НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

До захисту допущено
В.о. завідувача кафедри

_____ Микола ГРАЙВОРОНСЬКИЙ
(підпис)

«_____»_____2021 р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною программою «Системи, технології та математичні методи кібербезпеки»
спеціальності: 125 «Кібербезпека»

на тему: Проблема вибору ознакового простору кіберзагроз методом факторного аналізу за допомогою k-блочної перехресної перевірки

Виконав: здобувач вищої освіти **IV** курсу, групи ФБ-73
(шифр групи)

Деркач Вячеслав Віталійович
(прізвище, ім'я, по батькові) (підпис)

Керівник д.т.н., професор, Качинський Анатолій Броніславович
(посада, науковий ступінь, вчене звання, прізвище, ім'я, по батькові) (підпис)

Рецензент _____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище, ім'я, по батькові) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Здобувач вищої освіти _____
(підпис)

Київ - 2021 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

Рівень вищої освіти – перший (бакалаврський)
Спеціальність – 125 «Кібербезпека»
Освітньо-професійна програма «Системи, технології та математичні методи кібербезпеки»

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри

_____ Микола ГРАЙВОРОНСЬКИЙ
(підпис)

«___»_____2021 р.

ЗАВДАННЯ
на дипломну роботу здобувачу вищої освіти

Деркачу Вячеславу Віталійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи: Проблема вибору ознакового простору кіберзагроз методом факторного аналізу за допомогою k-блочної перехресної перевірки,
керівник роботи: д.т.н., професор, Качинський Анатолій Броніславович,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від « » _____ 2021 р. №

2. Термін подання здобувачем вищої освіти роботи 07 червня 2021 р.

3. Вихідні дані до роботи _____
_____
_____

4. Зміст роботи_____
_____
_____
_____
_____

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо)
_____
_____
_____

6. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів дипломної роботи	Примітка
1	Постановка задачі, огляд літератури	03.10.20 - 16.12.21	
2	Вибір даних для дослідження	17.12.20 – 20.01.21	
3	Вибір методів дослідження	11.01.21 – 15.02.21	
4	Підготовка вхідних даних	16.02.21 – 16.04.21	
5	Створення нової системи показників з використанням факторного аналізу	17.04.21 – 30.04.21	
6	Перевірка якості декомпозиції за допомогою перехресної перевірки	01.05.21 – 16.05.21	
7	Перевірка ефективності побудованої моделі за допомогою кластеризації	17.05.21 – 22.05.21	
8	Оформлення дипломної роботи	23.05.21 – 7.06.21	

Здобувач вищої освіти _____ Вячеслав ДЕРКАЧ
(підпис) (Власне ім’я, ПРІЗВИЩЕ)

Керівник роботи _____ Анатолій КАЧИНСЬКИЙ
(підпис) (Власне ім’я, ПРІЗВИЩЕ)

(підпис) (ініціали, прізвище)

РЕФЕРАТ

Дипломна робота має обсяг 69 сторінок, містить 20 рисунків, 12 таблиць та 5 додатків, а також 12 посилань

В умовах різкого зростання попиту на автоматизацію процесів моніторингу мережевого трафіку виникає гостра необхідність пошуку якісних моделей, що дозволяють проводити виявлення аномальної мережевої активності в режимі реального часу.

Метою роботи є формування системи показників для аналізу перехоплених пакетів мережевого трафіку засобами дослідницького факторного аналізу. Реалізація перевірки якості моделювання в умовах k-блочної перехресної перевірки для пошуку факторної моделі з найкращими оцінками якості в середньому.

У цьому дослідженні було побудовано факторну модель як систему показників для аналізу перехоплених пакетів мережевому трафіку засобами дослідницького факторного аналізу. Факторні навантаження знайдено як оцінки максимальної правдоподібності на основі SVD підходу. Для закріплення положення матриці навантажень застосовано метод ротації факторів Quartimax. Реалізовано алгоритм k-блочної перехресної перевірки (k=10) для оцінки якості факторних моделей. За результатами перевірки на основі показників повної та залишкової дисперсії виявилось, що факторні моделі з найкращими оцінками якості в середньому були виконані на навчальних наборах C_2, C_4, C_8. На основі побудованої моделі було проведено аналіз трафіку з наявною в ньому атакою, методом кластеризації. За результатами проведеної кластеризації було визначено, що унікальними особливостями для атаки класу DDoS SYN Flood є висока щільність спостережень та характерна відстань між зв'язками у просторі (< 0,1).

Ключові слова: DDoS SYN flood, метод факторного аналізу, метод Кайзера, метод максимальної правдоподібності, Singular Value Decomposition (SVD), ротація факторів, Quartimax, k-блочна перехресна перевірка.

ABSTRACT

The scope of work is 69 pages, 20 illustrations, 12 tables, 5 appendix, 12 sources of literature.

In the conditions of sharp growth of demand for automation of processes of monitoring of network traffic there is an urgent need of search of the qualitative models allowing to carry out detection of abnormal network activity in real time.

The aim of the work is to form a system of indicators for the analysis of intercepted packets of network traffic by means of research factor analysis. Implementation of simulation quality testing in terms of k-fold cross-validation to find the factor model with the best quality estimates on average.

In this study, a factor model was built as a system of indicators for the analysis of intercepted packets to network traffic by means of research factor analysis. Factor loads were found as estimates of maximum likelihood based on the SVD approach. The Quartimax factor rotation method was used to fix the position of the load matrix. The k-fold cross-validation algorithm (k = 10) for quality assessment of factor models is implemented. According to the results of the inspection based on the indicators of complete and residual variance, it was found that the factor models with the best quality scores were performed on average on training sets C_2, C_4, C_8. On the basis of the constructed model the analysis of traffic with the attack present in it, a clustering method was carried out. According to the results of the clustering, it was determined that the unique features for the attack of the DDoS SYN Flood class are the high density of observations and the characteristic distance between the connections in space (<0.1).

Keywords: DDoS SYN flood, factor analysis method, Kaiser method, maximum likelihood method, Singular Value Decomposition (SVD), factor rotation, Quartimax, k-fold cross-validation.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

PCAP – Packet Capture;

SVD (Singular Value Decomposition) – сингулярний розклад;

DDoS (Distributed Denial of Service) – розподілена атаки відмови в обслуговуванні;

IDS (Intrusion Detection System) – система виявлення вторгнень.

ВСТУП

Завдяки стрімкому розвитку мережних технологій та широкому використанню Інтернет-послуг, обсяг світового мережного трафіку зростає з кожним днем. Разом зі зростанням мережного трафіку збільшується і кількість аномалій. Природою їх виникнення може бути неправильне налаштування мережного обладнання, або ж реалізація певного виду кібератаки.

Для виявлення аномалій в режимі реального часу застосовуються автоматизовані системи моніторингу мережевого трафіку, однак через великі об’єми даних швидкість роботи даних систем може значно сповільнюватись. На ряду з чим виникає гостра необхідність у створенні якісних моделей для опису мережевого трафіку.

Одним з варіантів вирішення цієї проблеми є використання дослідницького факторного аналізу. Його застосування дозволяє не тільки зменшити розмірність простору початкових параметрів, а й детермінувати структуру взаємозв’язків між змінними та виявити приховані зв’язки. Зменшення розмірності допомагає пришвидшити роботу алгоритмів аналізу даних, а формування структури взаємозв’язків між змінними – краще інтерпретувати отримані в ході аналізу результати.

В свою чергу, проведення оцінки якості декомпозиції факторної моделі за допомогою тест-валідації в умовах k-блочної перехресної перевірки надає можливість оцінити наскільки добре вона описує мережевий трафік, в умовах обмеженої кількості вхідних даних.

Метою роботи є формування системи показників для аналізу перехоплених пакетів мережевого трафіку засобами дослідницького факторного аналізу. Реалізація перевіркі якості моделювання в умовах k-блочної перехресної перевірки для пошуку факторної моделі з найкращими оцінками якості в середньому.

Для досягнення поставленої мети необхідно вирішити наступні завдання:

- Підготувати набори даних для подальшого дослідження;
- Обрати та обрахувати початкові параметри;
- На основі підготовлених даних описати алгоритм формування факторної моделі;
- Провести перевірку якості декомпозиції моделей, побудованих на навчальних наборах, за допомогою тест-валідації з використанням k-блочної перехресної перевірки;
- Продемонструвати приклад використання моделі до іншого набору даних.

Об’єктом дослідження є система показників, яка використовується для аналізу мережевого трафіку.

Методи дослідження: Опрацювання літератури за даною темою. Формування наборів даних мережевого трафіку для подальшого їх аналізу. Побудова факторних моделей на основі сформованих даних. Перевірка якості побудованих моделей за допомогою тест-валідації в умовах k-блочної перехресної перевірки.

Апробація результатів роботи

Дослідження було представлено у вигляді доповіді на XIX Всеукраїнській науково-практичній конференції студентів, аспірантів та молодих вчених «Теоретичні і прикладні проблеми фізики, математики та інформатики».

Публікації

Робота була опублікована у збірнику матеріалів [1] XIX Всеукраїнській науково-практичної конференції студентів, аспірантів та молодих вчених «Теоретичні і прикладні проблеми фізики, математики та інформатики».

1 АНАЛІЗ ДАНИХ ПРИ ВИЯВЛЕННІ КІБЕРЗАГРОЗ В МЕРЕЖЕВОМУ ТРАФІКУ

1.1 Типи аномалій в мережевому трафіку

Завдяки стрімкому розвитку мережних технологій та широкому використанню Інтернет-послуг, обсяг світового мережного трафіку зростає з кожним днем. Разом зі зростанням мережного трафіку збільшується і кількість аномалій, які можуть виникати внаслідок неправильного налаштування мережевих пристроїв, сканування портів під час підготовки майбутніх атак, присутніх в мережі вірусів та хробаків, , а також атак відмови в обслуговуванні (DoS).

Будь-яка подія, або операція в мережі, яка відхиляється від затверджених норм, може бути охарактеризована як аномалія. Незалежно від причини виникнення, кожна аномалія може нести загрозу для нормального функціонування системи мережевих пристроїв. Мережеві аномалії в основному поділяються на два типи [2]:

1. Аномалії, які впливають на продуктивність системи. Даний тип аномалій може спричинити збої в роботі мережі, зменшити швидкість її роботи, або вивести з ладу деякі сервіси на певний час.
2. Аномалії, які втручаються у безпеку системи. Вони виникають коли поведінка мережевого трафіку суттєво відрізняється від звичайної. Загалом можна вирізнити три типи даних аномалій:
    - Точкові аномалії – у випадку, якщо одне зі спостережень суттєво відрізняється від інших.
    - Контекстуальні аномалії – у випадку, якщо якесь зі спостережень може бути визнане аномальним, лише у випадку якогось певного контексту.

- Групові аномалії – у випадку, якщо присутня ціла група спостережень, яка значно відрізняється від решти.

Виявлені аномалії в мережевому трафіку можуть виявитися звичайною помилкою в налаштуванні пристроїв, однак існує і велика імовірність того, що знайдені аномалії є наслідком проведення кібератаки на дану мережу.

1.2 Види мережевих атак

Атака – це зазвичай послідовність операцій, яка виконується з метою нанесення шкоди, шляхом викрадення приватної інформації, або втручанням у роботу внутрішніх сервісів.

Мережеві атаки часто поділяють на активні та пасивні [3]. Пасивні атаки направлені на моніторинг мережі та збір даних, що в ній передається, який пізніше може бути використаний під час проведення активних атак. Прикладом пасивної атаки є аналіз мережевого трафіку, який доволі важко помітити. Активні атаки, в свою чергу, безпосередньо втручаються та вносять зміни в саму систему.

Нижче приведено коротку характеристику деяких мережевих атак

DoS – атака

Атака відмови в обслуговуванні (англ. Denial of service) – атака направлена на спробу блокування доступу до наявних системних, або мережевих ресурсів. Реалізація даної атаки відбувається шляхом примушення цільових комп’ютерів до споживання великої кількості системних ресурсів. Як результат, законні користувачі атакованого ресурсу не зможуть отримати доступ до необхідних сервісів.

Приклади реалізації атаки: Ping of death, Smurf-атака,TCP SYN-флуд та інші.

Сканування мережі

Сканування мережі використовується для збору інформації стосовно цільової мережі, або для знаходження наявних в системі вразливостей шляхом сканування чи зондування мережі. В результаті проведення даної атаки зловмисники можуть отримати критичну інформацію стосовно функціонування мережі, яка в подальшому може бути використана для проведення інших видів атак.

Приклади реалізації атаки: SYN-сканування, FIN-сканування, XMAS-сканування та інші.

Атака R2L

Атака R2L (Remote to local) направлена на отримання віддаленого доступу до комп'ютера з боку незареєстрованого користувача. Реалізація даної атаки надає зловмиснику можливість надсилати мережеві пакети у віддалену систему, не маючи жодного облікового запису в цій мережі, та отримати доступ до системи в якості користувача, або рута. Також дана атака надає можливість зловмисникам можливість реалізувати її по відношенню до публічних сервісів, таких як HTTP та FTP.

Приклади реалізації атаки: ftp_write, imap, Warezclient та інші.

Зондування мережі

Зондування мережі використовується для виявлення дійсних IP-адрес та збору всієї можливої інформації стосовно хостів, наприклад, які сервіси доступні на даному хості, яка операційна система встановлена та інше. Отримана в ході зондування інформація, дозволяє зловмисник більш чітко спланувати вектор подальшої атаки, беручи до уваги наявні слабкі місця системи

Приклади реалізації атаки: IP-sweep, Port-sweep та інші.

Реалізація більшості мережевих атак потребує відносно значного часу, тому є досить важливим вчасно помітити наявні аномалії в мережі, викликані процесом проведення атаки, та відреагувати на них відповідним чином.

1.3 Виявлення аномалій в мережевому трафіку

Своєчасне виявлення аномалій в мережевому трафіку грає вирішальну роль у забезпеченні безпеки мережі, або окремого хоста. Аномалії можуть бути спричинені багатьма причинами та нести серйозні наслідки.

Основним інструментом для виявлення аномалій слугує система виявлення мережевих вторгень на основі аномалій (англ. ANIDS – anomaly-based network intrusion detection system) [4]. Вона проводить класифікацію мережевого трафіку на нормальний та аномальний. Класифікація може виконуватися на основі певних припущень, або як порівняння трафіку з наявними шаблонами атак. Правильно налаштована система зможе виявляти як відомі, так і невідомі атаки, без потреби у попередніх знаннях щодо цих атак.

Загалом розрізняють два основі види системи виявлення вторгнень:

- Host-based IDS (HIDS) – система виявлення вторгнень розташована на певному кінцевому хості. Основною її задачею є контроль та аналіз внутрішніх елементів хоста. Вона відстежує будь-яку несанкціоновану діяльність, направлену на отримання незаконного доступу до тих чи інших системних ресурсів комп’ютера. Дана система виступає в ролі другої ланки захисту, занотовуючи всі випадки коли хтось, віддалено або локально, оминає політику безпеки операційної системи.
- Network-based IDS (NIDS) – система виявлення вторгнень, яка аналізує безпосередньо мережевий трафік. Основною її задачею є виявлення наявних в мережевих зв’язках аномалій, які можуть

виникати внаслідок реалізації якогось типу мережевої атаки. NIDS аналізує всі вхідні пакети, або потоки, намагаючись при цьому знайти підозрілі закономірності. Наприклад, якщо система помітить, що за короткий час відбулася велика кількість TCP запитів на підключення до різних портів, то вона висуне припущення про те, що можливо хтось виконує сканування портів на одному з хостів у мережі. Приклад архітектури NIDS зображено на рисунку 1.1

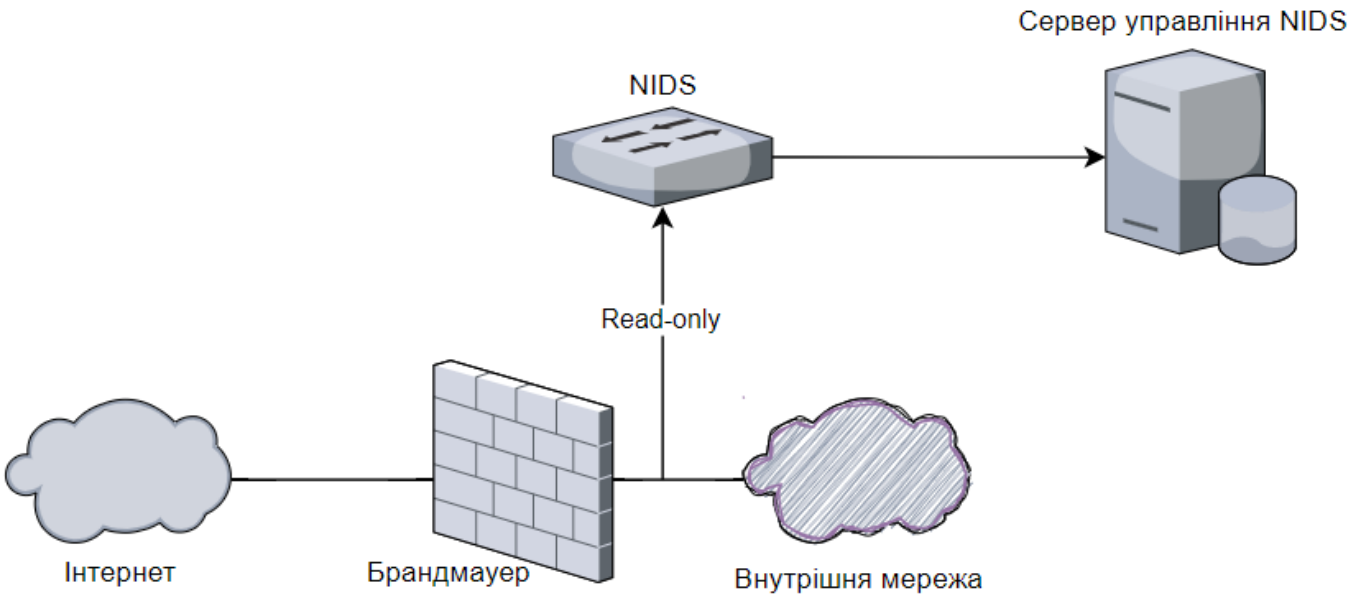

Рисунок 1.1 – Архітектура NIDS

Одним зі способів роботи ANIDS є використання попередньо отриманих знань стосовно трафіку [4]. Тобто модель прогнозування будується, як для нормального трафіку, так і для аномального, і відповідно в подальшому проводиться порівняння нових спостережень з моделлю прогнозування, та їх класифікація.

1.4 Підготовка мережевого трафіку для аналізу

Перехоплення мережевого трафіку

Проведення аналізу трафіку неможливе без предмету дослідження, тому досить важливим етапом виявлення кіберзагроз, є саме перехоплення мережевих пакетів, без втрат та з точними відмітками часу. Для реалізації цієї задачі використовуються різні програмні засоби, наприклад Wireshark, tcpdump, NetFlow Traffic Analyzer та інші.

Більш детально розглянемо перехоплення трафіку за допомогою Wireshark, так як саме даний інструмент застосовувався у подальшій роботі. Wireshark – досить відома програма для аналізу трафіку, однією з функцій якої є саме захоплення пакетів в режимі реального часу. Інтерфейс програми продемонстровано на рисунку 1.2.

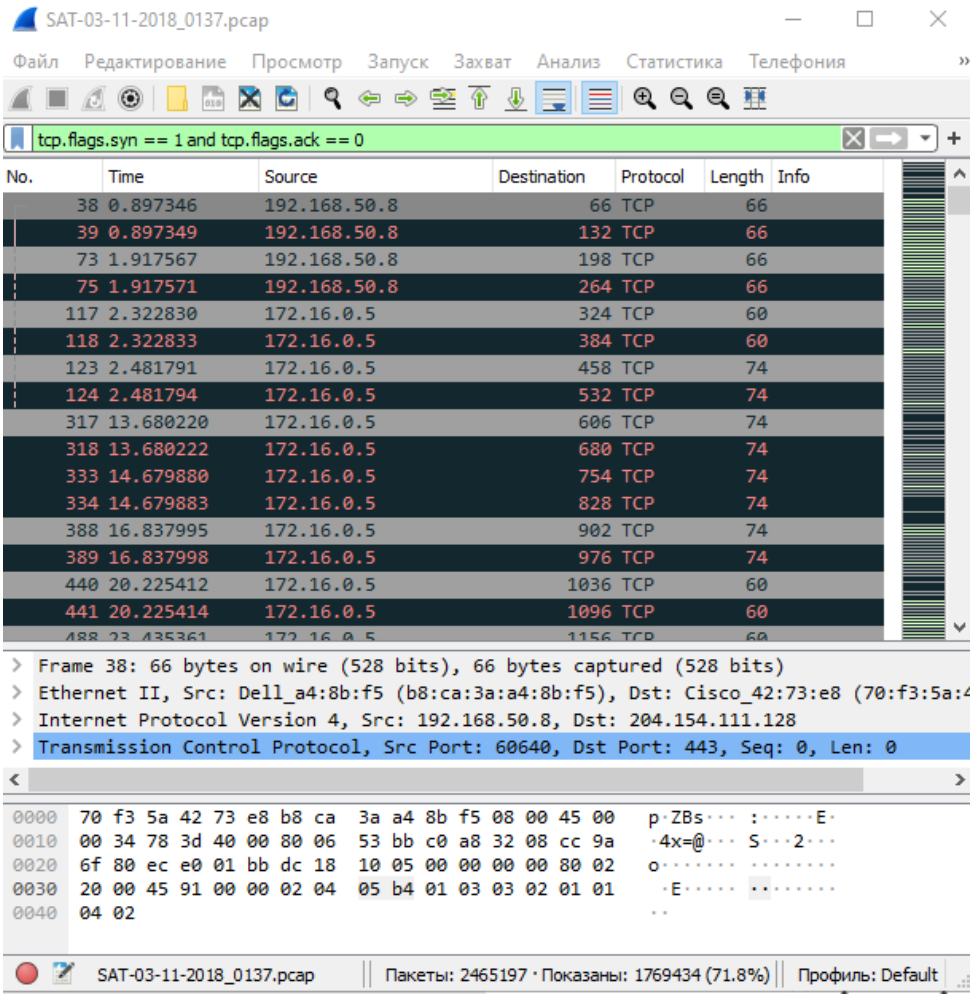

Рисунок 1.2 – Інтерфейс Wireshark

На рисунку 1.2 представлено фрагмент датасету, який використовувався у подальшому дослідженні в роботі. Wireshark представляє перехоплені пакети у форматі PCAP файлу, що значно полегшує проведення подальшого їх аналізу. Наприклад, одним з доступних інструментів для аналізу є фільтрування пакетів, яке продемонстроване на даному рисунку. У обраному наборі даних присутня атака типу DDoS SYN Flood, під час цієї атаки зловмисник відправляє велику кількість SYN запитів на встановлення TCP з'єднання, встановлюючи при цьому прапорець ACK рівним 0, що дозволяє йому не очікувати відповіді жертви на запит. Фільтр «tcp.flags.syn == 1 and tcp.flags.ack == 0» каже Wireshark вивести лише ті пакети, які за структурою схожі на даний тип атаки.

Формування простору початкових параметрів

Після перехоплення мережевих пакетів, наступним важливим кроком є вибір початкових параметрів з даних.

Вибір параметрів зменшує обчислювальну складність алгоритмів аналізу, усуває надмірність інформації та полегшує розуміння даних. Загалом існують чотири типи параметрів

- Базові параметри: тип протоколу, встановлені прапорці, кількість байтів переданих в прямому та зворотному напрямках.
- Параметри за змістом: кількість невдалих спроб входу
- Параметри за часом: кількість підключень до певного хосту за певний проміжок часу
- Параметри за зв'язками: кількість пакетів в прямому напрямку за певний проміжок часу

Більшість необхідних для аналізу параметрів може бути отримана безпосередньо з Wireshark, однак інколи виникає необхідність у додаткових змінних. В такому випадку для їх отримання можуть використовуватися

програмні реалізації відповідних функцій для обрахування даних змінних, написані, наприклад, на мовах програмування Python або R, які досить добре працюють з наборами даних.

Пошук наборів даних для проведення аналізу

Інколи виникають ситуації в яких необхідні набори даних відсутні. Наприклад потрібно змоделювати якусь атаку та перевірити чи зможе система її розпізнати, однак даних щодо цього типу атаки немає. У такому випадку існують різні підходи щодо отримання необхідного набору даних:

- Використання відомих наборів даних. До них відносяться набори мережевого трафіку, перехопленого під час лабораторних досліджень, різного роду кіберзмагань, або конференцій стосовно кібербезпеки. Наприклад одним з таких наборів даних є KDDcup99 [5], який був отриманий під час міжнародного конкурсу KDD Cup 1999 та базувався на даних з тестування системи виявлення вторгнень DARPA у 1998 році. Загалом даний набір навчальних даних складається приблизно з 4 900 000 спостережень, кожне з яких описано за допомогою 41 параметра та має позначення звичайного трафіку, або атаки певного типу.
- Використання відкритих наборів даних з реальним трафіком. До них відносяться набори даних отримані в результаті повсякденної роботи мережі, протягом значного періоду часу. Прикладом такого набору даних може слугувати UNIBS-2009 [6], який був утворений під час перехоплення трафіку на кінцевому маршрутизаторі мережі кампусу університету Brescia в Італії, протягом трьох робочих днів. Він містить 79000 мережевих зв’зків, серед яких є значна кількість різних протоколів (HTTP/HTTPS, TCP, UDP, POP3, IMAP, SMTP та інші)

- Штучне створення наборів даних. Останнім варіантом отримання необхідного набору даних є його утворення власноруч. Існує доволі велика кількість інструментів які дозволяють створити та надіслати мережеві пакети з необхідними для дослідження параметрами.

1.5 Класифікація методів виявлення аномалій в мережевому трафіку

Статистичні методи

Зазвичай статистичні методи аналізу базуються на розподілі мережевого трафіку, тобто висувається припущення що мережевий трафік за структурою належить до певного конкретному розподілу. У такому випадку найпростішим способом побудови статистичної моделі трафіку є обчислення значень функції розподілу імовірностей для кожного відомого класу мережевого трафіку. При побудові функції зазвичай беруться до уваги два різні профілі, що відповідають звичайному трафіку, та трафіку під час реалізації мережевої атаки. У подальшому проводиться перевірка певної нової вибірки спостережень на належність до одного з визначених класів, на основі утвореної функції розподілу імовірностей.

На практиці існує два основні підходи побудови функції розподілу імовірностей:

- Параметричні, які висувають припущення, що мережевий трафік за структурою належить до певного конкретному розподілу.
- Непараметричні, в яких не робиться додаткових припущень щодо природи розподілу мережевого трафіку, а побудова функції розподілу імовірностей проводиться безпосередньо на наявних даних.

Методи на основі класифікації

Класифікація – це проблема визначення до якої з категорій належить те чи інше нове спостереження, на основі навчального набору з маркованими даними.

Класифікаційні методи засновані на утворенні моделі мережевого трафіку, яка в свою чергу дозволяє розділити всі наявні спостереження на певну кількість класів. Однак при використанні даного типу методів виникає необхідність у наявності маркованих наборів даних, що вимагає залучення значної кількості обчислювальних ресурсів.

Для виявлення аномалій в мережевому трафіку використовуються такі методи, засновані на класифікації:

- Метод k-найближчих сусідів
- Метод опорних векторів
- Метод застосування дерева рішень

Методи на основі наявних викидів

Викиди – це ті точки в наборі даних, які суттєво відрізняються за значеннями від решти спостережень та мають специфічну природу походження.

Даний тип методів пошуку аномалій базується саме на виявленні викидів у наборах даних, сформованих на основі перехопленого мережевого трафіку, які можуть бути викликані як внаслідок збоїв у роботі мережевого обладнання, так і внаслідок реалізації якогось типу мережевої атаки. Загалом розрізняють такі методи пошуку викидів:

- Методи на основі відстані між спостереженнями
- Методи на основі щільності спостережень

- Методи з використання машинного навчання

Методи на основі кластеризації

Кластеризація – це завдання поділу набору даних на так звані кластери, в межах яких спостереження більш подібні між собою, аніж до спостережень в інших кластерах.

Методи виявлення аномалій на основі кластеризації досить часто використовуються у кібербезпеці. Наразі можна виділити такі алгоритми кластеризації:

- Алгоритми однозв’язкової кластеризації
- Алгоритми кластеризації k-середніх
- Алгоритми ієрархічної кластеризації

Використання алгоритмів кластеризації для виявлення аномалій в мережевому трафіку має певні переваги:

- надає змогу швидкого формування відповіді стосовно природи спостережень;
- зменшує обчислювальну складність при використанні великих наборів даних;
- забезпечує стабільну ефективність у порівнянні з класифікаторами та статистичними методами.

Та певні недоліки

- більшість алгоритмів здатні обробляти лише безперервні параметри;
- використання некоректної оцінки близькості спостережень може суттєво вплинути на результативність;
- динамічне оновлення кластерів займає багато часу.

Висновки до розділу 1

В даному розділі було наведено загальні відомості стосовно природи походження аномалій в мережевому трафіку. Було проведено короткий огляд мережевих атак та їх особливостей.

Проведено опис ключових складових підготовки до аналізу мережевого трафіку на наявність аномалій, а саме: перехоплення трафіку, вибір початкових параметрів та пошук навчальних наборів даних. Наведено класифікацію основних методів виявлення аномалій в мережевому трафіку.

2 ФАКТОРНИЙ АНАЛІЗ СИСТЕМИ ПОКАЗНИКІВ ДЛЯ АНАЛІЗУ PCAP-ФАЙЛІВ УМОВАХ ПЕРЕХРЕСНОЇ ПЕРЕВІРКИ

2.1 Опис дослідження

Важливим етапом на шляху організації ефективної системи моніторингу подій інформаційної безпеки є розробка якісних факторних моделей мережевого трафіку, що в поєднанні зі статистичними методами дослідження забезпечують виявлення аномальної мережевої активності в режимі реального часу. Застосування факторного аналізу дозволяє зменшити розмірність простору початкових параметрів шляхом їх заміни на меншу кількість нових факторів, обрахованих за допомогою лінійного перетворення. Для утворення факторів використовується спільна дисперсія кожного параметру, що дозволяє детермінувати структуру взаємозв’язків між змінними та виявити наявні приховані зв’язки.

2.2 Підготовка наборів вхідних даних

Дослідження проводиться на основі набору лабораторних зразків пакетів мережевого трафіку, захоплених в ході атаки класу DDoS SYN Flood, що був опублікований Канадським інститутом кібербезпеки [7, 8] для подальшого вивчення.

Загалом для подальшого аналізу було відібрано 400 000 пакетів мережевого трафіку. Характеристикою кожного мережевого з’єднання вважалися IP-адреса та номер порту з яких був відправлений пакет і IP-адреса та номер порту призначення.

Спираючись на матеріали публікації Костянтиноса Демерціса, Нікоса Цирітоса та інших [9] було сформовано простір початкових параметрів. Таким чином кожне мережеве з’єднання описувалося за допомогою 21 параметра, які наведені у таблиці 2.1. Була виконана програмна реалізація функцій для

обрахунку кожного з цих параметрів. Слід зауважити що у їх визначеннях використовуються поняття прямого та зворотного напрямків передачі пакетів. Вважалося що напрямок передачі задається першим пакетом у з'єднанні.

Також варто зазначити що поділ на підпотоки виконувався наступним чином: у разі якщо різниця в часі між двома послідовними мережевими пакетами у одному з'єднанні перевищувала 0.5 секунд, то вважалося що другий пакет вже відноситься до іншого підпотоку.

Таблиця 2.1 – Вхідні параметри

ID	**Ім'я**	**Інтерпретація**
1	**total_fpackets**	***Загальна кількість пакетів, що рухаються в прямому напрямку.***
2	**total_bpackets**	***Загальна кількість пакетів, що рухаються в зворотному напрямку.***
3	**min_fpktl**	***Мінімальна довжина пакета (у байтах) в прямого напрямку.***
4	**max_fpktl**	***Максимальна довжина пакета (у байтах) в прямого напрямку.***
5	**min_bpktl**	***Мінімальна довжина пакета (у байтах) в зворотному напрямку.***
6	**max_bpktl**	***Максимальна довжина пакета (у байтах) в зворотному напрямку.***
7	**min_fiat**	***Мінімальний інтервал (у мікросекундах) між двома пакетами у прямому напрямку.***
8	**max_fiat**	***Максимальний інтервал (у мікросекундах) між двома пакетами у прямому напрямку.***
9	**min_biat**	***Мінімальний інтервал (у мікросекундах) між двома пакетами у зворотному напрямку.***
10	**max_biat**	***Максимальний інтервал (у мікросекундах) між двома пакетами у зворотному напрямку.***
11	**duration**	***Час, що минув (у мікросекундах) від першого пакета до останнього.***
12	**min_active**	***Мінімальний час, що минув (у мікросекундах) в підпотоці).***
13	**max_active**	***Максимальний час, що минув (у мікросекундах) в підпотоці.***
14	**min_idle**	***Мінімальний час (у мікросекундах) затримки потоку.***
15	**max_idle**	***Максимальний час (у мікросекундах) затримки потоку.***
16	**sflow_fpackets**	***Середня кількість пакетів в прямому напрямку в підпотоках.***
17	**sflow_fbytes**	***Середня кількість байтів в прямому напрямку в підпотоках.***
18	**sflow_bpackets**	***Середня кількість пакетів в зворотному напрямку в підпотоках..***
19	**sflow_bbytes**	***Середня кількість байтів в зворотному напрямку в підпотоках.***
20	**total_fhlen**	***Загальна довжина заголовка (мережевий і транспортний рівень) пакетів, що рухаються в прямому напрямку.***
21	**total_bhlen**	***Загальна довжина заголовка (мережа та транспортний рівень) пакетів, що рухаються у зворотному напрямку.***

Для побудови початкової факторної моделі обраний набір вхідних даних включав 360 000 пакетів мережевого трафіку, з якого було виділено 123329 мережевих з'єднань виду <IP-адреса джерела, номер порту джерела>, <IP-адреса призначення, номер порту призначення>, описаних значеннями 21 параметра (таблиця 2.1). На основі запропонованих даних було сформовано матрицю спостережень $X = ||x_{ij}||,\ i = \overline{1,n},\ j = \overline{1,m}$, де n = 123329 – кількість мережевих з'єднань, m = 21 – кількість параметрів. Частина отриманої матриці спостережень наведена у додатку А.

2.3 Методика факторного аналізу

Концепція дослідницького факторного аналізу базується на визначенні певної кількості прихованих змінних (факторів), що пояснюють кореляцію між початковими параметрами. Тобто виконується дослідження багатовимірного простору ознак, описаних набором початкових параметрів, в ході якого виконується аналіз дисперсії та кореляцій між початковими параметрами з точки зору меншої кількості прихованих змінних. Дослідницька факторна модель може бути представлена у наступному вигляді:

$$X - \mu = \Lambda F + e, \tag{2.1}$$

де X — початкові параметри (спостережуванні змінні), μ — середні значення спостережень, Λ — матриця факторних навантажень, F — фактори(приховані змінні), e — помилки(залишки).

Матриця факторних навантажень Λ визначається шляхом максимального наближення матриці коваріацій Σ до матриці коваріацій $Cov(X - \mu)$ з використання методу оцінки максимальної правдоподібності.

Матриця Σ визначається наступним чином:

$$\Sigma = \Lambda\Lambda^T + \Psi, \text{ де } \Psi = Cov(e) \tag{2.2}$$

У цьому рівнянні доданок $\Lambda\Lambda^T$ відповідає тій частині дисперсії, що пояснюється лінійною комбінацією факторів, а Ψ – це невідома частина загальної дисперсії, яку неможливо пояснити за допомогою лінійної комбінації факторів. Детально ознайомитись з методикою факторного аналізу можна на електронному ресурсі [10].

Визначення факторних навантажень як оцінок максимальної правдоподібності на основі SVD підходу

Метод отримання оцінок максимальної правдоподібності спирається на ітеративний алгоритм, в ході якого використовуються наступні припущення:

a) Початкова матриця невідомих дисперсій є одиничною: $\widehat{\Psi} = \mathrm{E}_{\mathrm{n_f}}$;
b) Бажаний рівень точності алгоритму задається константою δ (приймемо $\delta = 1 * 10^{-12}$).

Визначимо, що n_s – кількість спостережень, n_f – кількість параметрів, n_c – кількість факторів, тоді в ході кожної ітерації алгоритму обраховуватимемо наступне:

1. $\widehat{\Psi}_{\mathrm{sqrt}} = \sqrt{\widehat{\Psi}} + \delta$
2. $\widehat{\mathrm{X}} = \mathrm{X} / (\widehat{\Psi}_{\mathrm{sqrt}} \sqrt{\mathrm{n_s}})$;
3. Застосуємо SVD розклад до $\hat{X}$: $\hat{X} = USV^*$,
   де U – матриця лівих сингулярних векторів, S – матриця сингулярних чисел та V^* - транспонована матриця правих сингулярних векторів. Надалі використовуватимемо матрицю сингулярних чисел $\hat{S}$ та транспоновану матриця правих сингулярних векторів $\hat{V}^*$, обмежені обраною кількістю факторів.
4. Розрахуємо приблизні значення матриці навантажень $\hat{\Lambda}$ та матриці невідомих дисперсій $\widehat{\Psi}$:

$$\hat{\Lambda} = \sqrt{\hat{S}^2 - E}\hat{V}^*\widehat{\Psi}_{sqrt};$$

$$\widehat{\Psi} = max\ (\sum_{j=1}^{n_f} {x_j}^2 - \sum_{j=1}^{n_c} {\lambda_j}^2, \delta).$$

5. Обчислимо логарифмічну функцію правдоподібності:

$$\ln L(x, \mathrm{S}, \Psi) = -\frac{n_s}{2}(n_s ln(2\pi) + n_c + \sum_{i=1}^{n_c} ln\left({\hat{s}_i}^2\right) +$$
$$\sum_{i=1}^{n_s} \sum_{j=1}^{n_c} {x_{ij}}^2 - \sum_{i=1}^{n_c} {\hat{s}_i}^2 + \sum_{i=1}^{n_f} ln(\hat{\psi}_i)).$$

Після кожної ітерації обчислене значення логарифмічної функції правдоподібності порівнюється зі значенням, отриманим на попередній ітерації. У разі якщо їх різниця менше δ, робота алгоритму завершується, а матрицям Λ та Ψ присвоюються значення, отримані під час передостанньої ітерації алгоритму.

Важливо зазначити, що під час використання факторного аналізу, лінійну факторну модель неможливо однозначно ідентифікувати. Тобто отримана матриця факторних навантажень визначена з точністю до ортогонального перетворення. Тому виникає необхідність у виборі саме тих розв'язків, які найкраще інтерпретують отриману модель.

Методи ротації факторів

Для вирішення цієї проблеми в факторному аналізі використовуються алгоритми ротації. Загалом існує два типи ротації факторів:

- Ортогональна ротація. Даний метод дозволяє зберегти незалежність факторів, тобто основною умовою є некорельованість факторів між собою.

- Косокутна ротація. На відміну від ортогональної, вона дозволяє факторам корелювати між собою.

Нижче наведено список алгоритмів ротації які наразі використовуються в сучасних мовах програмування R та Python.

- Varimax (ортогональна ротація)
- Quartimax (ортогональна ротація)
- Oblimax (ортогональна ротація)
- Equamax (ортогональна ротація)
- Geomin_ort (ортогональна ротація)
- Promax (косокутна ротація)
- Oblimin (косокутна ротація)
- Quartimin (косокутна ротація)
- Geomin_obl (косокутна ротація)

Varimax максимізує навантаження для кожного фактору, в результаті чого деякі навантаження стають якомога більшими, а решта залишається на помірному, або низькому рівнях. Однак використання даного методу може призвести до появи факторів помірно пов'язаних з одними і тими ж параметрами, що в результаті може вилитись у проблеми з інтерпретацією даної моделі.

Quartimax має схожий принцип роботи, однак значною відмінністю є те, що основною задачею метода ставиться виділення якомога більшої кількості параметрів зі значними навантаженнями на фактори. За результатами використання даного методу, в матриці факторних навантажень залишаються в основному лише значні та низькі навантаження, що зменшує імовірність появи складних для інтерпретації факторів.

Ротація факторів з використання методу Quartimax

Як вже було зазначено в попередньому пункті, основна проблема при виконанні факторного аналізу полягає в тому, що отримана матриця навантажень може бути не єдиною. В даному випадку для закріплення положення матриці навантажень Λ застосовується метод обертання факторів Quartimax, алгоритм якого є також ітеративним.

Початкова матриця ротації визначається як одинична матриця: $R = E_{n_c}$, тоді на кожній ітерації алгоритму виконується наступне:

1. Ротація факторів: $\hat{\Lambda} = \Lambda^T(\Lambda R)^3$;
2. SVD розклад до $\hat{\Lambda}$: $\hat{\Lambda} = USV^*$;
3. Розраховується нова матриця ротації: $R = UV^*$.

Після кожної ітерації обчислюється сума сингулярних чисел S, що порівнюється з попереднім значенням (початкове значення приймається за 0). Якщо сума менша за попереднє значення, то алгоритм завершує роботу, а матриці ротації присвоюється значення, отримане на передостанній ітерації.

2.3 Формування моделі мережевого трафіку на основі методу факторного аналізу

Стандартизація

Над матрицею спостережень X було виконано процедуру стандартизації, в ході якої сформовано стандартизовану матрицю спостережень Z (Додаток Б):

$$z_{ij} = \frac{x_{ij} - \overline{x_j}}{\sigma_j}, \text{ де}$$

$$\overline{x_j} = \frac{1}{n}\sum_{i=1}^{n} x_{ij}, \ \sigma_j = \sqrt{\frac{\sum_{i=1}^{n}(x_{ij} - \overline{x_j})^2}{n-1}}. \quad (2.3)$$

Оцінка якості вибірки

Перед початком формування факторної моделі необхідно визначити, чи можливе проведення факторного аналізу на обраному наборі даних. Для оцінки якості вибірки використовувалися формальні тести Кайзера-Мейера-Олкіна (КМО) та Бартлетта.

Критерій КМО застосовується для перевірки адекватності вибірки. Він з’ясовує чи дійсно зв’язки між парами параметрів обумовлюються впливом

інших параметрів, а не якоюсь третьою стороною. Проводиться порівняння значень коефіцієнтів кореляції між змінними зі значеннями часткових коефіцієнтів кореляції. В залежності від отриманого значення робляться висновки стосовно якості початкового набору даних, чим більше дане значення, в межах від 0 до 1, тим краще модель підходить для застосування факторного аналізу (таблиця 2.2).

Таблиця 2.2 – Оцінка адекватності вибірки за критерієм КМО

Значення КМО	Оцінка адекватності застосування факторного аналізу для даної моделі
$\geq 0{,}9$	Безумовна адекватність
$\geq 0{,}8$	Висока адекватність
$\geq 0{,}7$	Прийнятна адекватність
$\geq 0{,}6$	Задовільна адекватність
$\geq 0{,}5$	Низька адекватність
$< 0{,}5$	Факторний аналіз недоцільно застосовувати

Було розраховане значення КМО для даного набору даних, яке склало:

$$\text{КМО} = 0{,}63$$

Відповідно до таблиці 2.2 було визначено, що адекватність даного набору даних за КМО є задовільної, тому до нього можна застосовувати факторний аналіз.

Критерій сферичності Бартлетта застосовується для перевірки початкової гіпотези H_0, яка стверджує що між параметрами в вибірці не існує кореляційних зв’язків, тобто кореляційна матриця є діагональною (таблиця 2.3).

Таблиця 2.3 – Гіпотеза H_0

Коефіцієнти кореляції	X_1	X_2	X_3	⋯	X_m
X_1	1	0	0	⋯	0
X_2	0	1	0	⋯	0
X_3	0	0	1	⋯	0
⋯	⋯	⋯	⋯	⋯	⋯
X_m	0	0	0	⋯	1

Була проведена перевірка гіпотези H_0, результати наведені в таблиці 2.4.

Таблиця 2.4 – Результати оцінки за критерієм Бартлетта

Хі-квадрат	4260023
Ступені свободи	120
Рівень значимості *p-value*	0,000

За результатами перевірки значимість тесту Бартлетта виявилась рівною 0.000, що означає, що початкова гіпотеза H_0 може бути відкинута з імовірністю помилки 0.000. Замість неї приймається гіпотеза H_1, яка стверджує що між параметрами існують кореляційні зв'язки

Таким чином, спираючись на отримані, в ході оцінки якості вибірки, результати, було з'ясовано що початкові дані є придатними для проведення факторного аналізу.

Визначення кількості факторів

На основі стандартизованої матриці Z була розрахована матриця кореляцій R, яка частково продемонстрована у додатку Б. Наступним кроком були обраховані власні числа кореляційної матриці R та сформовано вектор власних чисел (таблиця 2.5).

Таблиця 2.5 – Власні числа

№	Власне число
1	6.69415531702237310
2	4.05213231184624512
3	3.30036475715912125
4	2.00071543852928535
5	1.91930552297437629
6	0.99375025679372320
7	0.91234231592044290
8	0.50598944571315052
9	0.29891020470433238
10	0.21275936247609525
11	0.07722407156035657
12	0.01657649427059764
13	0.01018711831883819
14	0.00441605158818699
15	0.00111852584040829
16	0.00005123884406233
17	0.00000155120583072
18	0.00000001523193727
19	0.00000000000061468
20	0.00000000000000006
21	0.00000000000000004

Для визначення кількості прихованих факторів використовувався метод Кайзера. Згідно з даним методом, брались до уваги лише ті фактори, значення власних чисел яких більші за одиницю:

- $\lambda_1 \approx 6.69$;
- $\lambda_2 \approx 4.05$;
- $\lambda_3 \approx 3.30$;
- $\lambda_4 \approx 2.00$;
- $\lambda_5 \approx 1.92$.

Таким чином, сумарний відсоток дисперсії, що пояснюють відібрані 5 факторів, склав:

$$\frac{\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 + \lambda_5}{21} * 100 \approx 85.55\%$$

Отримана дисперсія може здаватися досить значною, однак при проведенні перехресної перевірки було з’ясовано, що при зменшенні кількості факторів до 4, втрати інформативності під час застосуванні моделі до тестових наборів даних значно зростають. Саме тому, задля збереження рівня інформативності, було прийнято рішення використовувати 5 факторів.

Після вибору кількості шуканих факторів, виконано розрахунок матрицю навантажень та застосовано до неї метод ротації Quartimax. В результаті отримано фінальну матрицю навантажень (таблиця 2.6), що зокрема описує факторну модель.

Таблиця 2.6 – Матриця факторних навантажень

	Фактор 1	Фактор 2	Фактор 3	Фактор 4	Фактор 5
total_fpackets	0.005506	0.210778	-0.138297	0.967642	0.005256
total_bpackets	-0.003235	0.037833	-0.960765	0.238259	0.005019
min_fpktl	0.999908	0.002673	0.003595	0.012404	0.001506
max_fpktl	0.999908	0.002673	0.003595	0.012404	0.001506
min_bpktl	-0.000619	-0.019698	-0.99319	0.036904	-0.000050
max_bpktl	-0.000619	-0.019698	-0.99319	0.036904	-0.000050
min_fiat	0.001173	-0.016789	0.004755	-0.071709	-0.000086
max_fiat	-0.002781	-0.05729	-0.123133	0.867199	-0.034417
min_biat	0.000435	-0.01681	-0.269282	-0.035837	-0.005518
max_biat	-0.007387	-0.027866	-0.164596	0.605558	-0.013301
duration	0.000866	-0.056591	-0.124352	0.878629	-0.032645
min_active	-0.002861	0.836995	-0.038454	0.042942	0.001691
max_active	-0.003747	0.838163	-0.050685	0.11552	0.003211
min_idle	0.001415	0.013627	0.012655	-0.190796	0.973015
max_idle	-0.001811	0.003964	0.031939	-0.14209	-0.980745
sflow_fpackets	-0.003996	0.99419	-0.044018	0.09806	0.002949
sflow_fbytes	0.013105	0.994095	-0.04395	0.098259	0.002974
sflow_bpackets	-0.000145	0.056558	-0.998288	-0.013781	0.004968
sflow_bbytes	-0.000145	0.056557	-0.998288	-0.013782	0.004971
total_fhlen	0.025164	0.210639	-0.138112	0.967384	0.005288
total_bhlen	-0.003235	0.037832	-0.960766	0.238255	0.005024

В наслідок застосування ротації, факторні навантаження на змінні були максимізовані в межах кожного фактору, що полегшує їх інтерпретацію у відношенні до початкових параметрів. Таким чином на основі матриці

факторних навантажень було виявлено які саме параметри згрупував в собі кожен фактор. Варто зазначити що до уваги бралися лише факторні навантаження значення яких перевищують 0.5 по модулю. Результати зображені на рисунку 2.1.

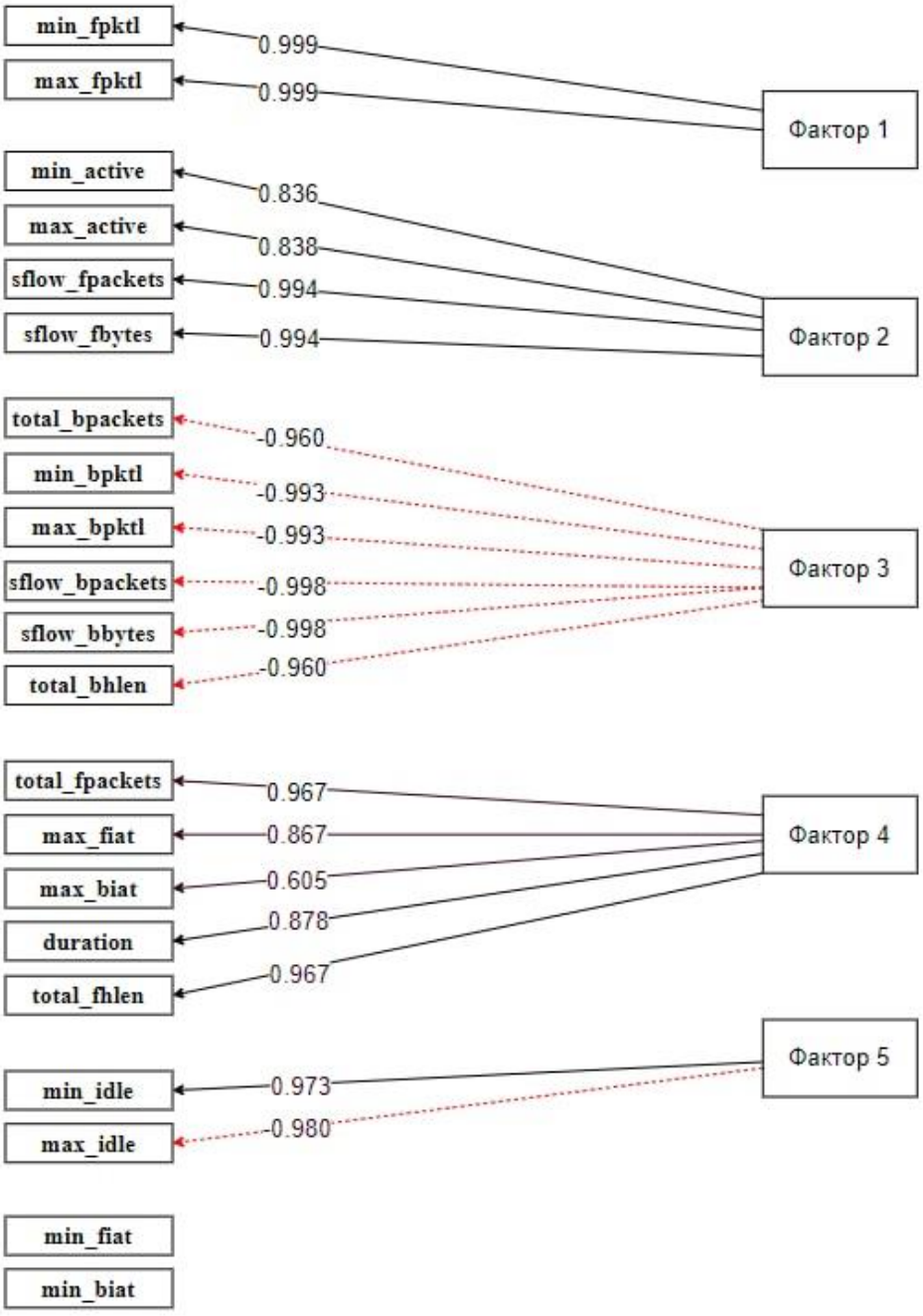

Рисунок 2.1 – Зв’язок факторів з початковими параметрами

В умовах отриманої моделі було розраховано частки дисперсії, що пояснюються кожним з факторів:

- для Фактора 1: 9.529%;
- для Фактора 2: 16.436%;
- для Фактора 3: 28.51%;
- для Фактора 4: 19.059%;
- для Фактора 5: 9.07%.

Таким чином, загальна дисперсія, описана моделлю, складає 82.7%.

Після аналізу зв'язків факторів з початковими параметрами було виявлено, що обраховані фактори мають незначні факторні навантаження на деякі початкові параметри, тому було вирішено обрахувати матрицю залишкових дисперсій (таблиця 2.7).

Таблиця 2.7 – Таблиця залишкових дисперсій

Початковий параметр	Залишкова дисперсія
total_fpackets	0.000049503996
total_bpackets	0.018687819713
min_fpktl	0.000000000001
max_fpktl	0.000000000001
min_bpktl	0.011815099620
max_bpktl	0.011815099620
min_fiat	0.994543818993
max_fiat	0.228321726329
min_biat	0.925881796160
max_biat	0.605191277236
duration	0.208270063748
min_active	0.296097767793
max_active	0.281536574874
min_idle	0.016490178970
max_idle	0.016895793573
sflow_fpackets	0.000000000001
sflow_fbytes	0.000000000001
sflow_bpackets	0.000000169217
sflow_bbytes	0.000000168985
total_fhlen	0.000054074872
total_bhlen	0.018689189627

При аналізі залишкових дисперсій, було з’ясовано, що отримані фактори недостатньо інтерпретують два параметри, а саме: min_fiat та min_biat. Значення їх залишкових дисперсій складають 0.99454 та 0.92588 відповідно. Тобто частки їх дисперсій, описані знайденими факторами, складають лише 0.546% та 7.412% відповідно.

Тож, якщо параметри min_fiat та min_biat будуть враховуватись на рівні з обрахованими факторами, загальна дисперсія, описана факторною моделлю, складатиме 90.8%. Запропонований підхід доцільно використовувати у випадках, коли критично важливо максимізувати інформативність моделі, проте з метою оптимізації швидкодії алгоритмів у випадку обробки великих обсягів даних такими факторами допускається знехтувати.

Трансформування початкової матриці спостережень у простір обрахованих факторів

Перехід від простору початкових параметрів до простору обрахованих факторів проводиться за даною формулою:

$$F = (X - \mu)(\Lambda/\Psi)^T(E + (\Lambda/\Psi)\Lambda^T)^{-1} \quad (2.4)$$

Розрахована в попередньому пункті модель була застосована до початкової матриці спостережень, відповідно до формули (2.4), частина отриманої матриці спостережень в просторі обрахованих факторів продемонстрована у додатку В.

2.4 Методика тест-валідації

Отримана в ході дослідження модель добре описує початковий набір даних, однак для підтвердження її валідності необхідно провести загальну оцінку якості декомпозиції моделі.

Одним з підходів до оцінки якості декомпозиції моделі є тест-валідація [11]. Даний метод розуміє під собою використання двох різних наборів даних. На першому наборі даних формується модель, він називається навчальним. Інший виступає в ролі тестового та використовується для перевірки моделі.

Алгоритм проведення тест-валідації можна описати наступним чином:

1. Позначимо матрицю спостережень навчального набору даних як X_c, а тестового набору як X_t.
2. Застосуємо, сформовану на навчальних даних, факторну модель до навчального та тестового наборів, відповідно до формули (2.4), в результаті чого отримаємо значення матриць спостережень X_c та X_t в просторі прихованих факторів.
3. Виконаємо обернене перетворення над матрицями в просторі прихованих факторів, в наслідок чого отримаємо значення $\hat{X}_c$ та $\hat{X}_t$ в просторі початкових параметрів.
4. Обрахуємо матриці залишків в навчанні $- E_c$, та тестуванні $- E_t$, за формулами:

$$E_c = X_c - \hat{X}_c$$

$$E_t = X_t - \hat{X}_t \quad (2.5)$$

5. Для оцінки якості декомпозиції обрахуємо наступні величини:
   - для навчального набору:

$$TRVC = \frac{1}{n_c m} \sum_{i=1}^{n_c} \sum_{j=1}^{m} {e_c}_{ij}^2$$

$$ERVC = 1 - \frac{\sum_{i=1}^{n_c} \sum_{j=1}^{m} {e_c}_{ij}^2}{\sum_{i=1}^{n_c} \sum_{j=1}^{m} {x_c}_{ij}^2}, \text{де} \quad (2.6)$$

n_c – кількість спостережень в навчальному наборі даних, m – кількість початкових параметрів, $e_{c_{ij}}$ – елементи матриці

залишків в навчанні, $x_{c_{ij}}$ – елементи матриці спостережень навчального набору.

- для тестового набору:

$$TRVC = \frac{1}{n_t m}\sum_{i=1}^{n_t}\sum_{j=1}^{m} e_{t_{ij}}^2$$

$$ERVC = 1 - \frac{\sum_{i=1}^{n_t}\sum_{j=1}^{m} e_{t_{ij}}^2}{\sum_{i=1}^{n_t}\sum_{j=1}^{m} x_{t_{ij}}^2}, \text{де} \qquad (2.7)$$

n_t – кількість спостережень в тестовому наборі даних, m – кількість початкових параметрів, $e_{t_{ij}}$ – елементи матриці залишків в тестуванні, $x_{t_{ij}}$ – елементи матриці спостережень тестового набору.

2.5 Перехресна перевірка

Проведено перевірку якості факторної моделі за методом тест-валідації для розширеного набору лабораторних зразків пакетів мережевого трафіку в умовах алгоритму базової k-блочної перехресної перевірки (англ., K-Fold Cross Validation), що описана в матеріалах [12].

Алгоритм k-блочної перехресної перевірки застосовується в умовах обмеженої кількості вхідних даних. Для його використання необхідно розділити початковий набір даних на k-частин, кожна з яких послідовно використовується в якості тестового набору, а інші k-1 частин виступають в ролі навчального.

Для розбиття початкової множини даних на тестові та навчальні набори розглянемо розширений набір лабораторних зразків пакетів мережевого трафіку, захоплених в ході атаки класу DDoS SYN Flood, як універсальну множину T, що складається з 400 000 пакетів. Виконаємо розбиття даного

набору на 10 послідовних рівних блоків по 40 000 пакетів відповідно: $\{T_0, T_1, \dots, T_9\}$. Таким чином, в ході перехресної перевірки елементи T_i, i = $\overline{0,9}$ такого розбиття по черзі виконуватимуть роль тестового набору. Далі визначимо навчальні набори для кожного тестового як абсолютне доповнення до множини T: $C_i = \overline{T_i} = T \backslash T_i$, i = $\overline{0,9}$.

Факторна модель, сформована на основі навчального набору даних, застосовується до даних відповідного перевірочного набору, після чого отримані результати порівнюють з перевірочними значеннями. Таким чином, приймається рішення щодо якості та точності запропонованої моделі.

Наступні величини слугують для оцінки якості моделювання в середньому:

- TRVC – повна дисперсія залишків в навчанні;
- ERVC – пояснена дисперсія залишків в навчанні;
- TRVP – повна дисперсія залишків в тестуванні;
- ERVP – пояснена дисперсія залишків в тестуванні.

Результати тест-валідації, обраховані за формулами (2.6) та (2.7), в умовах перехресної перевірки для всіх пар навчальних та тестових наборів (C_i, T_i), i = $\overline{0,9}$ наведено в таблиці 2.8.

Таблиця 2.8 – Повна та пояснена дисперсії в навчанні та тестуванні

	(C_0, T_0)	(C_1, T_1)	(C_2, T_2)	(C_3, T_3)	(C_4, T_4)	(C_5, T_5)	(C_6, T_6)	(C_7, T_7)	(C_8, T_8)	(C_9, T_9)
TRVC	0.17705	0.17497	0.17227	0.17314	0.17341	0.17243	0.17375	0.17459	0.17502	0.17558
ERVC	0.82295	0.82503	0.82773	0.82686	0.82659	0.82757	0.82625	0.82541	0.82498	0.82442
TRVP	0.19998	0.23296	0.208	0.20959	0.20773	0.20188	0.21012	0.20512	0.20733	0.20634
ERVP	0.77896	0.76703	0.79199	0.76833	0.79226	0.77685	0.76775	0.77327	0.79266	0.77193

Слід зазначити, що найкращі результати якості моделювання були отримані факторними моделями, сформованими на наступних навчальних наборах даних:

- C_2 з показниками: $\mathrm{TRVC} = 0.17227$, $\mathrm{ERVC} = 0.82773$, $\mathrm{TRVP} = 0.20800$, $\mathrm{ERVP} = 0.79199$;
- C_4 з показниками: $\mathrm{TRVC} = 0.17341$, $\mathrm{ERVC} = 0.82659$, $\mathrm{TRVP} = 0.20773$, $\mathrm{ERVP} = 0.79226$;
- C_8 з показниками: $\mathrm{TRVC} = 0.17502$, $\mathrm{ERVC} = 0.82498$, $\mathrm{TRVP} = 0.20733$, $\mathrm{ERVP} = 0.79266$.

Для отримання більш чіткої картини стосовно можливої кількості факторів, додатково була проведена перехресна перевірка моделі, що містить чотири фактори. Отримані результати представлені в таблиці 2.9.

Таблиця 2.9 – Повна та пояснена дисперсія в навчанні та тестуванні для моделі з чотирма факторами

	(C_0, T_0)	(C_1, T_1)	(C_2, T_2)	(C_3, T_3)	(C_4, T_4)	(C_5, T_5)	(C_6, T_6)	(C_7, T_7)	(C_8, T_8)	(C_9, T_9)
TRVC	0.26957	0.26706	0.2641	0.26468	0.26466	0.26368	0.2654	0.26666	0.26743	0.26833
ERVC	0.73043	0.73294	0.7359	0.73532	0.73534	0.73632	0.7346	0.73334	0.73257	0.73167
TRVP	0.26133	0.29459	0.26995	0.2719	0.26996	0.26375	0.27208	0.2671	0.26972	0.2679
ERVP	0.71115	0.70539	0.73003	0.69946	0.73002	0.70846	0.69926	0.70477	0.73026	0.70388

Аналізуючи дану таблицю можна помітити, що в деяких випадках значення поясненої дисперсії залишків в тестуванні виявилися в межах 70%, що може вилитися у значні втрати інформативності при використанні даної моделі.

Також було проведено аналіз залишкових дисперсій, значення яких продемонстровані в таблиці 2.10

Таблиця 2.10 – Таблиця залишкових дисперсій для моделі з чотирма факторами

Початковий параметр	Залишкова дисперсія
total_fpackets	0.000049708564
total_bpackets	0.018688175993
min_fpktl	0.000000000001
max_fpktl	0.000000000001
min_bpktl	0.011840425449
max_bpktl	0.011840425449
min_fiat	0.994543915926
max_fiat	0.229807865120
min_biat	0.925926000643
max_biat	0.605465171862
duration	0.209625773675
min_active	0.296098000315
max_active	0.281536827372
min_idle	0.964812314148
max_idle	0.977251919161
sflow_fpackets	0.000000000001
sflow_fbytes	0.000000000001
sflow_bpackets	0.000000169221
sflow_bbytes	0.000000168989
total_fhlen	0.000053870652
total_bhlen	0.018689537821

Беручи до уваги таблицю залишкових дисперсій, можна сказати, що модель побудована на чотирьох факторах неякісно описує значну кількість параметрів, а саме:

- min_fiat, з поясненою часткою загальної дисперсії – 0,54%
- min_biat, з поясненою часткою загальної дисперсії – 7,41%
- min_idle, з поясненою часткою загальної дисперсії – 3,52%
- max_idle, з поясненою часткою загальної дисперсії – 2,28%

Отже, як вже і було вказано раніше, більш доцільно використовувати модель побудовану на п’яти факторах, з міркувань збереження належного рівня інформативності.

Висновки до розділу 2

Виконано дослідження системи показників для аналізу захоплених пакетів мережевого трафіку за допомогою методу факторного аналізу, в ході якого:

- За критерієм Кайзера визначено 5 прихованих факторів, що в сумі пояснюють 85.38% дисперсії даних;
- За методом максимальної правдоподібності та алгоритмом ротації Quartimax розраховано матрицю навантажень, що описує зв’язок між факторами 1-5 та вхідними параметрами.
- Проведено оцінку якості декомпозиції з використанням тест-валідації в умовах перехресної перевірки

Дослідження показало, що завдяки застосуванню алгоритму ротації Quartimax отримані в ході перехресної перевірки моделі демонструють однаково високі результати як на навчальних, так і на тестових наборах. В свою чергу, моделі з найкращими оцінками якості в середньому були виконані на навчальних наборах C_2, C_4, C_8.

3 ВІЗУАЛІЗАЦІЯ МОДЕЛІ ТА ПРОВЕДЕННЯ КЛАСТЕРНОГО АНАЛІЗУ

3.1 Використання візуалізації

Важливим кроком будь-якого дослідження є візуалізація отриманих результатів. Великі об’єми даних досить складно засвоюються людиною, тому представлення результатів у вигляді графіків, діаграм та схем є більш сприйнятливим. Для проведення візуалізації використовувалися засоби мов програмування R та Python, а саме бібліотеки: Plotly, Matplotlib та Psych.

Проведення візуалізації також допомагає більш чітко зрозуміти описану факторної моделлю структуру зв’язків між початковими параметрами.

3.2 Візуальне представлення матриці факторних навантажень

Для кращого розуміння факторної моделі, можна представити відношення між кожним з отриманих факторів у вигляді попарних графіків, зображених на рисунку 3.1.

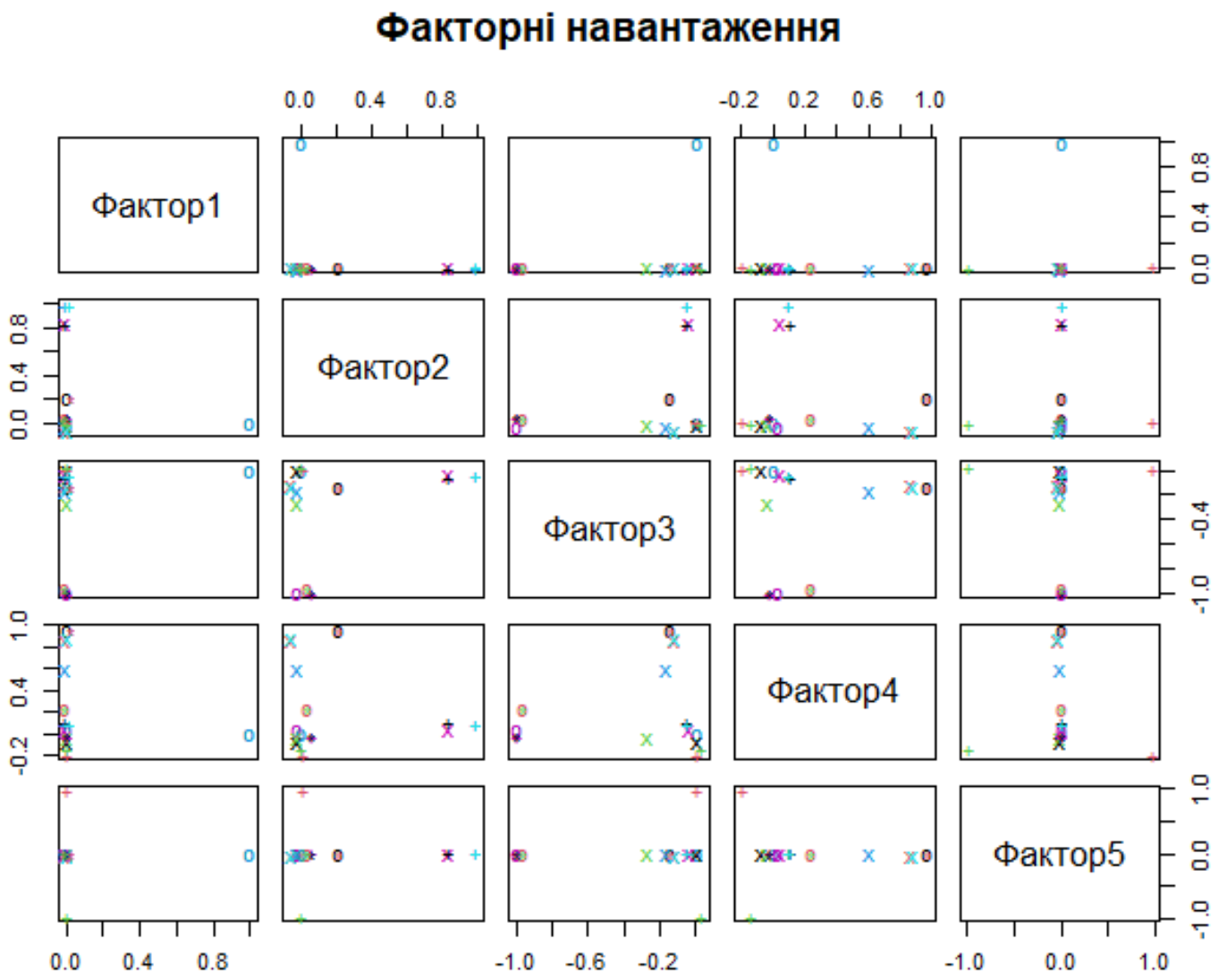

Рисунок 3.1 – Графіки факторних навантажень

o total_fpackets	x max_fiat	+ max_idle
o total_bpackets	x min_biat	+ sflow_fpackets
o min_fpktl	x max_biat	+ sflow_fbytes
o max_fpktl	x duration	+ sflow_bpackets
o min_bpktl	x min_active	* sflow_bbytes
o max_bpktl	+ max_active	* total_fhlen
x min_fiat	+ min_idle	* total_bhlen

Рисунок 3.2 – Позначення для параметрів

На даних графіках продемонстровано, як саме розподілилися факторні навантаження на початкові змінні в межах кожної пари факторів.. Для кожного початкового параметру призначене відповідне графічне позначення, згідно таблиці зображеній на рисунку 3.2.

Графіки розташовані у вигляді матриці, таким чином кожна клітинка з координатами (i, j), де i, j = $\overline{1,5}$, відповідає відношенню між навантаженнями i-го та j-го факторів. Наприклад, для пари факторів Фактор 1 та Фактор 5, можна сказати що більшість факторних навантажень мають координати близькі до (0, 0) і лише для параметрів sflow_bpackets – (0, 1), max_fpktl – (1, 0) та sflow_fbytes – (0, -1).

Іншим способом графічного представлення матриці факторних навантажень є використання теплової карти (рисунок 3.3).

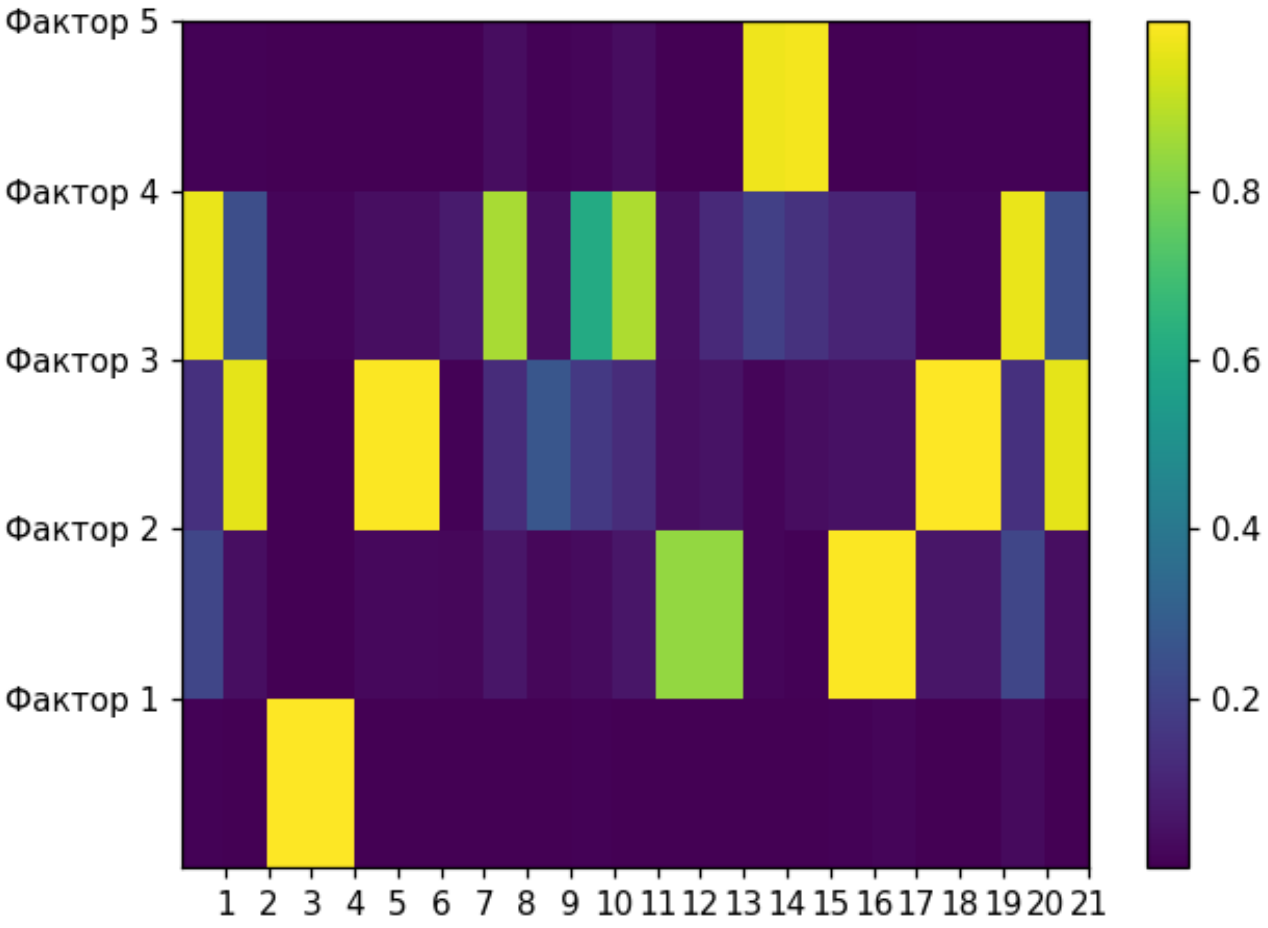

Рисунок 3.3 – Теплова карта факторних навантажень

На даній діаграмі зв’язок кожного фактору з початковими параметрами представлено у вигляді різнокольорових клітинок. Відповідно до шкали, розміщеній у правій частині діаграми, кожний колір вказує на силу зв’язку в межах від 0 до 1 за модулем. На осі X знаходяться номера параметрів відповідно до таблиці 2.1.

Наприклад, розглянемо Фактор 5. Відповідно до теплової карти, він має лише дві жовті клітинки під номерами 14 та 15, що відповідає значенням факторних навантажень близьким до 1 за модулем, у відношенні до параметрів min_idle та max_idle. В свою чергу, всі інші клітинки темно-сині, тобто факторні навантаження на дані параметри близькі до 0.

3.3 Візуалізація факторної моделі

Для кращого сприйняття отриманих результатів було вирішено представити факторну матрицю спостережень у вигляді графіків. Для осей координат було відібрано 3 фактори, які пояснюють найбільші відсотки дисперсії та введено позначення для них FA1, FA2, FA3:

- FA1 – Фактор 4
- FA2 – Фактор 3
- FA3 – Фактор 2

Загалом на кожному з графіків зображено 123 469 спостережень, майже всі з яких відносяться до атаки типу DDoS SYN Flood.

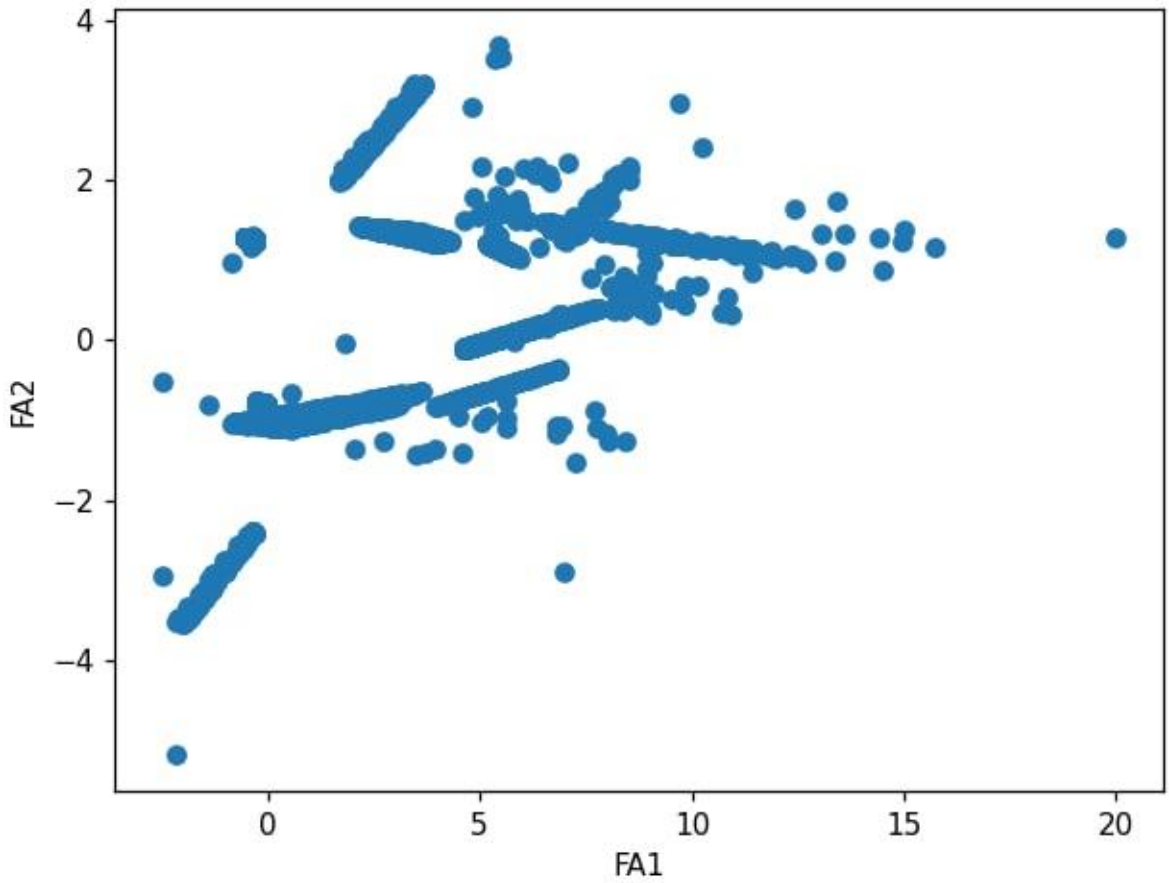

Рисунок 3.4 – Візуалізація значень FA1 та FA2

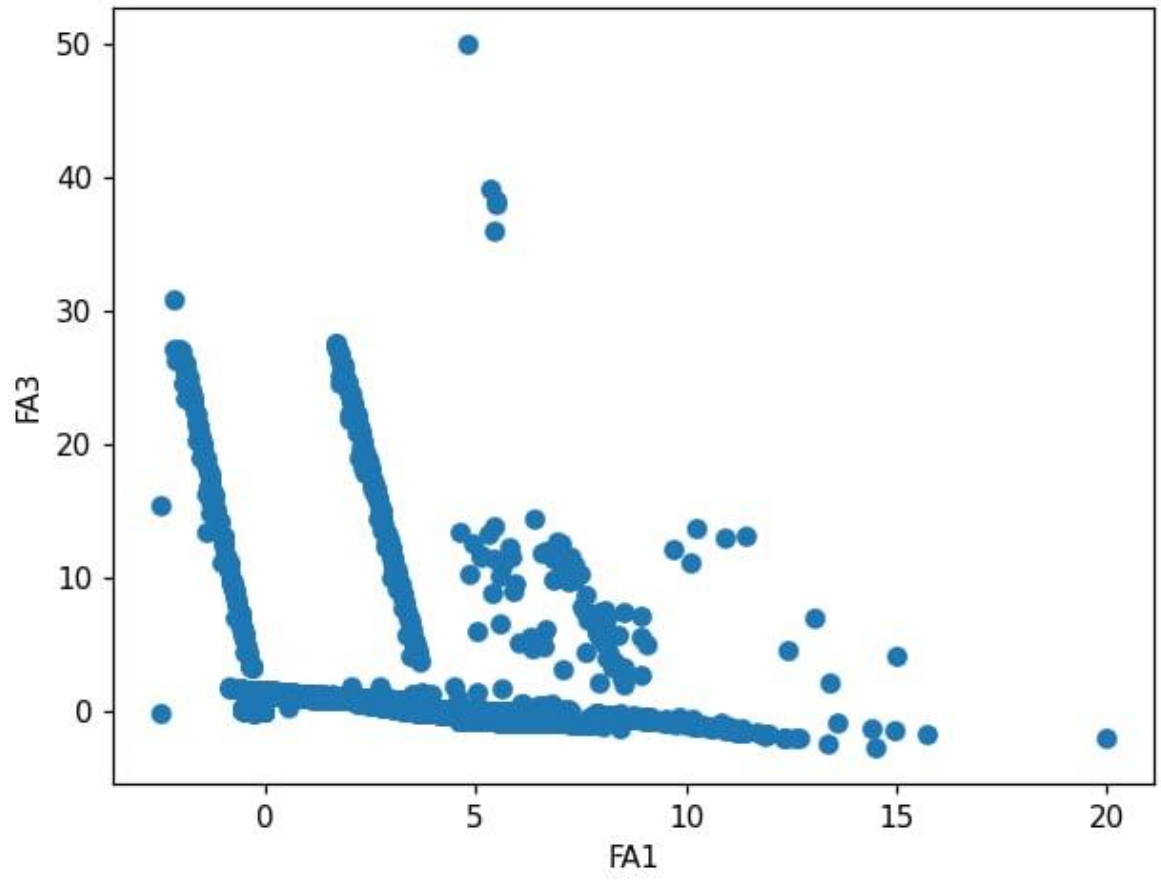

Рисунок 3.5 – Візуалізація значень FA1 та FA3

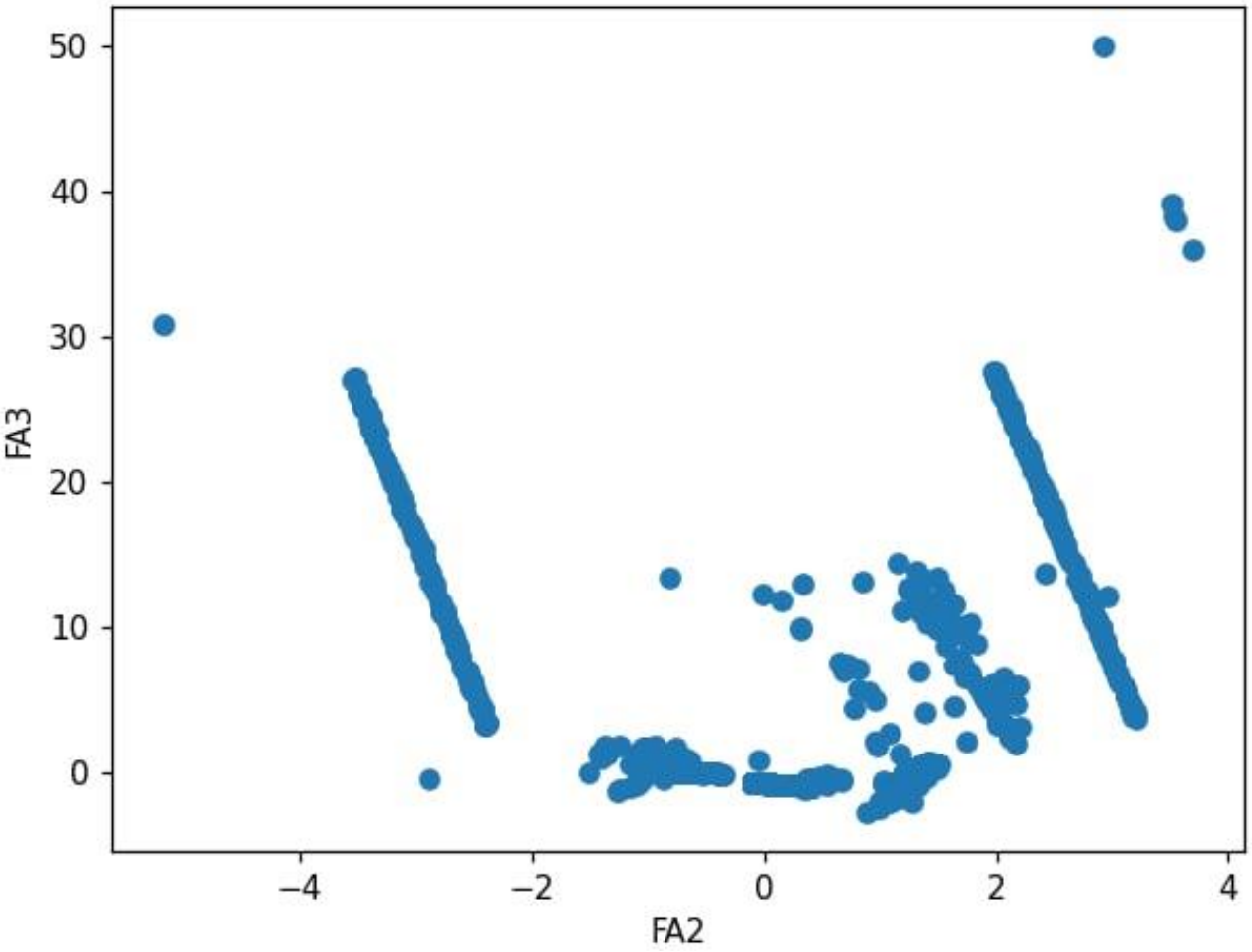

Рисунок 3.6 – Візуалізація значень FA2 та FA3

На рисунках 3.4–3.6 продемонстровані попарні візуальні представлення факторів (FA1, FA2), (FA1, FA3) та (FA2, FA3). На двох останніх графіках чітко відслідковуються два стовпчика з точок, саме в них зосереджена більша частина спостережень. Значна відстань між ними обумовлена наявністю зворотного потоку лише в половині зв’язках, тобто майже на половину з них не прийшла відповідь.

Для більш наглядної демонстрації мережевого трафіку було вирішено провести 3D візуалізацію трьох основних факторів. При цьому синім кольором були відмічені зв’язки на які не прийшла відповідь, а червоним, в свою чергу, зв’язки з наявним зворотним потоком. Отримані результати візуалізації продемонстровані на рисунку 3.7.

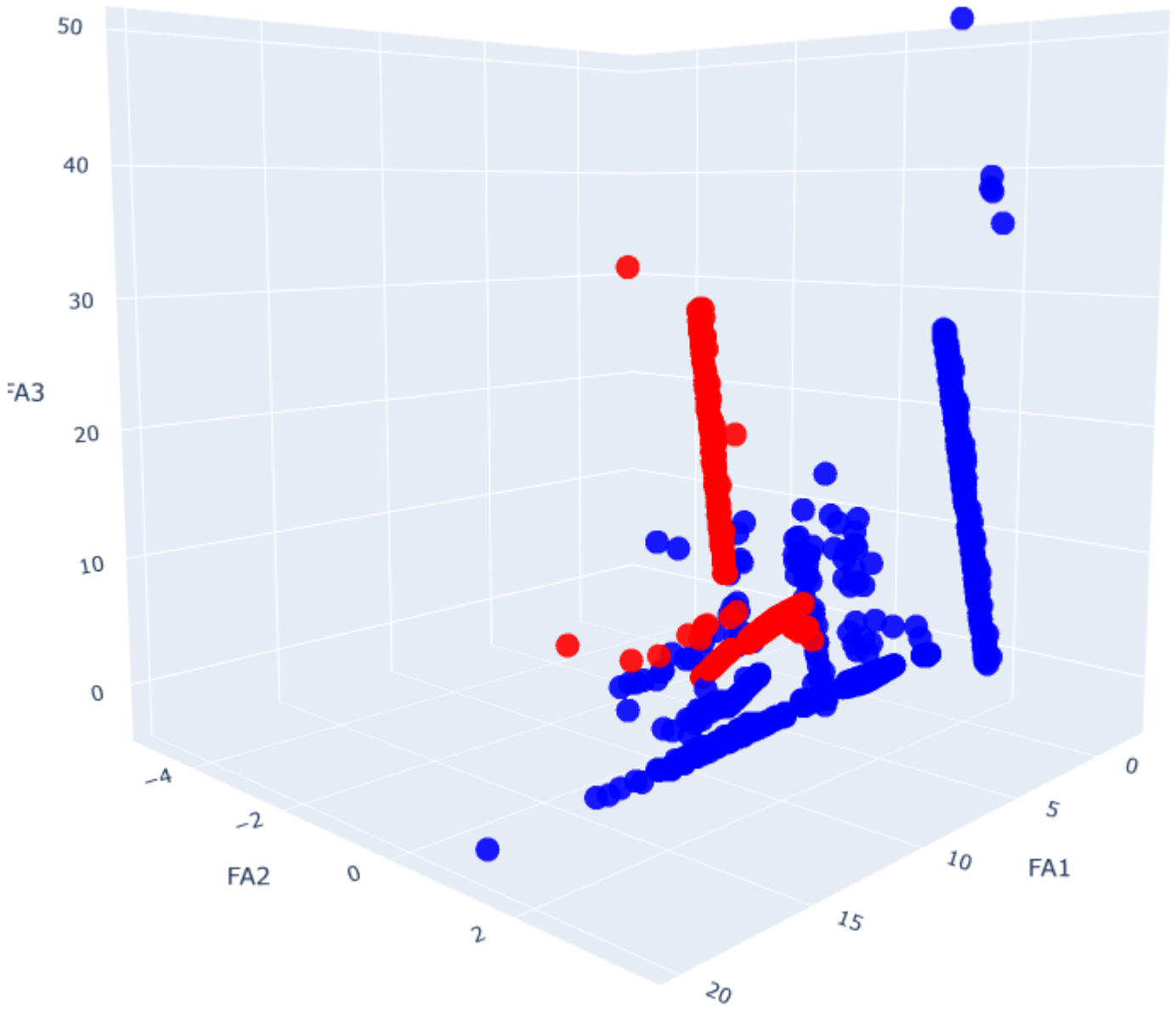

Рисунок 3.7 – Візуалізація значень FA1, FA2 та FA3

Аналізуючи графік зображений на рисунку 3.7 одразу можна помітити, що саме значення FA2, тобто Фактора 3, відповідають за наявність зворотних потоків в зв’язках.

Після проведення візуалізації, була виконана спроба інтерпретації факторів. Для цього, відповідно до матриці факторних навантажень, до уваги бралися зв’язки між обрахованими факторами та початковими параметрами зі значеннями більше за 0.5 по модулю. На основі відібраних зв’язків було

згруповано параметри в межах кожного фактору, та описано їх відповідно до таблиці 2.1. Отримані результати представленні у таблиці 3.1.

Таблиця 3.1 – Зв’язки обрахованих факторів з початковими параметрами

Обрахований фактор	Група відповідних параметрів
Фактор 1	Мінімальна та максимальна довжина пакетів в прямому напрямку.
Фактор 2	Мінімальний та максимальний час у підпотоках та середня кількість пакетів/байтів у прямому напрямку в підпотоці.
Фактор 3	Кількість пакетів в зворотному напрямку, довжина пакета в зворотному напрямку, середня кількість пакетів/байтів в зворотному напрямку в підпотоці, загальна довжина заголовку в зворотному напрямку.
Фактор 4	Кількість пакетів в прямому напрямку, максимальний інтервал між двома пакетами у прямому напрямку, час передачі, загальна довжина заголовку у прямому напрямку.
Фактор 5	Мінімальний час та максимальний час затримки потоку.

Беручи до уваги результати отримані в ході проведення візуалізації моделі та таблицю зв’язків факторів з початковими параметрами, було висунуто припущення стосовно природи трьох основник факторів (таблиця 3.2).

Таблиця 3.2 – Інтерпретація основних факторів

Обрахований фактор	Назва фактору
Фактор 3	Активність у зворотному напрямку
Фактор 4	Об'ємність трафіку у прямому напрямку
Фактор 2	Інтенсивність трафіку у прямому напрямку

3.4 Процес кластеризації

Відбір наборів даних

Для проведення кластеризації трафіку та виявлення наявних в ньому аномалій застосовувався тестовий набір даних, який налічує 35 000 пакетів мережевого трафіку, 5000 з яких відносяться до атаки типу DDoS SYN flood, а решта 30 000 до іншого трафіку.

У дослідженні застосовувалась факторна модель, сформована на одному з навчальних наборів даних, відібраних після проведення перехресної перевірки. В наслідок застосування даної моделі до обраного тестового набору даних була отримана матриця спостережень

Проведення кластеризації

Для аналізу тестового набору даних та пошуку присутнього в ньому фрагменту атаки типу DDoS SYN Flood, було вирішено застосовувати візуальний метод кластеризації.

Беручи до уваги те, що зв’язки DDoS досить щільно розміщуються на графіку та тісно пов’язані між собою, критерієм розділення спостережень на кластери було обрана відстань між сусідніми точками в координатах (FA1, FA2, FA3).

Якщо відстань між двома сусідніми точками не перевищувала значення 0.1, то вони заносились до кластеру DDoS та позначалися червоним кольором. В свою чергу, всі інші спостереження відносилися до іншого трафіку та позначалися синім кольором. Результати проведеної кластеризації зображені на рисунку 3.8

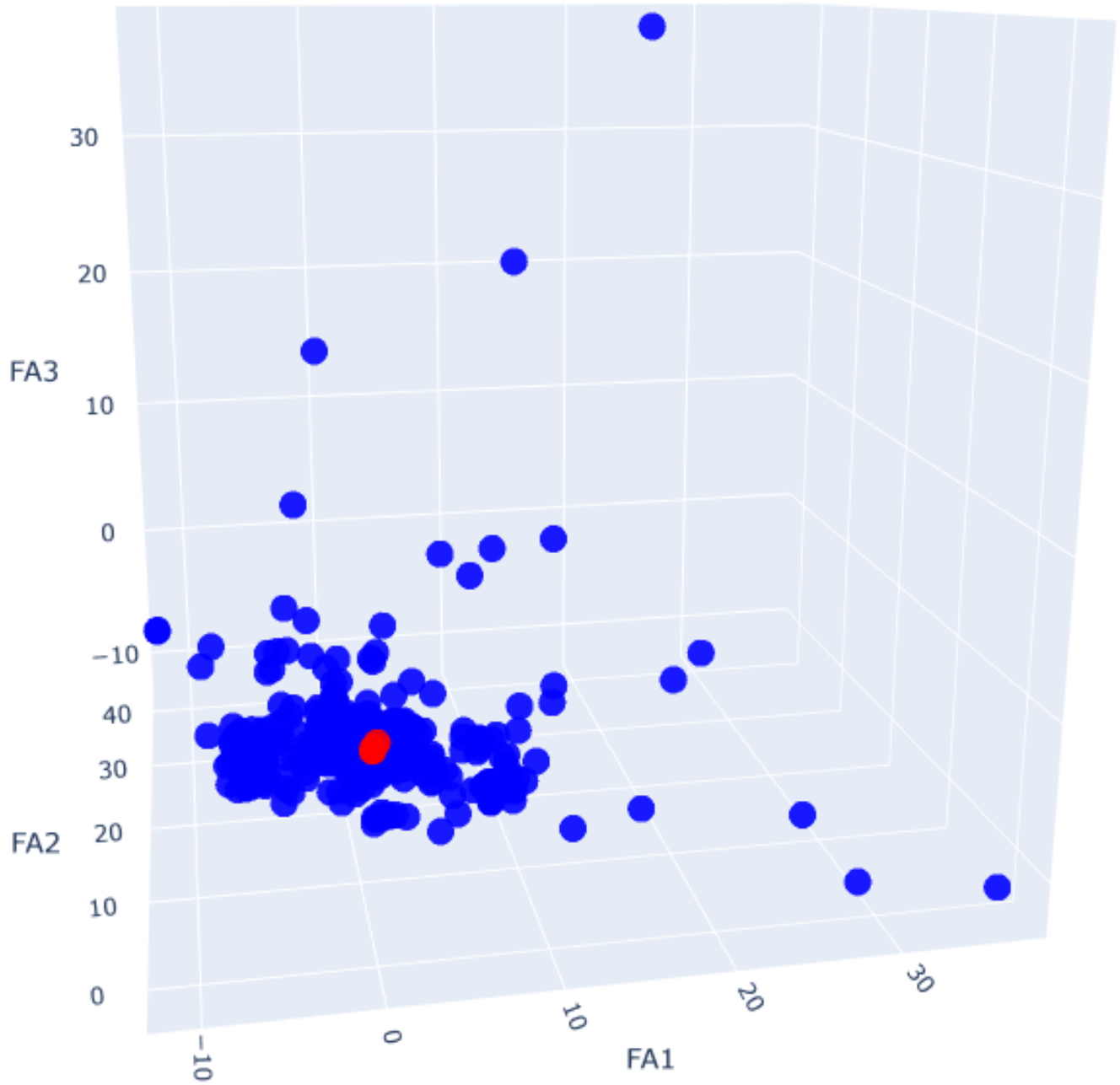

Рисунок 3.8 – Візуальне представлення кластеризації

Необхідно зауважити, що незважаючи на досить малу частину червоних точок на графіку, зображеному на рисунку 3.8, основна частина зв’язків знаходиться саме в них. Серед 2235 спостережень до кластеру атаки DDoS SYN Flood було віднесено 1803 точок, а до кластеру іншого трафіку – 432.

Для кращої демонстрації кластеру DDoS було проведено масштабування частини графіку, яка містить дані спостереження. Результати масштабування зображені на рисунку 3.9.

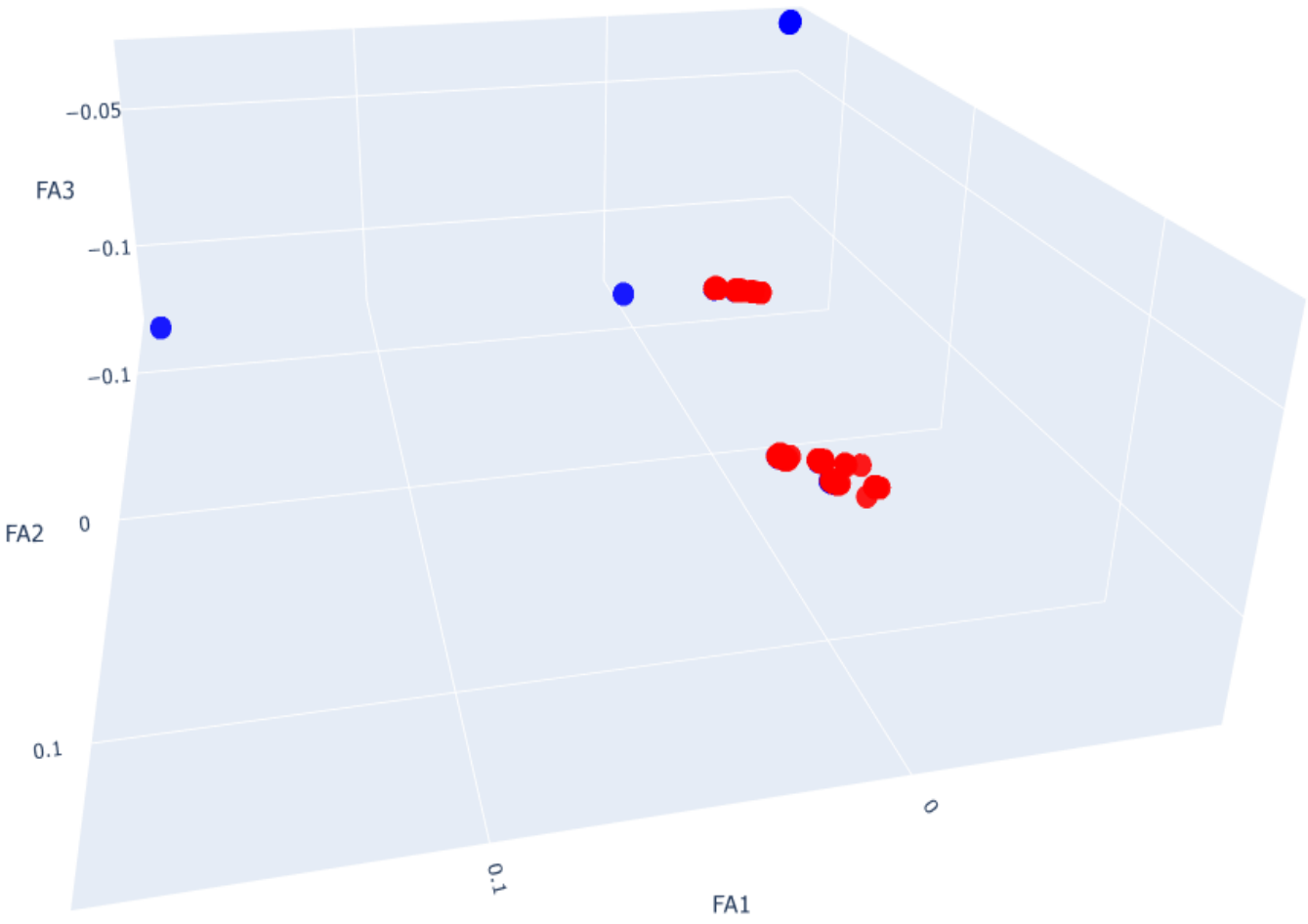

Рисунок 3.9 – Візуальне представлення кластеризації у масштабі

Проте отриманий графік все ще не надає чіткого розуміння стосовно розподілу спостережень. Зважаючи на високу щільність точок у просторі, було

вирішено використовувати теплові карти та додатковий параметр для виміру щільності точок у просторі.

Даний параметр оцінює кількість точок що знаходяться в безпосередній близькості до інших, та в залежності від отриманого значення присвоює відповідний розмір спостереженню. Результати побудови представлено на рисунку 3.10.

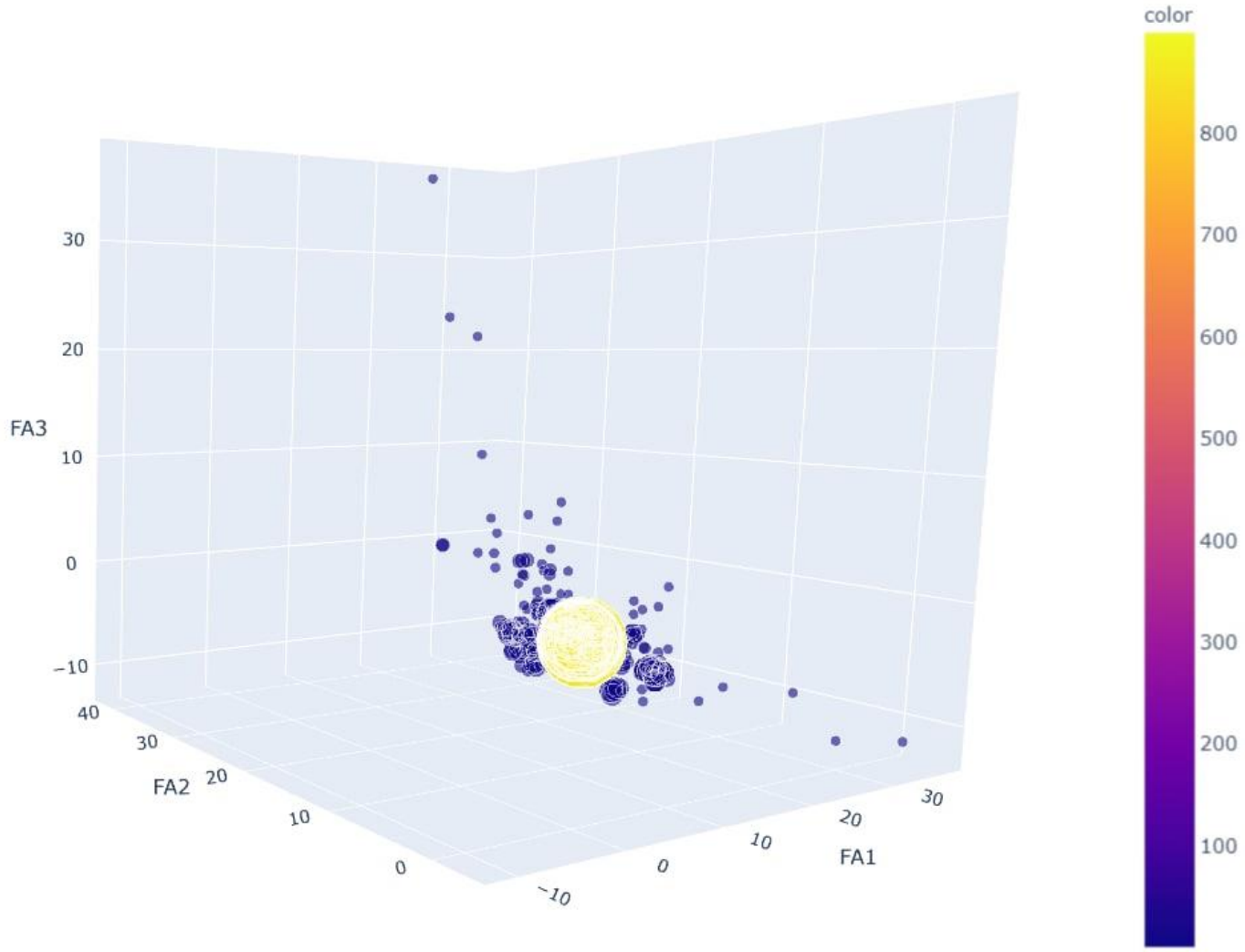

Рисунок 3.10 – Візуалізація кластеризації з використанням теплової карти

За результатами побудови, було проведено розбиття множини точок на два кластера, так як для спостережень DDoS в даній моделі характерна висока

щільність, то групи точок близькі до жовтого кольору були віднесені до кластеру атаки типу DDoS SYN Flood, а решта відповідно до іншого трафіку.

Також, для кращого сприйняття було проведено масштабування графіка зображеного на рисунку 3.10. Результати масштабування представлені на рисунку 3.11

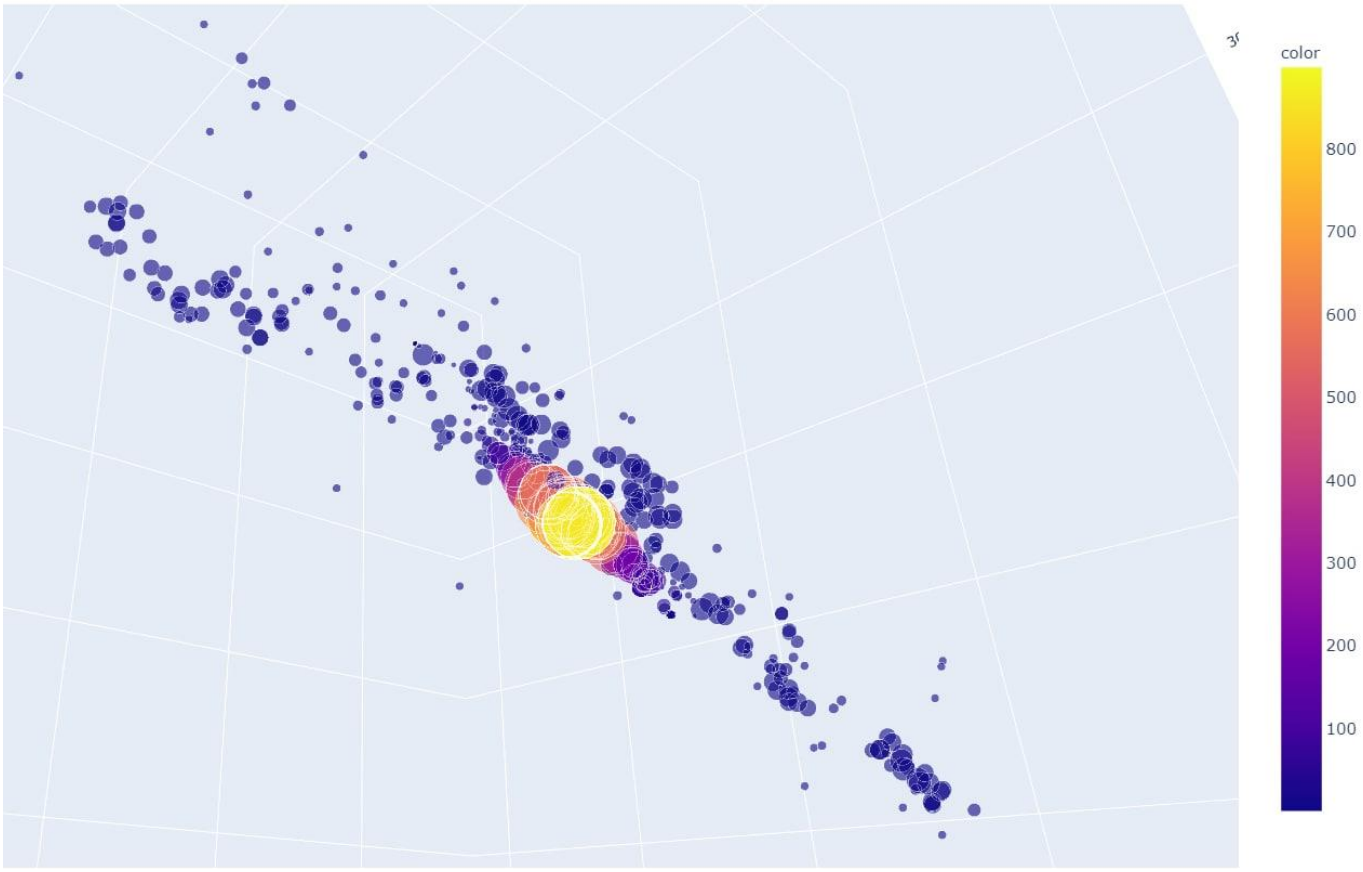

Рисунок 3.11 – Візуалізація кластеризації з використанням теплової карти при масштабуванні

Висновки до розділу 3

Проведено візуалізацію матриці факторних навантажень та моделі в цілому, із застосуванням графічних бібліотек мов програмування Python та R. На основі результатів проведеної візуалізації та значень матриці факторних навантажень було виконано опис структури взаємозв’язків між початковими параметрами.

Виконано перевірку ефективності моделі на змішаному наборі даних, який містив 30 000 пакетів звичайного трафіку та 5 000 пакетів атаки типу DDoS SYN Flood. Методом перевірки слугувало проведення кластеризації на множині даних, отриманих після застосування факторної моделі. За результатами проведеної кластеризації було визначено, що унікальними особливостями для атаки класу DDoS SYN Flood є висока щільність спостережень та характерна відстань між зв'язками у просторі (< 0,1).

ВИСНОВКИ

У ході проведеного дослідження було виконано наступне:

1. Розглянуто основну теоретичну інформацію стосовно природи походження аномалій в мережевому трафіку та необхідних дій щодо їх виявлення.
2. На основі лабораторних зразків, перехопленого в ході атаки типу DDoS SYN Flood мережевого трафіку, сформовано набір даних та описано його за допомогою 21-го початкового параметра..
3. Використовуючи методи дослідницького факторного аналізу розраховано значення п’яти прихованих факторів, на основі яких сформовано факторну модель мережевого трафіку. За проведеними підрахунками утворена модель пояснює близько 82% загальної дисперсії та належно описує зв’язки між 19-тьма початковими параметрами.
4. Проведено оцінку якості моделювання, в умовах k-блочної перехресної перевірки, за допомогою аналізу залишкових дисперсій в навчальних та тестових наборах даних. За результатами перевірки виділено три набори даних з найкращими показниками оцінки якості в середньому: C_2, C_4, C_8.
5. Додатково, було проаналізовано якість моделювання для факторної моделі, побудованої на чотирьох прихованих факторах. За результатами проведеного аналізу, отримані показники поясненої дисперсії залишків коливалися в межах 70% від загальної дисперсії, що свідчить про значні втрати інформативності.
6. Виконано перевірку ефективності моделі на наборі даних зі змішаним мережевим трафіком, де завдяки кластеризації вдалося виявити наявну атаку типу DDoS SYN flood. За результатами проведеної кластеризації було визначено, що унікальними

особливостями для атаки класу DDoS SYN Flood є висока щільність спостережень та характерна відстань між зв'язками у просторі (< 0,1)

Таким чином, факторний аналіз є дієвим методом для скорочення кількості змінних, необхідних для опису поведінки мережевого трафіку, та виявленню прихованих взаємозв’язків між ними, що забезпечує сприятливі умови для формування простих та ефективних механізмів виявлення підозрілої мережевої активності.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1) Деркач В.В. Факторний аналіз системи показників для аналізу PCAP-файлів в умовах перехресної перевірки [Текст] / Деркач В.В., Стремецька М.С. // XIX Всеукраїнська науково-практична конференція студентів, аспірантів та молодих вчених «Теоретичні і прикладні проблеми фізики, математики та інформатики», 13−14 травня 2021 р., м. Київ. – 2021. – С. 208-211.

2) Chandola V. Anomaly Detection: A Survey / V. Chandola, A. Banerjee, V. Kumar. [Текст] // ACM Computing Surveys 41(3). – 2009. – 72 p.

3) Wu Z. A taxonomy of network and computer attacks based on responses. In: Proceedings of the International Conference on Information Technology, Computer Engineering and Management Sciences [Текст] / Z. Wu, Y. Ou, Y. Liu. // Computer Society, Nanjing, Jiangsu. – 2011. – С. 26–29.

4) Thottan M. Anomaly detection in IP networks [Текст] / M. Thottan, C. Ji. // IEEE Transactions on Signal Processing. – 2003. – С. 2191 – 2204.

5) KDDcup99: Knowledge Discovery in Databases DARPA Archive. [Електронний ресурс]. – 1999. – Режим доступу до ресурсу: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

6) UNIBS: University of Brescia Dataset [Електронний ресурс] // 2009 – Режим доступу до ресурсу: http://www.ing.unibs.it/ntw/tools/traces/

7) DDoS Evaluation Dataset (CIC-DDoS2019) [Електронний ресурс] / UNB, 2019. – Режим доступу до ресурсу: https://www.unb.ca/cic/datasets/ddos-2019.html

8) "Developing Realistic Distributed Denial of Service (DDoS) Attack Dataset and Taxonomy" [Текст] / I. Sharafaldin, A. Habibi Lashkari, S. Hakak, A. A. Ghorbani, // IEEE 53rd International Carnahan Conference on Security Technology. – Chennai, India. – 2019. – 8 p

9) Demertzis, The Next Generation Cognitive Security Operations Center: Adaptive Analytic Lambda Architecture for Efficient Defense against Adversarial Attacks. [Текст] / Demertzis, Tziritas K., Kikiras N., Sanchez P., Iliadis S., Lazaros. // Big Data and Cognitive Computing, MDPI. – 2019. – 21 p.

10) Hartmann, K. Statistics and Geospatial Data Analysis [Електронний ресурс] / Hartmann, K., Krois, J., Waske, B. // Freie Universität Berlin. – 2018. – Режим доступу до ресурсу: https://www.geo.fu-berlin.de/en/v/soga/Geodata-analysis/factor-analysis/index.html

11) Алексей Померанцев, Калибровка (Градуировка) [Електронний ресурс] / Алексей Померанцев – Режим доступу до ресурсу: https://rcs.chemometrics.ru/old/Tutorials/calibration.htm

12) Unpingco J. Python for Probability, Statistics, and Machine Learning [Текст] / Jos Unpingco // Springer Publishing Company, 2016. – pp 215-218.

Додаток А Початкова матриця спостережень

2	0	60	60	0	0	1	1	0	0	1	1	1	500000	9402849	2.0	120.0	0.0	0.0	80	0
2	2	60	60	60	60	1	1	0	0	51	51	51	500000	9402797	2.0	120.0	2.0	120.0	80	80
2	0	60	60	0	0	1	1	0	0	0	1	1	500000	9402798	2.0	120.0	0.0	0.0	80	0
2	0	60	60	0	0	1	1	0	0	0	1	1	500000	9402721	2.0	120.0	0.0	0.0	80	0
4	2	60	60	60	60	1	4989979	1	1	4989980	1	95	500000	4989885	2.0	120.0	1.0	60.0	160	80
2	2	60	60	60	60	1	1	1	1	108	108	108	500000	9402578	2.0	120.0	2.0	120.0	80	80
2	0	60	60	0	0	1	1	0	0	0	1	1	500000	9402624	2.0	120.0	0.0	0.0	80	0
2	0	60	60	0	0	1	1	0	0	1	1	1	500000	9402622	2.0	120.0	0.0	0.0	80	0
2	0	60	60	0	0	0	0	0	0	0	0	0	500000	9402577	2.0	120.0	0.0	0.0	80	0
2	0	60	60	0	0	0	0	0	0	0	0	0	500000	9402506	2.0	120.0	0.0	0.0	80	0
2	0	60	60	0	0	1	1	0	0	0	1	1	500000	9402458	2.0	120.0	0.0	0.0	80	0
2	2	60	60	60	60	1	1	1	1	70	71	71	500000	9402386	2.0	120.0	2.0	120.0	80	80
2	0	60	60	0	0	0	0	0	0	0	0	0	500000	9402388	2.0	120.0	0.0	0.0	80	0
2	2	60	60	60	60	47	47	0	0	113	114	114	500000	9402271	2.0	120.0	2.0	120.0	80	80
2	0	60	60	0	0	0	0	0	0	0	0	0	500000	9402337	2.0	120.0	0.0	0.0	80	0
2	2	60	60	60	60	1	1	1	1	73	74	74	500000	9402199	2.0	120.0	2.0	120.0	80	80
2	2	60	60	60	60	2	2	0	0	55	55	55	500000	9402145	2.0	120.0	2.0	120.0	80	80
2	2	60	60	60	60	0	0	1	1	100	101	101	500000	9401806	2.0	120.0	2.0	120.0	80	80
4	4	60	60	60	60	0	8324958	1	8325001	8325106	100	103	500000	8324903	2.0	120.0	2.0	120.0	160	160
2	0	60	60	0	0	1	1	0	0	1	1	1	500000	9401856	2.0	120.0	0.0	0.0	80	0
2	0	60	60	0	0	1	1	0	0	1	1	1	500000	9401804	2.0	120.0	0.0	0.0	80	0
2	0	60	60	0	0	45	45	0	0	44	45	45	500000	9401757	2.0	120.0	0.0	0.0	80	0
2	0	60	60	0	0	50	50	0	0	49	50	50	500000	9401707	2.0	120.0	0.0	0.0	80	0
2	0	60	60	0	0	0	0	0	0	0	0	0	500000	9401706	2.0	120.0	0.0	0.0	80	0
2	0	60	60	0	0	0	0	0	0	0	0	0	500000	9401640	2.0	120.0	0.0	0.0	80	0
2	0	60	60	0	0	48	48	0	0	47	48	48	500000	9401591	2.0	120.0	0.0	0.0	80	0
2	2	60	60	60	60	1	1	1	1	66	66	66	500000	9401525	2.0	120.0	2.0	120.0	80	80

Додаток Б Стандартизована матриця

-0.204	1.076	-0.004	-0.004	1.234	1.234	2.2335	-0.185	-0.22	-0.124	-0.185	-0.039	-0.047	-1.386	1.769	-0.058	-0.058	1.2467	1.247	-0.204	1.076
-0.204	-0.78	-0.004	-0.004	-0.811	-0.81	-0.47	-0.185	-0.22	-0.124	-0.185	-0.047	-0.054	-1.386	1.769	-0.058	-0.058	-0.802	-0.802	-0.204	-0.775
-0.204	1.076	-0.004	-0.004	1.234	1.234	-0.413	-0.185	-0.093	-0.124	-0.185	-0.042	-0.049	-1.386	1.769	-0.058	-0.058	1.2467	1.247	-0.204	1.076
-0.204	1.076	-0.004	-0.004	1.234	1.234	-0.355	-0.185	-0.22	-0.124	-0.185	-0.043	-0.05	-1.386	1.769	-0.058	-0.058	1.2467	1.247	-0.204	1.076
-0.204	1.076	-0.004	-0.004	1.234	1.234	-0.47	-0.185	-0.093	-0.124	-0.185	-0.04	-0.048	-1.386	1.768	-0.058	-0.058	1.2467	1.247	-0.204	1.076
4.4116	2.927	-0.004	-0.004	1.234	1.234	-0.47	7.8606	-0.093	12.15	7.787	-0.04	-0.048	-1.386	0.983	-0.058	-0.058	1.2467	1.247	4.4084	2.927
-0.204	-0.78	-0.004	-0.004	-0.811	-0.81	-0.413	-0.185	-0.22	-0.124	-0.185	-0.047	-0.053	-1.386	1.769	-0.058	-0.058	-0.802	-0.802	-0.204	-0.775
-0.204	-0.78	-0.004	-0.004	-0.811	-0.81	-0.413	-0.185	-0.22	-0.124	-0.185	-0.047	-0.053	-1.386	1.768	-0.058	-0.058	-0.802	-0.802	-0.204	-0.775
-0.204	-0.78	-0.004	-0.004	-0.811	-0.81	2.1185	-0.185	-0.22	-0.124	-0.185	-0.044	-0.051	-1.386	1.768	-0.058	-0.058	-0.802	-0.802	-0.204	-0.775
-0.204	-0.78	-0.004	-0.004	-0.811	-0.81	2.4061	-0.185	-0.22	-0.124	-0.185	-0.043	-0.051	-1.386	1.768	-0.058	-0.058	-0.802	-0.802	-0.204	-0.775
-0.204	-0.78	-0.004	-0.004	-0.811	-0.81	-0.47	-0.185	-0.22	-0.124	-0.185	-0.047	-0.054	-1.386	1.768	-0.058	-0.058	-0.802	-0.802	-0.204	-0.775
-0.204	-0.78	-0.004	-0.004	-0.811	-0.81	-0.47	-0.185	-0.22	-0.124	-0.185	-0.047	-0.054	-1.386	1.768	-0.058	-0.058	-0.802	-0.802	-0.204	-0.775
-0.204	-0.78	-0.004	-0.004	-0.811	-0.81	2.2911	-0.185	-0.22	-0.124	-0.185	-0.044	-0.051	-1.386	1.768	-0.058	-0.058	-0.802	-0.802	-0.204	-0.775
-0.204	1.076	-0.004	-0.004	1.234	1.234	-0.413	-0.185	-0.093	-0.124	-0.185	-0.042	-0.05	-1.386	1.768	-0.058	-0.058	1.2467	1.247	-0.204	1.076
-0.204	1.076	-0.004	-0.004	1.234	1.234	-0.413	-0.185	-0.093	-0.124	-0.185	-0.043	-0.051	-1.386	1.768	-0.058	-0.058	1.2467	1.247	-0.204	1.076
4.4116	1.076	-0.004	-0.004	1.234	1.234	-0.413	5.9435	-0.093	-0.124	5.887	-0.047	-0.046	-1.386	-0.46	-0.058	-0.058	0.2225	0.223	4.4084	1.076
-0.204	1.076	-0.004	-0.004	1.234	1.234	-0.413	-0.185	-0.093	-0.124	-0.185	-0.04	-0.047	-1.386	1.768	-0.058	-0.058	1.2467	1.247	-0.204	1.076
-0.204	-0.78	-0.004	-0.004	-0.811	-0.81	-0.413	-0.185	-0.22	-0.124	-0.185	-0.047	-0.053	-1.386	1.768	-0.058	-0.058	-0.802	-0.802	-0.204	-0.775
4.4116	1.076	-0.004	-0.004	1.234	1.234	-0.413	7.6936	-0.093	-0.124	7.621	-0.047	-0.047	-1.386	0.857	-0.058	-0.058	0.2225	0.223	4.4084	1.076
-0.204	-0.78	-0.004	-0.004	-0.811	-0.81	-0.355	-0.185	-0.22	-0.124	-0.185	-0.047	-0.053	-1.386	1.768	-0.058	-0.058	-0.802	-0.802	-0.204	-0.775
-0.204	1.076	-0.004	-0.004	1.234	1.234	2.0034	-0.185	-0.22	-0.124	-0.185	-0.04	-0.048	-1.386	1.768	-0.058	-0.058	1.2467	1.247	-0.204	1.076
-0.204	-0.78	-0.004	-0.004	-0.811	-0.81	-0.413	-0.185	-0.22	-0.124	-0.185	-0.047	-0.053	-1.386	1.768	-0.058	-0.058	-0.802	-0.802	-0.204	-0.775
-0.204	-0.78	-0.004	-0.004	-0.811	-0.81	2.2335	-0.185	-0.22	-0.124	-0.185	-0.044	-0.051	-1.386	1.768	-0.058	-0.058	-0.802	-0.802	-0.204	-0.775
-0.204	-0.78	-0.004	-0.004	-0.811	-0.81	-0.413	-0.185	-0.22	-0.124	-0.185	-0.047	-0.053	-1.386	1.768	-0.058	-0.058	-0.802	-0.802	-0.204	-0.775
-0.204	1.076	-0.004	-0.004	1.234	1.234	-0.413	-0.185	-0.22	-0.124	-0.185	-0.043	-0.051	-1.386	1.768	-0.058	-0.058	1.2467	1.247	-0.204	1.076
-0.204	1.076	-0.004	-0.004	1.234	1.234	-0.413	-0.185	-0.22	-0.124	-0.185	-0.039	-0.047	-1.386	1.768	-0.058	-0.058	1.2467	1.247	-0.204	1.076
-0.204	-0.78	-0.004	-0.004	-0.811	-0.81	-0.413	-0.185	-0.22	-0.124	-0.185	-0.047	-0.053	-1.386	1.767	-0.058	-0.058	-0.802	-0.802	-0.204	-0.775
9.0269	4.778	-0.004	-0.004	1.234	1.234	-0.47	5.1311	-0.22	7.9854	7.273	-0.04	-0.047	-0.046	-1.08	-0.058	-0.058	1.2467	1.247	9.0204	4.778
-0.204	-0.78	-0.004	-0.004	-0.811	-0.81	-0.413	-0.185	-0.22	-0.124	-0.185	-0.047	-0.053	-1.386	1.767	-0.058	-0.058	-0.802	-0.802	-0.204	-0.775

Додаток В Кореляційна матриця

1.0	0.371	0.018	0.018	0.169	0.169	-0.074	0.844	-0.001	0.603	0.855	0.223	0.295	-0.178	-0.146	0.311	0.311	0.137	0.137	1.0	0.371
0.371	1.0	-0.004	-0.004	0.958	0.958	-0.022	0.321	0.244	0.372	0.327	0.076	0.111	-0.051	-0.068	0.103	0.103	0.958	0.958	0.371	1.0
0.018	-0.004	1.0	1.0	-0.004	-0.004	0.0	0.007	-0.001	-0.001	0.011	-0.0	-0.0	0.001	-0.005	-0.0	0.017	-0.004	-0.004	0.037	-0.004
0.018	-0.004	1.0	1.0	-0.004	-0.004	0.0	0.007	-0.001	-0.001	0.011	-0.0	-0.0	0.001	-0.005	-0.0	0.017	-0.004	-0.004	0.037	-0.004
0.169	0.958	-0.004	-0.004	1.0	1.0	-0.007	0.18	0.272	0.148	0.18	0.023	0.032	-0.02	-0.038	0.028	0.028	0.99	0.99	0.169	0.958
0.169	0.958	-0.004	-0.004	1.0	1.0	-0.007	0.18	0.272	0.148	0.18	0.023	0.032	-0.02	-0.038	0.028	0.028	0.99	0.99	0.169	0.958
-0.074	-0.022	0.0	0.0	-0.007	-0.007	1.0	-0.064	0.008	-0.043	-0.064	-0.018	-0.021	0.014	0.011	-0.024	-0.024	-0.005	-0.005	-0.074	-0.022
0.844	0.321	0.007	0.007	0.18	0.18	-0.064	1.0	0.011	0.66	0.998	0.008	0.051	-0.202	-0.095	0.033	0.034	0.108	0.108	0.844	0.321
-0.001	0.244	-0.001	-0.001	0.272	0.272	0.008	0.011	1.0	-0.011	0.011	-0.005	-0.006	-0.002	0.002	-0.008	-0.008	0.268	0.268	-0.001	0.244
0.603	0.372	-0.001	-0.001	0.148	0.148	-0.043	0.66	-0.011	1.0	0.663	0.01	0.059	-0.129	-0.076	0.039	0.039	0.154	0.154	0.602	0.372
0.855	0.327	0.011	0.011	0.18	0.18	-0.064	0.998	0.011	0.663	1.0	0.008	0.054	-0.201	-0.097	0.035	0.035	0.109	0.109	0.855	0.327
0.223	0.076	-0.0	-0.0	0.023	0.023	-0.018	0.008	-0.005	0.01	0.008	1.0	0.918	0.001	-0.009	0.838	0.838	0.085	0.085	0.223	0.076
0.295	0.111	-0.0	-0.0	0.032	0.032	-0.021	0.051	-0.006	0.059	0.054	0.918	1.0	-0.008	-0.017	0.847	0.847	0.096	0.096	0.295	0.111
-0.178	-0.051	0.001	0.001	-0.02	-0.02	0.014	-0.202	-0.002	-0.129	-0.201	0.001	-0.008	1.0	-0.927	-0.003	-0.003	-0.004	-0.004	-0.178	-0.051
-0.146	-0.068	-0.005	-0.005	-0.038	-0.038	0.011	-0.095	0.002	-0.076	-0.097	-0.009	-0.017	-0.927	1.0	-0.014	-0.014	-0.035	-0.035	-0.146	-0.068
0.311	0.103	-0.0	-0.0	0.028	0.028	-0.024	0.033	-0.008	0.039	0.035	0.838	0.847	-0.003	-0.014	1.0	1.0	0.099	0.099	0.31	0.103
0.311	0.103	0.017	0.017	0.028	0.028	-0.024	0.034	-0.008	0.039	0.035	0.838	0.847	-0.003	-0.014	1.0	1.0	0.099	0.099	0.311	0.103
0.137	0.958	-0.004	-0.004	0.99	0.99	-0.005	0.108	0.268	0.154	0.109	0.085	0.096	-0.004	-0.035	0.099	0.099	1.0	1.0	0.136	0.958
0.137	0.958	-0.004	-0.004	0.99	0.99	-0.005	0.108	0.268	0.154	0.109	0.085	0.096	-0.004	-0.035	0.099	0.099	1.0	1.0	0.136	0.958
1.0	0.371	0.037	0.037	0.169	0.169	-0.074	0.844	-0.001	0.602	0.855	0.223	0.295	-0.178	-0.146	0.31	0.311	0.136	0.136	1.0	0.371
0.371	1.0	-0.004	-0.004	0.958	0.958	-0.022	0.321	0.244	0.372	0.327	0.076	0.111	-0.051	-0.068	0.103	0.103	0.958	0.958	0.371	1.0

Додаток Г Матриця спостережень в просторі обрахованих факторів

Фактор 1	Фактор 2	Фактор 3	Фактор 4	Фактор 5
-0.00415	-0.01165	0.79466	-0.07675	-1.59405
0.00687	-0.07467	-1.25221	-0.35547	-1.62059
-0.00415	-0.01165	0.79466	-0.07675	-1.59403
-0.00415	-0.01165	0.79466	-0.07675	-1.59401
-0.06224	-0.5507	-0.31854	4.83419	0.14091
0.00687	-0.07467	-1.25221	-0.35547	-1.62052
-0.00415	-0.01165	0.79466	-0.07675	-1.59397
-0.00415	-0.01165	0.79466	-0.07675	-1.59397
-0.00415	-0.01165	0.79466	-0.07675	-1.59395
-0.00415	-0.01165	0.79466	-0.07675	-1.59393
-0.00415	-0.01165	0.79466	-0.07675	-1.59391
0.00687	-0.07467	-1.25221	-0.35547	-1.62045
-0.00415	-0.01165	0.79466	-0.07675	-1.59388
0.00687	-0.07466	-1.2522	-0.35548	-1.6204
-0.00415	-0.01165	0.79466	-0.07675	-1.59387
0.00687	-0.07467	-1.25221	-0.35547	-1.62038
0.00687	-0.07467	-1.2522	-0.35547	-1.62036
0.00687	-0.07467	-1.2522	-0.35547	-1.62024
-0.05494	-0.57942	-1.34802	4.70088	-1.10279
-0.00415	-0.01165	0.79466	-0.07675	-1.59369
-0.00415	-0.01165	0.79466	-0.07675	-1.59367
-0.00415	-0.01165	0.79466	-0.07676	-1.59365
-0.00415	-0.01165	0.79466	-0.07676	-1.59363
-0.00415	-0.01165	0.79466	-0.07675	-1.59363
-0.00415	-0.01165	0.79466	-0.07675	-1.59361

Додаток Е Програмний код

```
%matplotlib
matplotlib.use('nbagg')

df = pd.read_csv('big_data2_result.csv')
a = list(df.loc[df['total_bpackets'] == 0]['Unnamed: 0'].T)
df.drop('Unnamed: 0', axis=1, inplace=True)
df.drop('srcip', axis=1, inplace=True)
df.drop('dstip', axis=1, inplace=True)
df.drop('srcport', axis=1, inplace=True)
df.drop('dstport', axis=1, inplace=True)
df.drop('protocol', axis=1, inplace=True)
df.drop('furg_cnt', axis=1, inplace=True)
df.drop('burg_cnt', axis=1, inplace=True)
df.drop('dscp', axis=1, inplace=True)

df = df.fillna(0)
df_z = (df-df.mean())/(df.std(ddof=1))
df_z.corr().round(decimals=3).to_csv("cor.csv")

ev,_ = np.linalg.eig(df_z.corr())
ev = np.array(["%14.22f" % v for v in ev])
pd.DataFrame(ev, [i for i in range(1,df_z.shape[1] + 1)],
columns = ["Власне число"])

from sklearn.decomposition import FactorAnalysis
fa = FactorAnalysis(n_components=5,rotation = "quartimax")
fa.fit(df_z)
pd.DataFrame(np.round(fa.components_,6),columns=df_z.column
s,index = ['Фактор 1','Фактор 2', 'Фактор 3', 'Фактор 4',
'Фактор5']).T.to_csv("fa_4.csv")

fa.components_1 = fa.components_ ** 2
variance = np.sum(fa.components_1, axis=1)
proportional_variance = variance / fa.components_.shape[1]
cumulative_variance = np.cumsum(proportional_variance,
axis=0)
pd.DataFrame(np.round(np.append(proportional_variance,
np.sum(proportional_variance)), 5), index = ['Фактор
1','Фактор 2', 'Фактор 3','Фактор 4','Фактор 5', 'Загальна
дисперсія'], columns = ['Дисперсія'])
```

```
pd.DataFrame(np.array(["%14.12f" % v for v in
fa.noise_variance_]), df_z.columns, columns = ["Залишкова
дисперсія"]).to_csv("var_5.csv")
scores = pd.DataFrame(fa.transform(df_z), columns=['Factor
1', 'Factor 2', 'Factor 3', 'Factor 4', 'Factor 5'])

x = list(scores_m.values[0])
y = list(scores_m.values[1])
z = list(scores_m.values[2])
d = list(scores_m.values[3])
e = list(scores_m.values[4])

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(x, y, z)
ax.set_xlabel('FA2')
ax.set_ylabel('FA3')
ax.set_zlabel('FA4')

plt.figure()
plt.scatter(x, y)
plt.xlabel('FA1')
plt.ylabel('FA2')

plt.figure()
plt.scatter(x, z)
plt.xlabel('FA1')
plt.ylabel('FA3')

plt.figure()
plt.scatter(y, z)
plt.xlabel('FA2')
plt.ylabel('FA3')
plt.show()

xyz = np.vstack([x,y,z])
clor = [i for i in gaussian_kde(xyz)(xyz)]
clors = [i * 1000 for i in gaussian_kde(xyz)(xyz)]
imax = max(clors)
for i in range(len(clors)):
    if (imax/clors[i]) > 200:
        clors[i] = clors[i] * 20
fig1 = px.scatter_3d(scores_m, x="FA1", y="FA2", z="FA3",
color = clor, size = clors, opacity = 0.6, size_max = 100)
fig1.show()
```

```
fig1.write_html("file.html")

Z=np.abs(fa.components_)
fig, ax = plt.subplots()
c = ax.pcolor(Z)
fig.colorbar(c, ax=ax)
ax.set_yticks(np.arange(1, fa.components_.shape[0] + 1),
minor=False)
ax.set_yticklabels(["Фактор 1", "Фактор 2", "Фактор 3",
"Фактор 4", "Фактор 5"])
ax.set_xticks(np.arange(1, fa.components_.shape[1] + 1),
minor=False)
plt.show()
fig.savefig('Loadings')

D_L = []
D_T = []
TRVC = 0
TRVP = 0
ERVC = 0
ERVP = 0
temp_L = 0
temp_D = 0

for i in range(10):
    df_L = pd.read_csv('big_data%s_result.csv' % i)
    df_L.drop('Unnamed: 0', axis=1, inplace=True)
    df_L.drop('srcip', axis=1, inplace=True)
    df_L.drop('dstip', axis=1, inplace=True)
    df_L.drop('srcport', axis=1, inplace=True)
    df_L.drop('dstport', axis=1, inplace=True)
    df_L.drop('furg_cnt', axis=1, inplace=True)
    df_L.drop('burg_cnt', axis=1, inplace=True)
    df_L.drop('fpsh_cnt', axis=1, inplace=True)
    df_L.drop('bpsh_cnt', axis=1, inplace=True)


    df_L = df_L.fillna(0)
    df_L = (df_L-df_L.mean())/(df_L.std(ddof=1))
    df_L = df_L.fillna(0)

    fa = FactorAnalysis(n_components=4,rotation =
'quartimax', random_state=0)
    fa.fit(df_L)
    scores = fa.transform(df_L)
```

```
    scores_ = np.dot(scores, fa.components_)

    error_L = df_L - scores_
    TRVC = np.sum(np.power(error_L, 2))
    TRVC = np.sum(TRVC)
    ERVC = 1 - (TRVC / np.sum(np.sum(np.power(df_L, 2))))
    TRVC /= (df_L.shape[0]*df_L.shape[1])



    df_T = pd.read_csv('data%s_result.csv' % i)
    df_T.drop('Unnamed: 0', axis=1, inplace=True)
    df_T.drop('srcip', axis=1, inplace=True)
    df_T.drop('dstip', axis=1, inplace=True)
    df_T.drop('srcport', axis=1, inplace=True)
    df_T.drop('dstport', axis=1, inplace=True)
    df_T.drop('furg_cnt', axis=1, inplace=True)
    df_T.drop('burg_cnt', axis=1, inplace=True)
    df_T.drop('fpsh_cnt', axis=1, inplace=True)
    df_T.drop('bpsh_cnt', axis=1, inplace=True)

    df_T = df_T.fillna(0)
    df_T = (df_T-df_T.mean())/(df_T.std(ddof=1))
    df_T = df_T.fillna(0)

    scores_T = fa.transform(df_T)
    scores_T_ = np.dot(scores_T, fa.components_)
    error_T = df_T - scores_T_
    TRVP = np.sum(np.power(error_T, 2))
    TRVP = np.sum(TRVP)
    ERVP = 1 - (TRVP / np.sum(np.sum(np.power(df_T, 2))))
    TRVP /= (df_T.shape[0]*df_T.shape[1])
    D_L.append([TRVC, ERVC, TRVP, ERVP])

 pd.DataFrame(np.round(np.array(D_L).T, 5), columns =
['data%s' % i for i in range(1, 11)], index = ['TRVC',
'ERVC', 'TRVP', 'ERVP']).to_csv("FA5_p.csv")
```