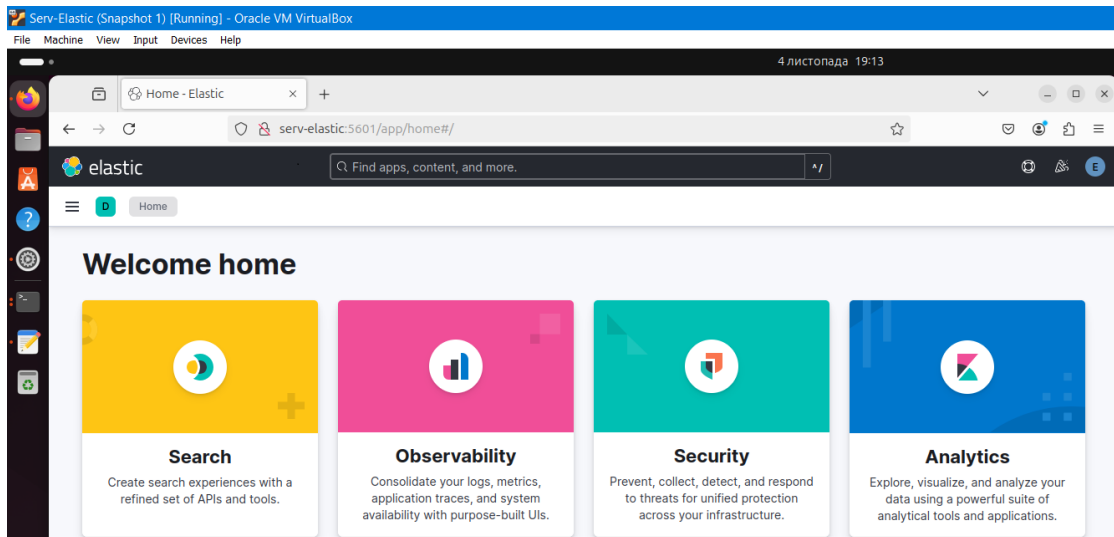


Рисунок 3.20 – Веб-інтерфейс Kibana

Налаштування Elastic Fleet та встановлення агентів

Переходимо до налаштувань Elastic Fleet з метою впровадження централізованого пункту керування агентами та інтеграціями. Перш за все створюємо Fleet Server, який також буде розміщуватись на даному комп'ютері, але за портом 8220, рис.3.21.

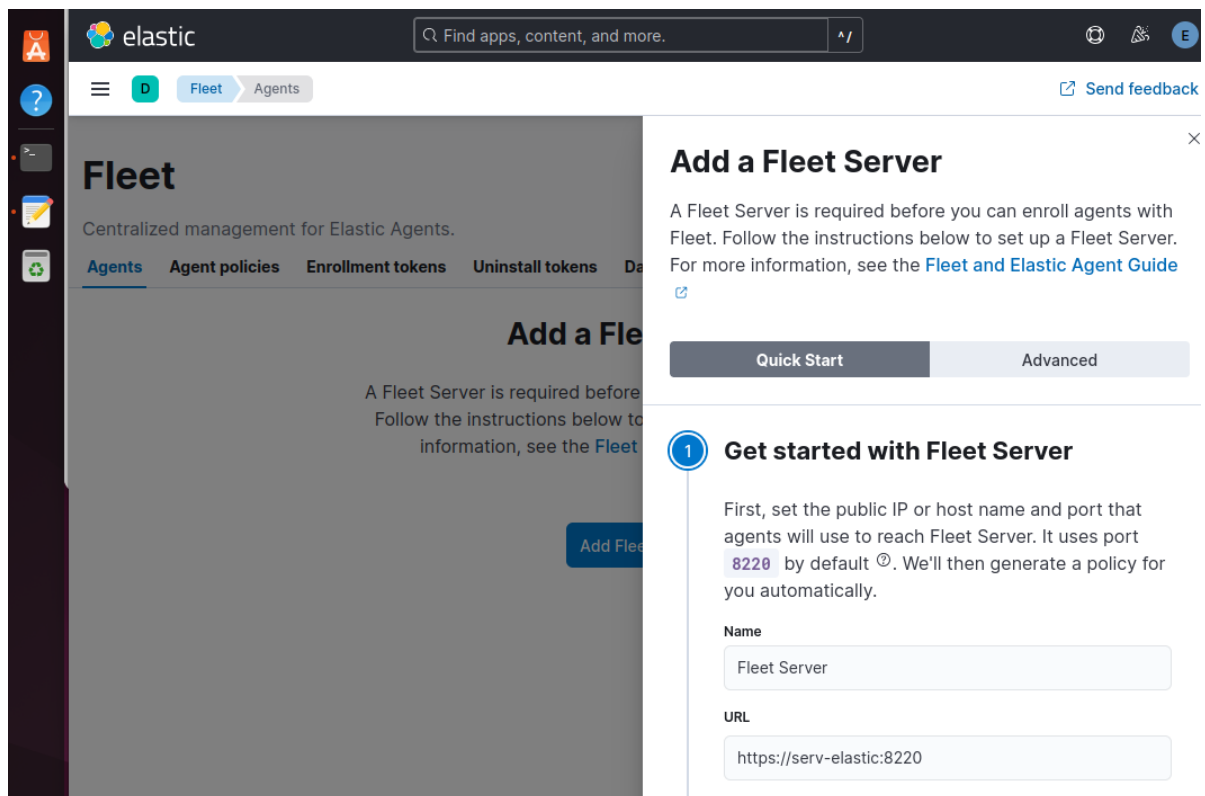


Рисунок 3.21 – Ініціалізації Fleet Server

Далі майстер налаштувань пропонує автоматично запропонує ввести наступні команди, які встановлять агент на даний пристрій і тим самим налаштують роботу Fleet Server, рис.3.22.

```

student@Serv-Elastic: ~$ curl -L -O https://artifacts.elastic.co/downloads/beats/elastic-agent/elastic-agent-8.15.3-linux-x86_64.tar.gz
student@Serv-Elastic: ~$ tar xzvf elastic-agent-8.15.3-linux-x86_64.tar.gz
student@Serv-Elastic: ~$ cd elastic-agent-8.15.3-linux-x86_64
student@Serv-Elastic: ~$ sudo ./elastic-agent install \
--fleet-server-es=https://192.168.231.241:9200 \
--fleet-server-service-token=AAEAAWVsYXN0aWVzMmxlZXQtc2VydMvYl3Rva2VuLTE3MzA1NzYwMDMwOTA6STNXU2ZuSm9UT2ExMXdxU0R0eU10QQ \
--fleet-server-policy=fleet-server-policy \
--fleet-server-es-ca-trusted-fingerprint=ae2dbd1b81dab799a4b63f996547392d298dd89a1de5b1085d74d977b924aa37 \
--fleet-server-port=8220
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left   Speed
100 299M  100 299M    0     0  2887k      0  0:01:46  0:01:46 --:--:-- 2854k
elastic-agent-8.15.3-linux-x86_64/elastic-agent.reference.yml
elastic-agent-8.15.3-linux-x86_64/.build_hash.txt
elastic-agent-8.15.3-linux-x86_64/data/elastic-agent-619758/package.version
  
```

Рисунок 3.22 – Виконання команд для ініціалізації Fleet Server

Якщо все пройшло успішно, то побачимо даний комп’ютер в меню Fleet зі статусом «Healthy» і навіть вже зараз можемо побачити певні отримані дані, рис.3.23. Це значить, що тепер можна почати розгортання агентів на кінцевих пристроях.

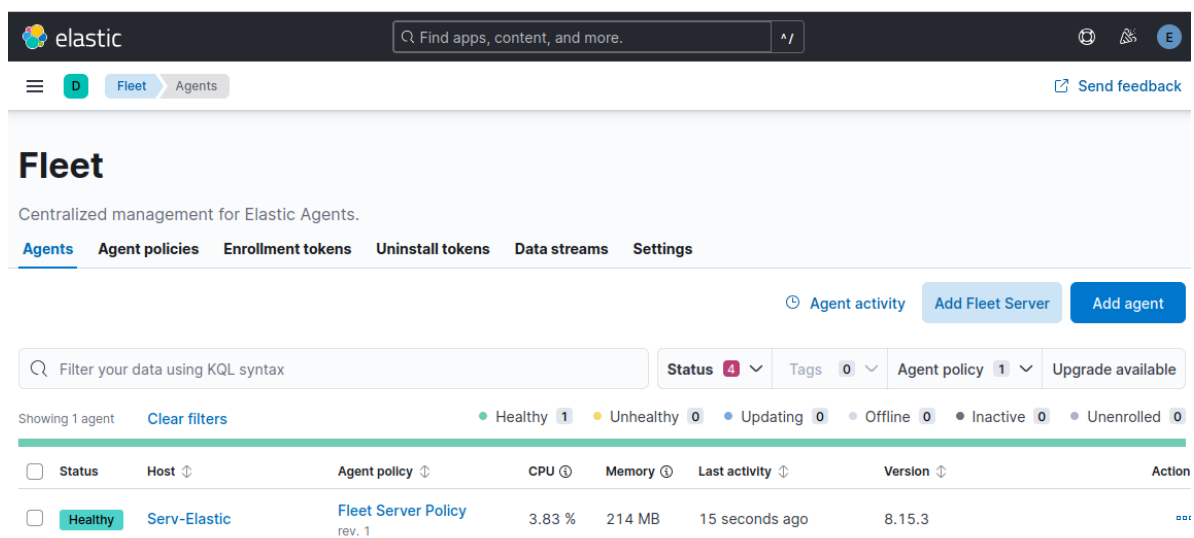


Рисунок 3.23 – Fleet Server успішно налаштовано

Додання агентів на кінцеві вузли досить схоже на попередні кроки. Головною відмінністю є те, що вузлам з ОС Windows буде призначено і створено окрему політику агентів – «Agent-policy-windows», рис.3.24. Це дозволяє кожній умовній групі пристроїв додавати лише необхідний для них набір інтеграцій.

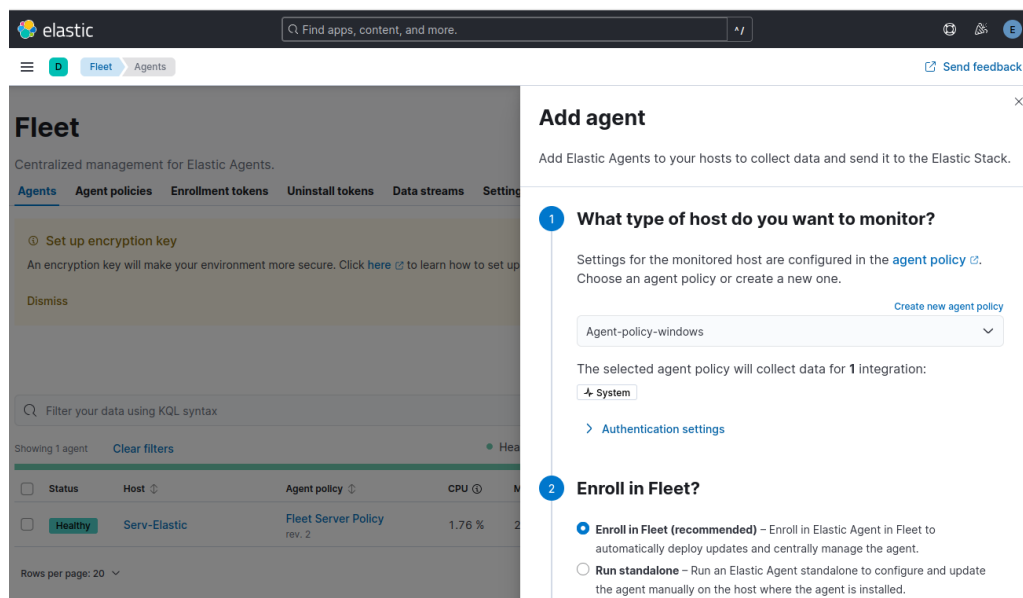


Рисунок 3.24 – Створення політики агентів для Windows

І тепер на кінцевому пристрої, в даному випадку Serv-WS19-1, виконуємо в терміналі набір команд які запропонував майстер налаштувань. В результаті агент було успішно встановлено, рис.3.25.

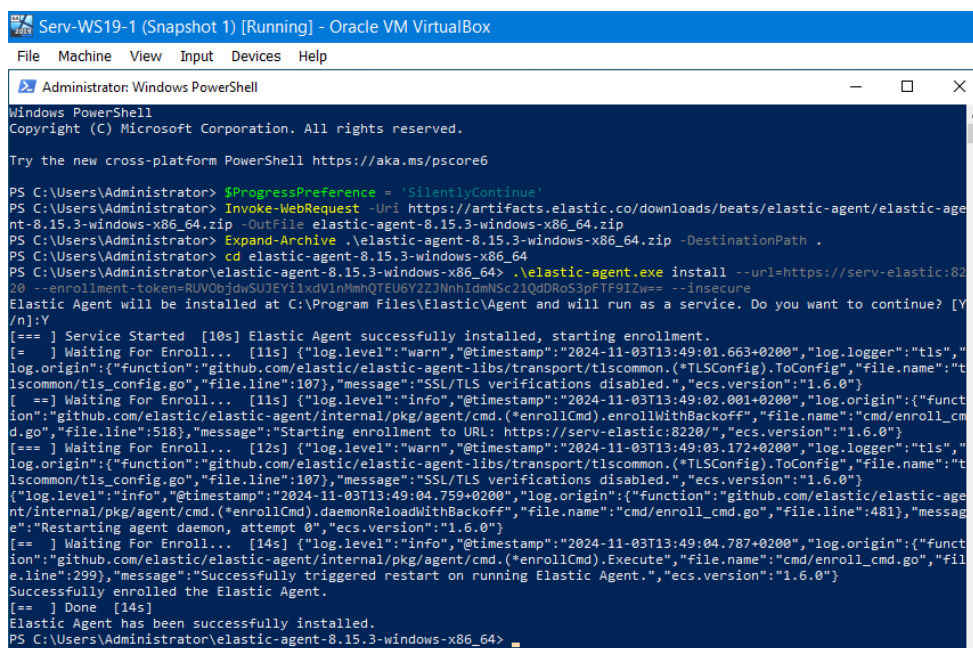


Рисунок 3.25 – Успішне встановлення агента на Serv-WS-19-1

Аналогічно до Serv-WS-19-1 виконується додавання агентів для всіх інших кінцевих пристроїв. В даному випадку це було зроблено для WS-Access-1. В результаті маємо наступний вигляд пункту Fleet, де статус «Healthy» означає успішне встановлення і роботу агента, рис.3.26. Після цього все керування агентами буде здійснюватися лише з даного меню без прямого втручання на кінцевих пристроях.

The screenshot shows the Elastic Fleet management page. At the top, there's a search bar and navigation tabs for Fleet and Agents. Below the 'Fleet' header, there are tabs for Agents, Agent policies, Enrollment tokens, Uninstall tokens, Data streams, and Settings. The 'Agents' tab is active, showing a list of agents. The list includes columns for Status, Host, Agent policy, CPU, Memory, Last activity, Version, and Actions. Three agents are listed, all with a 'Healthy' status: WS-Access-1, Serv-WS19-1, and Serv-Elastic. Each agent row has a dropdown menu for actions.

Status	Host	Agent policy	CPU	Memory	Last activity	Version	Actions
Healthy	WS-Access-1	Agent-policy-windows rev. 1	11.17 %	150 MB	24 seconds ago	8.15.3	...
Healthy	Serv-WS19-1	Agent-policy-windows rev. 1	0.98 %	169 MB	35 seconds ago	8.15.3	...
Healthy	Serv-Elastic	Fleet Server Policy rev. 2	1.63 %	213 MB	35 seconds ago	8.15.3	...

Рисунок 3.26 – Доступні агенти

Встановлення інтеграцій

Інтеграції в Elastic Stack – це готові налаштування та інструменти для збору, обробки та візуалізації даних із різних джерел. На даний момент для агентів Elastic доступно 376 інтеграцій для різних сервісів та цілей, рис.3.27. Їх встановлення не вимагає будь-яких дій на комп'ютерах клієнтів. Додавання чи видалення бажаних інтеграцій відбувається автоматично на всіх агентах. Це значно спрощує управління корпоративною мережею та моніторинг інфраструктури, оскільки інтеграції дозволяють централізовано налаштовувати збір логів, метрик та інших даних з пристроїв мережі.

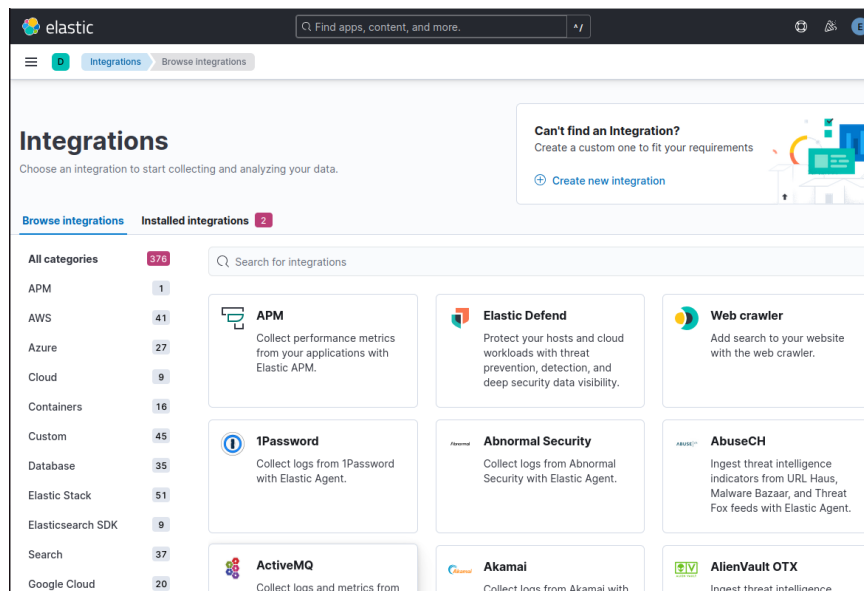


Рисунок 3.27 – Перелік доступних інтеграцій для агентів Elastic

В межах даної роботи для станцій на ОС Windows нас цікавлять наступні інтеграції, які будуть встановлені для політики «Agent-policy-windows», рис.3.28:

Elastic Defend – виконує збір подій пов’язаних з безпекою і надає функції EDR (Endpoint Detection and Response).

Network Packet Capture – збір мережових пакетів що надходять або виходять з пристроїв.

System – збір журналів та метрик з комп’ютерів (вже встановлений за замовчуванням для всіх агентів).

Windows – збір журналів та метрик з ОС Windows.

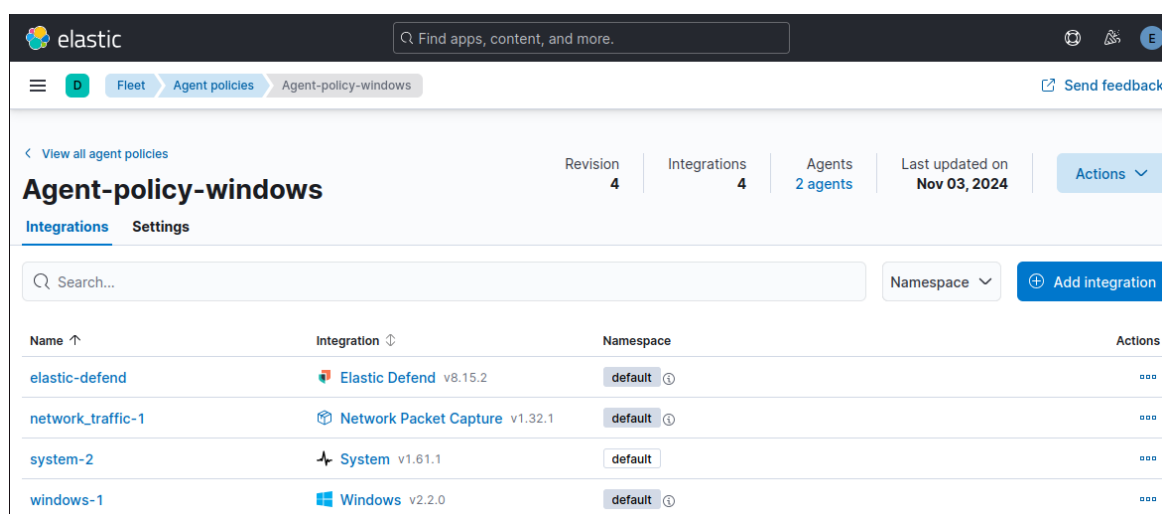


Рисунок 3.28 – Встановленні інтеграції для Agent-policy-windows

3.3 Тестування роботи мережі та Elastic Stack

Перш за все виконаємо вхід на одну з робочих станцій з використання облікового запису користувача, що був створений на контролері домену.

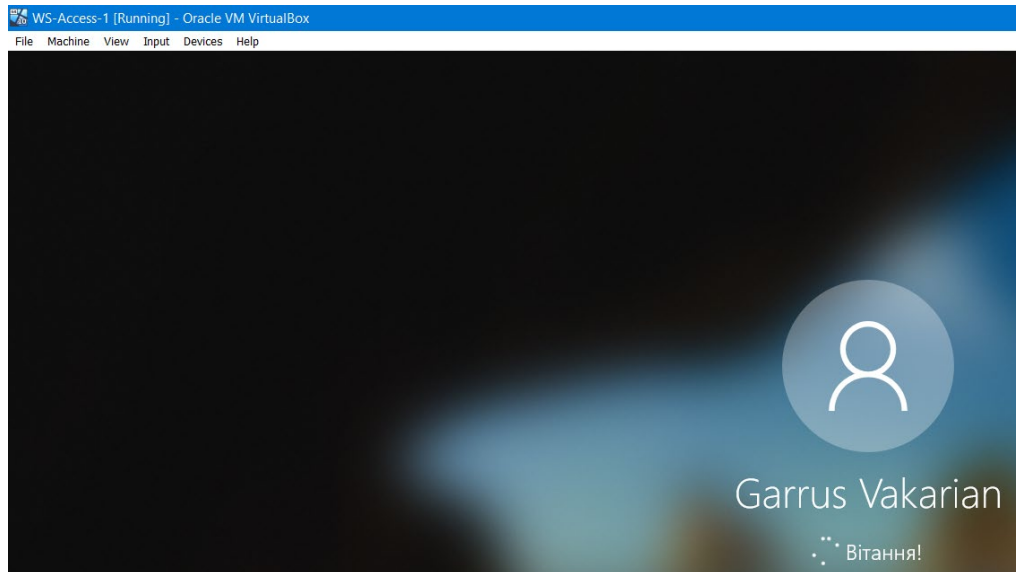


Рисунок 3.29 – Вхід в систему

Далі перевіримо наявність зв'язку між пристроями мережі та роботу DNS-серверу виконавши пінги за їх доменними іменами, рис.3.30.

```
C:\Windows\system32\cmd.exe
C:\Users\user-vakarian>ping serv-elastic

Pinging serv-elastic.armax.net [192.168.231.241] with 32 bytes of data:
Reply from 192.168.231.241: bytes=32 time=1ms TTL=64
Reply from 192.168.231.241: bytes=32 time=1ms TTL=64
Reply from 192.168.231.241: bytes=32 time=1ms TTL=64
Reply from 192.168.231.241: bytes=32 time=1ms TTL=64

Ping statistics for 192.168.231.241:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 1ms, Maximum = 1ms, Average = 1ms

C:\Users\user-vakarian>ping serv-ws19-1

Pinging Serv-WS19-1.armax.net [192.168.231.240] with 32 bytes of data:
Reply from 192.168.231.240: bytes=32 time=2ms TTL=128
Reply from 192.168.231.240: bytes=32 time=1ms TTL=128
Reply from 192.168.231.240: bytes=32 time=2ms TTL=128
Reply from 192.168.231.240: bytes=32 time=1ms TTL=128

Ping statistics for 192.168.231.240:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 1ms, Maximum = 2ms, Average = 1ms
```

Рисунок 3.30 – Виконання пінгів до пристроїв локальної мережі

Тепер слід перевірити доступ до мережі Інтернет. Для цього в браузері перейдемо за адресою ztu.edu.ua і побачимо наступний результат, рис.3.31.

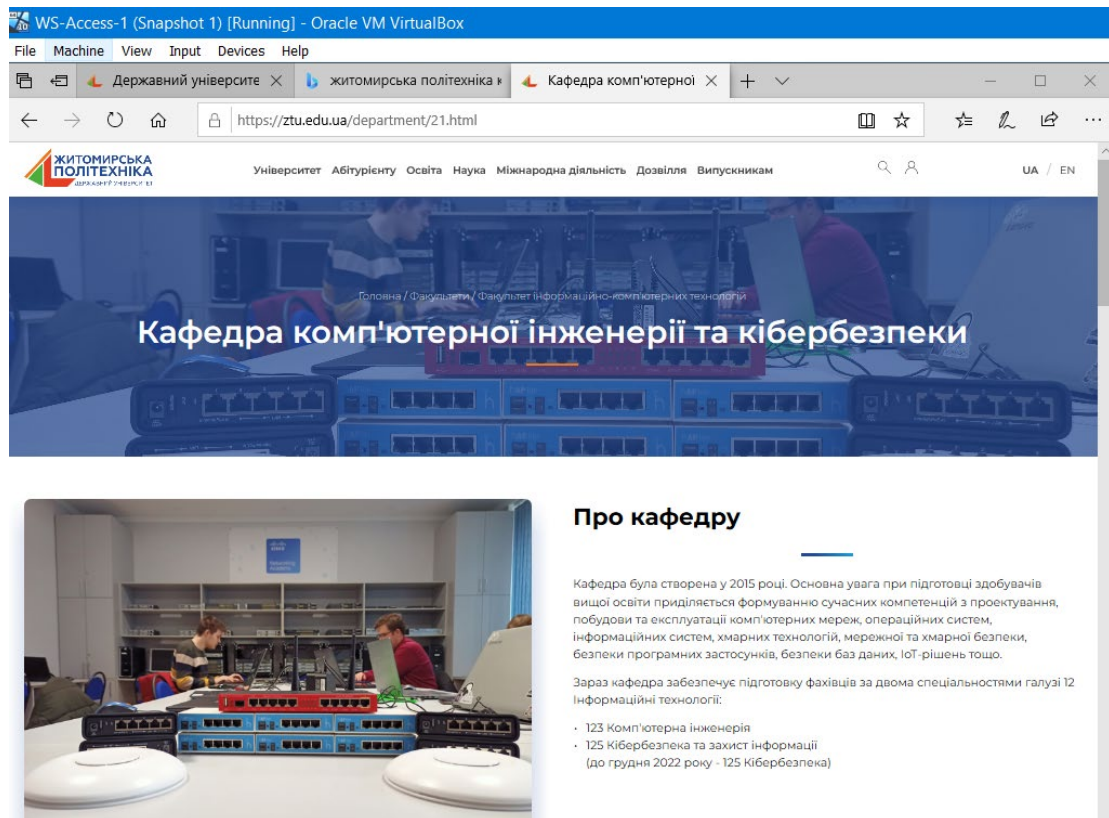


Рисунок 3.31 – Доступ до Інтернету наявний

І на Windows Server 2019 можна перевірити яким пристроям були призначені адреси з DHCP пулу, рис.3.32. На цьому базову перевірку роботи мережі можна вважати завершеною.

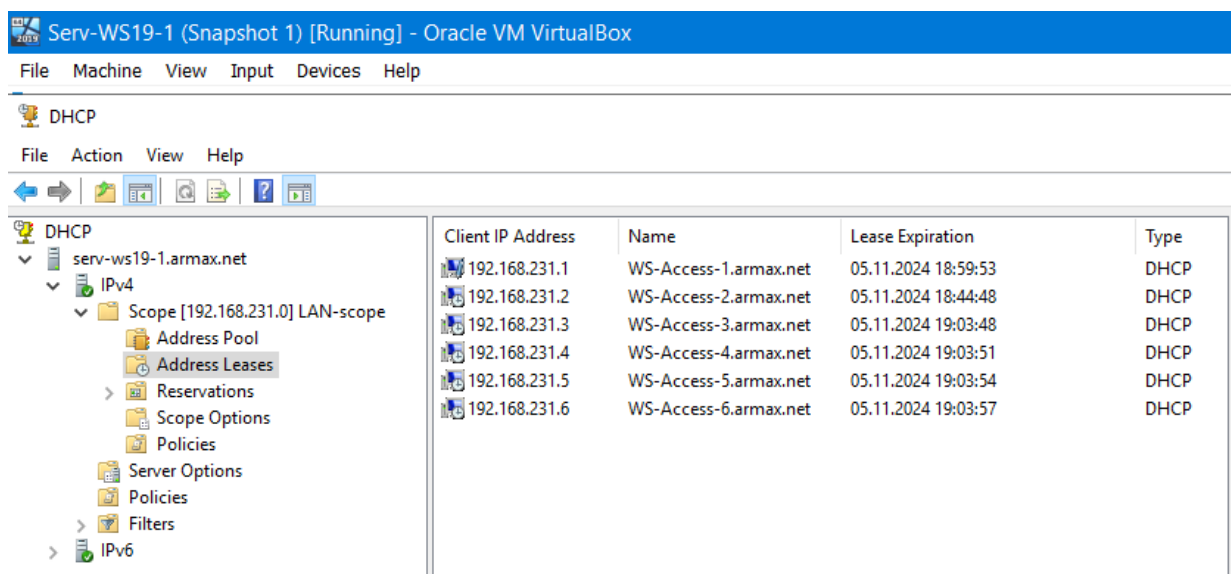


Рисунок 3.32 – Перелік призначених адрес кінцевим пристроям
Безпека передачі даних в Elastic Stack

Elastic Stack використовується для аналізу логів, моніторингу подій та збору телеметрії, що містять чутливі дані, які при перехопленні можуть бути використані в злочинних цілях. Тому важливо переконатись що дані до Elastic передаються лише в шифрованому вигляді. Для цього виконаємо перехоплення пакетів між WS-Access-1 та Serv-Elastic-1 і відфільтруємо результат за адресою призначення. Як можна побачити нижче передача зашифрованих пакетів з агента Elastic (WS-Access-1) здійснюється з використанням криптографічного протоколу TLS (Transport Layer Security) рис.3.33. Тому за можливе перехоплення та розшифрування даних можна не перейматись.

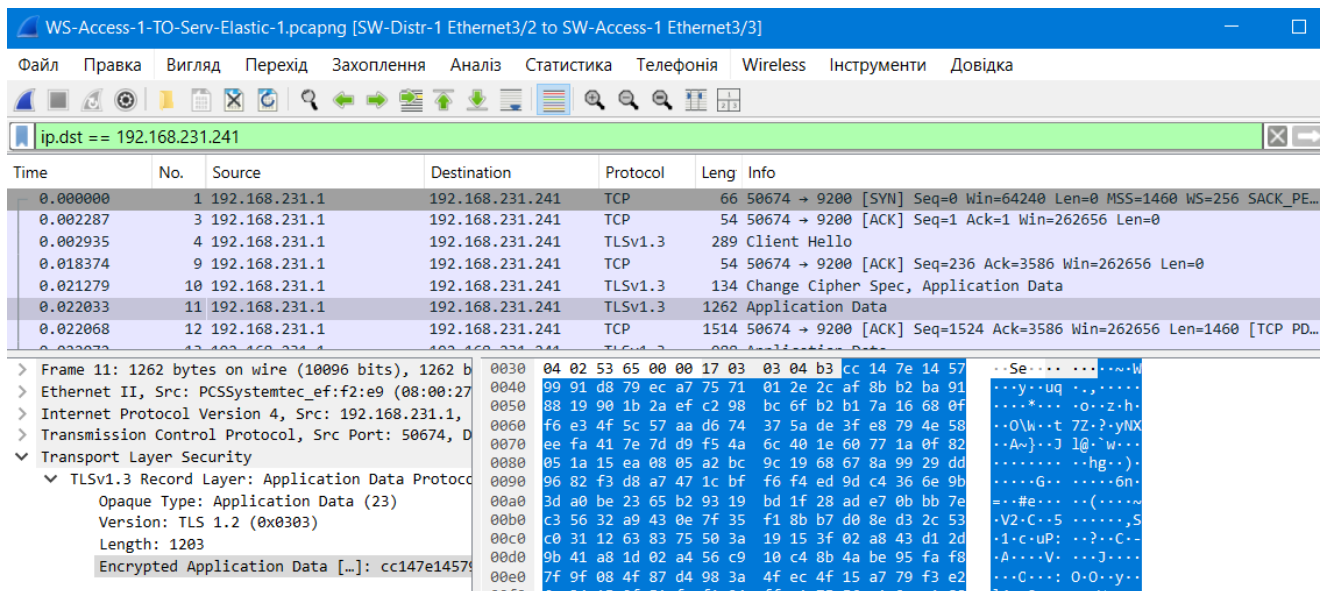


Рисунок 3.33 – Перехоплений мережевий трафік між WS-Access-1 та Serv-Elastic-1

Говорячи про кінцеві вузли важливо, щоб користувачі не могли навмисно або випадково зупинити роботу служби агента і тим самим перервати передачу даних до серверу Elastic. Для запобігання таких випадків встановлення агентів виконувалося з правами адміністратора. Це значить, що звичайний користувач немає будь-яких прав щодо редагування даної папки, рис.3.34.

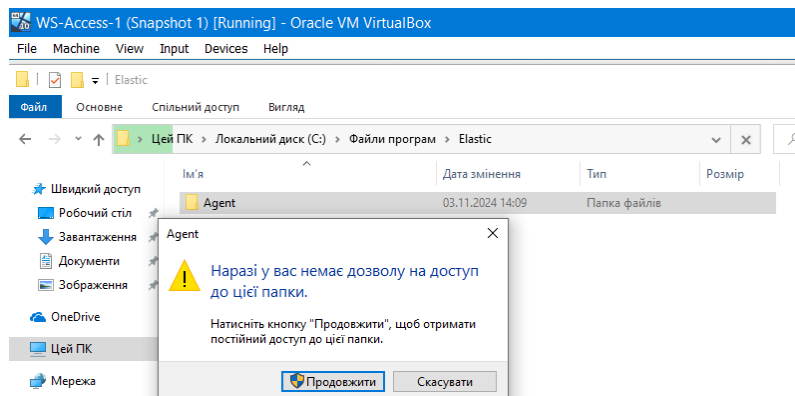


Рисунок 3.34 – Для доступу до папки агента необхідні права адміністратора

Аналіз та візуалізація даних в Elastic

Через певний проміжок роботи мережі до серверу Elastic надійшло достатньо даних для їх візуалізації та аналізу.

Перш за все розглянемо можливість пошуку логів з використанням запитів. Для цього за замовчуванням використовується мова KQL (Kibana Query Language). Під час пошуку можна задати часовий інтервал та вказати інші параметри. Нижче відображено отримані логи від хоста Serv-WS-19-1 що надійшли за останні 15 хв, рис.3.35. Бажані поля для відображення даних можна змінити під свої потреби.

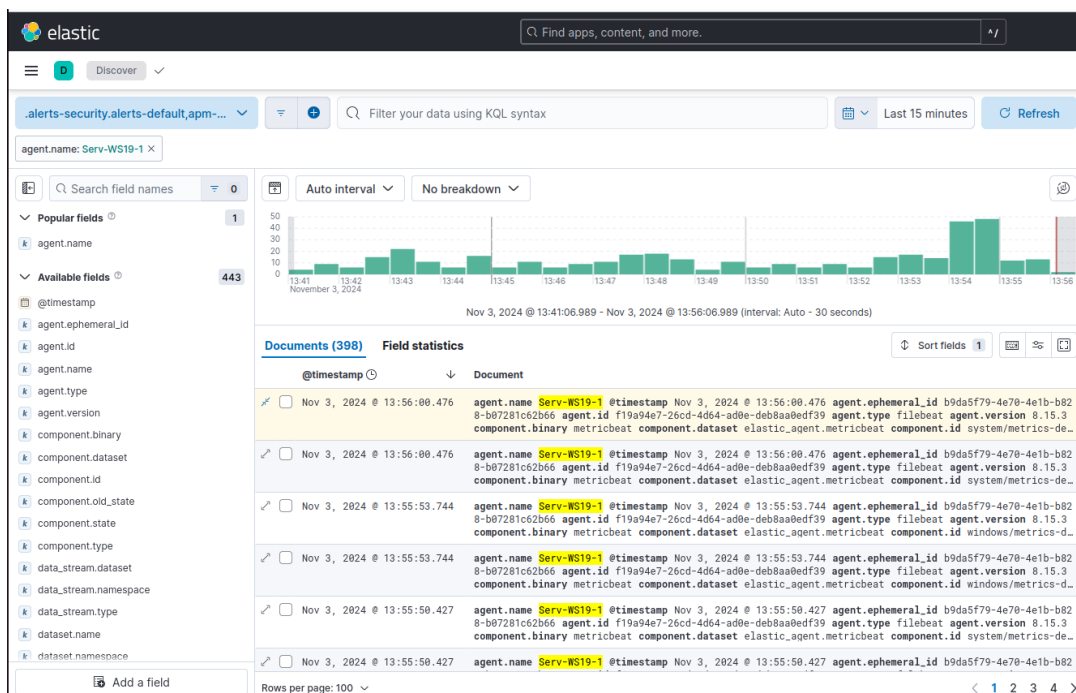


Рисунок 3.35 – Використання запитів для фільтрації та пошуку даних

Однак найбільш зручно переглядати дані у вигляді графіків і візуалізацій. Kibana вже надає ряд доступних візуалізацій [20], однак за потреби можна їх можна видозмінювати під свої бажання або навіть повністю створювати нові вікна [21]. На рис.3.36 наведена інформація по використанню системних ресурсів на пристроях де встановлені агенти. А на рис.3.37 представлена інформація лише для пристрою WS-Access-1. На додатках В.1-В.3 представлений повні варіант скріншотів графіків.

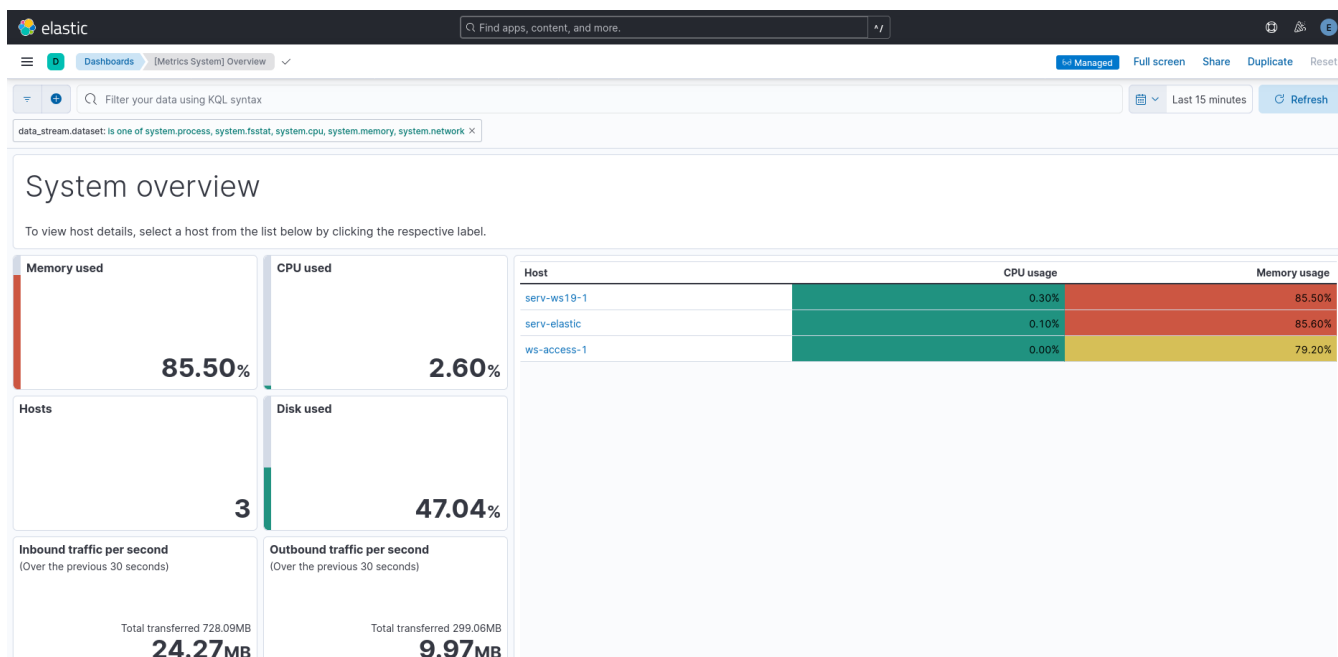


Рисунок 3.36 – Загальна інформація по використанню системних ресурсів

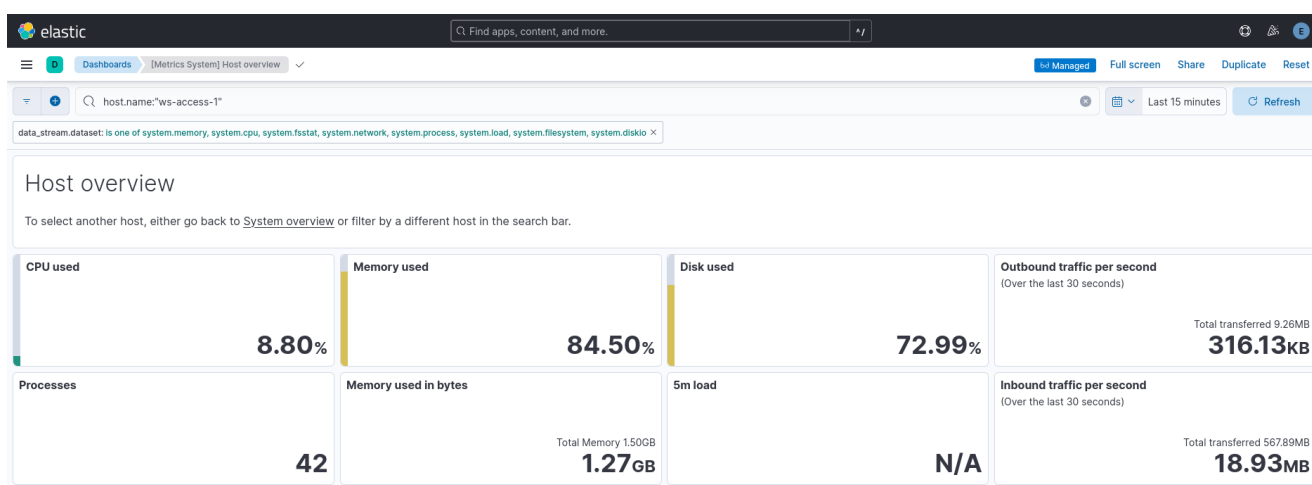


Рисунок 3.37 – Візуалізація використання системних ресурсів WS-Access-1

Схожим чином можна переглянути інформацію про стани роботи служб та запущені процеси на комп'ютера WS-Access, рис.3.38. Подібних візуалізацій в Kibana можна знайти майже під будь-які параметри або навіть створити власні.

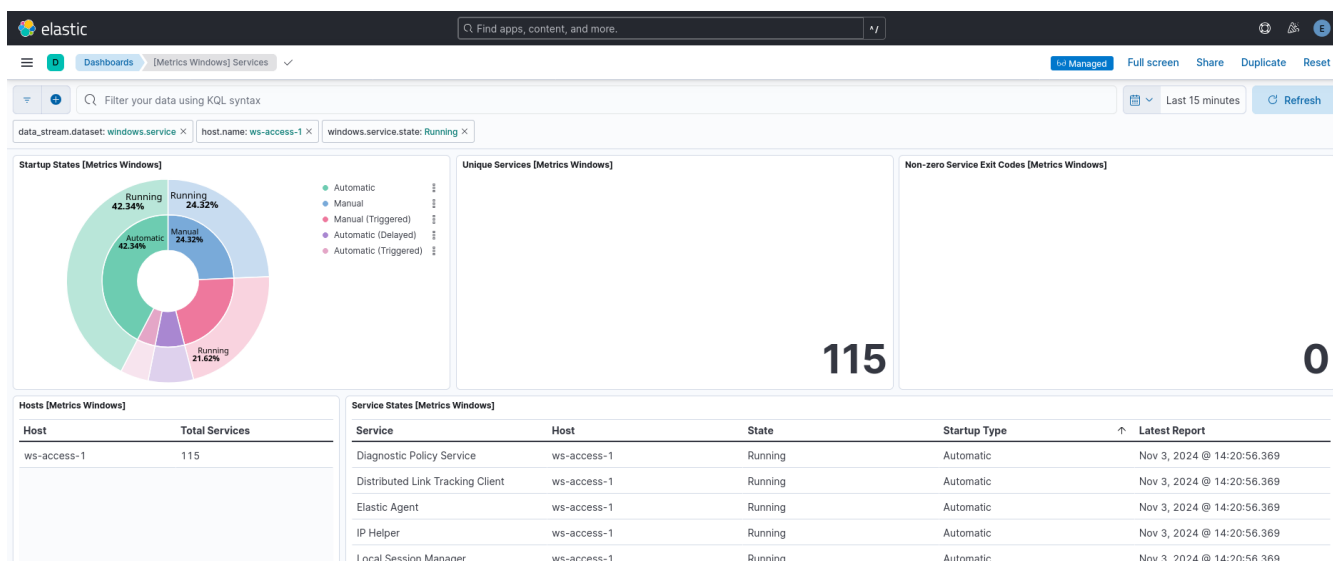


Рисунок 3.38 – Візуалізація станів роботи служб на WS-Access

Elastic Security

Elastic Security – це вбудований компонент Elastic Stack призначений для забезпечення захисту даних, виявлення загроз та реагування на них у реальному часі. Даний модуль об'єднує у собі різні безпекові інструменти, що призначені для виявлення загроз і захисту кінцевих точок. Elastic Security дозволяє аналізувати великі обсяги даних, автоматизувати виявлення атак і централізовано реагувати на загрози. Даний компоненти пропонує наступні можливості, проте деякі з них можуть надаватись лише за платною підпискою:

- Потужний механізм для виявлення широкого спектра загроз.
- Інструменти для оперативного аналізу подій та розслідування інцидентів.
- Захист кінцевих точок від загроз (EDR).
- Налаштування сповіщень.
- Інтерактивні інструменти для візуалізації даних.
- Інтеграції для збору даних із різноманітних джерел.

Отже, перш за все даний компонент пропонує велике різноманіття вже вбудованих візуалізацій. Для прикладу на рис.3.39 наведена візуалізація статистики виконаних входів в систему. Повні версії всіх наступних рисунків наведені на додатка В.4-В.6.

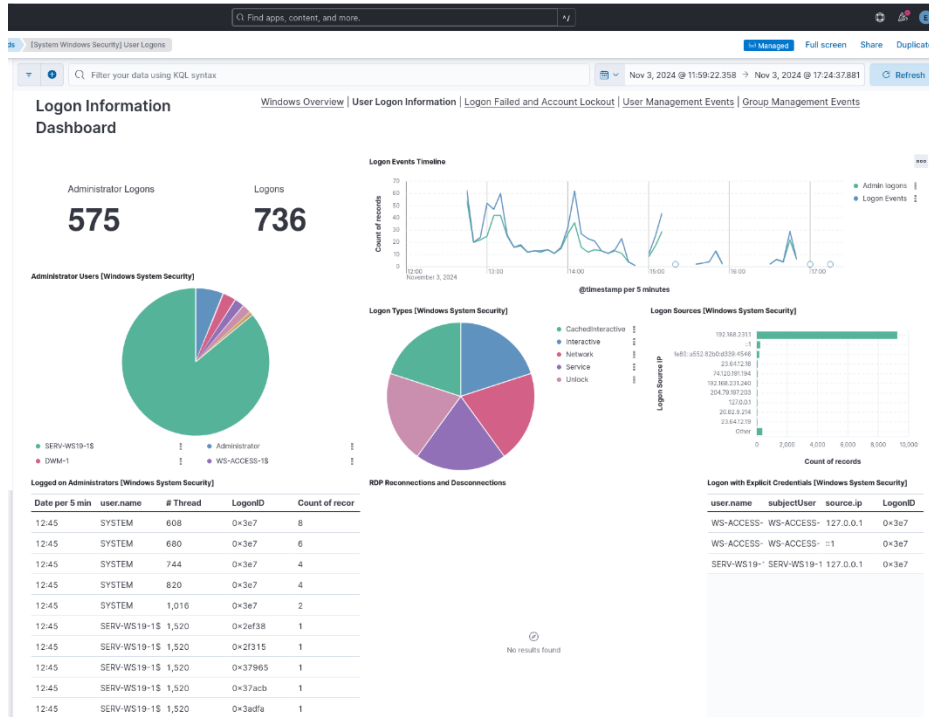


Рисунок 3.39 – Візуалізація виконаних входів в систему

Або візуалізація типів мережевих пакетів, рис.3.40. Подібні візуалізації є майже під будь-який сценарії та потреби.

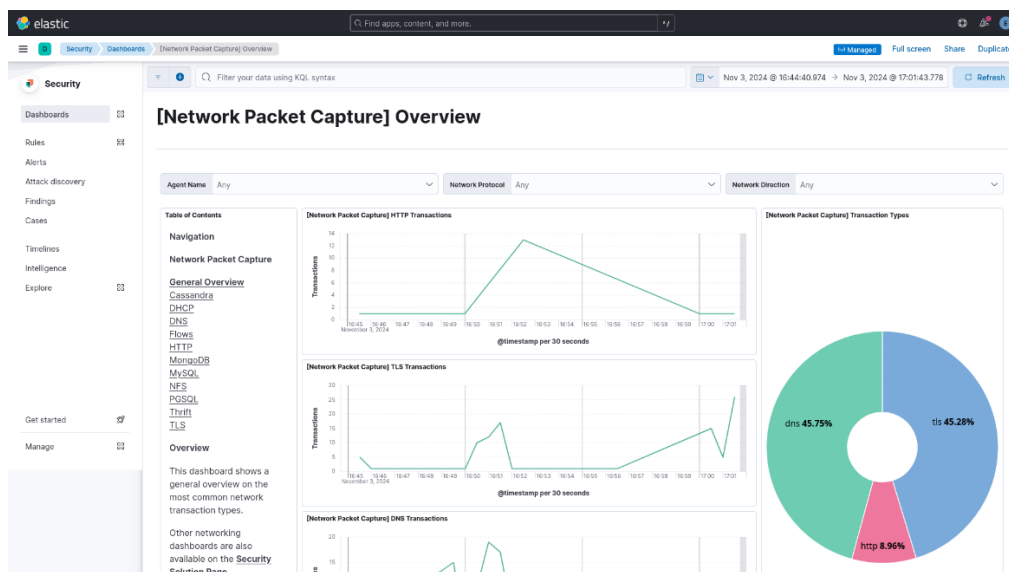


Рисунок 3.40 – Візуалізація типів мережевих пакетів

Окремо слід відмітити пункт Network, який надає широкі можливості аналізу та візуалізації міжмережових з'єднань пристроїв з використанням інтерактивної карти, рис.3.41-3.42.

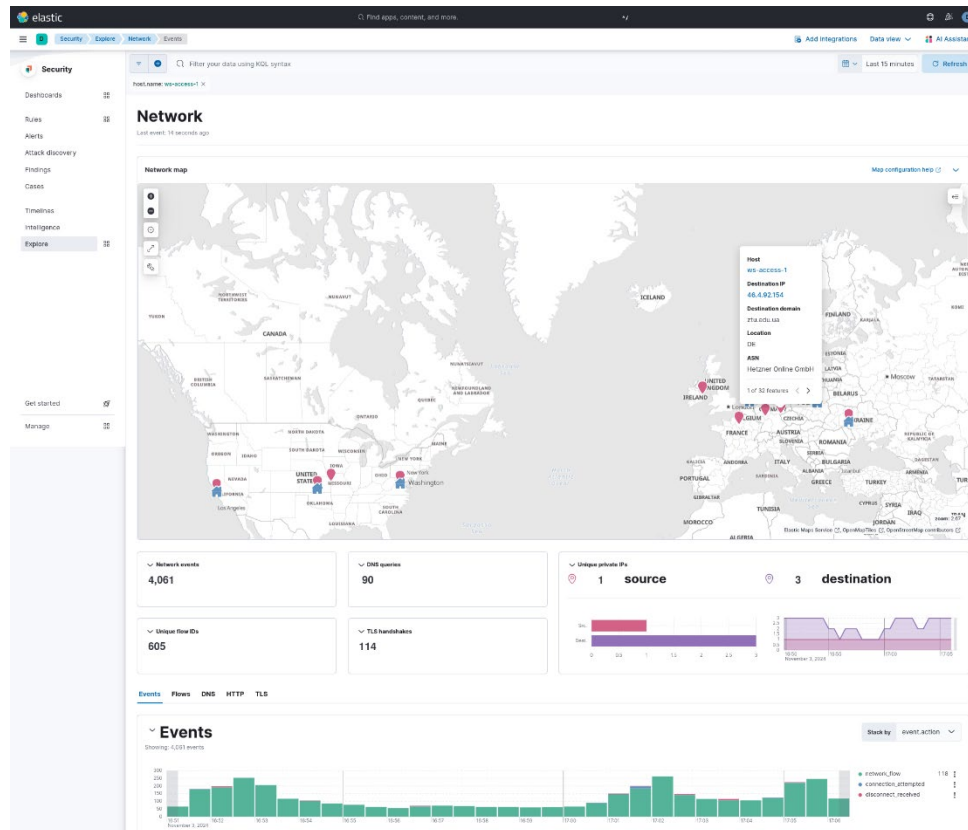


Рисунок 3.41 – Вікно Network

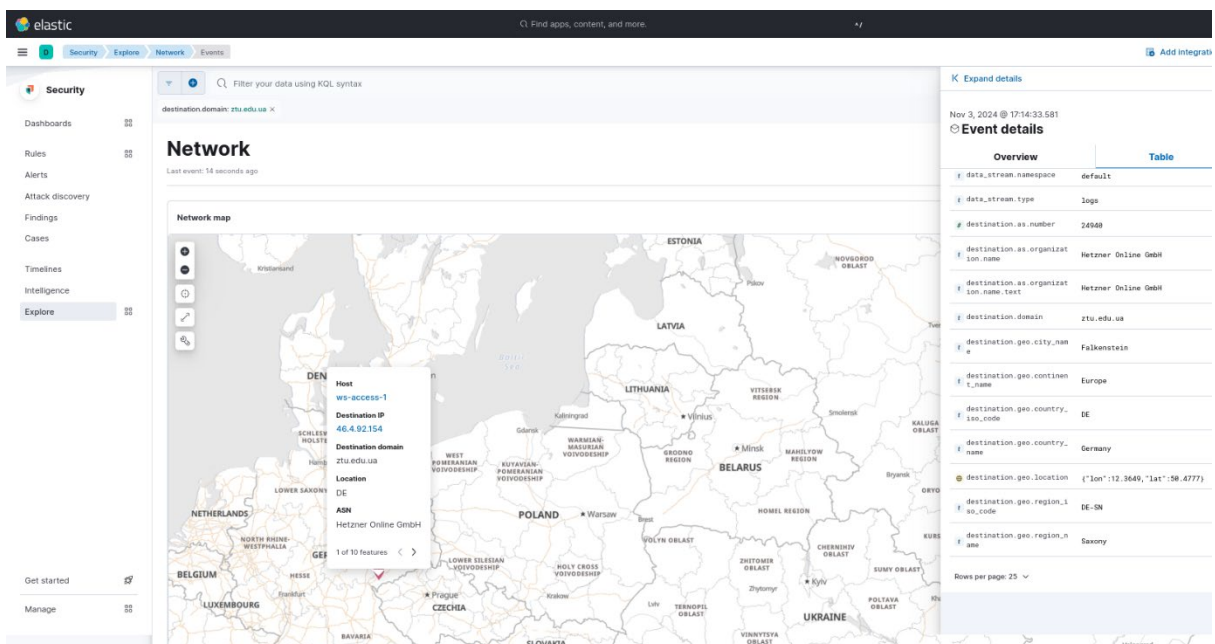


Рисунок 3.42 – Вікно Network

Висновки до третього розділу

В ході розробки даного розділу було виконано проєктування та налаштування корпоративної мережі в середовищі GNS3 з використання гіпервізора VirtualBox. Наведено сценарій базового налаштування маршрутизатора, комутаторів, серверу Windows Server 2019 та кінцевих робочих станій. На сервері виконано розгортання Active Directory, створено домен, користувачів, налаштовано роботу DNS та DHCP серверу. Всі мережеві пристрої та кінцеві станції додано до домену.

Далі було виконано розгортання та налаштування Elastic Stack і агентів для збору та обробки даних із корпоративної мережі. Було встановлено компоненти Elasticsearch, Kibana та Fleet Server, а також налаштовано політики роботи агентів Elastic. Здійснені налагодження забезпечують автоматизований збір логів і метрик з кінцевих комп'ютерів, а також їх подальшу обробку й візуалізацію у Kibana.

І наприкінці проведено тестування роботи корпоративної мережі та Elastic Stack. Перевірено доступність зв'язку між пристроями корпоративної мережі з використання доменних імен та доступ до мережі Інтернет. Виконано вхід в систему на робочій станції з використанням облікового запису користувача. Проведено тестування роботи та можливостей Elastic Stack

Перевірено коректність маршрутизації, стабільність роботи мережі та здатність Elastic Stack ефективно системи збирати, обробляти та візуалізувати дані. Результати тестування підтвердили працездатність проєкту мережі та системи збору та обробки логів Elastic Stack.

ВИСНОВКИ

В ході виконання кваліфікаційної роботи було досліджено методи та технології збору, аналізу й візуалізації даних із використанням Elastic Stack у корпоративних мережах. Розглянуто принципи функціонування ключових компонентів Elastic Stack, їх взаємодію, переваги й недоліки при обробці даних з пристроїв мережі, а також практичні аспекти впровадження даної системи в корпоративній мережі для забезпечення ефективного моніторингу та аналітики.

У першому розділі було розглянуто теоретичні основи призначення лог-файлів та цілей збору, аналізу й зберігання журналів подій. Особливу увагу приділено централізованим системам для обробки логів, які оптимізують роботу з великими обсягами даних та полегшують моніторинг безпеки. Було здійснено огляд безкоштовних рішень для централізованого збору і обробки даних, зокрема таких продуктів, як Elastic Stack, Graylog, SigNoz та Grafana Loki, що надають ефективні альтернативи комерційним рішенням. Порівняльний аналіз показав, що Elastic Stack найбільше відповідає вимогам корпоративної мережі, завдяки своїй універсальності та широким можливостям. Розглянуто ключові компоненти Elastic Stack та його архітектуру.

У другому розділі було детально розглянуто цілі, функції та переваги використання корпоративних мереж, а також основні компоненти, які забезпечують їх роботу. Проаналізовано структуру та переваги використання трирівневої моделі мережі. Розроблено топологію мережі, що включає маршрутизатор, комутатори, сервери та клієнтські станції. Розроблено схему системи збору та обробки даних на базі Elastic Stack з використанням Elastic Agents і Fleet Server. Проведено вибір оптимального обладнання та програмного забезпечення, зокрема маршрутизаторів, комутаторів, серверів і кінцевих пристроїв, що відповідають потребам корпоративної мережі.

У третьому розділі наведено розробку проєкту корпоративної мережі, що включає її проєктування, налаштування та тестування. Було створено мережеву інфраструктуру із впровадженням Active Directory, DNS та DHCP на базі

Windows Server 2019, а також додано всі пристрої до домену. Реалізовано систему моніторингу та збору на базі Elastic Stack і виконано налаштування агентів для автоматизованого збору логів і метрик із кінцевих пристроїв. Проведене тестування підтвердило роботу мережі, доступність внутрішніх і зовнішніх ресурсів, а також ефективність впровадженого рішення..

Розроблена система збору та аналізу логів на основі Elastic Stack характеризується мінімальним впливом на продуктивність корпоративної мережі, забезпечуючи ефективний моніторинг подій та швидку обробку без значного навантаження на сервери й клієнтські пристрої. Висока ефективність Elastic Stack підтверджена успішною обробкою всіх тестових подій без втрат, що свідчить про її надійність у зборі, аналізі та візуалізації даних. Інтеграція з Kibana значно полегшує аналіз інцидентів, надаючи зрозумілі й наочні візуалізації. Впроваджене рішення відповідає сучасним вимогам щодо продуктивності, масштабованості та надійності.

Напрямами подальших досліджень є вдосконалення програмного комплексу шляхом інтеграції інших засобів моніторингу та впровадження інструментів на базі штучного інтелекту для прогнозування подій та виявлення аномалій.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. What are log files? [Електронний ресурс]. – Режим доступу: <https://aws.amazon.com/what-is/log-files/>
2. Different Types of Log Files [Електронний ресурс]. – Режим доступу: <https://sematext.com/glossary/log-file/>
3. NIST SP 800-92, Guide to Computer Security Log Management
4. What is log monitoring? [Електронний ресурс]. – Режим доступу: <https://www.elastic.co/what-is/log-monitoring>
5. Log Analysis Tools [Електронний ресурс]. – Режим доступу: <https://www.exabeam.com/explainers/log-management/log-analysis-tools-key-capabilities-and-5-tools-you-should-know/>
6. Log monitoring for IT: Challenges and best practices [Електронний ресурс]. – Режим доступу: <https://www.manageengine.com/products/eventlog/log-monitoring.html>
7. Кушнір І. В., Фальковський І.Г. Розробка рекомендацій щодо впровадження інфраструктури керування логами // Тези доповідей VII Всеукраїнської науково-технічної конференції «Комп'ютерні технології: інновації, проблеми, рішення», м. Житомир, 02-03 грудня 2024 року – Житомир: Житомирська політехніка, 2024. – С. 27–28.
8. 12 Best Open Source Logging Tools 2024 [Електронний ресурс]. – Режим доступу: <https://logit.io/blog/post/open-source-logging-tools/>
9. Кушнір І. В., Фальковський І.Г. Використання стеку програм Elastic Stack для забезпечення функцій моніторингу, аналізу та візуалізації даних в мережі // Тези доповідей XIV Міжнародної науково-технічної конференції «Інформаційно-комп'ютерні технології – 2024», м. Житомир, 02-03 грудня 2024 року – Житомир: Житомирська політехніка, 2024.
10. Elasticsearch Components [Електронний ресурс]. – Режим доступу: <https://medium.com/huawei-developers/huawei-cloud-search-service-elasticsearch-ea867ca46ebc>

11. Elasticsearch Integrations [Электронный ресурс]. – Режим доступа: <https://www.elastic.co/integrations>
12. Everything You Need to Know about Enterprise Networking [Электронный ресурс]. – Режим доступа: <https://www.perimeter81.com/blog/network/enterprise-networking>
13. Enterprise Networking Explained: Types, Concepts & Trends [Электронный ресурс]. – Режим доступа: <https://blogs.bmc.com/enterprise-networking/?print-posts=pdf>
14. Beats and Elastic Agent capabilities What is Hierarchical Network Design? [Электронный ресурс]. – Режим доступа: <https://www.auvik.com/franklyit/blog/hierarchical-network-design/>
15. Beats and Elastic Agent capabilities [Электронный ресурс]. – Режим доступа: <https://www.elastic.co/guide/en/fleet/current/beats-agent-comparison.html>
16. Fleet and Elastic Agent overview [Электронный ресурс]. – Режим доступа: <https://www.elastic.co/guide/en/fleet/current/fleet-overview.html>
17. Cisco 4221 Integrated Services Router [Электронный ресурс] – Режим доступа: <https://www.cisco.com/c/en/us/support/routers/4221-integrated-services-router-isr/model.html>
18. Cisco Catalyst 2960-X Series Switches [Электронный ресурс] – Режим доступа: <https://www.cisco.com/c/en/us/support/switches/catalyst-2960-x-series-switches/series.html>
19. Installing the Elastic Stack [Электронный ресурс] – Режим доступа: <https://www.elastic.co/guide/en/elastic-stack/current/installing-elastic-stack.html>
20. Srivastava A. Mastering Kibana 6.x: Visualize your Elastic Stack data with histograms, maps, charts, and graphs. Packt Publishing, 2018. 365 p.
21. Dashboards | Kibana Guide [Электронный ресурс] – Режим доступа: <https://www.elastic.co/guide/en/kibana/current/dashboard.html>

ДОДАТКИ

Нормативні акти і стандарти, що стосуються логування

1) GDPR (General Data Protection Regulation). Європейський закон захисту даних, який вимагає від компаній реєструвати дії з персональними даними, зокрема доступи, зміни та видалення. GDPR також зобов'язує вести журнали про зловживання доступом і повідомляти про порушення конфіденційності у визначені строки (72 години). Логи мають зберігатись у захищеному вигляді, щоб гарантувати конфіденційність.

2) ISO/IEC 27001. Міжнародний стандарт для управління інформаційною безпекою, який вимагає від компаній створення системи управління журналами подій. Логи повинні збиратися, зберігатися та аналізуватися, щоб запобігати виникненню подальших інцидентів безпеки. Журнали подій мають зберігатись відповідно до вимог безпеки, що гарантує їхню доступність для аудиту.

3) SOX (Sarbanes-Oxley Act). Цей закон США стосується переважно компаній, які котируються на біржах. SOX передбачає регулярний аудит фінансових журналів, що включає моніторинг доступу до фінансових систем, фіксування спроб несанкціонованого доступу та зміни критичних файлів. А розділ 404 стосується внутрішнього контролю та процедур звітності, включаючи моніторинг доступу і ведення журналів аудиту. Це гарантує прозорість та можливість відстежити всі зміни в системах для запобігання шахрайству.

4) PCI DSS (Payment Card Industry Data Security Standard). Стандарт безпеки для захисту даних платіжних карток, який вимагає вести журнали про доступ до систем обробки платежів, а також фіксувати дії адміністраторів і користувачів. Вимога 10 даного стандарту передбачає відстеження та контроль доступу до мережі та даних власників карток. PCI DSS регулює зберігання журналів щонайменше на рік, з обов'язковим збереженням активних логів на останні три місяці.

5) HIPAA (Health Insurance Portability and Accountability Act). Закон США, що вимагає захисту медичних даних. HIPAA встановлює вимоги до журналів,

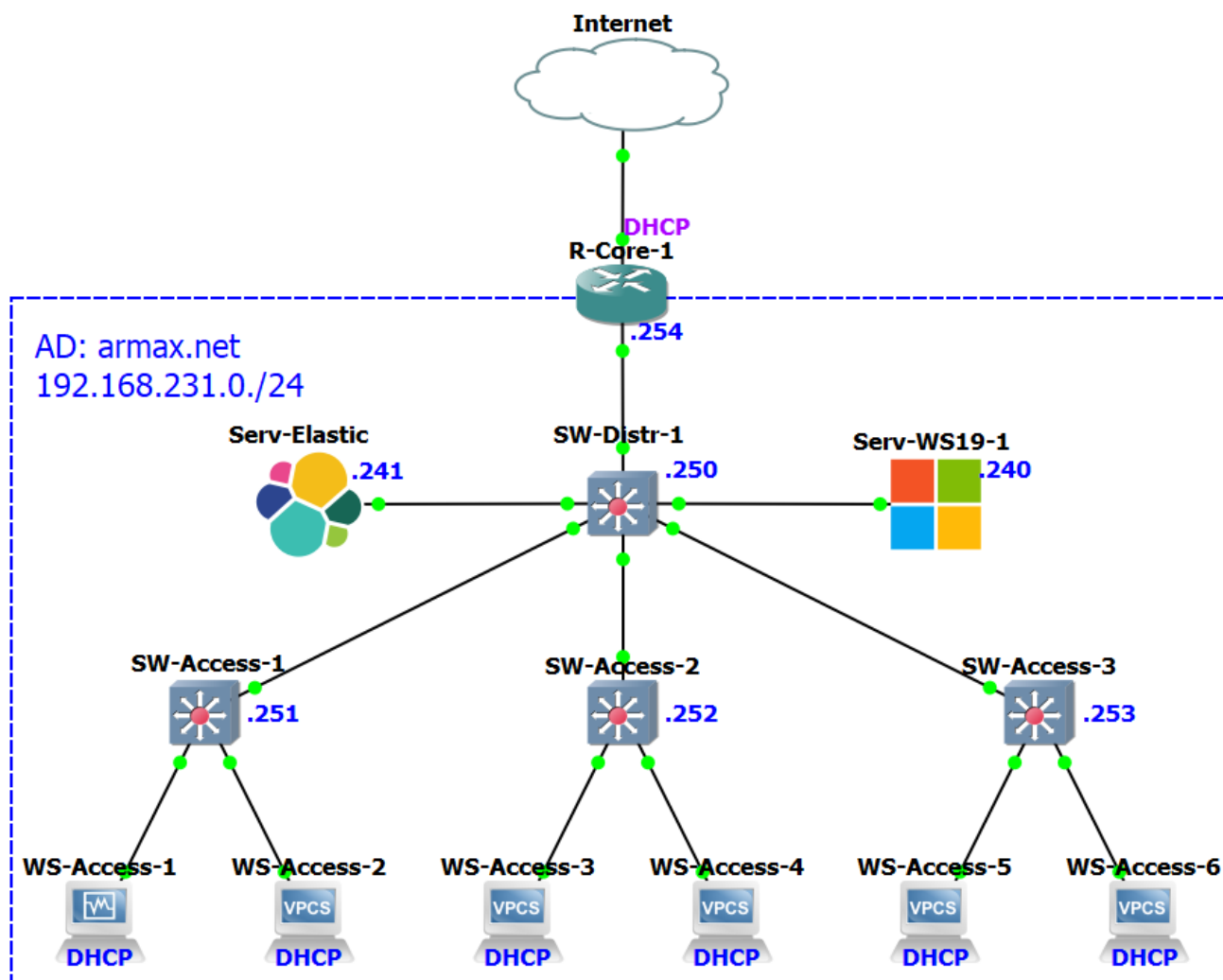
щоб забезпечити доступ лише авторизованих осіб до медичних записів. Логування має фіксувати дії користувачів з медичними даними, а журнали повинні зберігатися в зашифрованому вигляді для захисту конфіденційності пацієнтів.

6) NIST SP 800-92. Рекомендації Національного інституту стандартів і технологій США, які містять найкращі практики для журналювання та аналізу безпеки. NIST SP 800-92 допомагає компаніям ідентифікувати ключові події для моніторингу, встановлювати налаштування для збору логів і виявляти загрози. Це керівництво також включає вказівки для обробки та аналізу журналів, щоб покращити безпеку систем.

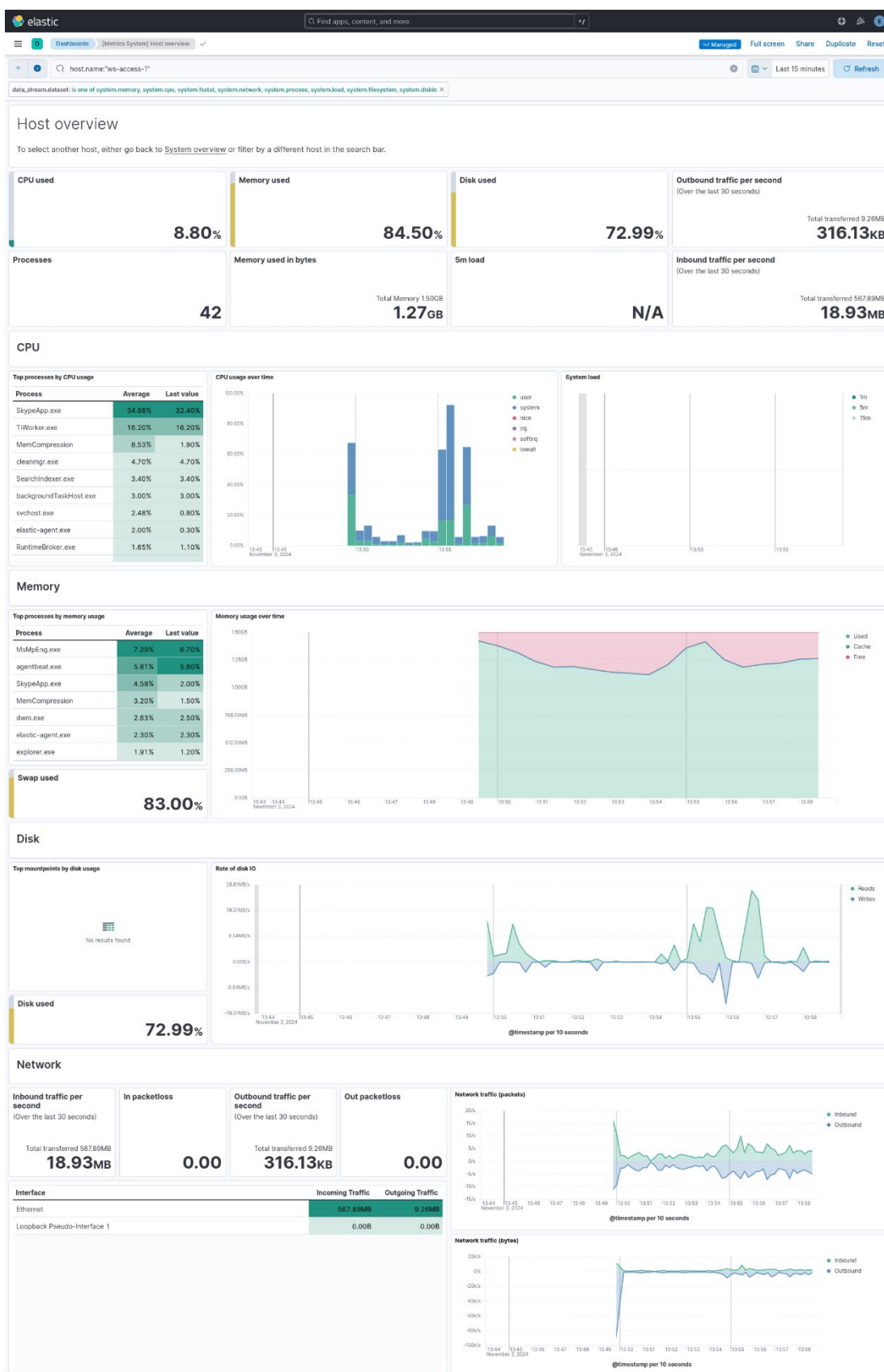
7) FISMA (Federal Information Security Management Act). Закон США, що встановлює вимоги до кібербезпеки державних установ. FISMA зобов'язує організації документувати всі події в інформаційних системах, зокрема доступи, зміни в даних та спроби порушення безпеки. Логи повинні зберігатись і бути доступними для аудиту, щоб забезпечити прозорість і відповідність вимогам кібербезпеки.

8) COBIT (Control Objectives for Information and Related Technology). Фреймворк управління ІТ, що рекомендує логування для контролю над інформаційними активами та управління ризиками. COBIT передбачає ведення журналів подій і доступів для моніторингу систем, щоб компанії могли ефективно оцінювати загрози, мінімізувати ризики та досягати відповідності бізнес-цілям та вимогам безпеки.

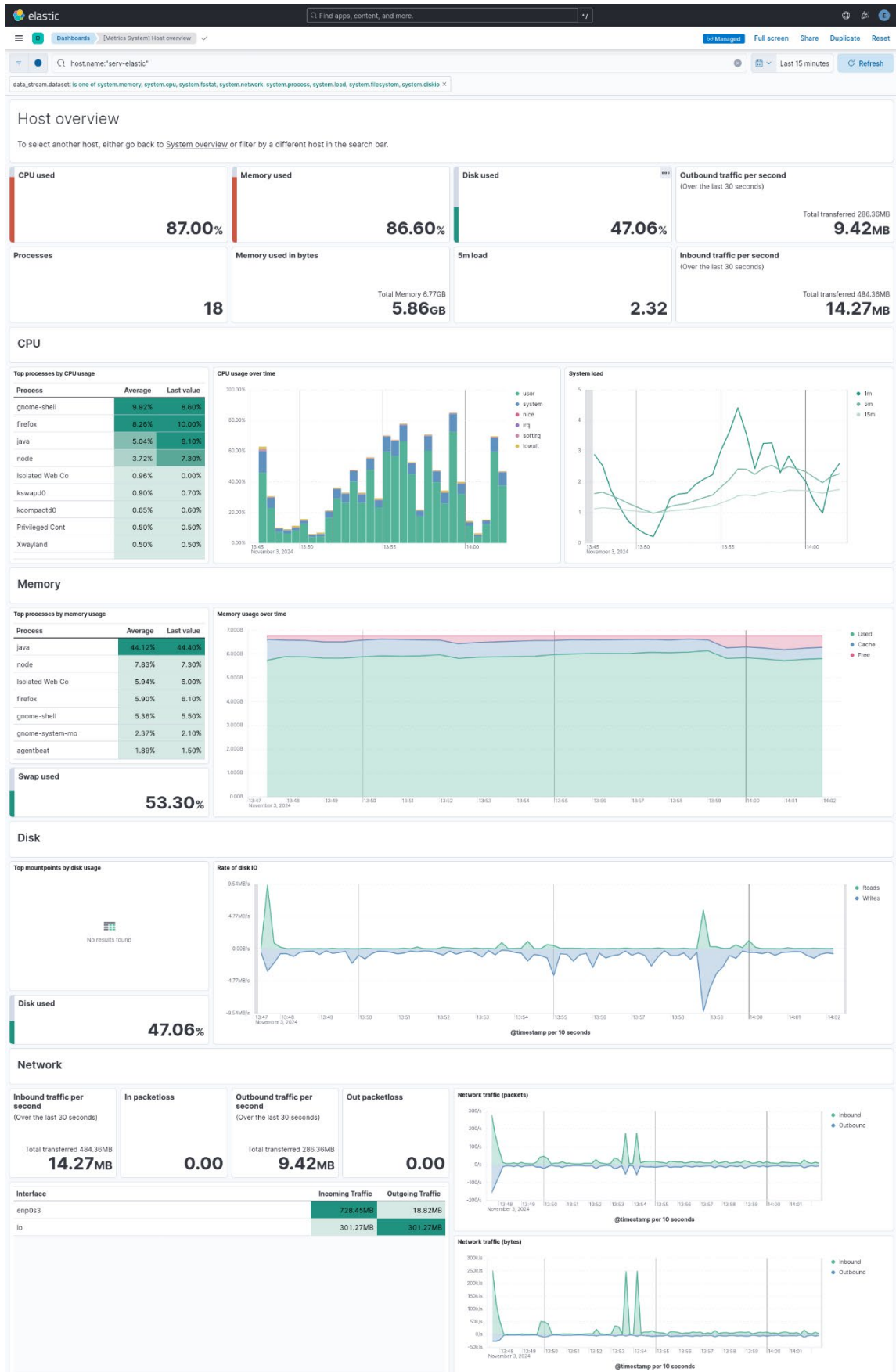
Схема спроектованої мережі в GNS3



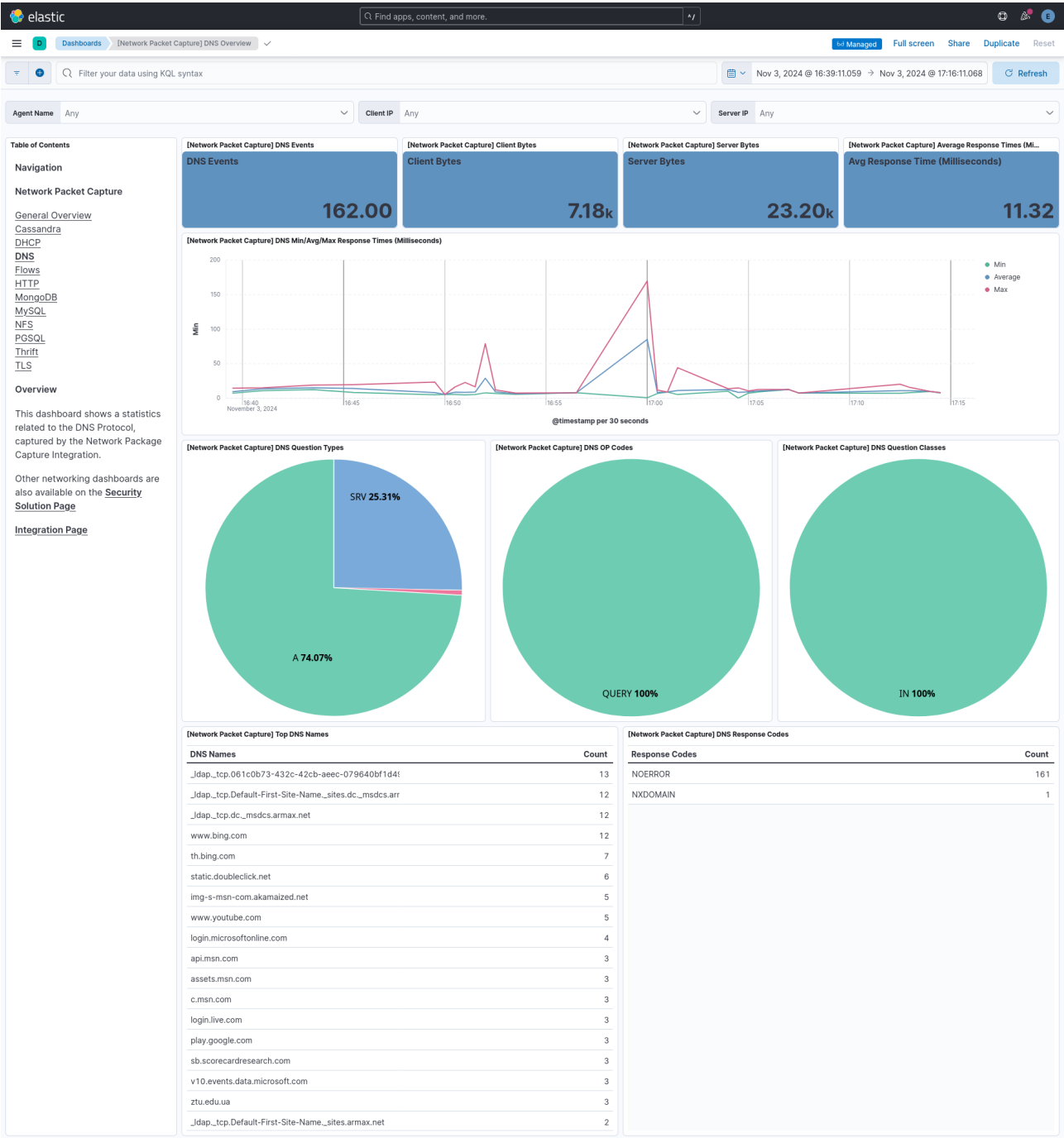
Візуалізація використання системних ресурсів WS-Access-1



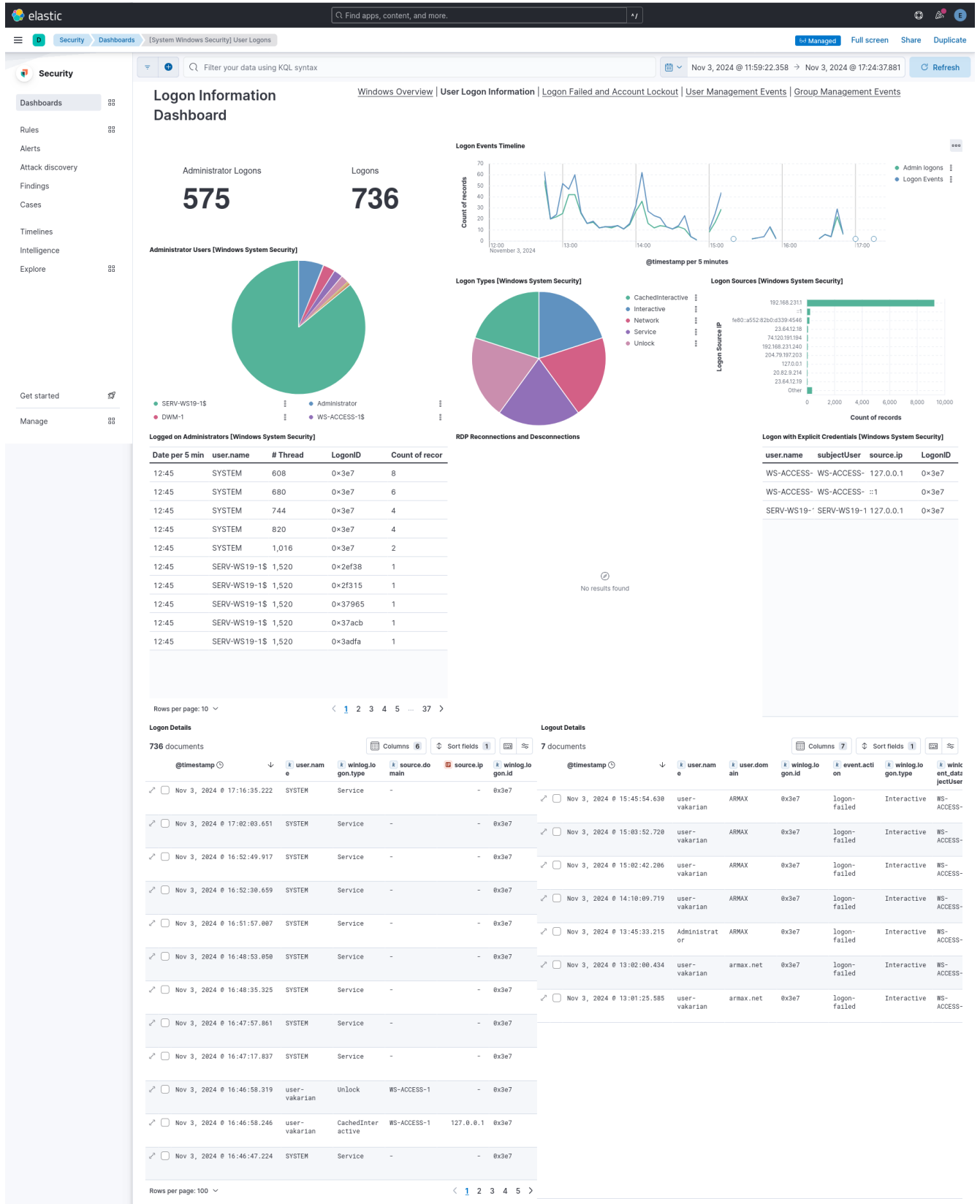
Візуалізація використання системних ресурсів Serv-Elastic



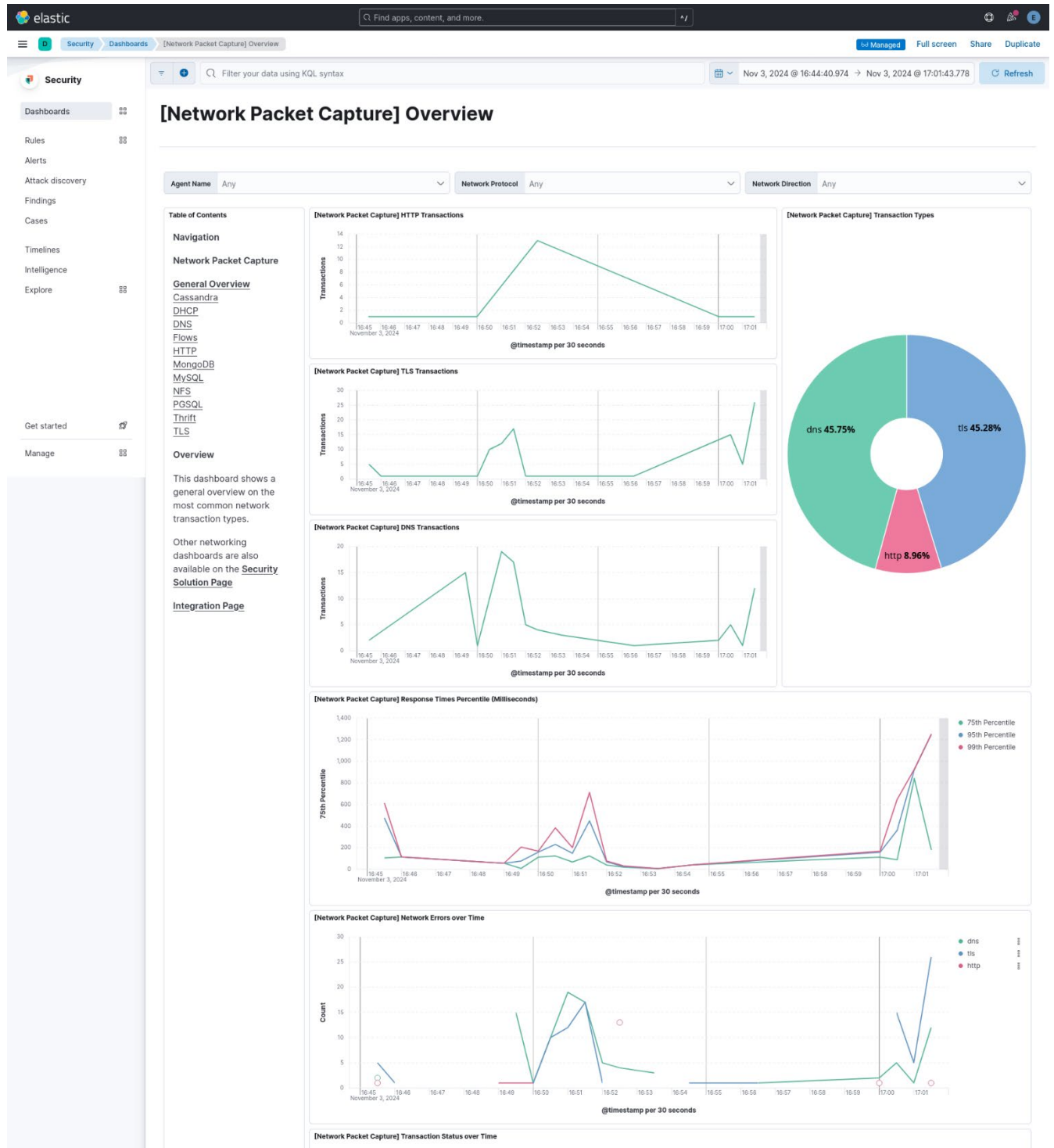
Візуалізація виконаних DNS запитів



Візуалізація виконаних входів в систему



Візуалізація надісланих і отриманих типів мережових пакетів на вузлах





Лістинг конфігураційних файлів пристроїв мережі

```

R-Core-1:
!
!
!
upgrade fpd auto
version 12.4
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname R-Core-1
!
boot-start-marker
boot-end-marker
!
logging message-counter syslog
!
no aaa new-model
ip source-route
no ip icmp rate-limit unreachable
ip cef
!
!
!
!
!
ip domain name armax.net
ip name-server 8.8.8.8
ip name-server 192.168.231.240
no ipv6 cef
!
multilink bundle-name authenticated
!
!
!
!
!
archive
 log config
  hidekeys
!
!
!
!
!
ip tcp synwait-time 5
!
!
!
!
interface FastEthernet0/0
description link-to-internet
ip address dhcp
ip nat outside
ip virtual-reassembly
duplex auto
speed auto
!
interface FastEthernet0/1
description link-to-sw-distr-1
ip address 192.168.231.254 255.255.255.0
ip nat inside
ip virtual-reassembly
duplex auto
speed auto
!
ip forward-protocol nd
no ip http server
no ip http secure-server
!
!
!
ip dns server
ip nat inside source list 1 interface
FastEthernet0/0 overload
!
access-list 1 permit 192.168.231.0 0.0.0.255
no cdp log mismatch duplex
!
!
!
!
!
control-plane
!
!
!
!
gatekeeper
shutdown
!
!
!
line con 0
 exec-timeout 0 0
 privilege level 15
 logging synchronous
 stopbits 1
line aux 0
 exec-timeout 0 0
 privilege level 15
 logging synchronous
 stopbits 1
line vty 0 4
 login
!
End

```

SW-Distr-1:

```
!
! Last configuration change at 16:56:08 UTC Fri
Nov 1 2024
!
version 15.2
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
service compress-config
!
hostname SW-Distr-1
!
boot-start-marker
boot-end-marker
!
!
logging discriminator EXCESS severity drops 6
msg-body drops EXCESSCOLL
logging buffered 50000
logging console discriminator EXCESS
!
no aaa new-model
!
!
no ip routing
no ip icmp rate-limit unreachable
!
!
!
ip domain-name armax.net
ip name-server 192.168.231.254
ip name-server 192.168.231.240
no ip cef
no ipv6 cef
!
!
!
spanning-tree mode pvst
spanning-tree extend system-id
!
!
!
interface Ethernet0/0
!
interface Ethernet0/1
!
interface Ethernet0/2
!
interface Ethernet0/3
!
interface Ethernet1/0
!
interface Ethernet1/1
!
interface Ethernet1/2
!
interface Ethernet1/3
!
interface Ethernet2/0
!
interface Ethernet2/1
!
interface Ethernet2/2
!
interface Ethernet2/3
!
interface Ethernet3/0
!
interface Ethernet3/1
!
interface Ethernet3/2
!
interface Ethernet3/3
!
interface Vlan1
ip address 192.168.231.250 255.255.255.0
no ip route-cache
!
ip default-gateway 192.168.231.254
ip forward-protocol nd
!
ip tcp synwait-time 5
ip http server
!
ip ssh server algorithm encryption aes128-ctr
aes192-ctr aes256-ctr
ip ssh client algorithm encryption aes128-ctr
aes192-ctr aes256-ctr
!
!
!
control-plane
!
!
!
line con 0
exec-timeout 0 0
privilege level 15
logging synchronous
line aux 0
exec-timeout 0 0
privilege level 15
logging synchronous
line vty 0 4
login
!
!
!
end
```

SW-Access-1:

```
!  
! Last configuration change at 16:42:58 UTC Fri  
Nov 1 2024  
!  
version 15.2  
service timestamps debug datetime msec  
service timestamps log datetime msec  
no service password-encryption  
service compress-config  
!  
hostname SW-Access-1  
!  
boot-start-marker  
boot-end-marker  
!  
!  
logging discriminator EXCESS severity drops 6  
msg-body drops EXCESSCOLL  
logging buffered 50000  
logging console discriminator EXCESS  
!  
no aaa new-model  
!  
!  
!  
!  
no ip routing  
no ip icmp rate-limit unreachable  
!  
!  
ip domain-name armax.net  
ip name-server 192.168.231.254  
ip name-server 192.168.231.240  
no ip cef  
no ipv6 cef  
!  
!  
!  
spanning-tree mode pvst  
spanning-tree extend system-id  
!  
!  
!  
interface Ethernet0/0  
!  
interface Ethernet0/1  
!  
interface Ethernet0/2  
!  
interface Ethernet0/3  
!  
interface Ethernet1/0  
!  
interface Ethernet1/1
```

```
!  
interface Ethernet1/2  
!  
interface Ethernet1/3  
!  
interface Ethernet2/0  
!  
interface Ethernet2/1  
!  
interface Ethernet2/2  
!  
interface Ethernet2/3  
!  
interface Ethernet3/0  
!  
interface Ethernet3/1  
!  
interface Ethernet3/2  
!  
interface Ethernet3/3  
!  
interface Vlan1  
ip address 192.168.231.251 255.255.255.0  
no ip route-cache  
!  
ip default-gateway 192.168.231.254  
ip forward-protocol nd  
!  
ip tcp synwait-time 5  
ip http server  
!  
ip ssh server algorithm encryption aes128-ctr  
aes192-ctr aes256-ctr  
ip ssh client algorithm encryption aes128-ctr  
aes192-ctr aes256-ctr  
!  
!  
!  
control-plane  
!  
!  
line con 0  
exec-timeout 0 0  
privilege level 15  
logging synchronous  
line aux 0  
exec-timeout 0 0  
privilege level 15  
logging synchronous  
line vty 0 4  
login  
!  
!  
!  
end
```