ВСТУП

Актуальність роботи. Велика кількість комерційних та державних підприємств становляться жертвами зловмисників з точки зору безпеки інформації. І як результат – дані компанії втрачають як і фінансові ресурси так і моральні, наприклад, довіру користувачів. Задля забезпечення безпеки взаємодії інформації в системі було розроблено велику кількість різних засобів захисту. На жаль, більшість засобів сфокусовано на виконання однієї задачі, наприклад пошук шкідливого ПЗ в системі. Дана конкретизація з точки зору захисту інформації являється дуже незручною через купу факторів:

- Наявність конкретизованих ПЗ може привести до зайвого технічного навантаження. Дане навантаження може негативно вплинути на систему у результаті реалізації атаки, які спрямовані на відмову в обслуговуванні, блокуванні циркуляції інформації в системі тощо;
- Конфлікти ПЗ. У результаті наявності великої кількості різного ПЗ, яке забезпечує захист інформації в системі можуть виникати конфлікти, якщо дані працюють одночасно. Конфлікти можуть привести до втрати ефективності одного із засобів захисту або блокування його роботи. Як наслідок – це може привести до реалізації вразливості в системі.
- Фінансовий аспект. Більшість компаній намагається підпорядковувати власні системи захисту згідно міжнародних або державних стандартів. Підтримка систем захисту потребує великої кількості фінансових ресурсів. У результаті вузької спрямованості кожного із механізмів захисту, фінансові ресурси можуть витрачатися не раціонально.

Найбільш критичними для компаній вважаються атаки, які блокують середовище інформаційної системи і які важко або неможливо відстежити. Дані

атаки блокують середовище взаємодії інформації і як наслідок – компанія витрачає велику кількість ресурсів на відновлення роботи системи.

Компанії, які займаються питаннями інформаційної безпеки намагаються створити універсальний ресурс, який зможе ліквідувати, або мінімізувати наслідки реалізації зовнішніх або внутрішніх атак.

Сучасний стан проблеми. У результаті розвитку технологій виникають і негативні сторони. До таких негативних пунктів можна віднести відсутність часу для налаштування рівня захисту інформації до рівня розвитку технологій. Тобто, сучасні механізми безпеки не мають достатнього рівня розвитку щоб зафіксувати всі недоліки в системі, де циркулює інформація. Не виключним фактором необхідно вважати специфічні атаки, які спрямовані не на викрадення інформації, а на блокування систем.

На сьогоднішній день атаки відмови в обслуговуванні являються актуальною проблемою для великих підприємств, таких як SoftServe, GitHub, Google, Amazon тощо, оскільки дані атаки являються пріоритетними для зловмисників через множину факторів;

- Анонімність. Даний тип атаки реалізується шляхом нескінченного відправлення запитів ботнет-мережі до конкретного мережевого ресурсу і даний тип атаки важко або неможливо відстежити;
- Непередбачуваність. Дані атаки можуть проходити незалежно від часу реалізації атаки, кількості запитів або витривалості ботнет мережі. Зловмисники можуть реалізувати атаку у будь-який період часу. Виходячи з даного пункту спеціалістам із питань захисту інформації важко встановити «портрет зловмисника»;
- Слабкість або нестача технічних ресурсів компанії. Компанія, яка становиться об’єктом реалізації даного типу атак, частіше всього не має достатню кількість технічних засобів для знищення або мінімізації загрози.

Організації, які займаються питаннями безпеки інформації розробляють універсальні системи, які будуть гібридними в контексті захисту інформації. Даний продукт називається системою виявлення вторгнень.

Дана система була реалізована із технічної і економічної точки зору. Технічна сторона полягає на універсальності даних систем. Наявність інструментарію у даному типі ПЗ дозволяє проводити аналіз, дослідження та знищення атак.

Економічна точка зору полягає у втраті необхідності використання великої кількості спеціалізованого ПЗ, що веде до економії фінансових ресурсів та їх раціонального управління.

Проте дана система не має достатню кількість ресурсів для знищення реалізації атак відмови в обслуговуванні. Тобто, у даній системі практично відсутні методи, які дозволяють відслідковувати факт наявності атаки і засобів, які будуть знищувати або проводити аналіз даного типу атак.

Метою даної роботи можна вважати наступні пункти:

- аналіз сучасних систем виявлення вторгнень;
- аналіз методів систем виявлення вторгнень, що використовуються для знаходження атак;
- аналіз видів і категорій атак відмови в обслуговуванні та дослідження методики їх роботи;
- аналіз ефективності існуючих методів систем виявлення вторгнень у контексті аналізу атак відмови в обслуговуванні;
- аналіз нових методів систем виявлення вторгнень та дослідження їх поведінки в контексті аналізу атак відмови в обслуговуванні;

Об’єктом дослідження являються DDoS-атаки під наглядом методів аналізу систем виявлення вторгнень.

Предметом дослідження являються результати ефективності методів та алгоритмів нечіткої логіки в контексті аналізу і виявлення DDoS-атак.

Теоретичне значення отриманих результатів. У ході роботи було запропоновано використання алгоритмів нечіткої логіки та їх поліпшення, завдяки яким можна проаналізувати періодичність атак та їх вплив на систему.

Практична цінність отриманих результатів:

- проведено процедуру вдосконалення існуючих алгоритмів нечіткої логіки для виявлення DDoS-атак;
- проведено зрівняльну характеристику існуючих алгоритмів нечіткої логіки та вдосконалену версію на прикладі виявлення поведінки DDoS-атак.

РОЗДІЛ 1. СТАН ПИТАННЯ. ПОСТАНОВКА ЗАДАЧІ.

1.1 Системи виявлення вторгнень

В обчислювальній техніці система виявлення мережевих вторгнень являє собою спеціальний пристрій, який відстежує вхідний мережевий трафік. Дане програмне забезпечення зчитує пакети повідомлень, які відправляються через мережу, і визначає, чи є вони шкідливими чи ні. Багато компаній та інші організації мають потребу в цих системах для забезпечення безпеки своїх комп'ютерних мереж. Систему виявлення мережевих вторгнень часто вважають першою лінією захисту комп'ютерної мережі. Дана система може фільтрувати вхідний мережевий трафік на основі заданих параметрів загроз кібербезпеки. Дана система може відстежувати мережу на предмет багатьох типів кіберзагроз. До них відносяться атаки на відмову в обслуговуванні, віруси, черв'яки і шкідливий спам.

Система виявлення мережевих вторгнень відрізняється від мережевого екрану можливістю відслідковувати і відстежувати вектор реалізованої атаки. Мережевий екран тільки блокує несанкціанований мережевий трафік.

Більшість систем виявлення вторгнень відстежують вхідний і вихідний мережевий трафік компанії. Дане програмне забезпечення безпеки зчитує пакети повідомлень, які передаються по всій компанії в пошуках шкідливої активності. При виявленні підозрілого повідомлення воно зазвичай реєструється і блокується в мережі. Система виявлення мережевих вторгнень може вчитися на основі виявлених загроз. Оскільки повідомлення блокуються в мережі, вони додаються в дерево відповідей майбутніх потенційних загроз. Це забезпечує швидке додавання нових вірусів в систему виявлення, тим самим блокуючи шкідливу активність. [1]

Заснована на протоколі система виявлення мережевих вторгнень – це особлива форма виявлення, яка шукає певні типи повідомлень на основі протоколу. Це програмне забезпечення безпеки шукає повідомлення на основі протоколу на місці. Деякі приклади розглянутих протоколів включають протокол передачі гіпертексту (HTTP), безпечний протокол передачі гіпертексту (HTTPS) і простий протокол передачі пошти (SMTP).

Деякі програми забезпечення безпеки можуть фільтрувати шкідливі дії на основі певних IP-адрес. Цей тип системи виявлення мережевих вторгнень вважається менш складним інструментом, оскільки багато хакерів кібербезпеки підробляють IP-адреса, намагаючись сховатися від програмного забезпечення забезпечує захист мережі. Фільтрація IP-адрес аналогічна реєстру. Система шукає запити з певних IP-адрес і забороняє доступ до мережі при виявленні підозрілого адреси.

У систему виявлення вторгнень зазвичай включений обхідний перемикач. Цей комутатор представляє собою апаратний пристрій, який забезпечує шлюз для програмного забезпечення, моніторингу та перевірки пакетів в мережі. Обхідний перемикач знаходиться в точці входу в мережу, щоб забезпечити фільтрацію шкідливих повідомлень.

Багатоскладні системи виявлення вторгнень можуть відстежувати і ловити кіберзлочинців. Ці системи встановлюють внутрішні сигнали тривоги і надають метод вилову і реєстрації шкідливої активності. Контролюючи пристрій таким чином, фахівці з безпеки можуть виявити і відключити хакерів. [2]

1.1.1 Класифікація систем та їх особливості.

Дані системи умовно поділяються на декілька наступних категорій:

- Статичні системи;
- Динамічні системи;
- Мережеві системи;
- Хостові системи.

Статичні системи виявлення вторгнень фіксують середовище та виконують їх аналіз, знаходячи шкідливе ПЗ, віруси, помилки в конфігураціях системи тощо. Дані системи також можуть перевіряти встановлене ПЗ на наявність вразливостей, слабких паролів, перевірку вмісту файлів і здійснюють перевірку конфігурації мережевих сервісів.

Динамічні системи виявлення вторгнень сканують середовище у режимі реального часу всіх дій, які виконуються в середовищі, аналізуючи мережвеі пакети, файли аудиту, що передаються за певний проміжок часу.

Мережеві системи виявлення вторгнень встановлюються у місцях з активним мережевим трафіком всіх пристроїв в системі. Дана система сканує мережевий трафік і підмережі та порівнює отримані дані з бібліотекою даних, які вважаються системою «зловмисними». У результаті виявлення атаки, система надсилає спеціалісту з питань безпеки інформації повідомлення про реалізацію атаки.

Хостові системи виявлення вторгнень налаштовуються у хостовому середовищі, тобто на хості. Головна ідея даної системи – порівняння цілісності файлів, отриманих через мережу, здійснення контролю цілісності файлів,

створених мережевими службами та сервісами, ведення системи аудиту і внесення змін в файли (логування змін, внесених в файли).

За методами виявлення системи розподіляються на дві категорії:

- Аналіз сигнатур та протоколів;
- Аналіз аномалій.

Перша категорія проводить аналіз сигнатур та протоколів. Сигнатура базується на понятті збігу послідовності зі зразком. У пакеті проводиться стеження байта за байтом і отриминий результат порівнюється з сигнатурою (підписом) —рядком програми, що вказує на характеристику шкідливого трафіку. Такий підпис може містити ключову фразу або команду, яка пов'язана з нападом. Якщо збіг знайдено, оголошується тривога. Аналіз протоколів полягає в розгляді даних трафіку мережі. Кожен пакет супроводжується різними протоколами. Кожен протокол має кілька полів із нормальними параметрами. Якщо система зіштовхується із «незвичайними» даними, то система розцінює проаналізований об’єкт як ймовірну зловмисність. IDS переглядає кожне поле всіх протоколів вхідних пакетів: IP, TCP, і UDP. Якщо є порушення протоколу, наприклад, якщо він містить несподівані значення в одному з полів, оголошується тривога.

Система виявлення аномалій базується на знаходженні нових та невідомих видів атак через активний розвиток шкідливого ПЗ. Головний підхід даної системи полягає у використанні алгоритмів машинного навчання, щоб створити правдоподібну діяльність, з якою можна зрівняти незвичайну та нову поведінку. Даний метод дозволяє знаходити нові і невідомі атаки, проте з іншої сторони головним недоліком даного методу являється наявність хибно позитивних спрацювань, тобто невідома законна діяльність може бути розцінена як шкідлива. [3]

1.1.2 Проблеми сучасних систем виявлення вторгнень

Системи виявлення вторгнень розподіляються на декілька категорій, кожна з яких має власні позитивні і негативні сторони в залежності від технології, проекту і методів їх роботи. Дані проблеми розподіляються на дві головні категорії:

- Зовнішні;
- Внутрішні.

Перша категорія проблем стосується чинників, які можуть бути реалізовані за межами компанії, підприємства, організації тощо. Дані проблеми залежать від досвіду зловмисників, які реалізують атаку. Друга категорія проблем стосується внутрішнього середовища компанії, методи протидії яких будуть описані далі. У даній категорії буде проаналізовано наступні системи: [2]

- Системи, які відповідають за захист мережевих вузлів;
- Системи, які відповідають за захист серверу;

У СВВ, які спрямовані на захист мережевих вузлів присутні такі недоліки, як:

- Відсутність механізму виявлення атаки, яка розпочата у момент високого завантаження мережі. Крім того, необхідність швидко аналізувати пакети змушує розробників виявляти напад із мінімальними витратами обчислювальних ресурсів, що серйозно знижує ефективність виявлення;
- Більшість комутаторів не забезпечують універсальні порти керування, що скорочує контрольний діапазон СВВ датчика. У таких комутаторах окремий порт часто не може відобразити весь трафік, що проходить через комутатор;
- Відсутність механізму аналізу зашифрованої інформації;
- Відсутність механізму аналізу ступені проникнення атаки, у результаті реалізації самої атаки.

У СВВ, які спрямовані на захист серверу присутні наступні проблеми:

- Необхідність налаштування СВВ на кожному сервері;
- Механізми СВВ можуть бути заблоковані, завдяки освіченості зловмисника;
- Дані механізми лише аналізують трафік і не мають «механізмів контролю», тому що дані процеси «бачать» весь трафік, який надходить до серверу;
- СВВ не мають достатню кількість механізмів захисту, які зможуть протидіяти DDOS-атакам;
- Для протидії даним атакам використовуються лише серверні ресурси, що знижує працездатність СВВ. [3]

1.2 Розповсюдженість атак

Атака – це варіант стрімкої і активної дії проти опонента. У контексті захисту інформації даний термін визначає активну дію, яка негативним шляхом впливає на діяльність організації, структури або інших ресурсів, які мають технічне, соціальне, економічне, стратегічне та інші значення особливої важливості.

Розвиток суспільства, держави та інших галузей її життєдіяльності прийшов до необхідності створення, підтримки та реалізації різних аспектів, процедур, технологій тощо спрямованих на захист інформації та безпеку її ресурсів.

Атаки з боку зловмисників можуть розподілятися на декілька категорій з різними наслідками, цілями і процедурами. Серед даних атак можна виділити такі основні як:

- Атака відмови в обслуговуванні (Denial of Service Attack);
- Перехоплення і перенаправлення трафіку;
- Впровадження шкідливого ПЗ в технічні засоби;

- Троянські програми;
- Віруси;
- Шпигунські програми;
- Спам;
- Мережеві черв’яки.

Дані вектори атак мають різні фактори і призначення, які спрямовані на отримання доступу, знешкодження і інший негативний вплив на ресурс, такі як:

- Фактор часу;
- Освіченість;
- Швидкість виконання;
- Необхідність додаткових ресурсів тощо.

Виходячи із даних пунктів, самим легким і доступним видом атаки можна вважати саме атаку відмови в обслуговуванні, оскільки інші види атак потребують соціальної взаємодії з суб’єктами інформаційного ресурсу, наприклад, співробітниками організації. [4]

Атака відмови в обслуговуванні направляється на інформаційні сервери підприємства, функціонування яких являється критично важливою умовою для працездатності і ведення будь-якої діяльності організації. Об’єктами даних атак являються:

- Веб-сервери;
- Файлові і поштові сервери;
- Сервери системи доменного ім’я.

У наступній частині буде детально розглянуто види даної атаки, їх спрямованість і вплив на інформаційні ресурси компанії і методи реалізації даної атаки. [5]

1.2.1 DDOS-атаки як основний тип атаки

Даний тип атаки буде поставлено як основний через декілька факторів, які представляють собою найбільш ймовірну для імплементації:

- Безкарність. Зловмисники, використовуючи даний тип атаки частіше всього можуть оставатися анонімними для компанії, яка являється жертвою інциденту. Даний тип атаки використовує бот-нет мережі, які відправляють велику кількість запитів на сервери компанії.

- Доступність. Зловмисники отримують доступ до пристрою завдяки необізнаності користувачів. Встановлене шкідливе ПЗ не відслідковується через байдужість користувача та досвідченості зловмисника.

- Варіативність. DDOS-атака може бути реалізована різними варіантами і на різні ресурси. Завдяки даній розгалуженості атаки, зловмисник може змінювати стратегію атаки на ресурс із різною періодичністю та часом. Таким чином, системи аналізу атакованої організації не мають уявлення про точну поведінку зловмисника.

- Технічна обмеженість. Більшість компаній мають низький рівень технічної захищеності від даного типу атаки. Так як дана атака виконується за допомогою бот-нет мережі, то компанії не мають достатній рівень протидії.

У наступному розділі буде проведено детальний огляд різних типів даної атаки і методів їх роботи. [5]

1.2.2 Типи DDOS-атак

Умовно атаки можна розподілити на дві категорії: Атаки «третього і четвертого рівня» та атаки «сьомого рівня»

Перша категорія - це атаки на мережевому і транспортному рівнях. Зловмисники «забивають» канали передачі великою кількістю переданих пакетів. Канал не може впоратися з навантаженням і видає помилку «відмова в

доступі». DDoS-атаки можуть використовувати недоліки мережевих протоколів, перевантажуючи сервер, в результаті чого сервер перестає відповідати на запити.

Друга категорія – це атаки на рівні додатків, які споживають не тільки мережеві ресурси, а й ресурси сервера. Сервер не витримує навантаження, що призводить до його недоступності. Атака при цьому спрямована безпосередньо на операційну систему або додаток з метою змусити їх перевищити ліміт обчислювальної потужності сервера. При цьому атакам піддаються конкретні програми, сервіси та служби. Атаки можуть тривати довгий час (кілька годин або днів).

Далі будуть розглянуті більш детально типи DDoS-атак:

- HTTP-флуд. Найпростіший вид запитів, наприклад, за допомогою HTTP-заголовку пакета, який вказує запитуваний ресурс сервера. Зловмисник може використовувати скільки завгодно заголовків, надаючи їм потрібні властивості. При DDoS-атаці HTTP-заголовки можуть змінюватися, роблячи їх складними для виявлення. Крім того, HTTP-запити атаки можуть передаватися по захищеному протоколу HTTPS. У цьому випадку надіслані дані між клієнтом (зловмисником) і сервером шифруються. При цьому «захищеність» йде атакуючому тільки на користь: щоб виявити зловмисний запит, сервер повинен спочатку розшифрувати його. Тобто розшифровувати доводиться весь потік запитів, яких під час DDoS-атаки надходить дуже інтенсивно. Це створює додаткове навантаження на сервер-жертву, що призводить до вичерпання його обчислювальної потужності.
- SYN-флуд (TCP / SYN). Дана атака встановлює напіввідкриті з'єднання з вузлом. Коли сервер-жертва приймає SYN-пакет синхронізації через відкритий порт, він повинен послати у відповідь на сервер-джерело запиту пакет підтвердження SYN-ACK і відкрити з'єднання. Після цього запиту сервер-джерело повинен послати на сервер пакет підтвердження ACK, проте

зловмисник не посилає це підтвердження. З'єднання залишаються напіввідкритими до закінчення тайм-ауту. Черга на підключення переповнюється і нові клієнти не можуть з'єднатися з сервером.

- MAC-флуд. Зловмисник посилає потік порожніх фреймів Ethernet з різними MAC-адресами в кожному. Комутатори мережі розглядають кожну MAC-адресу окремо і резервують ресурси під кожен з них. Коли вся пам'ять комутатора використана, він або перестає відповідати, або зупиняє роботу. Іноді атака MAC-флудом може видаляти таблиці маршрутизації на вузлах мережі, таким чином порушуючи роботу всієї мережі, а не тільки одного сервера.
- Ping of Death. Майстер-бот Ping of Death посилає зловмисні пінг-запити через різні IP-протоколи на сервер, що призводить до його перевантаження. В даний час такі атаки рідкісні. Пропускна здатність мереж значно підвищилася, а для таких атак потрібно багато ресурсів - заражених спамерських пошукових роботів.
- Smurf Attack. Різновид Ping of Death. Цей вид атаки використовує протокол Інтернет (IP) і протокол керуючих повідомлень ICMP, на яких працює шкідлива програма Smurf. Вона підміняє IP-адресу атакуючого і пінг IP-адреси в корпоративній мережі, залишаючи на них напіввідкриті з'єднання.
- Fraggle Attack. Використовує великі обсяги трафіку UDP на мовній мережі маршрутизатора. Ця атака працює аналогічно Smurf-атаці, але замість протоколу ICMP використовує мовний протокол UDP.
- Slowloris. Атака Slowloris спрямована на веб-сервер. Атакуючий сервер підключається до цільового сервера і залишає це підключення напіввідкритим настільки довго, наскільки це можливо, а потім розмножує ці напіввідкриті з'єднання. Закриття таких з'єднань атакується сервером, а таймаут відбувається повільніше, ніж установка нових з'єднань. Протидіяти цій атаці досить складно, оскільки фаткично неможливо відстежити джерело атаки.

- NTP-посилення. Атака використовує сервери протоколу мережевого часу NTP, який використовується для синхронізації комп'ютерних годин в мережі. Мета атаки - перевантаження трафіком UDP. При цій атаці сервер відповідає, посилаючи UDP-трафік на підмінений зловмисником IP-адрес. «Посилення» означає, що обсяг відповідного UDP-трафіку набагато вище запиту. Тому мережа швидко перевантажується марним трафіком, а її вузли зупиняють роботу.
- Pulse Wave. Головна небезпека пульсуючих атак Pulse wave полягає в методиці періодичних сплесків трафіку. Звичайна DDoS-атака виглядає як поступово наростаючий потік шкідливого трафіку. Pulse wave є серією коротких, але потужних імпульсів, що відбуваються з певною періодичністю.
- APDoS. Удосконалена повторювана DoS-атака. APDoS націлена на нанесення максимальної шкоди до серверу. Він використовує механізми HTTP-флуду, SYN-флуда і інших видів атак. При цьому посилаються мільйони запитів в секунду. Така атака може тривати тижнями, оскільки зловмисник весь час змінює тактику атаки, типи атаки, щоб обдурити засоби протидії даним атакам. [6]

1.3 Методи пошуку і аналізу атак

Архітектура системи виявлення вторгнень містить купу методів та засобів, які застосовані для аналізу, дослідженню або перешкоджанню атак. В методах системи виявлення зловживань можна віднести наступні прийоми, які використовуються для перешкоджанню реалізації атак, такі як:

- Умовні ймовірності;
- Продукційні і експертні системи;
- Аналіз зміни станів;
- Спостереження за натисканням клавіш;
- Методи, які сфокусовані на моделі поведінки зловмисника.

У наступних розділах буде детально розглянуто кожний метод і описано способи їх використання.

1.3.1 Умовні ймовірності.

Для виявлення зловживань необхідно встановити умовну ймовірність за формулою 1.1:

$$P(AoA|Pat) \tag{1.1}$$

де P – умовна ймовірність, AoA – кількість атак або вторгнень, Pat – патерн подій. Тобто, головна ідея даного методу – це встановити ймовірність того, що будь-яка множина або множини подій являються діями зловмисника.

Наступним етапом необхідно використати формулу 1.2:

$$P(I|A_1, A_2, \dots, A_n) = P(A_1, A_2, \dots, A_n|I)\frac{P(I)}{P(A_1, A_2, \dots, A_n)} \tag{1.2}$$

де I – вторгнення, а A1…An – послідовність подій. Кожна подія – це сукупність параметрів оцінки захищеності системи.

1.3.2 Продукційні та експертні системи

Головна особливість використання продукційних систем являється у можливості розподілення причин і вирішень проблем, які виникають при реалізації атак зловмисником.

Даний метод кодує інформацію про вторгнення у правила формату «if(якщо) причина then(тоді) вирішення», при цьому додавання правил «причина» співпадає із подіями, які реєструються підсистемою збору інформації системи виявлення вторгнень. У частині «якщо» правила кодуються у стан, необхідний для атаки. Коли всі умови у лівій частині задоволені, виконуються дії (вирішення) які задані у правій частині.

Головні проблеми додатків, які використовують даний метод, виникають при практичному використанні даного методу:

- Недостатня ефективність при роботі з великою купою інформації;

- Важко враховувати природу залежності параметрів оцінки.

При використанні продукційних систем для виявлення вторгнень можна встановити проявлення вторгнень за допомогою отриманої інформації.

Проблеми:

- Відсутність вкладеної або природної оброки порядку послідовності проаналізованих даних. База фактів, яка відповідає за ліву частину «продукції» використовується для правої частини. У лівій частині продукційного правила всі елементи об'єднуються за допомогою «і» зв'язку;
- Вкладена експертиза буде ефективною тільки у тих випадках, якщо навички адміністратора безпеки не будуть заважати роботі;
- У даному випадку будуть зафіксовані тільки відомі вразливості;
- В системі існує певне ПЗ, яке підтримує і оновлює базу знань. При додаванні або видаленні будь-якого з правил множина правил теж повинна змінюватися.
- Об'єднання різних векторів вторгнень та створення залежності між ними може привести до того, що окремі випадки (причини) становляться невизначеними.

1.3.3 Аналіз зміни станів

Сигнатура вторгнення описується як послідовність переходів між станами системи захисту. Патерни атаки (сукупність значень параметрів оцінки) співпадають із будь-яким станом системи, яка потребує захисту і має підключену з ним логічну функцію. Якщо та функція виконується, то вважається, що система перейшла в наступний стан. Послідовні стани зв'язані з активними лініями, які являють собою необхідні події для подальших переходів. Типи можливих подій вмонтовані в модель і відповідають (хоча і не обов'язково), значенням параметрів оцінки по принципу один до одного.

Патерни атаки можуть тільки задати послідовність подій, тому більш важкий спосіб аналізу подій не підтримується. Також, відсутній загальний механізм цілей, який можна було б використовувати для розподілення часткового співпадіння атак. Замість цього можна використати вкладену логічну функцію.

1.3.4 Спостереження за натисканням клавіш

Для виявлення атак у даної технології використовується моніторинг за натисканням клавіш клавіатури. Основна ідея – це аналіз послідовності натискань користувача, який може задати патерн атаки. Недоліком даного методу являється відсутність вправного механізму зчитування роботи з клавіатурою без підтримки операційної системи, а також великої кількості можливих варіантів представлення однієї і тієї самої атаки. Крім того, без семантичного аналізатору натискань, різні варіанти псевдокоманд можуть легко ліквідувати дану технологію. Оскільки вона спрямована на аналіз натискань клавіш, а автоматизовані атаки, які являються результатом виконання програм зловмисника, також можуть бути не знайдені.

1.3.5 Методи, які сфокусовані на моделюванні поведінки зловмисника

Одним із варіантів виявлення зловживань являється метод об’єднання моделі зловживання із об’єктивними причинами. Сенс даного методу розкривається наступним чином: в системі існує база даних сценаріїв атак, кожна з яких формує послідовність поведінок, які формують атаку. У будь-який момент часу існує можливість того, що в системі має місце одне із підмножин сценаріїв атак. Далі робиться спроба перевірки варіанту стосовно їх наявності шляхом пошуку інформації в записах аудиту. Результатом пошуку являється будь-яка кількість фактів, якої достатньо для підтвердження або відхиленні гіпотези. Перевірка виконується в одному процесі, який називається «антисипатор». Антисипатор, базуючись на стані активної моделі, формує наступні можливі

множини подій, які необхідно перевірити в записах аудиту та передає отримані результати «планувальнику». Планувальник визначає, яким чином передбачувана поведінка виражається в записах аудиту і трансформує їх в системно-аудитозалежний результат. Дані результат повинні містити такі структури, які можна було би просто знайти в записах аудиту та для яких існувала б достатньо висока ймовірність появи в записах аудиту. Внаслідок того, як підстави для підозр деяких сценаріїв накопичуються, а для інших - знижуються, список моделей активностей зменшується. Обчислення причин вбудовано в систему і дозволяє оновлювати ймовірність появи сценаріїв атак в списку моделей активності.

Переваги:

- З'являється можливість зменшити кількість істотних обробок, необхідних для одного запису аудиту; спочатку спостерігаються більш «грубі» події в пасивному режимі, і далі, як тільки одне з них виявлено, спостерігаються більш точні події;
- Планувальник забезпечує незалежність подання від форми даних аудиту.

Недоліки:

- При застосуванні даного підходу у особи, відповідальної за створення моделі виявлення вторгнення, з'являється додаткове навантаження, пов'язане з призначенням змістовних і точних кількісних характеристик для різних частин графічного представлення моделі;
- Ефективність цього підходу була продемонстрована створенням програмного прототипу. З опису моделі не ясно, як поведінки можуть бути ефективно складені в планувальнику, і який ефект буде впливати на систему під час роботи;

- Цей підхід доповнює, але не замінює підсистему виявлення аномалій. [7]

1.4 Висновок до першого розділу

Отже, можна сказати, що будь-яка організація, яка займається будь-якою діяльністю може стати жертвою потенційних зловмисників через різні причини:

- Конкуренція;
- Політичні інтереси;
- Інформація, яка становить інтерес для зловмисників;
- Крадіжка фінансових ресурсів жертви тощо.

Задля запобігання реалізації різних векторів атаки, суб'єкт потребує захисту власних ресурсів шляхом створення «блокад» в контексті інформаційної безпеки компанії. Унікальним та універсальним засобом захисту інформації вважається СВВ, оскільки даний механізм має купу методів для аналізу, виявлення і можливої ліквідації атаки.

Розповсюдженим варіантом реалізації атаки на сьогоднішній день вважається атака відмови в обслуговуванні, головна ідея якої – перешкоджання, знищення або перенавантаження мережево-технічних ресурсів суб'єкта. Даний тип атаки має декілька ефектів впливу на підприємство:

- Компанія втрачає час на відновлення систем, а це може призвести до втрати ресурсів різного характеру;
- Організації починають панікувати, що може призвести до фатальних і критичних помилок з боку жертви;
- Дестабілізація структури підприємства, яке може паралізувати діяльність організації.

При дослідженні робіт різних авторів та існуючих засобів захисту можна сказати, що на сьогоднішній день існує велика кількість механізмів захисту, які використовують апаратні, технічні, організаційні та інші методи, які

забезпечують захист інформації в різних інформаційних системах. Головне питання виділяється в особливостях різних компаній, частоті реалізованих атак на компанію та їх інформаційного середовища. В залежності від даного фактору будується комплекс систем захисту, спрямований на безпечне існування та циркуляцію інформації.

Головна мета даної роботи полягає у покращені рівня якості систем безпеки, які спрямовані на зниження реалізації атак відмови в обслуговуванні, як один із основних та ефективних засобів зниження якості циркуляції інформації в системі компанії.

Для досягнення встановленої мети було вирішено наступні задачі:

- Проаналізувати існуючі системи протидії атакам відмови в обслуговуванні та їх обґрунтування з точки зору ефективності, надійності та функціональності;
- Запропонувати методи штучного інтелекту та нечіткої логіки для створення системи фільтрації шляхом розподілення підозрілого і надійного трафіку;
- Вдосконалити описану систему для підвищення її якості;
- Продемонструвати інструменти, спрямовані на тестування ефективності і витривалості існуючих систем та виділити необхідні дані як тестові вибірки, які спрямовані для навчання алгоритмів нечіткої логіки та штучного інтелекту.

У наступному розділі буде проведено детальний аналіз кожного методу виявлення реалізації атак відмови в обслуговуванні в механізмі системи виявлення вторгнень, виявлено позитивні і негативні сторони кожного із описаних методів в контексті аналізу даного типу атаки і зроблено висновки стосовно наступного розділу.

РОЗДІЛ 2. СПЕЦІАЛЬНА ЧАСТИНА

2.1 Математична модель виявлення початку атаки та шкідливого трафіку

Вигідним варіантом для виявлення початку атаки і шкідливого трафіку буде застосування механізмів, які базуються на аналізі аномалій, внаслідок якого реалізується порівняння поточного стану системи з її нормальним станом. Порівняльний аналіз станів системи з точки зору DdoS-атак можна реалізувати шляхом порівняння різних характеристик мережевої активності. До даних властивостей можна віднести наступні пункти:

- Кількість запитів;
- Типи запитів;
- Час реалізації запитів тощо.

Сучасні вектори атаки постійно розвиваються і зловмисники намагаються імітувати діяльність користувачів системи. У даному випадку при аналізі властивостей мережевої активності необхідно сконцентруватися на параметрах, які не можуть бути підроблені зловмисниками.

Можна сказати, що завдання по дослідженню та знаходженню шкідливого трафіку можна звести до їх класифікації на підставі особливостей мережевої активності. Вірним варіантом у даній ситуації можна вважати використання різноманітних класифікаторів і систем нейронних мереж з метою знаходження шкідливого трафіку. Негативною стороною у реалізації даної ідеї являється той факт, що для позитивної роботи класифікатора необхідно мати як мінімум дві актуальні навчальні вибірки, які стосуються шкідливого та коректного трафіку. Хоча до моменту реалізації атаки, процес отримання вибірок являється неможливим.

Задля знешкодження даної проблеми необхідно точно встановити точку реалізації атаки. Даний механізм відкриє можливість віднесення всього

попереднього трафіку до коректного і створить можливості щодо розподілення змішаного трафіку, який надходить після початку атаки. У даній ситуації методика виявлення шкідливого трафіку наближуватися до наступних завдань:

- Визначення актуальності сезонних періодів;
- Визначення точки початку атаки із урахуванням сезонності;
- Присвоєння коректного статусу всьому отриманому трафіку;
- Розподілення трафіку на коректний і шкідливий;
- Порівняння коректного трафіку із змішаним трафіком, який було отримано до початку атаки;
- Коригування вибірок згідно отриманих результатів у попередньому пункті;
- Аналіз вхідного трафіку із урахуванням отриманих даних.

Початок роботи атаки відмови в обслуговуванні сфокусований із збільшенням числа запитів до атакованого серверу. Як наслідок, для встановлення факту наявності атаки необхідно знайти межу по кількості запитів до сервера. Завдяки даній ідеї, можна встановити аномальність отриманих запитів, яка може привести до нештатної події. Даною границею можна вважати максимальну кількість запитів до сервера і обмежений запас можливих запитів. Варіант налаштування даної межі, після якої буде активовано механізми захисту систем безпеки може бути реалізований як і програмним, так і апаратним шляхом. Проте даний аспект має купу негативних факторів:

- Задля зниження кількості випадкових спрацьовувань, межа повинна бути вище максимального рівня кількості запитів, що може привести до наявності похибки у результаті виявлення атаки;
- Технічні засоби, які відповідають за підтримку роботи мережі можуть знаходитися під хаотичним навантаженням в залежності від періодичності атак. У такій ситуації, реалізовані атаки можуть бути зафіксовані

із запізненням. Якщо у СВВ наявний механізм використання класифікаторів та навчання фільтрів на підставі вхідного трафіку, існує варіант у їх негативному навчанні.

Задля знешкодження виявлених проблем необхідно застосовувати спеціальний параметр, який буде характеризувати активність мережі у режимі реального часу. На підставі даного параметру можна визначати активну межу, яка буде актуальна у результаті періоду почату атаки. Як приклад спеціального параметру можна використовувати середньоквадратичне відхилення, яке описано у формулі 2.1.

$$\sigma = \sqrt{\frac{1}{n} \cdot \sum_{i=1}^{n} (x_i - \bar{x})^2} \tag{2.1}$$

де σ - середньоквадратичне відхилення; n - кількість розглянутих часових періодів; і x - кількість запитів за і-період; x - середнє арифметичне запитів по всіх періодах. У результаті проведених досліджень на різних веб-ресурсах було встановлено, що параметр, необхідний для встановлення верхньої межі може відрізнятися і знаходитися в діапазоні від 2.2σ до 2.9σ. Через дану причину для більш динамічної роботи програмного забезпечення по виявленню атак відмови в обслуговуванні необхідно встановлювати даний параметр у лаконічній формі (значення повинно бути динамічним). У спеціаліста в налаштуванні даного ПЗ наявні можливості у виді регулювання даних значень. Проте даний прийом може негативно відповісти у вигляді потенційної вразливості, якою зловмисник може скористатися. Головна ідея даної вразливості – це хаотичне відхилення границі середньоквадратичного значення, оскільки зловмисник може пересувати дану межу шляхом поступового нарощування потужності атаки. Ліквідувати дану вразливість можливо завдяки механізмам обліку сезонних коливань. [8]

2.1.1 Виявлення початку атаки статичними методами

Статичні методи являють собою будь-які механізми захисту, які можуть бути як і окремим ПЗ, так і елементом СВВ, який надає можливість аналізу, знешкодження або попередження про шкідливі дії, які надходять із мережі. До статичних методів відносяться наступні засоби:

- Фільтрація та блекхоллінг;
- Розмежування;
- Звернена атака відмови в обслуговуванні;
- Знаходження вразливостей;
- Нарощування технічних ресурсів;
- Ухилення.

Фільтрація та блекхолінг дозволяє застосувати системи захисту та спрямувати їх ресурси на блокування трафіку. Ефективність даних засобів знижується, якщо механізми захисту наближуються до об’єкту атаки. У супротивному випадку – підвищуєься. До даних механізмів захисту можна віднести наступні технології:

- Мережевий екран;
- Лист ACL.

Мережеві екрани допомогають блокувати загальний трафік, який надходить від всіх засобів, які намагаються надіслати запит до мережевого ресурсу. Основним недоліком даної технології в контексті виявлення атак можна вважати відсутність механізму визначення трафіку. Тобто, дана технологія не має додаткових ресурсів для визначення надійного або шкідливого трафіку.

Лист контролю доступу (ACL) являє собою лист управління доступом, який допомагає визначити користувачів, які можуть отримати доступ до системи та її ресурсів. Не виключним варіантом можна вважати фільтрацію додаткових протоколів, окрім TCP-протоколів. В контексті знаходження потенційних атак

відмови в обслуговуванні даний механізм захисту може бути не ефективним у ситуаціях, коли зловмисник надсилає шкідливий трафік завдяки першорядним значенням запитів.

Розмежування трафіку надає можливість технічним та серверним ресурсам знизити ефективність атак відмови в обслуговуванні. Як приклад, можна віднести проксі-технології, які дозволяють розділити трафік між технічними засобами та розподілити відповідальність за обробку запитів. Технологія зверненого проксі-серверу представляє собою механізм розподілення трафіку між серверами, приховуючи технічні характеристики кожного мережевого ресурсу, який приймає участь в отриманні трафіку. Не виключним фактором являється розподілення трафіку між кількома серверами, які розташовані у мережі. Дана технологія має механізм мережевого екрану, який допомагає блокувати трафік. З іншої сторони, дана технологія залежить від кількості серверних та технічних ресурсів. Тобто, чим менше ресурсів приймає участь в обробці трафіку, тим не ефективнішим являється технологія в цілому.

Звернена атака відмови в обслуговуванні представляє собою комплекс програмно-апаратних засобів, які спрямовані на знешкодження атак відмови в обслуговуванні. Дана технологія надає можливість «парирувати» надіслані запити і надіслати їх зловмиснику. Задля реалізації даного механізму захисту необхідно мати велику кількість ресурсів та місце надходження атаки.

СВВ та інші механізми захисту, спрямованих на зниження реалізації атак відмови в обслуговуванні представляють собою засоби пошуку вразливостей в системі. Дані системи допомогають знизити ефективність реалізованих атак відмови в обслуговуванні, проте з іншої сторони дані засоби можна вважати не ефективними у випадках оновлення ПЗ та інших ресурсів, які використовуються в системі.

Нарощування технічних ресурсів дозволяє оброблювати велику кількість трафіку. У випадку реалізації атак відмови в обслуговуванні ідея поліпшення серверних ресурсів мінімізує ймовірність виведення робочих систем та серверних ресурсів з ладу.

Технології «ухилення» направлені на зниження ефективності атак відмови в обслуговуванні шляхом уникнення самого факту атаки. До таких механізмів захисту можна віднести honeypot-ресурси, підміна IP-адресів та інші засоби, які допомагають зберегти технічні ресурси системи. [9]

2.1.2 Виявлення алгоритму сезонності атак

Даний термін слід розуміти, як набір шкідливих дій зловмисника, який впливає на систему з певною періодичністю часу. Сезонність у даному контексті представляє собою наступні фактори:

- Інтервал (періодичність) атак;
- Частота атак;
- Ефективність атак.

Періодичність атак залежить від зовнішніх факторів, на які впливає атакована система. Якщо система має низький рівень захисту, то інтервал атак буде коротший. У супротивному випадку – довший.

Частота атак являється похідною від періодичності. Визначення даного параметру обирається із загального трафіку, який надходить до системи. Якщо в системі фіксуються аномальна кількість «недійсних» користувачів, то система розцінює даних суб'єктів як елементи реалізації атак. Дані елементи відносяться до шкідливого трафіку і фіксуються у статистичній системі. У ситуації, коли дані елементи повторно проводять атаку на ресурси системи, статистична вибірка оновлюється і таким чином встановлюється частота проведених атак.

Ефективність атак залежить від зовнішніх та внутрішніх чинників, які впливають на систему. До внутрішніх факторів можна віднести такі пункти, як:

- Якість мережі;
- Кількість технічних ресурсів;
- Ефективність систем захисту.

До зовнішніх чинників відносяться такі аспекти, як:

- Досвід зловмисника;
- Кількість технічних ресурсів зловмисника;
- Якість технічних ресурсів зловмисника.

З математичної точки зору, ефективність атак представляє собою співвідношення успішних та невдалих атак та їх вплив на систему.

2.1.3 Формальний опис сезонності

Формальний опис сезонності або аналіз часових рядів представляє собою сукупність математико-статичних методів аналізу, спрямованих на визначення структури числових рядів і заходів, щодо їх прогнозування. Дані заходи необхідні з метою побудування математичної моделі, основним джерелом якого являється проаналізований часовий ряд.

Даний підхід необхідний для вирішення наступних цілей:

- Визначення природи ряду;
- Прогнозування майбутніх значень часового ряду.

В аспекті виявлення атак, дані для вибірки будуть представляти час або інтенсивність атак.

Для дослідження часових рядів необхідно застосувати один або декілька методів, таких як:

- Спектральний аналіз;
- Кореляційний аналіз;
- Моделі авторегресії і ковзного середнього;
- Багатоканальні моделі авторегресії і ковзного середнього;
- Сезонна модель Бокса-Дженкінса;

- Прогноз експоненціально зваженим ковзаючим середнім.

Спектральний аналіз представляє собою метод аналізу часових рядів, головна ідея якого – це перетворення одномірного часового ряду у багатомірний ряд із послідовним використанням к отриманому результату методу головних компонент.

Спосіб перетворення одномірного ряду у багатомірний можна уявити як «обертання» часового ряду у матрицю, яка містить елементи часового ряду у відхиленій формі.

Даний метод можна використовувати для аналізу як і випадкових, так і нестаціонарних рядів. Головна ідея даного методу – це розподілення отриманого ряду на множини конкретизованих параметрів, наприклад шум, тренд, періодичні компоненти.

Для прикладу буде розглянуто числовий ряд $X = x_i, \dots, x_n$ довжини N. Нехай L $(1 < L < N)$ – умовне ціле значення, яке називається «довжиною вікна» та $M = N - L + 1$.

$$\mathbf{X} = [X_1 : \dots : X_M] = (x_{ij})_{i,j=1}^{L,} = \begin{bmatrix} x_1 & x_2 & x_3 & \dots & x_M \\ x_2 & x_3 & x_4 & \dots & x_{M+1} \\ x_3 & x_4 & x_5 & \dots & x_{M+2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_L & x_{L+1} & x_{L+2} & \dots & x_N \end{bmatrix}, \tag{2.2}$$

Наступним етапом при розкладенні траекторної матриці X являється сингулярне розкладення. Створимо параметр $S = XX^T$ і позначимо $\varepsilon_1, \dots, \varepsilon_L$ власні параметри S взяті у наступному нарощуваному порядку $(\varepsilon_1 \geq \dots \geq \varepsilon_L \geq 0)$ і $U_1, \dots, U_L$ ортонормовану систему власних векторів S, яка дорівнює власним параметрам.

Положимо $d = rankX = \max\{i: \varepsilon_i > 0\}$ та $V_i = X^T U_i / \sqrt{\varepsilon_i} (i = 1, \dots, d)$. При даних позначеннях сингулярне розкладення матриці X може бути записано як $X = X_1, \dots, X_d$.

У даному випадку, буде отримано результат матриці $X_i = \sqrt{\varepsilon_i} U_i V_i^T$, які будуть мати перший ранг і називаються елементарними матрицями. Набір $(\sqrt{\varepsilon_i}, U_i, V_i^T)$ називаються і-ю власною трійкою сингулярного розкладення. Вектори U_i і V_i називаються лівими і правими, тобто, сингулярними векторами матриці X числа $\sqrt{\varepsilon_i}$. Дані числа являються сингулярними, тому вектори$\sqrt{\varepsilon_i} V_i = X^T U_i$, називаються векторами головних компонент.

Наступний етап являється шляхом об'єднання трійок. Множина всіх індексів {1,..,*d*} розділяється на *m* неперетинальних підмножин $I_1, \dots, I_m$.

Уявимо, що $I = \{i_1, \dots, i_p\}$. Тоді отримана матриця X_i, яка відповідає групі *I*, визначається як $X = X_{I_1}, \dots, X_{I_p}$. Отримані матриці розраховуються по групам $I = I_1, \dots, I_m$. Тоді сингулярні вектори можна записати як $X = X_{I_1}, \dots, X_{I_m}$.

Останнім етапом у розрахунках являється діагональне усереднення. Кожна матриця X_{I_j} згрупованого розкладення ганкелізується. Отримана ганкелізована матриця трансформується у новий ряд довжини *N* на основі взаємно-однозначної віпдовідності між ганкелевими матрицями і часовими рядами. Діагональне усереднення, яке приймається до кожної матриці X_{I_k} створює встановлені ряди $\widetilde{X^{(k)}} = (\widetilde{x_1^{(k)}}, \dots, \widetilde{x_n^{(k)}})$. Таким чином, початковий ряд $x_1, \dots, x_N$ розкладується на суму *m* встановлених рядів і виражається у наступній формулі:

$$x_n = \sum_{k=1}^{m} \widetilde{x_n^{(k)}} \ (n = 1,2 \dots, N) \qquad (2.3)$$

Дане розкладення являється основним результатом описаного алгоритму для аналізу часового ряду. Алгоритм можна вважати ефективним, якщо кожен із його компонент може бути інтерпретований як тренд, шум, або коливання.

Кореляція представляє собою статичну взаємодію між двома або кількома випадковими параметрами. При цьому значення одного або кількох параметрів приєднуються до систематичних змін значень вказаних параметрів.

Математичною мірою взаємодії двох випадкових параметрів являється кореляційний коефіцієнт *R*. У випадку, якщо зміна однієї випадкової величини не може вести до закономірної зміни другої випадкової величини, але результат приводить до зміни другої статистичної характеристики даної випадкової величини, то дана взаємодія вважається кореляційною, проте і являється статичною.

Методи прорахування параметрів кореляції можна розподілити на дві основні категорії:

- Параметричні показники кореляції;
- Непараметричні показники кореляції.

Важливим аспектом параметричних показників кореляції можна вважати ковариацію (кореляційний момент). Ковариація являється сумісним центральним моментом другого порядку. Ковариацію, як параметр можна визначити результатом математичного очікування добутку відхилення випадкових величин.

$$cov_{XY} = M\big[\big(X - M(X)\big)\big(Y - M(Y)\big)\big] = M(XY) - M(X)M(Y) \qquad (2.4)$$

У даному випадку параметр M представляє собою математичне очікування.

Для знешкодження негативних сторін ковариації необхідно прорахувати отримані дані за допомогою лінійного коефіцієнту кореляції. Коефіцієнт кореляції змінюється у межах від -1 до +1.

$$r_{XY} = \frac{cov_{XY}}{\sigma_X \sigma_y} = \frac{\sum(X - \overline{X})(Y - \overline{Y})}{\sqrt{\sum(X - \overline{X})^2 \sum(Y - \overline{Y})^2}} \quad (2.5)$$

Параметри $\overline{X}$ та $\overline{Y}$ розраховуються за наступними формулами,

$$\overline{X} = \frac{1}{n}\sum_{t=1}^{n} X_t \quad (2.6)$$

$$\overline{Y} = \frac{1}{n}\sum_{t=1}^{n} Y_t \quad (2.7)$$

де лінійний коефіцієнт кореляції пов'язаний із коефіцієнтом регресії у вигляді залежності $R_{XY} = a_i \frac{\sigma_{x_i}}{\sigma_Y}$, де a_i – коефіцієнт регресії, σ_{x_i} – середньоквадратичне відхилення факторного признаку. Відношення коефіцієнта регресії до середньоквадратичного відхилення Y не залежить від параметру Y.

До непараметричних показників кореляції відносять параметри рангових кореляцій, наприклад коефіцієнт кореляції знаків Фехнера. Даний елемент підраховує всі можливі параметри за наступною формулою 2.8

$$i = \frac{C - H}{C + H} \quad (2.8)$$

де C – кількість пар, у яких знаки відхилень параметрів співпадають від їх середніх значень, H – кількість пар, у яких знаки відхилень параметрів не співпадають від їх середніх значень.

Отже, можна сказати, що кореляційний аналіз – це метод обробки статичних даних, за допомогою якого вимірюється наближеність зв'язку між двома або кількома змінними.

Авторегресійна модель представляє собою модель часових рядів, у якій параметри часового ряду в залежать від попередніх параметрів даного ряду. Авторегресійний процес описується наступною формулою,

$$X_t = c + \sum_{i=1}^{p} a_i X_{t-i} + \partial_{t1} \quad (2.9)$$

де $a_1, \dots, a_p$ – параметри моделі (коефіцієнти авторегресії), с – постійна (константа, яка частіше всього при вимірах дорівнює нулю), а ∂_t – білий шум (математичні похибки).

Виходячи із даної формули, можна вивести наступне рівняння авторегресійного процесу першого порядку.

$$X_t = c + rX_{t-1} + \partial_t \quad (2.10)$$

Для даного порядку, коефіцієнт авторегресії співпадає із коефіцієнтом автокореляції першого порядку.

Для визначення сезонності використовують сезонні авторегресійні моделі. Сезонність моделі визначається через наявність часового елементу (місяця, року, кварталу та ін.). Як приклад буде побудовано сезонну авторегресійну мс (2.11) квартал у фомулі 2.11:

$$y_t = a_4 y_{t-4} + \partial_t \quad (2.11)$$

Фактично, дана модель являється стандартною формою звичайної авторегресійної моделі. На практиці сезонність може співпадати із авторегресією, наприклад:

$$y_t = a_1 y_{t-1} + a_4 y_{t-4} + \partial_t \quad (2.12)$$

У деяких ситуаціях дані співвідношення можуть бути корисними, наприклад у вибірках і розрахунках, де випадкова помилка може підкорюється будь-якому авторегресійному процесу:

$$y_t = a_4 y_{t-4} + \partial_t \quad (2.13)$$

У випадку використання ковзного середнього, то даний елемент розраховується за наступною формулою,

$$X_t = \mu + \partial_t + \sum_{i=1}^{q} \theta_i \partial_{t-i} \quad (2.14)$$

де $\theta_1, \dots, \theta_q$ представляють собою параметри моделі, μ – математичне сподівання X_t (як правило, даний параметр може дорівнювати нулю).

Виходячи із даних формул, можна отримати рівняння АРКС:

$$X_t = c + \sum_{i=1}^{p} a_i X_{t-i} + \varphi_i X_{t-i} + \sum_{i=1}^{q} \theta_i \partial_{t-i} \quad (2.15)$$

Важливо пам'ятати, що математичні похибки ∂_t вважаються незалежними однаково розподіленими випадковими величинами, які відбираються нормальним розподілом із нульовим середнім $\partial_t \sim N(0, \sigma^2)$, де σ^2 представляється дисперсією. Дані припущення можна послабити, проте дане внесення коректівів може змінити властивості моделі. Слід враховувати, що зміна припущення про незалежні однаково розподілені випадкові величини може призвести до принципової відмінності.

Багатоканальні авторегресійні моделі відрізняються від лінійних властивістю «нелінійності». Якщо у лінійних моделях присутня система підпорядкованості параметрів (параметри виступають у ролі якості коефіцієнтів змінних) та змінних (змінні входять в період першого ступеня), то дані нелінійні системи можуть отримати хаотичний результат. Для коректного розрахування параметрів необхідно застосовувати логарифмічні формули та функції для знаходження і налаштування багатоканальної моделі.

Для наближення моделі до стану нелінійності можна зробити процес розрахунку із використанням лінійних параметрів та змінних, проте, отримані дані будуть не точні.

Сезонна модель Бокса-Дженкінса являється доповненою версією АРКС. Особливістю даної моделі вважається формування нестаціонарних часових рядів

у стаціонарну форму шляхом визначення різниці деякого порядку від вихідного часового ряду.

Модель можна описати формулою 2.16,

$$\Delta^d X_t = c + \sum_{i=1}^{p} a_i \Delta^d X_{t-i} \sum_{j=1}^{q} b_j \rho_{i-j} + \rho_t \qquad (2.16)$$

де ρ_t – стаціонарний часовий ряд, c, a_i, b_j – параметри моделі, Δ^d – оператор різниці часового ряду порядку d (послідовне визначення d раз різниць першого порядку – спочатку від часового ряду, а потім від отриманих різниць першого порядку, другого порядку тощо).

Підхід моделі Бокса-Дженкіса до часових рядів визначається оцінкою їх стаціонарності. Із використанням різних тестів виявляється наявність одиничних рядів та порядок інтегрованості часового ряду (використовується обмеження першого або другого порядку). Якщо необхідно, ряд перетворюється взяттям різниці відповідного порядку і для отриманої моделі будується відповідна авторегресійна модель із ковзним середнім. Дана процедура виконується з метою встановлення припущення, що отриманий процес являється стаціонарним у відмінності від нестаціонарного процесу.

Авторегресійна модель із використанням експоненціального згладжування дозволяє отримувати точні результати при аналізі часових рядів. Параметр згладжування розраховується за формулою 2.17,

$$f_k = f_{k-1} + l(x_{k-1} - f_{k-1}) \qquad (2.17)$$

де l – постійна згладжування (число в проміжку від 0 до 1, яке визначає ступінь згладжування), f_k – прогноз в момент часу, k – реальний показник в момент часу.

Необхідно враховувати даний факт, що в методі експоненціального згладжування важливу роль відіграє суб'єктивний вибір двох параметрів:

прогнозу на перший використовуваний момент часу f_1 і постійної згладжування l. Якщо вибір початкових даних прогнозу повністю залежить фізичних факторів (статистична вибірка, досвід тощо), то постійна згладжування може бути обрана методом проб і помилок, яка засновується на послідовному наближенні прогнозу до реальних значень. При використанні методу експоненціального згладжування для складання прогнозу використовується унікальна постійна згладжування.

2.1.4 Вибір та обґрунтування методу кластеризації

Метод кластеризації представляє собою розбиття сукупності даних на підножини (кластери), які можуть збиратися у групи завдяки шаблону специфічного параметра. В свою чергу шаблон являється учбовою статистичною вибіркою даних, головна мета якої – навчання моделі.

Кластерний аналіз виконує наступні задачі:

- Розробка типології або класифікації;
- Дослідження корисних концептуальних схем групування об’єктів;
- Створення гіпотез на підставі дослідження даних;
- Перевірка гіпотез або дослідження для визначення «дійсності» отриманих результатів.

Реалізація моделі потребує наступні етапи:

- Вибір даних (вибірки) для кластеризації;
- Визначення множини змінних, завдяки яким будуть оцінюватись об’єкти у вибірці, тобто признакового простору;
- Розрахування значень завдяки шаблонам співпадіння (належності до однієї множини об’єктів);
- Виконання моделі кластерного аналізу з метою створення груп схожості об’єктів;
- Перевірка коректності результатів кластерного рішення.

Пріоритетність вибору методу кластеризації залежить від множини факторів, у порівнянні з іншими методами аналізу даних, можуть бути не ефективними:

- Непередбачуваність атак впливає на якість збору даних. Внаслідок даного фактору, моделі завдання навчання «із вчителем» можуть бути неефективними, оскільки дані моделі враховують фіксовану поведінку даних;
- Особливість виявлення трафіку. У результаті специфічності атак відмови в обслуговуванні, система приймає як і корисний так і шкідливий трафік в одному джерелі. Метод кластеризації дозволяє розподілити трафік у певні групи, що дозволяє скоротити час як і для навчання моделі, так і у фільтруванні трафіку. Іншим методам необхідний додатковий час, що може бути критичним у певних ситуаціях.
- Математична відсутність. Під даним параметром слід розуміти неефективність або відсутність спеціалізованих методів для розподілення даних.

2.1.5 Метод кластеризації на основі алгоритму k-means

Уявимо що змінна T – це множина клієнтських запитів, які були надіслані до початку реалізації атаки. Множину запитів, які надходять після реалізації атаки – це сукупність множин H – шкідливих клієнтських запитів і T* - коректних клієнтських запитів. Тобто, для реалізації вибірки, яка знаходить шкідливий трафік треба розділити отримані дані на дві категорії.

Оскільки кількість кластерів являється відомою характеристикою, то можна припустити варіант використання алгоритму k-means. Проте з іншої сторони, використання даного алгоритму потребує велику кількість технічних засобів та ресурсів при аналізі та обробці великої кількості даних. Якщо уявити, що дана методика використовується на одному технічному ресурсі (тільки одному сервері, наприклад) який знаходиться в стані атаки, підключати даний алгоритм, який може використати фактично всі можливості ресурсу, буде

нераціональним рішенням. Також слід враховувати специфічність алгоритму. Справа у тім, що даний алгоритм може бути чутливим до викидів, які спотворюють середнє значення. Хоча при аналізі сезонності, ризик реалізації викиду може бути мінімальним, але з іншої сторони, дана особливість може привести до катастрофічних наслідків, в контексті роботи алгоритму.

Для ліквідації негативних наслідків або їх мінімізації необхідно застосовувати модифікацію алгоритму k-середніх, а саме – k-медіани (kmedoids). Даний тип алгоритму менше всього підпорядковюється похибкам та чутливості з точки зору наявності шумів, тому що медіана не піддається на викиди. Хоча алгоритм можна вважати нераціональним у використанні, якщо вибірка даних буде дуже великою.

Оптимальним варіантом із мінімізацією всіх зазначених ризиків буде використання алгоритму CLARA.

Даний алгоритм працює із множиною зразків, які зберігаються у базі даних. Кластеризація використовується для кожного зразку із множини даних і у результаті аналізу вибірки, алгоритм пропонує найбільш успішну кластеризацію. Для проектів, де використовується велика кількість інформації, даний алгоритм буде ефективним. Проте ефективність напряму залежить від специфікації вибірки даних, як зразка для набору даних. У випадку хаотичності передачі даних в мережі, фактичні результати можуть сильно відрізнятися від очікуваних.

З метою аналізу та дослідження різних алгоритмів кластеризації, було розглянуто наступні варіанти:

- Clarans;
- CURE;
- DBScan;
- WaveCluster;

Алгоритм кластеризації WaveCluster надає позитивні результати при аналізі малих об'ємів даних. При аналізі незначних даних, алгоритм не буде чутливим до похибок та шумів і розробляти кластери довільної структури. Проте при нарощуванні кількості проаналізованих даних ефективність WaveCluster сильно знижується.

Алгоритми кластеризації Clarans, CURE, DBScan не були проаналізовані внаслідок специфікації даної роботи і для оброки тестових завдань були неефективними. Оскільки особливість даних алгоритмів – це стартове налаштування певної щільності точок для покращення роботи із невеликими об'ємами даних. За підсумками розгляду поданих вище методів кластеризації, був обраний метод k-means. Даний методи був досліджений на предмет ефективності для вирішення поточної задачі кластеризації.

Для підтвердження гіпотези щодо існування двох класетрів було застосовано візуалізацію отриманих даних. Візуалізація представляє собою двомірну діаграму розсіювання. Зниження розмірності було проведено за допомогою методу компонент.

Наступним етапом для проведення дослідження було встановлення вибору оптимальних ознак для проведення кластеризації. У результаті отриманих даних можна сказати, що найбільшу ефективність роботи кластеризації забезпечують наступні ознаки:

- Джерело запитів;
- Швидкість надходження запитів;
- Сторінка призначення запиту;
- Офлайн дані;
- Браузер клієнта;
- Операційна система клієнта;
- Обсяг переданих і отриманих даних.

Зменшення розмірності вихідного масиву даних, дозволило покращити швидкість роботи кластеризації. Наступним етапом було встановлено вибір оптимальної кількості даних для аналізу. Якщо звернути увагу на те, що дані для аналізу беруться з файлу, і одному запиту може бути підпорядковано кілька записів з цього файлу, (при зверненні до однієї сторінки можуть завантажуватися різні зображення, файли каскадних таблиць стилів скриптів і т.д.), вказати точний інформацію про розмір файлів та даних (в кількості рядків), оптимальних для аналізу, неможливо встановити. У зв'язку з цим оптимальний розмір даних вказується в кількості періодів. Даний процес розглянуто на рисунку 2.1.

Приблизна і ефективна кількість періодів для середньоквадратичного відхилення дорівнює 22. Якщо знизити даний параметр, то отримані результати ймовірно будуть хибними. При збільшенні може відбуватися незначний ріст точності, який при подальшому підвищенні кількості періодів можуть змінюватися спадом, внаслідок зниження чутливості.

У результаті отриманих даних, метод продемонстрував якісну ефективність. Даний алгоритм забезпечує встановлену точність, мінімальне навантаження і швидкість роботи.

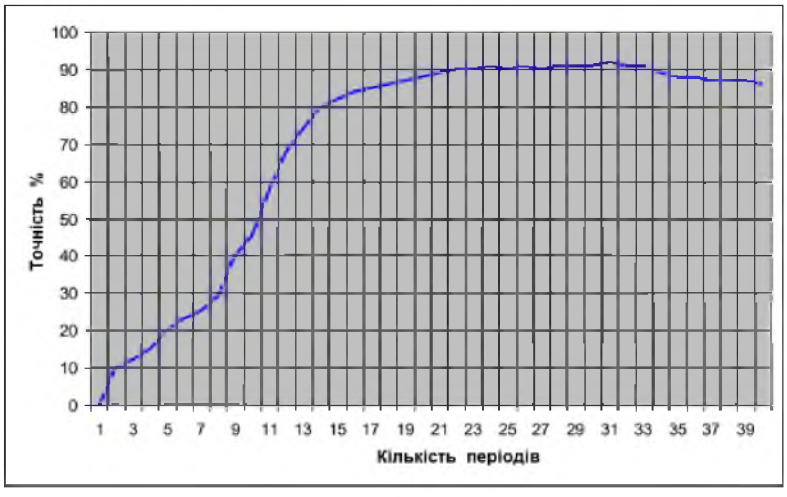


Рисунок 2.1 Графік залежності точності визначення початку атаки від кількості розглянутих періодів.

Для розділу трафіку на надійний і шкідливий вигідним варіантом буде використання алгоритмів k-means. Даний метод дає можливість скористатися заздалегідь при відомому числі кластерів. Метод містить встановлену точність, необхідну для поділу і більш ефективну швидкість роботи у порівнянні з іншими видами алгоритмів.

Основна ідея алгоритму закладається у виділенні окремих кластерів та обчисленні центрів їх мас. На наступних ітераціях буде відбуватися корекція кластерів і перерахування параметрів центрів мас.

$$V = \sum_{i=1}^{k} \sum_{x_j \in S_i} (x_j - \mu_i) \tag{2.18}$$

де k – кількість кластерів, S_i – отримані кластери, i – кількість ітерацій, μ_i – центри мас векторів $x_j \in S_i$.

У результаті роботи алгоритму комбінований трафік буде розподілений на шкідливий і надійний трафік. Внаслідок цього будуть проаналізовані наступні етапи, які доступні для аналізу і обробки:

- Надійний трафік, який надходить до початку атаки T;
- Надійний трафік, виділений із змішаної T*;
- Шкідливий трафік, виділений із змішаної H.

Критерії успішності і аналіз отриманих результатів. Для розрахунку результату ефективності кластеризації можна розглянути формули 2.19-2.23:

$$p_0\gamma = p_1\mu \tag{2.19}$$

$$(\gamma + i\mu)p_i = \gamma p_{i-1} + (i+1) \cdot \mu p_{i+1}, i = 1, \dots, K-2, \tag{2.20}$$

$$(\gamma + (K-1)\mu)p_{K-1} = \gamma p_{K-2} + KN \cdot \mu p_k, \tag{2.21}$$

$$(\gamma^* + KN\mu)p_K = \gamma p_{K-1} + KN \cdot \mu p_{k+1}, \tag{2.22}$$

$$(\gamma^* + i\mu^*)p_i = \gamma^* p_{i-1} + (i+1) \cdot \mu^* p_{i+1},\ i = K-1, \dots, N-1 \tag{2.23}$$

де, γ - інтенсивність навантаження, γL - інтенсивність навантаження, створювана легальними користувачами, S - інтенсивність шкідливого трафіку, μ - інтенсивність звільнення черги запитів, * μ - інтенсивність звільнення черги запитів при активованому фільтрі, E_1, E_2 - помилки першого і другого роду, K - межа активації фільтра, N - обсяг черги запитів, $\gamma = \gamma_L + S$ - навантаження в момент атаки, $\gamma^* = \gamma_L + S(1 - E_2)$ - навантаження при активованому фільтрі.

Оскільки аналіз запитів знижує ефективність і продуктивність серверу, то в окремих ситуаціях інтенсивність звільнення черги запитів може змінюватись. Задля цього можна провести оцінку і записати їх у параметри як Z_μ, де Z – коефіцієнт уповільнення.

При нормалізації $\sum_{i=1}^{N} p_i$ можна отримати ймовірність блокування запиту у формулі 2.24:

$$p_{BLK} = \frac{\frac{1}{N!}\left(\frac{\gamma}{\mu}\right)^{K-1}\frac{\gamma}{\mu^*}\left(\frac{\gamma^*}{\mu^*}\right)^{N-K}}{\sum_{i=0}^{K-1}\frac{\left(\frac{\gamma}{\mu}\right)^i}{i!} + \sum_{i=K}^{N}\frac{1}{i!}\left(\frac{\gamma}{\mu}\right)^{K-1}\frac{\gamma}{\mu^*}\left(\frac{\gamma^*}{\mu^*}\right)^{i-K}} \quad (2.24)$$

Внаслідок цього факту, оцінку ефективності надійного та шкідливого трафіку можна оцінити як $R = (1 - E_1) \cdot (1 - p_B)$.

Завдяки даній формулі було отримано наступні критерії успішності.

Наступний етап алгоритму проводить налаштування отриманих результатів (вибірок) з урахуванням таких факторів як:

- Критерій розмірності отриманих кластерів. Якщо в поточному періоді проаналізувати кількість запитів (n) і надійного трафіку (m), то можна отримати співвідношення, яке буде наближено до n-m. Даний параметр буде справедливий і для різних особливостей мережевої активності (кількість запитів до цільової сторінки, порту тощо);

- Критерій аналогічності надійних вибірок. Максимальна схожість вибірки, яка надходить від надійного трафіку до початку атаки і надійного трафіку, виділеного із змішаного трафіку.
- Критерій відповідності центрів мас. Центр маси надійної вибірки, який виділений із змішаного трафіку, повинен відповідати схожому сезонному періоду надійного трафіку, який був отриманий до початку атаки. Тобто, відстань між зазначеними центрами мас повинна бути наближена до нуля.

Для конкретизації можна знайти ймовірність належності кожного елемента своєму класу. Елементи із мінімальною ймовірністю переносяться в протилежні групи із розрахуванням фактору розмірності груп.

Для пошуку схожості і надійності кластерів і надалі для класифікації запитів, які отримуються у результаті розрахунків, можна скористатися «Байєсовським класифікатором». Для встановлення ймовірності моделі для класифікатора, можна використати умовну ймовірність $p(C|F_1, \dots, F_n)$ над залежною зміною класу C з малою кількістю або класів, що залежать від зазначених змінних $F_1, \dots, F_n$.

Задля вирішення даного питання необхідно проаналізувати теорему Байєса, описану у формулі 2.25:

$$p(C|F_1, \dots, F_n) = \frac{p(C) \cdot p(F_1, \dots, F_n|C)}{p(F_1, \dots, F_n)} \quad (2.25)$$

Умовний розподіл по класовій змінній C можна розрахувати за формулою 2.26:

$$p(C|F_1, \dots, F_n) = \frac{1}{Z} p(C) \prod_{i=1}^{n} p(F_i|C) \quad (2.26)$$

Виходячи із отриманих розрахунків, буде отримано формули 2.27 і 2.28 для надійних і не надійних користувачів:

$$P(T|D) = \frac{P(T)}{P(D)} \prod_{i=1}^{n} P(w_i|T) \quad (2.27)$$

$$P(H|D) = \frac{P(H)}{P(D)} \prod_{i=1}^{n} P(w_i|H) \quad (2.28)$$

В аспекті вибору навчальних розрахунків (вибірок) можна використати множини T і H. Після завершення даного кроку, всі елементи із множини T^* будуть віднесені до категорії шкідливого трафіку та будуть змінені із множиною H згідно зазначених розрахунків, які були описані вище. Дані процедури будуть повторятися до тих пір, поки всі елементи множини T не будуть позначені як надійні, або коли алгоритм досягне максимальної ітеративної межі.

Отримані стартові параметри для визначення надійного і шкідливого трафіку та їх механізми у сучасному стані можна використовувати з різними класифікаторами.

На рисунку 2.2 було продемонстровано фактичні схеми алгоритмів по визначенню початку атаки і виділенню шкідливого трафіку. Перший алгоритм пояснює процедури початку шкідливого трафіку, а наступні два алгоритми визначають початок атаки.

На першому кроці відбувається виклик механізмів, які займаються визначенням сезонних періодів, розрахунком допустимої межі, кількості запитів та початку атаки. У ситуації початку атаки необхідно, щоб алгоритм розділив весь трафік на два кластери де кожний отриманий результат буде розподілятися як шкідливий або надійний трафік. У результаті налаштування кластерів, отримані запити конкретизуються. Як наслідок, після виконання конфігураційних процедур, будь-який отриманий запит буде проаналізовано і віднесено до зазначеного кластеру. [9]

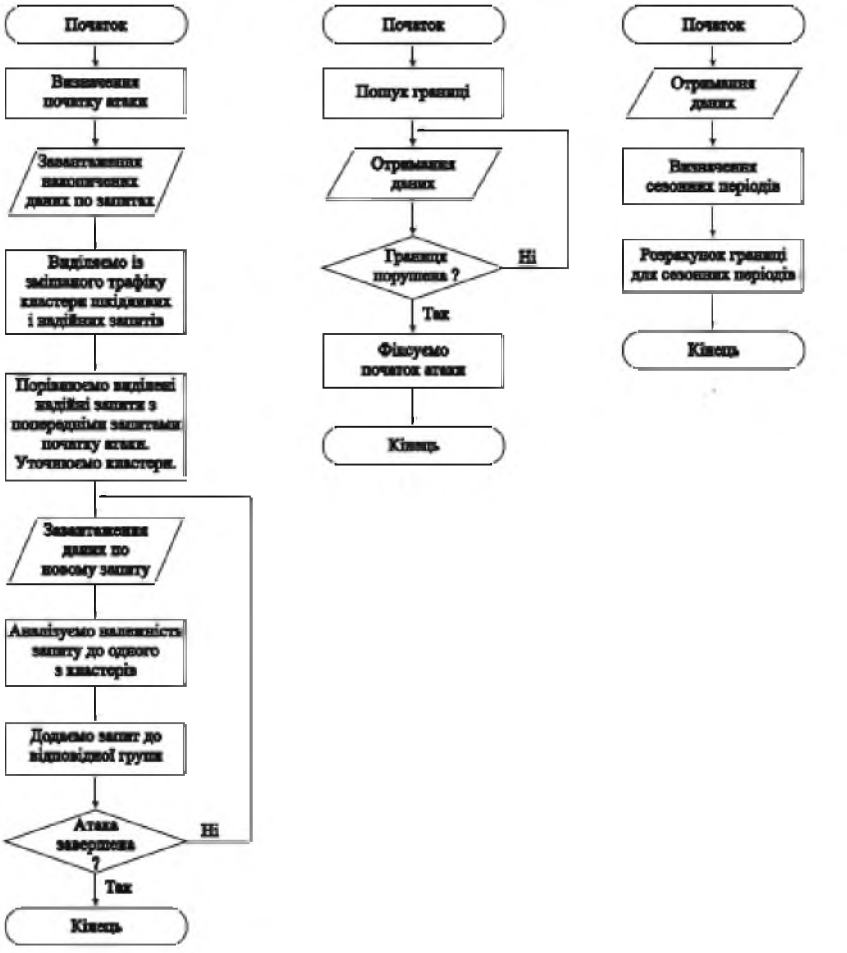


Рисунок 2.2 – Алгоритм по визначенню атаки і шкідливого трафіку

Отже, у контексті аналізу DdoS-атак було запропоновано механізм захисту, який спрямований на виявлення атак відмови в обслуговуванні або шкідливого трафіку. Даний алгоритм враховує наступні пункти:

- Сезонність відхилень та їх навантаження;
- Виявлення сезонних періодів і підтвердження типової сезонності періодів;

Сезонність відхилень та їх навантаження надає можливість отримання результатів виявлення атаки на ранніх стадіях і з найбільшою точністю.

Виявлення сезонів і підтвердження типової сезонності періодів визначає амплітуду надходження шкідливого трафіку, що надає можливість встановити періодичність отриманих атак.

Внаслідок отриманих результатів було визначено тижневу, добову і невизначену сезонність, а також причини їх виникнення.

Задля розподілення гібридного трафіку необхідно використовувати алгоритм кластеризації k-means. Для ефективної роботи алгоритму були підібрані оптимальні параметри та розмірність значень, а також визначені критерії успішності. Розроблені алгоритми складають основу узагальненого методу знаходження атаки відмови в обслуговуванні та шкідливого трафіку. Загальний варіант роботи алгоритму можна описати наступним шляхом:

- За допомогою отриманих даних визначаємо існуючі сезонні періоди;
- Для кожного сезонного періоду встановлюємо максимальну межу кількості запитів;
- У разі порушення межі необхідно встановити точку початку атаки;
- Відносимо весь отриманий трафік до загального кластеру. У даному кластері буде знаходитися весь трафік;
- Класифікуємо трафік на надійний і шкідливий за допомогою алгоритму k-means;
- Аналізуємо трафік, який був визначений на початку атаки з кластером надійного трафіку;
- Коригуємо кластери на підставі отриманих результатів;
- Весь трафік, що надходить, необхідно проаналізувати із урахуванням налаштованих значень. [10]

2.1.6 Критерії успіху. Корекція отриманих кластерів.

Критерії успіху можна розподілити на множини факторів, які впливають на отриманий результат:

- Фільтрація трафіку;
- Зміна межі фільтрації трафіку;
- Сезонність атак та її вплив на межу трафіку.

Метод кластеризації дозволяє фільтрувати дані по спеціальному шаблону (фільтру), завдяки якому будуть розмежовуватися інформація (у нашому випадку – отриманий трафік). В контексті виявлення аномальної кількості запитів, можна враховувати, що запит на ресурс реалізується завдяки прямому посиланню на ресурс. У даному випадку процес розмежовування трафіку здійснюється завдяки факту доступу до ресурсу (тобто, до негативного трафіку вносяться всі запити до ресурсу, які надходять по прямому посиланню).

Межа фільтрації трафіку встановлюється як пункт співвідношення кластерів і змінюється у результаті впливу запитів на систему. Як приклад, фактором, який впливає на межу шкідливого і надійного трафіку можна вважати дії користувачів в системі. Як правило, DdoS-атаки здійснюють прямий запит до ресурсу і не виконують активних дій, які робить користувач. Механізм відслідковування дій користувачів в системі можна розцінювати як межу, яка відділяє надійний трафік від шкідливого. Зміна межі трафіку залежить від фактору наявності користувача в системі (наприклад, чим більше часу проводить користувач в системі, тим ймовірніше даний користувач вважається «живим»). Таким чином, знижується ризик додання запитів користувачів до кластеру негативного трафіку.

Сезонність атак та їх періодичність може впливати на фільтрацію трафіку кластерних систем. Одним із негативних чинників, який першочергово впливає на стан фільтрації трафіку можна вважати непередбачуваність дій зловмисника в момент реалізації атаки. Межа трафіку буде змінюватись хаотично на перших етапах аналізу трафіку, оскільки система не має умовного «портрету» зловмисника і не може точно проаналізувати запити, які надходять до ресурсу.

Даний аспект можна визначити фактором часу і встановлення періодичності атак. На виході система отримує співвідношення тривалості атак та їх сезонності і отримує середнє значення, яке можна враховувати як точку початку атаки. У деяких ситуаціях, даний фактор може визначити технічні можливості зловмисника, що може бути корисним для мінімізації втрат з боку сторони, яка захищається.

Головною проблемою даної технології, можна вважати наявність шаблону, завдяки якому, система буде функціонувати ефективно. Коректна ефективність даної технології залежить від контексту шаблону (сфокусованість даних).

Технології аналізу систем і збору інформації відкривають наступні можливості:

- Дослідження витривалості систем;
- Визначення отриманих результатів, як шаблону навчання алгоритму;
- Вибір шаблону для навчання алгоритму на підставі отриманих результатів;
- Прорахування ефективності систем обробки трафіку;
- Аналіз технічних можливостей системи. [11]

2.2 Методи покращення алгоритму кластеризації на основі k-means

Основне завдання k-середніх полягає у пошуку центру кластерів, які мінімізують внутрішньокласову дисперсію, тобто суму квадратів відстаней від кожної точки даних, яка кластеризується, до її центру кластера. Процес пошуку точного рішення алгоритму k-середніх для довільного введення являється складним. Стандартна версія даного алгоритму надає чіткі і ефективні рішення.

Проте з іншої сторони, даний алгоритм має два основних недоліки:

- Ефективність алгоритму залежить від розміру вхідних даних;
- Проблеми апроксимації.

Алгоритм k-means буде ефективним, якщо система взаємодіє із невеликим обсягом інформації. Для комерційних систем, де існує велика кількість користувачів, швидкість обробки інформації являється головним критерієм ефективності систем захисту. У системі, де циркулює велика кількість трафіку, швидкість фільтрації буде знижена до критичного рівня, що може привести до його неефективності.

Наступна проблема полягає у знайденій апроксимації. Даний параметр може бути як завгодно поганим щодо цільової функції порівняно з оптимальною кластеризацією. Описана подія може призвести до некоректної фільтрації даних. Може виникнути ситуація, коли межа шкідливого і надійного трафіку «вузька», тобто отриманий параметр межі може бути максимальним. Як, наслідок, зміна значення може не відбутися і частина надійного трафіку може бути розцінена алгоритмом, як шкідливий трафік.

При застосуванні стандартної версії алгоритму кластеризації на основі k-means присуті ризики, які негативно будуть впливати на результат отриманої роботи. Приклад використання стандартного алгоритму із використанням довільно обраних точок збору кластерів можна розглянути на рисунку 2.3. Результати показують, що при паралельно віддаленних від центру кластери, не мають змоги «знайти» центр, тобто місце кластеризації даних і як наслідок – алгоритм не враховує отримані результати.

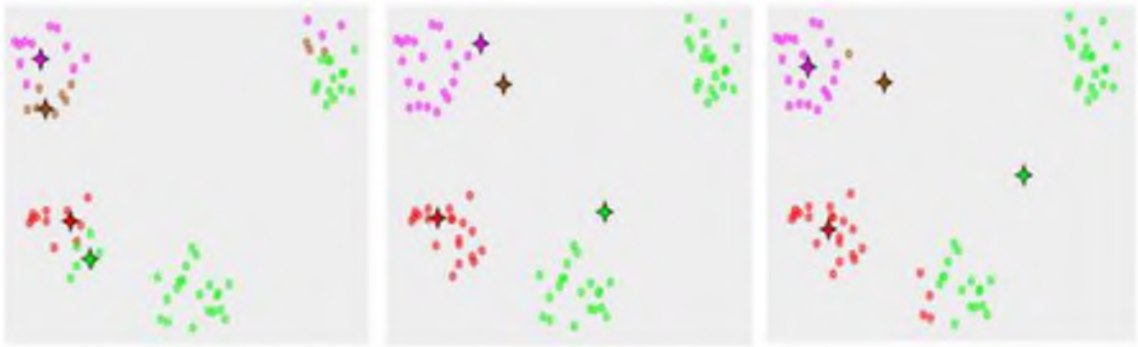


Рисунок 2.3 – Робота алгоритму кластеризації на основі k-means

Алгоритм k-means++ частково вирішує описані проблеми, вказуючи процедуру ініціалізації кластерів перед продовженням стандартних ітерацій оптимізації k-середніх. При використанні модифікованої версії (k-means++) алгоритм гарантовано знайде рішення.

Можна розглянути приклад неефективності стандартного алгоритму. Уявимо, що вибірка містить чотири точки, яка утворює вирівняний прямокутник, ширина якого більше за його висоту.

Якщо k = 2 і два початкових кластери знаходяться у серединах нижніх і верхніх сегментів прямокутника, утвореного чотирма точками даних, алгоритм k-середніх збігається миттєво, без процедури переміщення центрів кластерів. Отже, дві точки даних об'єднуються, а інші дві, що утворюють верхню частину прямокутника, групуються разом. Як наслідок алгоритм створює неоптимальну кластеризацію, оскільки ширина прямокутника більше за його висоту.

Далі буде умовно описано процес розтягнення прямокутника по горизонталі до довільної ширини. Стандартний алгоритм буде продовжувати групувати точки не оптимально, і збільшуючи горизонтальну відстань між двома точками даних у кожному кластері, можна зробити алгоритм неефективним щодо цільової функції k-середніх.

У покращеній версії даного алгоритму, річ полягає у тім, що розподіл k-початкових центрів кластерів являється ефективною процедурою. Перший центр кластера обирається рівномірно випадковим чином із точок даних, які кластеризуються, після чого кожен наступний центр кластера вибирається із решти точок даних, враховуючи ймовірність пропорційності його квадрату від найближчого існуючого центру скупчення точки.

Процес алгоритму описується наступними етапами:

- Вибирається один центр рівномірно випадковим чином серед точок даних;

- Для кожної необраної точки даних x, розраховується $D(x)^2$відстань між x і найближчим центром, який було обрано;
- Вибирається наступна точка, як новий центр, із використанням зваженого розподілу ймовірностей, де точка x обирається із ймовірністю, пропорційною $D(x)^2$;
- Попередні два етапи повторюються, доки k центри не будуть обрані;
- Після налаштування і встановлення позицій, використовується стандартна кластеризація k-середніх.

Даний метод висіву займає час на налаштування стартових точок даних, проте основна частина k-середніх ефективно збігається і як наслідок – алгоритм працює швидше. [12]

Повні порівняння k-means і k-means++ представлені в таблиці 2.2. Зауважимо, що k-means++ стабільно перевершував k-means, досягаючи нижчого потенціалу, в деяких випадках на кілька порядків, а також цикл роботи був ефективніше. Наприклад, метод k-середніх не працює належним чином, оскільки випадковий висів неминуче об’єднує кластери разом і алгоритм ніколи не зможе їх розділити. Обережний метод посіву k-means++ повністю уникає дану проблему і практично завжди досягає оптимальні результати на синтетичних наборах даних.

Оскільки всі перевірені алгоритми уявляються випадковими, було проведено випробування для кожного випадку. Далі необхідно створити мінімум і середній потенціал та середній час, необхідний для завершення роботи алгоритму. Реалізація являється стандартною без спеціальних оптимізацій

Різниця між k-means та k-means++ у реальних наборах даних також досить істотна. Алгоритм із набором даних Cloud k-means++ завершується майже вдвічі швидше, досягаючи потенціалу значення функції приблизно на 20%. Підвищення продуктивності являється більш різким у більшому наборі даних, де

потенційне значення, отримане за допомогою k-means++ ефективніше в 10-1000 разів та на 70% швидше. [17]

Таблиця 2.2 – Результати роботи алгоритму k-means++ із різними вхідними даними

k	Average ϕ		Minimum ϕ		Average T	
	k-means	k-means++	k-means	k-means++	k-means	k-means++
10	135512	126433	119201	111611	0.14	0.13
25	48050.5	15.8313	25734.6	15.8313	1.69	0.26
50	5466.02	14.76	14.79	14.73	3.79	4.21

k	Average ϕ		Minimum ϕ		Average T	
	k-means	k-means++	k-means	k-means++	k-means	k-means++
10	7553.5	6151.2	6139.45	5631.99	0.12	0.05
25	3626.1	2064.9	2568.2	1988.76	0.19	0.09
50	2004.2	1133.7	1344	1088	0.27	0.17

k	Average ϕ		Minimum ϕ		Average T	
	k-means	k-means++	k-means	k-means++	k-means	k-means++
10	$3.45 \cdot 10^8$	$2.31 \cdot 10^7$	$3.25 \cdot 10^8$	$1.79 \cdot 10^7$	107.5	64.04
25	$3.15 \cdot 10^8$	$2.53 \cdot 10^6$	$3.1 \cdot 10^8$	$2.06 \cdot 10^6$	421.5	313.65
50	$3.08 \cdot 10^8$	$4.67 \cdot 10^5$	$3.08 \cdot 10^8$	$3.98 \cdot 10^5$	766.2	282.9

Як шаблон до навчання систем фільтрації трафіку необхідно використовувати дані, які будуть виступати у ролі центрів точок збору даних. До даних центрів можна віднести наступні параметри;

- Кількість користувачів в системі;
- Час, який проводить користувач в системі;
- Дії, які виконує користувач в системі.

Прикладом технології, яка дозволяє отримати базову інформацію про ефективність технічних ресурсів, являється інструмент Jmeter.

2.3 Система Jmeter.

Jmeter представляє собою інструмент для реалізації навантажувальних тестів. Дана технологія дозволяє перевірити витривалість систем, шляхом створення штучного навантаження. Даний пристрій підтримує систему множинного навантаження (для її реалізації необхідно декілька комп'ютерів), для створення додаткових умов перевірки ефективності аналізованих систем.

Дана технологія застосовується спеціалістами, які займаються тестуванням ПЗ та додаткових систем, проте її функціонал можна використовувати і для машинного навчання.

2.3.1 Опис системи

Більшість технологій даного інструменту спрямовані на проведення тестування систем та перевірки їх ефективності і витривалості під час хаотичного або постійного навантаження.

Умовно, технології можна розподілити на дві групи

- Технології, спрямовані на створення документації, визначенню результатів;
- Технології, спрямовані на проведення тестування систем.

Перша категорія представляє собою сукупність організаційно-технічних ресурсів, спрямованих на визначення мети проведення тестування систем, а також планування шляху для проведення тесту. Перед створенням файлу-документації необхідно визначити формат збереження даних (XML або CSV). Для створення документації необхідно ввести наступну команду:

jmeter -n -t file_name.jmx

Дана команда створить файл-документацію для подальшого його редагування із файлом плану тестування і початковими параметрами.

План тестування описує шлях, необхідний програмі при запуску. Повний план тестування буде містити одну або декілька груп потоків, логічних

контролерів, контролерів генерації образів, слухачів, таймерів, систем виявлення помилок та елементів конфігурації.

Додавання нових елементів до плану тестування задається наступним шляхом, який відображається на рисунках 2.4 і 2.5

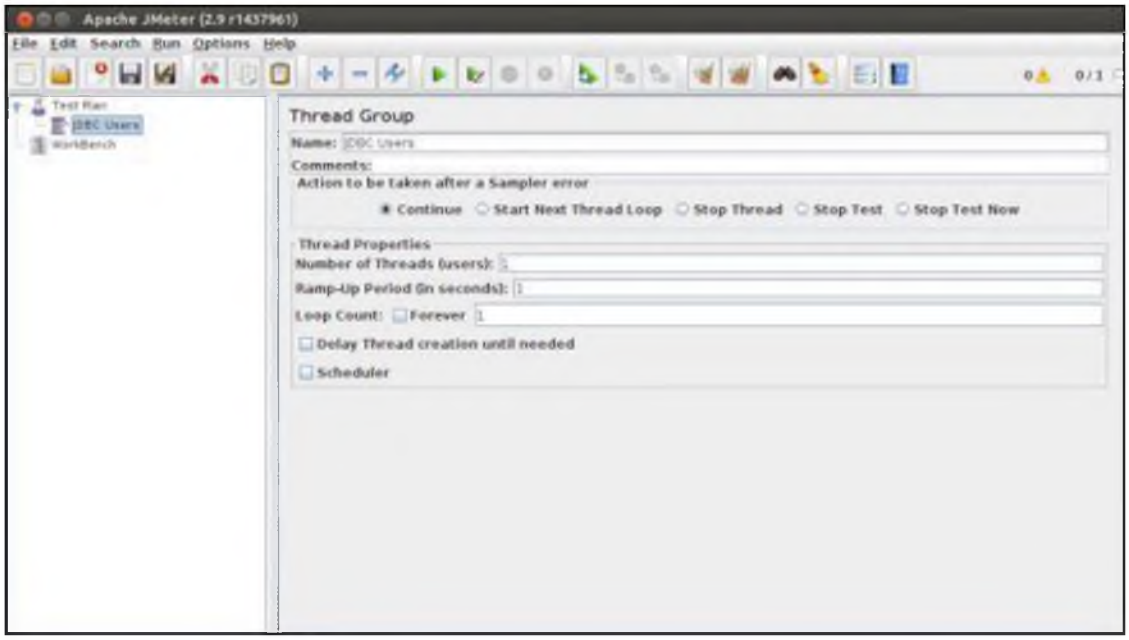


Рисунок 2.4 – Головне меню Jmeter

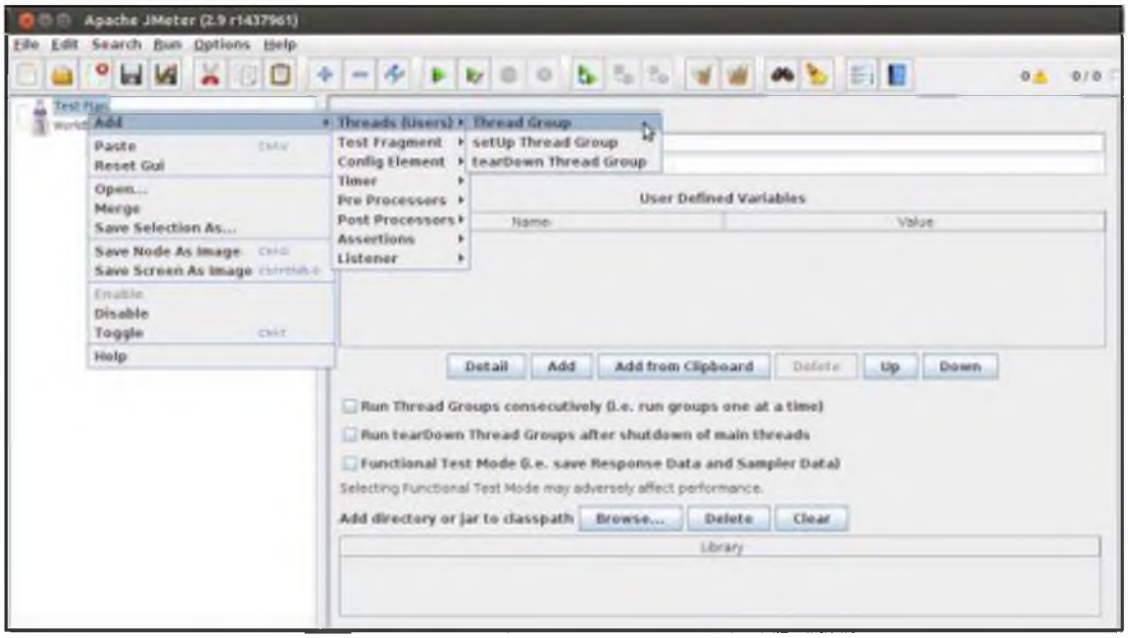


Рисунок 2.5 – Додавання або видалення задач

Група представляє собою набір тестів із використанням протоколу, який присутній в інструментарію Jmeter. Завдяки групам технологія може розподіляти тести для перевірки різних частин програми.

На рисунку 2.6 відображається процес збереження зміненої інформації в інструменті Jmeter.

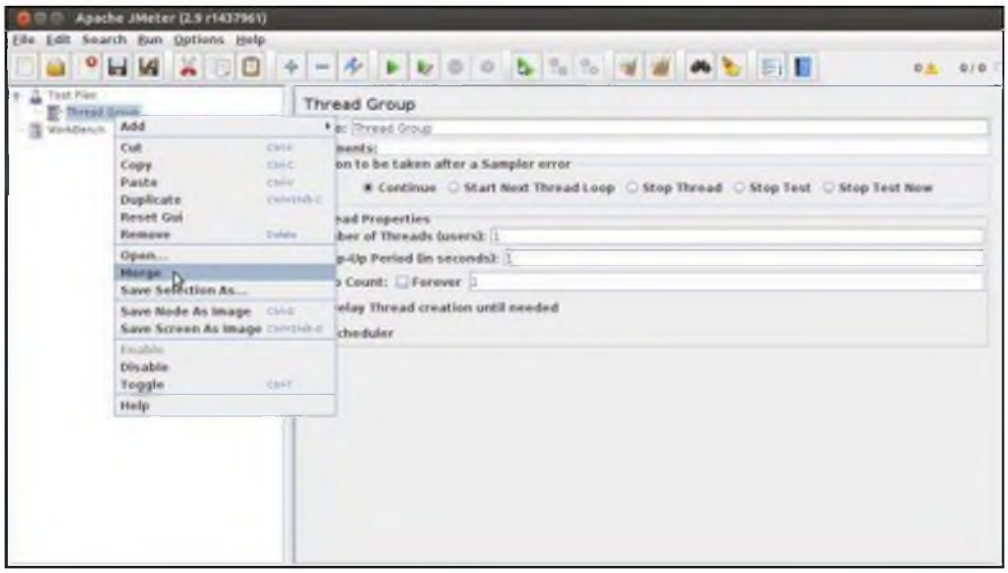


Рисунок 2.6 – Збереження або завантаження задач

Даний аспект дозволяє створити гілку для ініціалізації тестів. У деяких випадках необхідно специфічне налаштування тестів для гілки з метою перевірки реакції системи на запити. Дані елементи дозволяють налаштувати кожен тест в гілці окремо. Детальний процес розглянуто на рисунку 2.7.

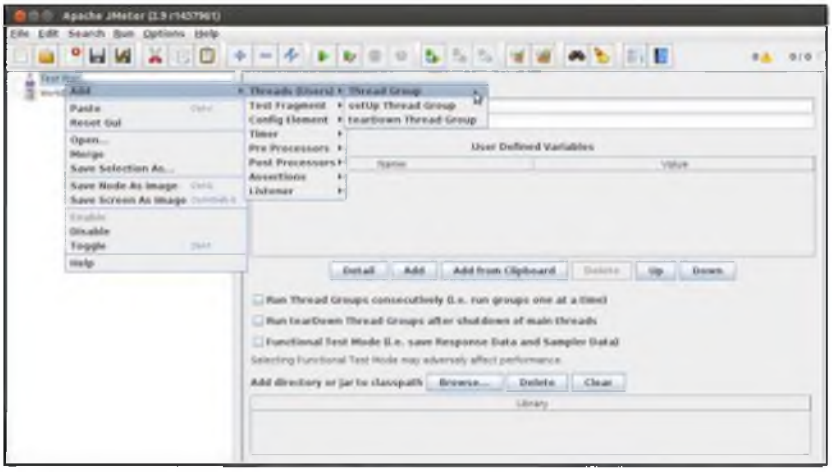


Рисунок 2.7 – Налаштування тесту

Після налаштування систем, необхідно зберегти змінену інформацію та запустити план тестування, згідно рисунку 2.8.

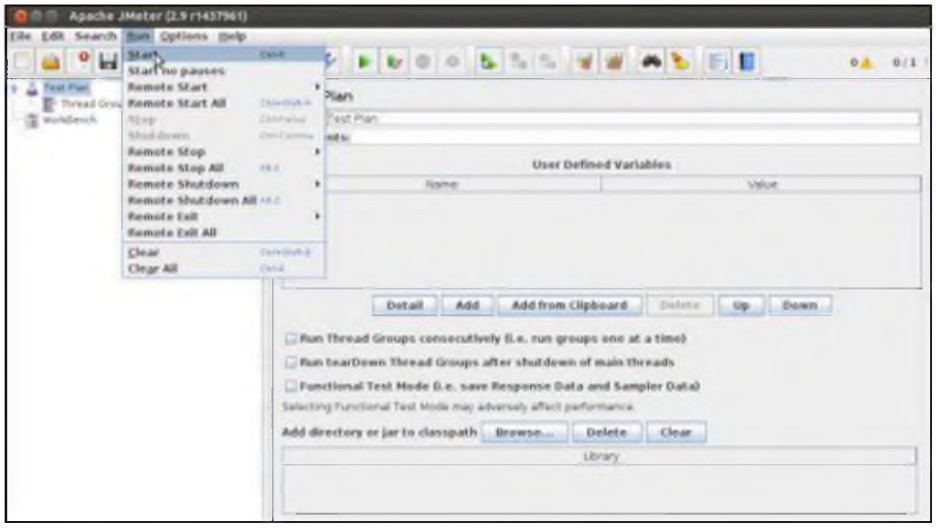


Рисунок 2.8 – Запуск тест-плану

Для відміни всіх процесів запуску плану тестування необхідно натиснути на клавішу Stop (Control + .), або вимкнення працюючого процесу (Control + .). [12]

2.3.2 Технології та механізми

Інструмент Jmeter ефективно взаємодіяти із наступними категоріями протоколів та веб-ресурсів:

- FTP;
- SOAP;
- Протоколи «повідомлень» POP3, IMAP;
- HTTP;
- TCP.

Для реалізації тестування FTP ресурсу необхідно створити план тестування, згідно якого будуть проведені процедури перевірки якості системи. Категорія «Групи тем» в інструменті Jmeter дозволяє створити групу

користувачів, які симулюють запити до системи в залежності від типу протоколу або технології. Детальна інформація відображена на рисунку 2.9.

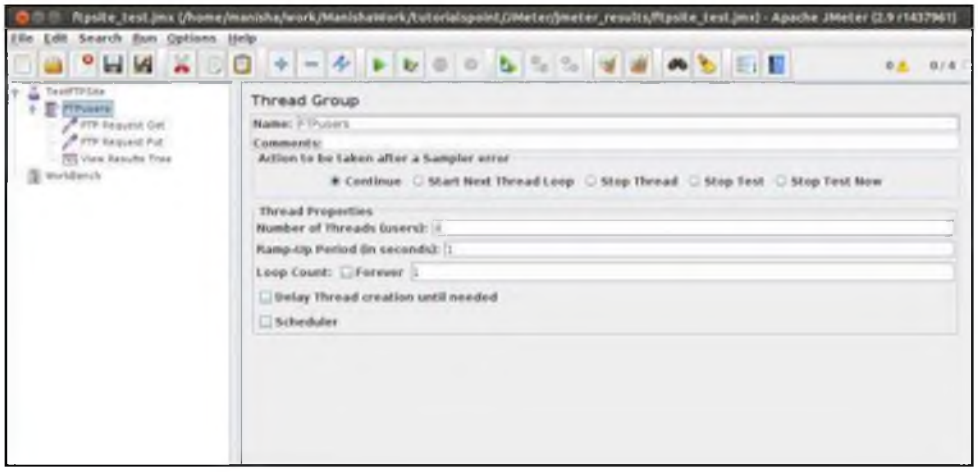


Рисунок 2.9 – Створення групи тем

Для створення групи тем необхідно застосувати наступні параметри:

- Ім’я групи;
- Кількість користувачів;
- Кількість циклів.

Для створення циклу запусків необхідно встановити семплер-параметр. Даний параметр встановлює шаблон запитів (протоколів, веб-технологій) для запуску тестів. У ситуації створення FTP-запиту необхідно заповнити інформацію у наступних полях:

- Ім’я;
- Назва серверу (IP);
- Видалений файл;
- Локальний файл;
- Ім’я користувача;
- Пароль.

Приклад завантажених даних відображено на рисунку 2.10 FTP-запит представляє собою протокол передачі даних у мережі, який використовує

системи та протоколи TCP/IP. Даний протокол використовується для розповсюдження ПЗ і доступу до віддаленних хостів.

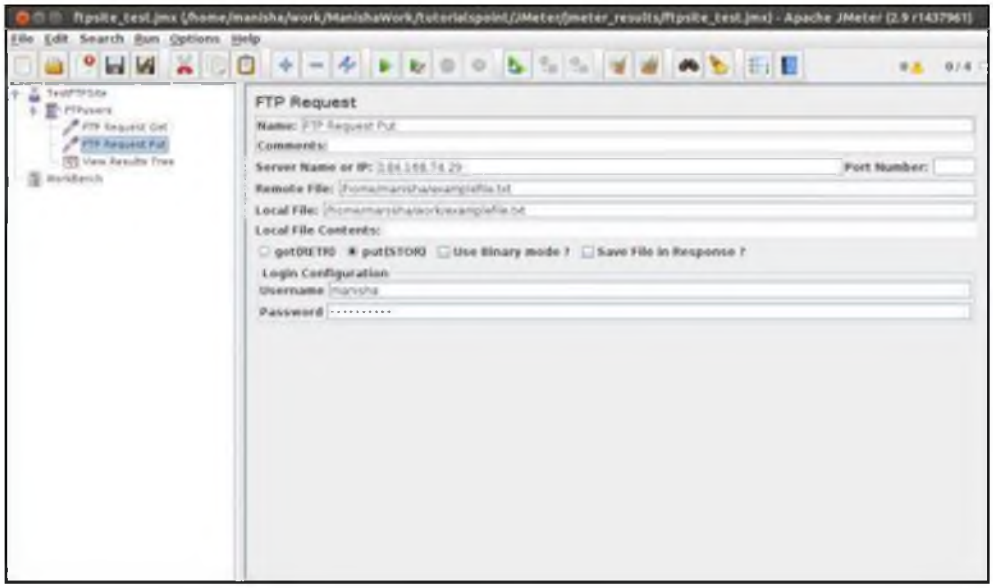


Рисунок 2.10 – Створення FTP-запиту

Далі необхідно створити аналогічний запит із зміненими даними. Інформація, яка була створена, відображається у таблиці 2.1.

Таблиця 2.1 – Інформація для створення FTP-запиту

Ім'я	FTP-Req
IP	184.168.74.29
Видалений сервер	/home/manisha/examplefile.txt
Локальний сервер	/home/manisha/work/examplefile.txt
Статус	STOR
Ім'я користувача	Маніша
Пароль	manisha123

Для завершення процедур налаштування необхідно встановити «прослуховування» процесу. Під даним терміном слід розуміти параметр, який відповідає за збереження всіх результатів ініціалізації FTP-запитів під час роботи інструменту та відображення отриманої інформації у вигляді візуальної моделі даних.

Даний процес детально відображається на рисунку 2.11.

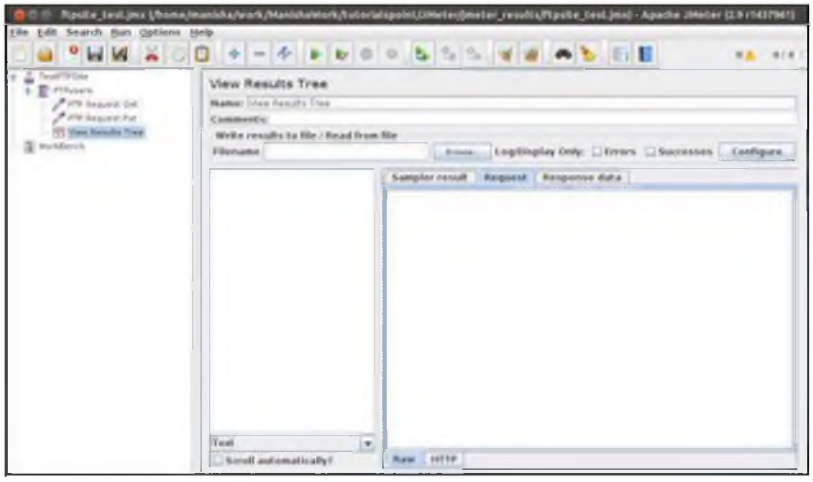


Рисунок 2.11 – Процес налаштування слухачів.

Далі необхідно зберегти всі налаштовані процедури у файлі (у даному випадку вся інформація збережена у файлі ftpsite_test.jmx). Далі ініціалізується процес запуску тестів із використанням різних методів FTP-запиту. Приклади результатів роботи можна розглянути на рисунках 2.12-2.14. У процесі виконання тесту використовувались PUT та GET методи. Всі отримані результати можна розглянути у створених об’єктах-слухачах.

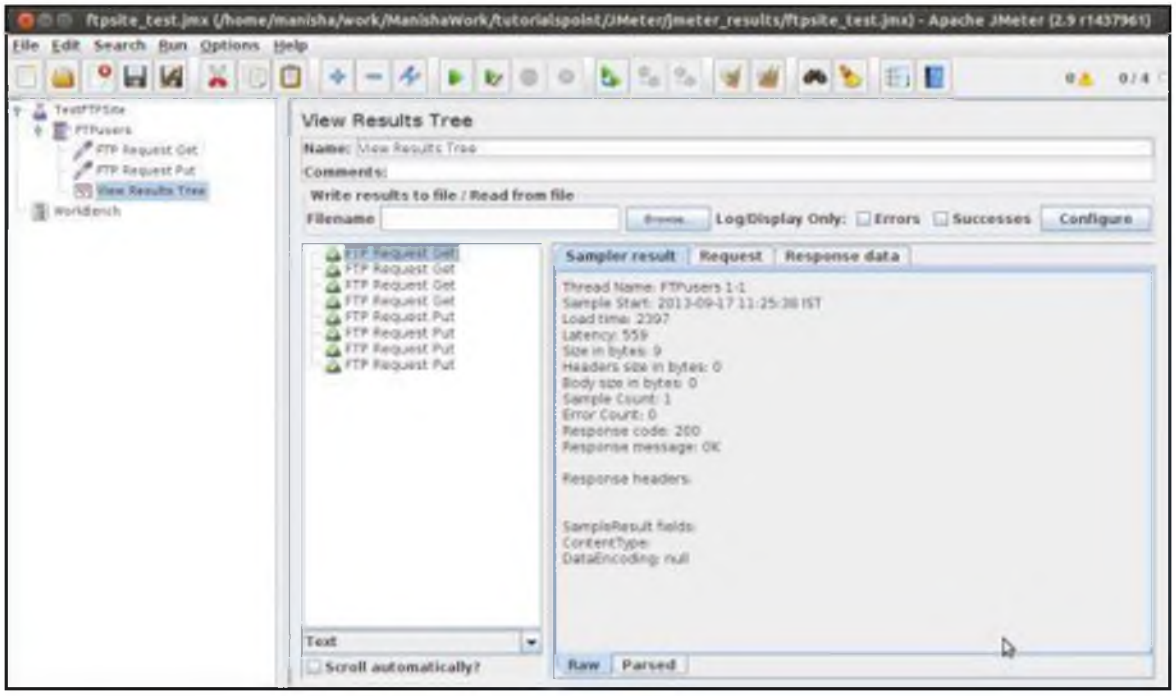


Рисунок 2.12 – Результат роботи при використанні GET-методу

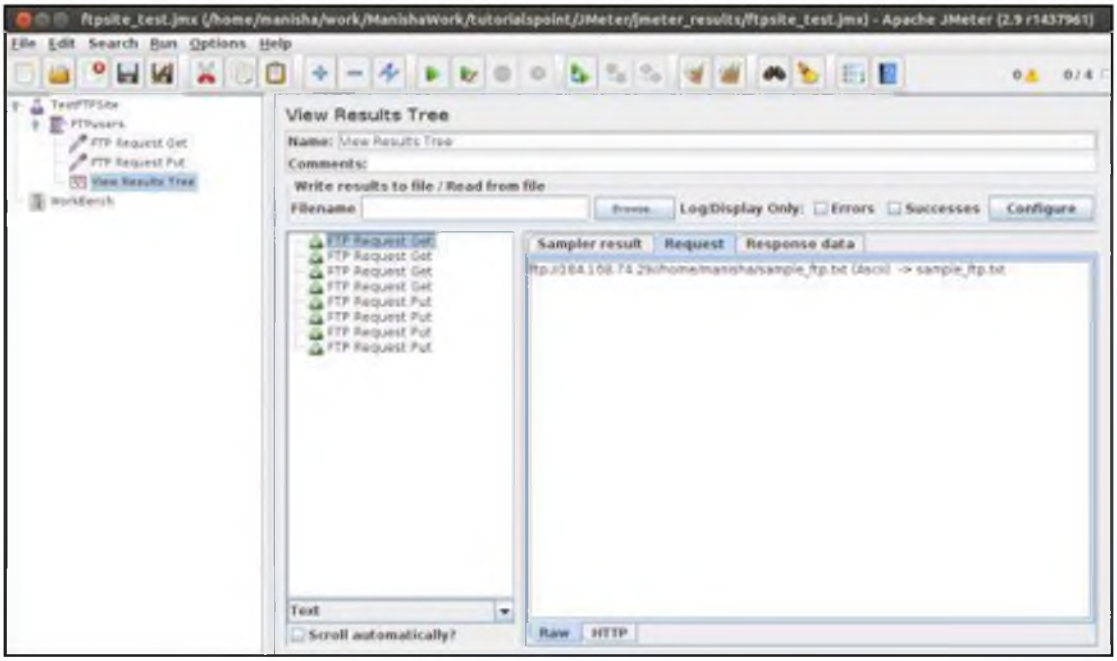


Рисунок 2.13 - Результат роботи при використанні GET-методу (інший тест)

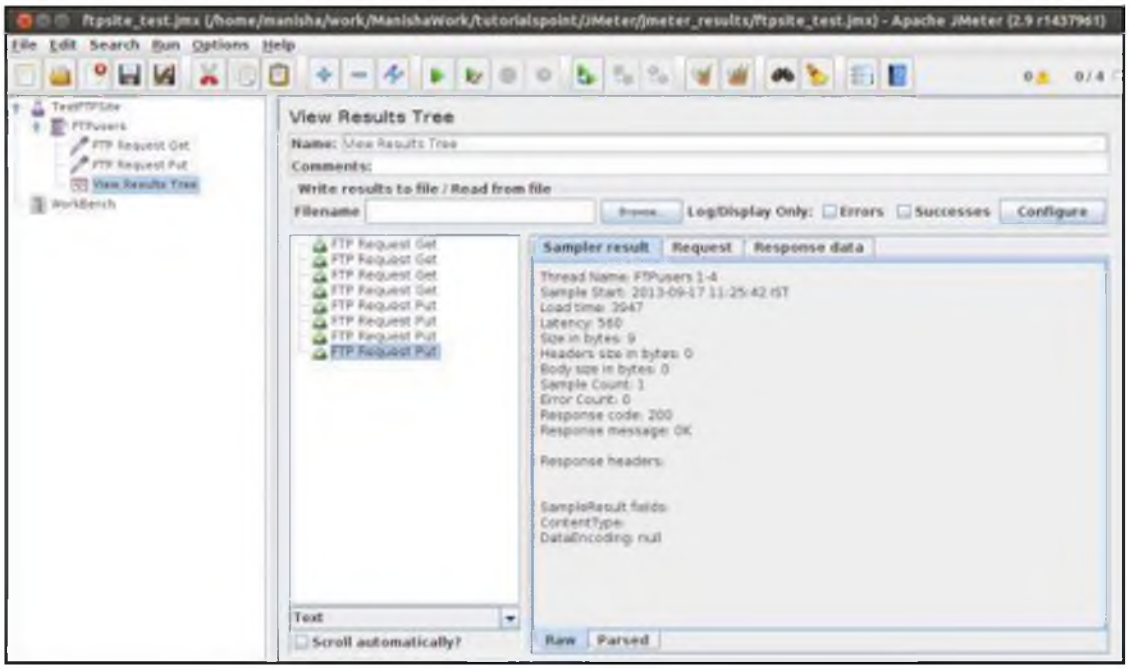


Рисунок 2.14 - Результат роботи при використанні PUT-методу

Всі отримані дані після запуску тестів можна розглянути по шляху path_to_apache/bin (Для PUT запитів дані зберігаються у окремому файлі).

Для створення проекту веб-сервісу необхідно використовувати спеціальну фреймворк систему Eclipse IDE. Для початку реалізації систем перевірки веб-сервісу необхідно створити інтерфейс кінцевої точки служби SomeExample у пакеті com.tutorialspoint.cws. У файлі SomeExample.java код має наступний вигляд, який відображається на рисунку 2.15

```
package com.tutorialspoint.ws;

import javax.jws.WebMethod;
import javax.jws.WebService;
import javax.jws.soap.SOAPBinding;
import javax.jws.soap.SOAPBinding.Style;

//Service Endpoint Interface
@WebService
@SOAPBinding(style = Style.RPC)

public interface HelloWorld {
   @WebMethod String getHelloWorldMessage(String string);
}
```

Рисунок 2.15 – Код файлу SomeExample.java

Умовний веб-сервіс містить метод getHelloWorldMessage, який може приймати параметр String.

Для подальшої роботи необхідно створити клас реалізації AnotherExample.java у пакеті com.tutorialspoint.ws. Повний код детально можна розглянути на рисунку 2.16

```
package com.tutorialspoint.ws;

import javax.jws.WebService;

@WebService(endpointInterface="com.tutorialspoint.ws.HelloWorld")
public class HelloWorldImpl  implements HelloWorld  {
   @Override
   public String getHelloWorldMessage(String myName) {
      return("Hello "+myName+" to JAX WS world");
   }
}
```

Рисунок 2.16 – Код файлу AnotherExample.java

Далі необхідно опублікувати веб-сервіс у локальному режимі, створивши видавця Endpoint та встановити сервіс на сервері.

Метод публікації фіксує два параметри:

- Строка URL кінцевої точки;
- Клас реалізатору, який представляється як веб-служба у кінцевій точці, яка ідентифікується як URL-параметр.

Далі необхідно створити файл-публікатор ExamplePublisher.java, який буде виконувати функцію кінцевої точки (Endpoint). Код приведено нижче на рисунку 2.17 Даний файл представляє собою вихідний елемент у результаті підключення користувача до веб-сервісу. Даний файл дозволяє отримати певний результат при виконанні запиту (наприклад, статус підключення, IP тощо).

```
package com.tutorialspoint.endpoint;

import javax.xml.ws.Endpoint;
import com.tutorialspoint.ws.HelloWorldImpl;

public class HelloWorldPublisher {
   public static void main(String[] args) {
      Endpoint.publish("http://localhost:9000/ws/hello", new HelloWorldImpl());
   }
}
```

Рисунок 2.17– Код файлу ExamplePublisher.java

Наступним етапом необхідно налаштувати файл web.xml. Даний документ буде зберігати інформацію про статус і процеси запиту, версію XML, мови програмування, а також дані, які будуть передаватися при реалізації тестових запитів. Описана інформація у даному документі дозволяє «користувачам» надсилати заповнені запити. У комерційних структурах даний процес поглиблюється з метою встановлення імітації дій «живого» користувача в системі. Запити описуються реалістично із урахуванням можливостей користувача щодо ведення діяльності у системі, яка тестується. Елемент XML файлу являє собою команду, яка виконує певний набір дій. Наприклад, команда <session-timeout> дозволяє зчитати виділений час, необхідний запиту для підключення до мережі. У результаті відсутності можливості підключення, дана команда надає час, який встановлюється користувачем. У деяких випадках дану команду коректно використовувати у ситуаціях, коли процес підключення до серверу може займати велику кількість часу, або система знаходиться під навантаженням. Описані процеси і кодову базу процесу створення XML-файлу детальним шляхом було відображено на рисунку 2.18.

```
<?xml version = "1.0" encoding = "UTF-8"?>
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems,
   Inc.//DTD Web Application 2.3//EN" "http://java.sun.com/j2ee/dtds/web-app_2_3.dtd">

<web-app>
   <listener>
      <listener-class>
         com.sun.xml.ws.transport.http.servlet.WSServletContextListener
      </listener-class>
   </listener>

   <servlet>
      <servlet-name>hello</servlet-name>
      <servlet-class>com.sun.xml.ws.transport.http.servlet.WSServlet</servlet-class>
      <load-on-startup>1</load-on-startup>
   </servlet>

   <servlet-mapping>
      <servlet-name>hello</servlet-name>
      <url-pattern>/hello</url-pattern>
   </servlet-mapping>

   <session-config>
      <session-timeout>120</session-timeout>
   </session-config>

</web-app>
```

Рисунок 2.18 – Код файлу web.xml

Задля відтворення додатку, як веб-сервісу необхідно створити файл конфігурації sun-jaxws.xml. Код описано на рисунку 2.19.

```
<?xml version = "1.0" encoding = "UTF-8"?>
<endpoints
   xmlns = "http://java.sun.com/xml/ns/jax-ws/ri/runtime"
   version = "2.0">

   <endpoint name = "HelloWorld"
      implementation = "com.tutorialspoint.ws.HelloWorldImpl"
      url-pattern = "/hello"/>
</endpoints>
```

Рисунок 2.19. – Код файлу sun-jaxws.xml

Даний код виконує можливість підключеня користувача до системи, заначеної в команді Endpoint. Як наслідок, до URL додається елемент, який вноситься користувачем. У даному випадку, до URL додається /hello/ значення. Тобто, запит буде надсилатися із урахуванням додаткової інформації, зазначеної в Endpoint (У даному випадку, www.tutorialspoint.com/hello/) Після налаштування і створення файлів, структура в інструменті Jmeter буде виглядати у наступному вигляді, як на рисунку 2.20.

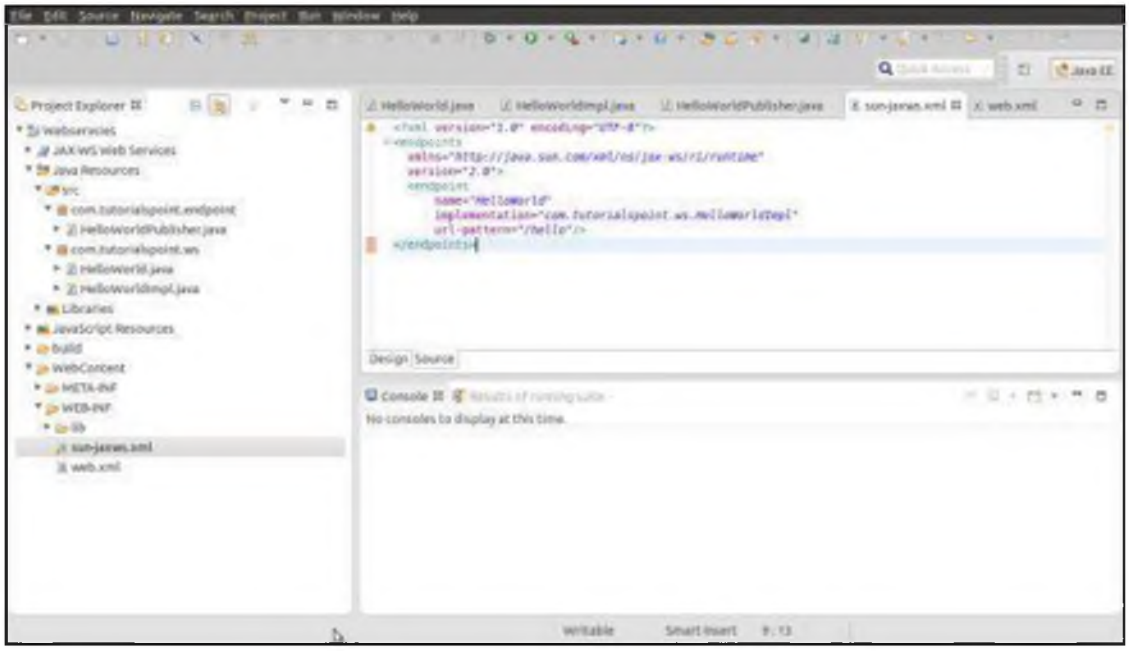


Рисунок 2.20 – Результат роботи після налаштування і створення файлів

Далі необхідно створити WAR-файл і зберегти його в серверній системі Tomcat. Після фінального налаштування, для підключення до серверу необхідно ввести https://localhost:8000/example/example. [13]

Задля початку процесу тестування системи необхідно створити план тестування в Jmeter і створити вузол. У даній ситуації, було створено вузол Webservicetest. Детальну інформацію можна розглянути на рисунку 2.21.

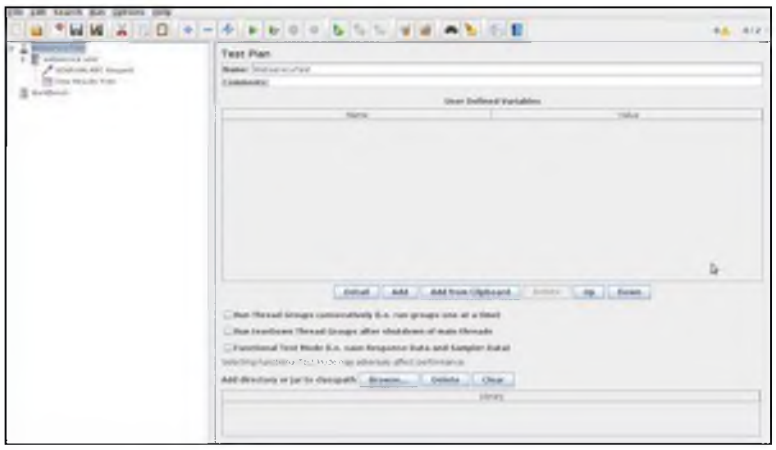


Рисунок 2.21– Вузол Webservicetest

Наступним етапом необхідно створити групу тем із використанням даних. Необхідно створити потік даних, який буде заповнювачем для всіх інших елементів, таких як Samplers, Controllers та Listeners.

Кількість запитів і користувачів можна розглянути у таблиці 2.3.

Таблиця 2.3 – Тестові дані для вузлу Webservicetest

Ім'я	Користувач веб-сервісу
Кількість користувачів	2
Кількість циклів	2

Детальну інформацію можна розглянути на рисунку 2.22. Кількість користувачів може варіюватися (для даного тестового прикладу було взято двох користувачів). Чим більше користувачів використовується для тестування, тим більше навантаження надходить до системи. Кількість циклів впливає на повторюваність запитів після проходження попередніх.

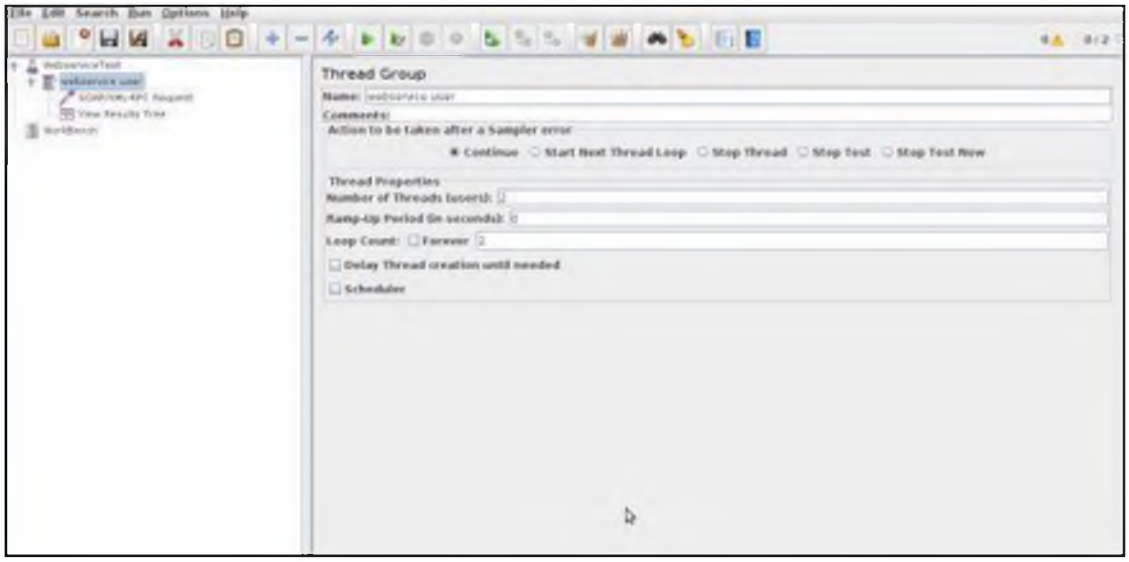


Рисунок 2.22 – Налаштування групи

Задля початку тестування необхідно створити запит до SOAP-сервісу. Запит SOAP/XML-RPC дозволяє створити запит та отримати результат у вигляді XML-форми. Для створення даного запиту необхідно застосувати наступні параметри:

- Ім’я запиту;
- URL;
- Інформація про запит.

Детальну інформацію про запит можна розглянути на рисунку 2.23

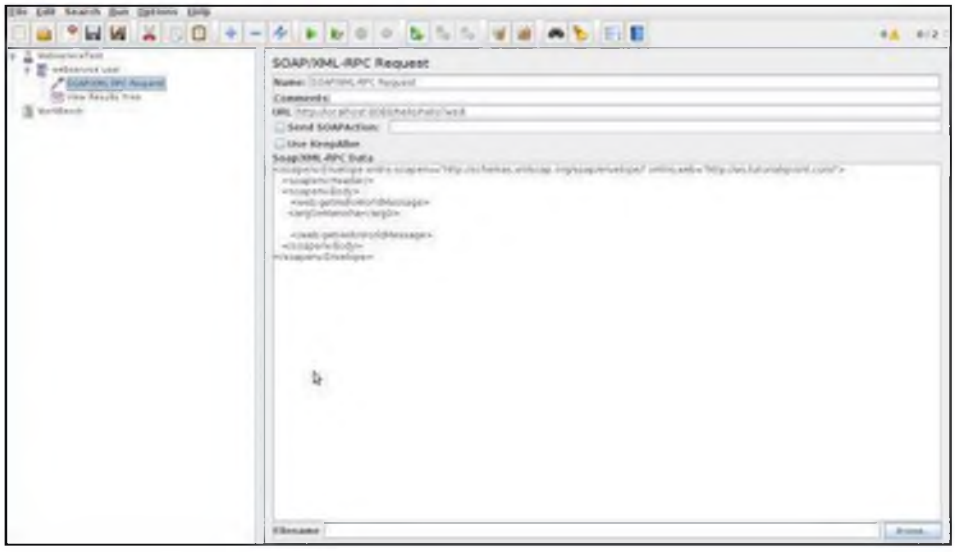


Рисунок 2.23 – SOAP/XML-RPC запит

Дану систему розподілення можна використовувати, як підсистему, для навчання алгоритму нечіткої логіки, в контексті внесення технології фільтрації трафіку. У результаті отримання невірної інформації (тобто, коли запит було створено невірно і повертаються помилки), пеленгатор відсіює всі невірні результати і повертає інформацію тільки про виконані запити. Система Jmeter дозволяє отримувати результати виконаних тестів та запланованих тестових завдань, незалежно від типу протоколу, налаштування системи, схеми реалізації тестових планів, використання BeanShell та інших засобів, які присутні у інструменті Jmeter. План налаштування системи слухачів та результати активованих запитів було продемонстровано на рисунку 2.24.

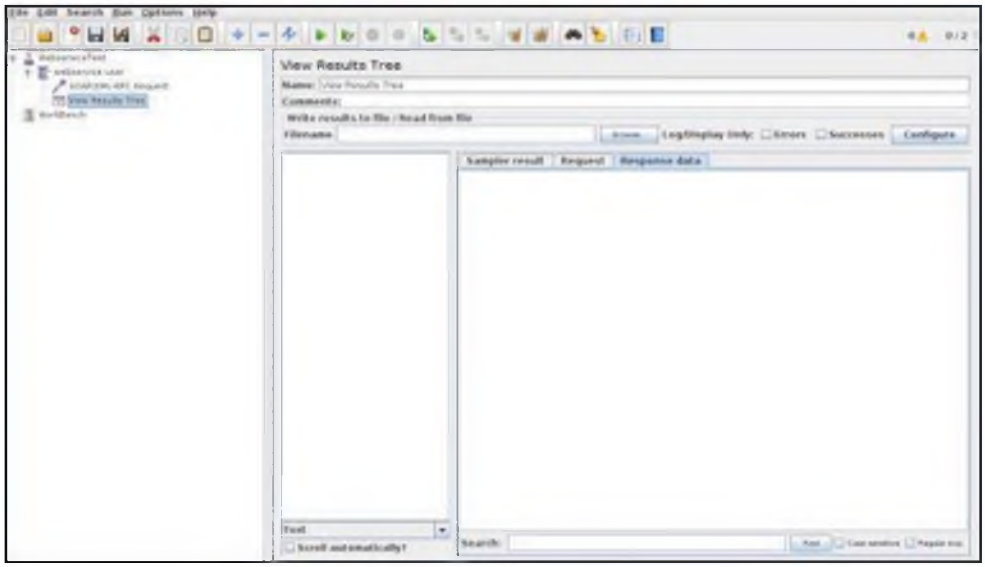


Рисунок 2.24 – Дерево результатів і процес створення слухачів.

У даному експерименті було створено та використано https://localhost:8000/example/example? Wsdl, як тестовий параметр. Далі необхідно створити слухача, який буде виконувати роль пеленгатору HTTP-запитів та зберігати отримані дані після роботи трафіку. У даному випадку, слухач необхнідний для «виловлювання» HTTP-запитів, оскільки у даному прикладі використовуються HTTP та HTTP-подібні запити, які не являються коректними. Пеленгатор і слухач виконують дві головні ролі, а саме – прослуховують весь трафік та фільтрують отримані запити на вірні і невірні.

Після запуску тестування із використанням описаних вище значень було отримано наступні результати. Дані результати зазначені у слухачах та відображені на рисунках 2.25-2.27.

Не виключним варіантом можна вважати спосіб відображення отриманих даних, оскільки для створення вибірки даних даний процес може бути конкретизованим, наприклад XML-файли можуть бути достатньо великими за обсягом, тому при отриманні інформації про трафік користувача необхідно виділяти тільки спеціальну інформацію для навчання вибірки, наприклад IP або дані про користувача.

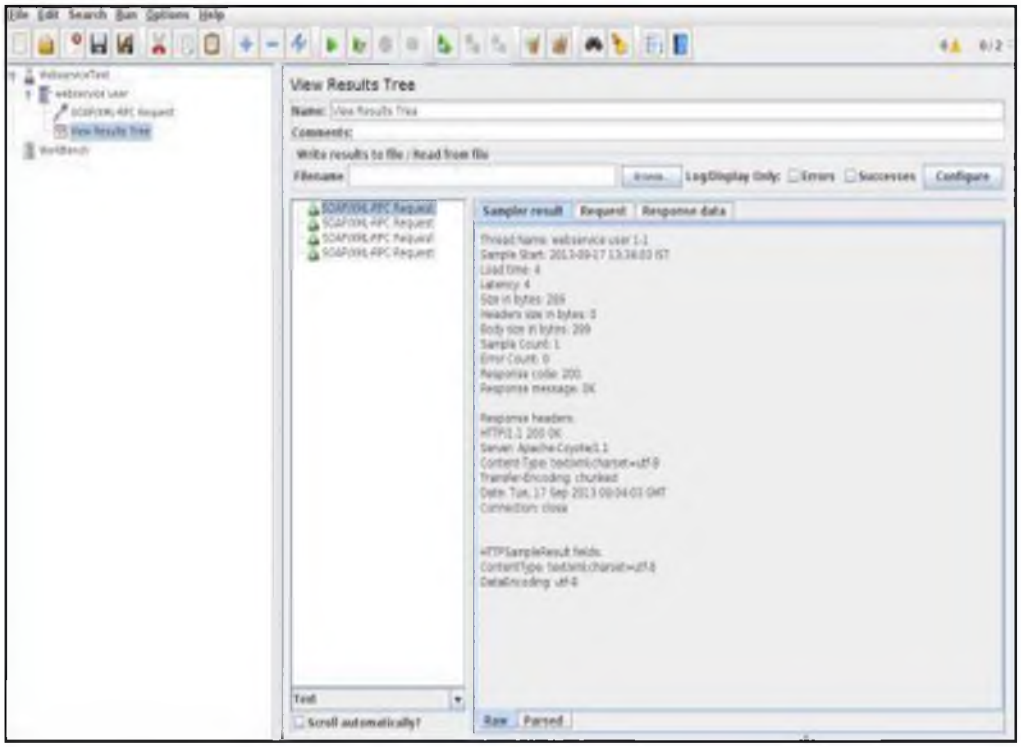


Рисунок 2.25 – Результат роботи при надсиланні XML-RPC запиту

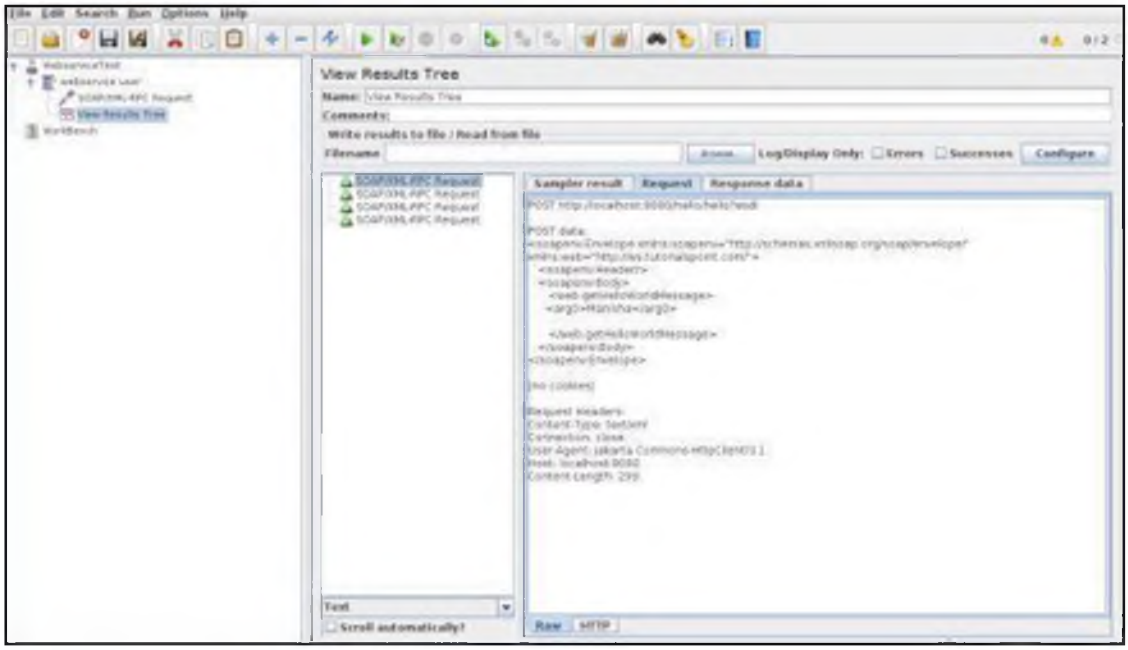


Рисунок 2.26 – Результат роботи при надсиланні XML-RPC запиту (2)

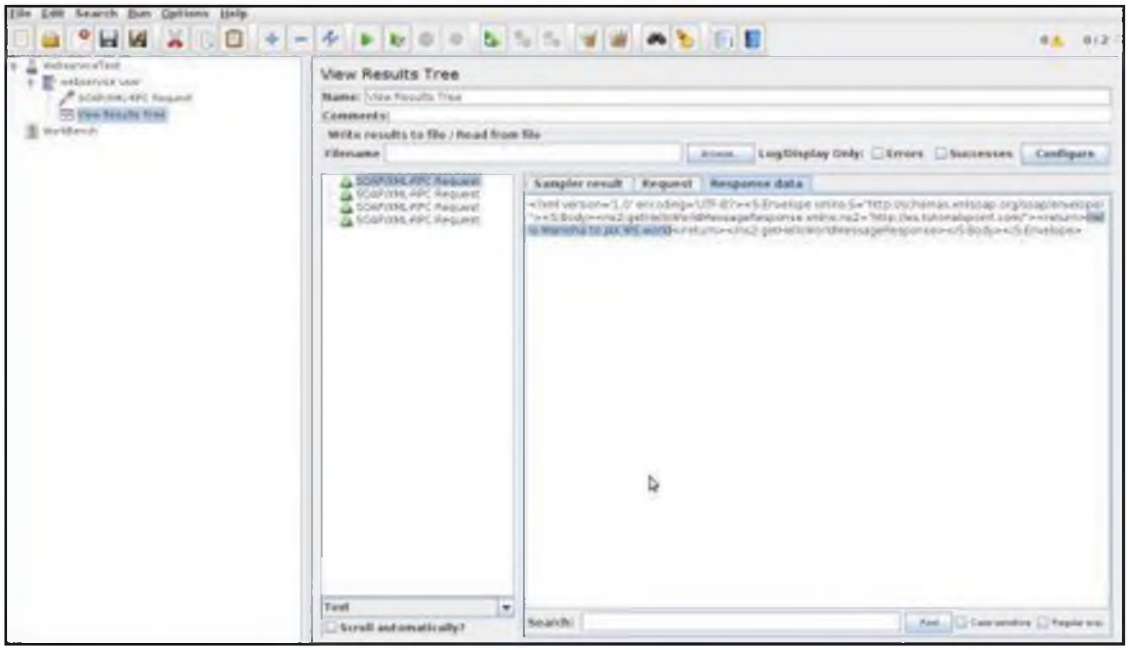


Рисунок 2.27 – Результат роботи при надсиланні XML-RPC запиту (3)

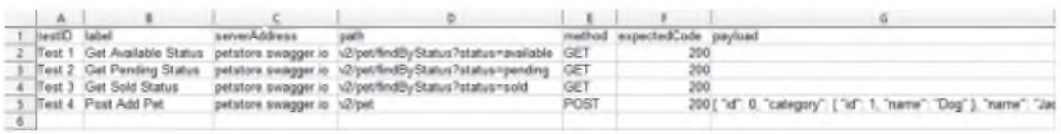
При виконанні останнього запиту можна розглянути повідомлення «Hello Manisha to JAX WS world». [14]

HTTP-запити використовуються при роботі із мережевими сервісами або сервісами взаємодії додатків та їх ресурсів. Першочергово, необхідно створити план тесту та встановити стратегію ініціалізації тестових процесів. Необхідно встановити функції і параметри, виходячи із потреб до реалізації тестів. До таких функцій можна віднести:

- Групи потоків;
- Логічні контролери;
- Таймери;
- Препроцеси і Постпроцеси;
- Елементи конфігурації

Для виконання HTTP-запитів необхідна робота хоча б одного потоку. Далі буде встановлено конфігураційну функцію CSV. Дана опція необхідна для створення формату збереження файлів. Особливості даного формату дозволяють вибирати інформацію у табличній формі. Необхідність використання даної

функції опціональна, проте її можливості дозволяють створити структуру збереження, що у деяких випадках може пришвидшити швидкість відправлення запитів. Детально процес відображення інформації можна розглянути на рисунку 2.28.



	A	B	C	D	E	F	G
1	testID	label	serverAddress	path	method	expectedCode	payload
2	Test 1	Get Available Status	petstore.swagger.io	v2/pet/findByStatus?status=available	GET	200	
3	Test 2	Get Pending Status	petstore.swagger.io	v2/pet/findByStatus?status=pending	GET	200	
4	Test 3	Get Sold Status	petstore.swagger.io	v2/pet/findByStatus?status=sold	GET	200	
5	Test 4	Post Add Pet	petstore.swagger.io	v2/pet	POST	200	{ "id": 0, "category": { "id": 1, "name": "Dog" }, "name": "Ja
6							

Рисунок 2.28 – Формат збереження даних CSV

Завдяки елементу конфігурації Data CSV Config можна налаштувати порядок запуску потоків та вибирати із CSV файлу збережені дані (у конкретній ситуації – вибір запитів) наступні параметри:

- Test ID (ID тесту);
- Label (Інформація про тест);
- serverAddress (Адрес серверу);
- path (Шлях до подачі запиту);
- method (Метод HTTP запиту);
- expected code (Очікуваний код-статус);
- payload (Інформація необхідна для запуску тесту).

Отримані результати можна оглянути на рисунку 2.29.

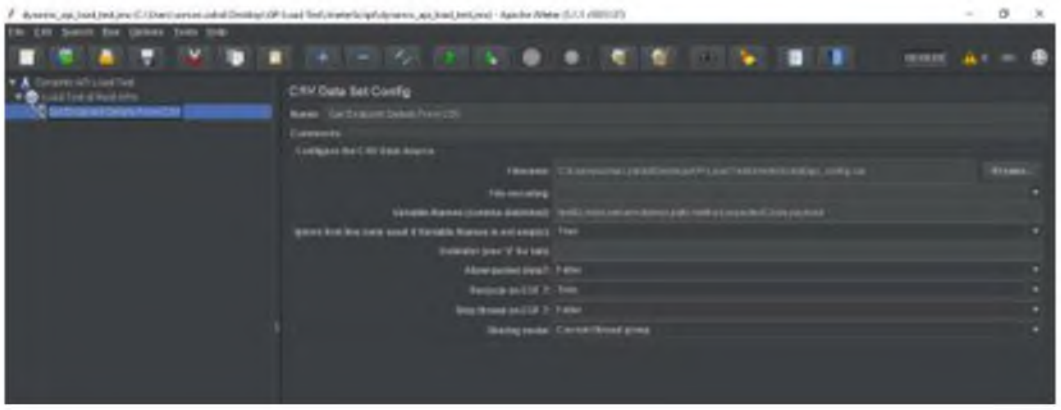


Рисунок 2.29 – Налаштування конфігурації CSV Data set

У процесі налаштування необхідно вказати шлях до даного файлу разом із іменами змінних, які передаються і зберігаються у файлі у форматі назв колонок.

Для коректної роботи HTTP Sampler необхідно використати інструмент Transaction Controller. Даний інструмент дозволяє контролювати процес надсилання запитів, оскільки дана архітектура побудування тестів вимагає декілька кінцевих точок згідно порядку, встановленому у CSV-файлі. Важливим елементів вважається встановлення імені згідно назв колонок. У даному прикладі було встановлено ${testID} — ${_ThreadGroupName}/(${label}. Процес налаштування було відображено на рисунку 2.30.

Рисунок 2.30 – Налаштування Transaction Controller

Для активізації запитів у HTTP Sampler, які будуть виконуватися паралельно, необхідно налаштувати процес запуску згідно змінних, встановлених у CSV-файлі. Дану процедуру можна розглянути на рисунку 2.32.

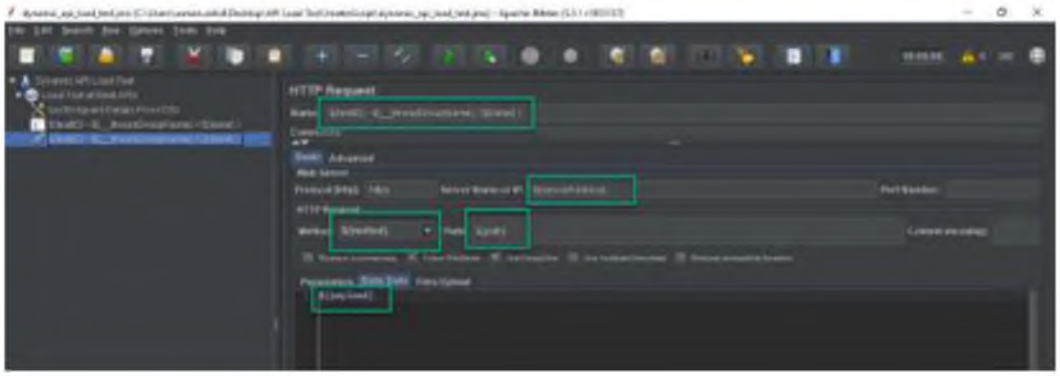


Рисунок 2.31 – Налаштування HTTP Sampler

Для виконання кінцевих точок POST нпотрібно передати Payload у форматі JSON із заголовками запитів для створення або оновлення будь-якого об'єкту.

Для реалізації процесу необхідно визначити Content_Type для JSON payload в HTTP Header Manager, встановивши значення як application/json. У супротивному випадку, payload буде переданий у вигляді текстового значення, що призведе до поганого запиту під час виконання кінцевої точки API. На рисунку 2.32 зображено процес налаштування payload.

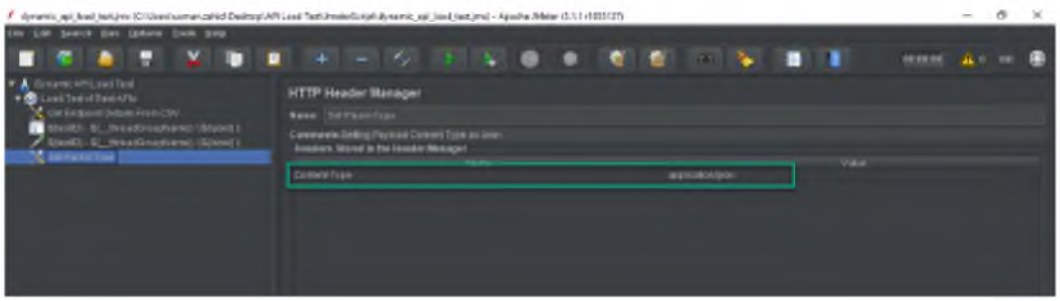


Рисунок 2.32 – Процес налаштування Payload

У тесті продуктивності можна застосувати затвердження на основі наданих бенчмарків. Дані твердження можна застосувати в тілі відповіді, коді відповіді, даних запиту тощо, але в даному випадку у тесті результат виходить у коді відповіді тільки щодо очікуваного коду відповіді, заданого в файлі CSV для кожної кінцевої точки. Процес налаштування зображено на рисунку 2.33.

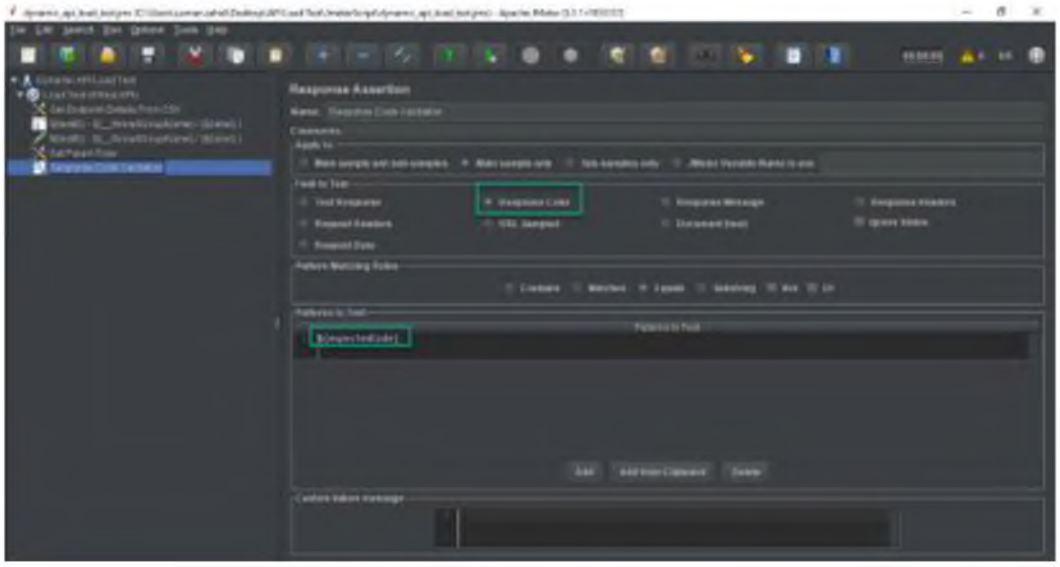


Рисунок 2.33 – Налаштування тестів продуктивності.

Далі необхідно налаштувати елемент Teardown Thread Group. Це особлива форма групи потоків, що використовується для здійснення необхідних дій після завершення виконання звичайної групи потоків. Поведінка потоків, зазначених під Teardown Thread Group, не відрізняється від стандартної.

У даному випадку, описана група потоків використовується для генерації HTML звіту. Відповідь можна отримати, використовуючи семплер BeanShell. Описані процеси відображені на рисунку 2.34.

Beanshell підтримує синтаксис Java і розширює його скриптовими можливостями, такими як: слабкі типи, команди та замикання методів. У даному процесі використовується Beanshell для створення HTML звіту, виконуючи команди cmd.

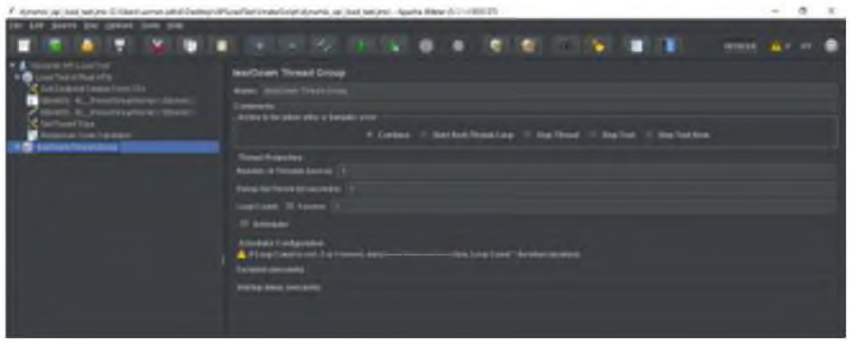


Рисунок 2.34 – Налаштування Teardown Thread Group

Дані команди містять URL-адреси директорій, що надходять із файлу конфігурації CSV, що надає змогу прочитати результати CSV та розмістити HTML звіт. Можливості даної технології дозволяють створити спеціальний скрипт, який створює стислий файл HTML звіту з поточною часовою міткою. Код можна розглянути на рисунку 2.35:

```
cmdObject = Runtime.getRuntime().exec("C:/Windows/System32/cmd.exe /c
${root_dir} & cd ${JMeterDir} & jmeter -g ${result_log_path} -o
${report_dir} & jar -cMf
${compress_report_dir}HTMLReport_%time:~0,2%%time:~3,2%%time:~6,2%_%d
ate:~-10,2%%date:~-7,2%%date:~-4,4%.rar ${report_dir}")
```

Рисунок 2.35 – Код скрипту BeanShell

Генерацію HTML звіту можна налаштувати в Jmeter. Процес відображено на рисунку 2.36.

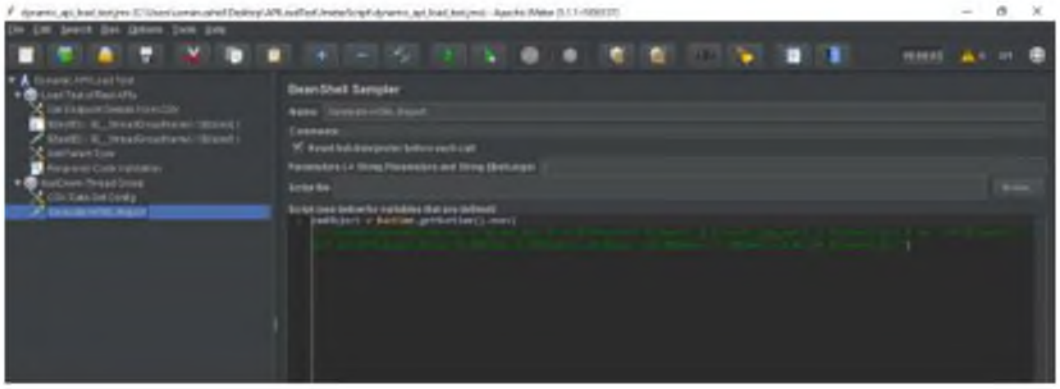


Рисунок 2.36 – Процес налаштування звіту

Наступним етапом необхідно налаштувати процес запуску тестів. У файлі групи потоків необхідно налаштувати кількість потоків, час, необхідний на виконання всіх потоків і кількість повторень тестів. Детальну інформацію можна розглянути на рисунку 2.37.

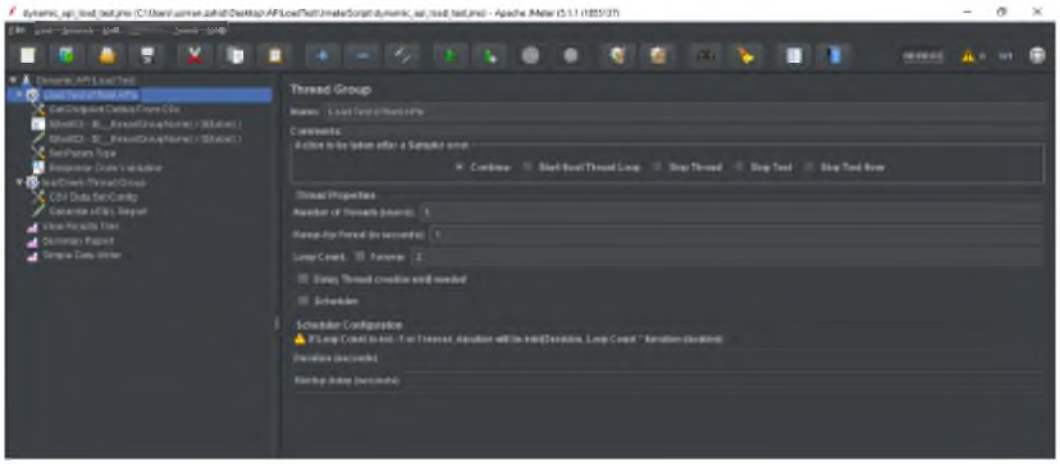


Рисунок 2.37 – Налаштування даних для HTTP Sampler

Після запуску навантажувальних тестів, було отримано наступні результати, які детальним шляхом демонструють витривалість серверних та технічних ресурсів системи. Інформацію можна розглянути на рисунках 2.38-2.41 [16]

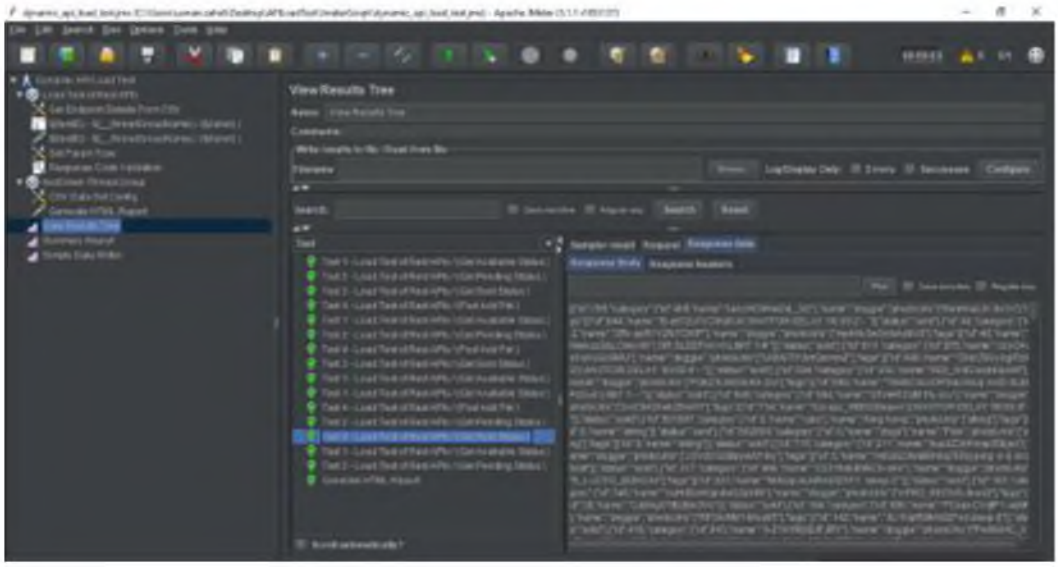


Рисунок 2.38 – Результат роботи навантажувального тесту

Рисунок 2.39 –Загальний звіт.

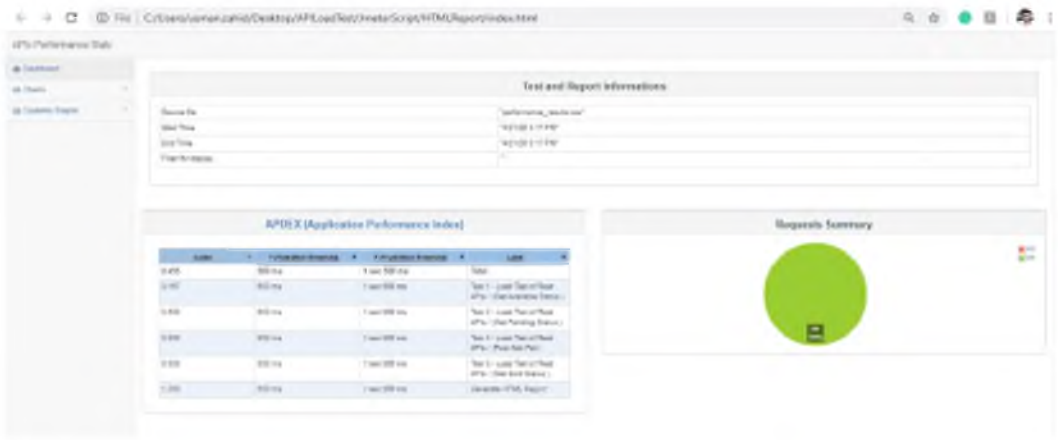


Рисунок 2.40 –HTML звіт

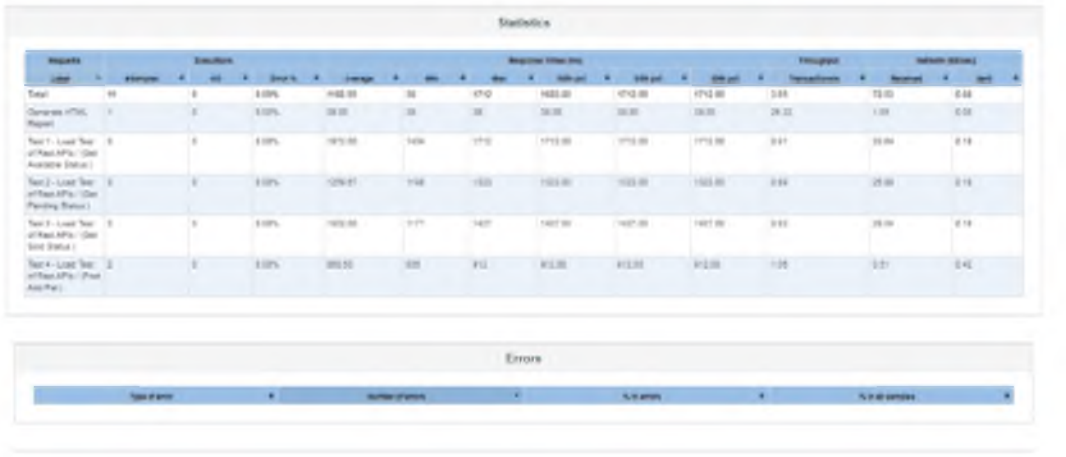


Рисунок 2.41 –HTML звіт (2)

Інструментарій Jmeter містить технології та ресурси, призначені для тестування сервісів, які використовують TCP протокол. Для реалізації тестових процесів необхідно застосувати функцію TCP Sampler для нашалтування тестової системи. Необхідно обрати даний інструмент, згідно рисунку 2.42. TCP-протокол дозволяє створити користувача, який надсилає інформацію у режимі очікування, тобто даний протокол не містить властивостей постійного надсилання пакетів до серверу.

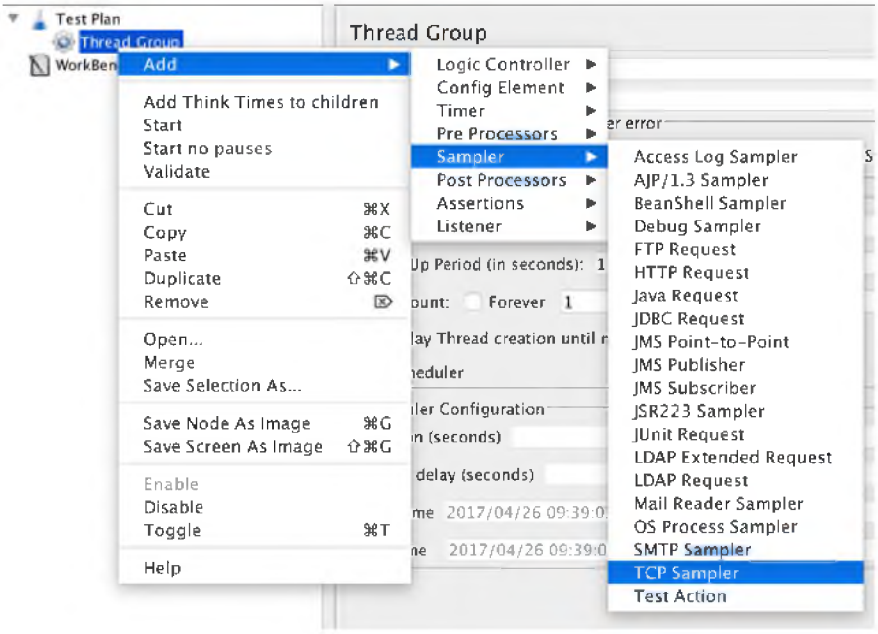


Рисунок 2.42 – Шлях до вибіру TCP Sampler.

Кожне поле TCP Sampler містить “шаблон”, тобто який являється умовним класом для групи об’єктів. Основними параметрами для налаштування системи, вважаються:

- TCPClientImpl – базовий клас для реалізації обміну текстовими повідомленнями;.
- BinaryTCPClientImpl – базовий клас для реалізації обміну текстовими повідомленнями із перекладом повідомлення у бінарний або шістнадцядковий вигляд;
- LengthPrefixedBinaryTCPClientImpl – базовий клас, аналогічний до попереднього із використанням функції префіксу двійкової довжини байта.

Слід зауважити, що для детального відображення повідомлення, буде використано функцію $(__iterationNum()}, яка буде відображати порядок ітерації (у даному випадку, номер повідомлення). Детальну інформацію відображено на рисунку 2.43.

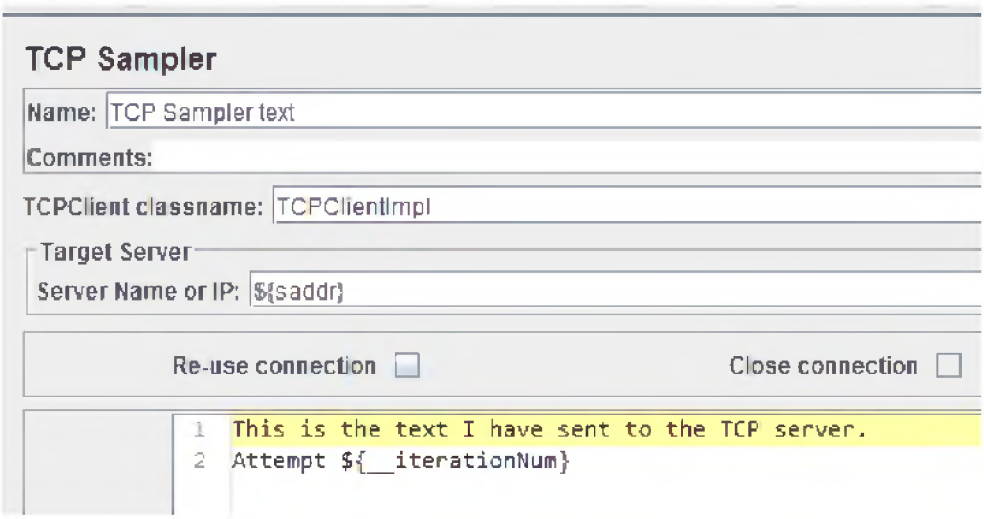


Рисунок 2.43 – Налаштування TCP Sampler

Далі необхідно заповнити наступні поля:

- TCPClient classname – ім’я класу, який буде використовуватись для запуску тестів;
- Server Name (IP) – ім’я станції, де знаходиться TCP сервер;

- Port field – порт, на якому знаходиться TCP сервер;
- Text message – текст повідомлення.

Результат налаштування можна розглянути на рисунку 2.44.

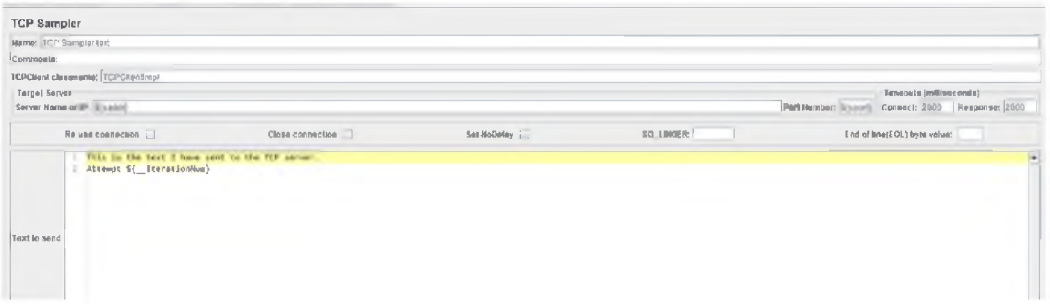


Рисунок 2.44 – Процес налаштування текстового повідомлення.

Після запуску тестів, було отримано наступні результати, які відображені на рисунку 2.45.

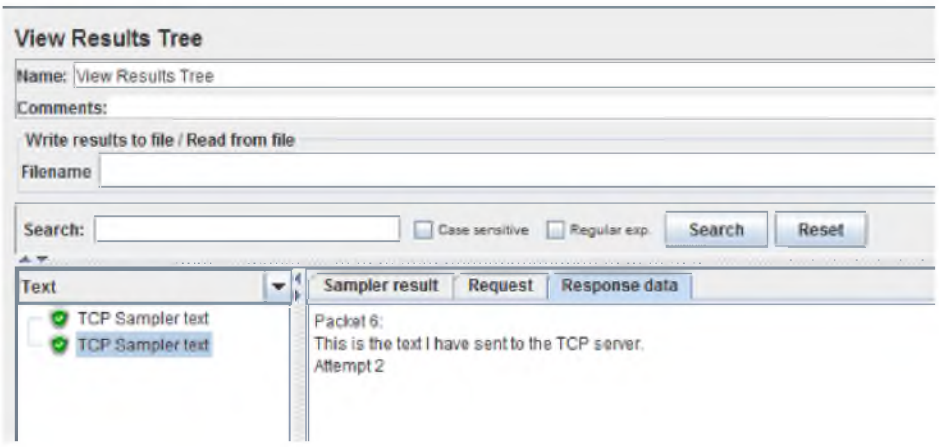


Рисунок 2.45 – Результат роботи TCP Sampler.

2.3.3 Протоколи та навантажувальні тести

Конфігураційна система Jmeter налаштована на вибір протоколу для початку процесу тестування, але для отримання результатів тестування необхідно застосовувати серверні механізми та ресурси. У даному випадку, для тестування зазначених протоколів необхідна система, яка буде виконувати роль поштового ресурсу (або серверу обробки повідомлень). Для подальшого тестування POP3 або IMAP технології, необхідно застосовувати SMTP протокол.

Задля виконання процесу тестування необхідно використовувати SMTP Sampler Jmeter для надсилання для надсилання повідомлень до електронної пошти. Jmeter не має технічних ресурсів або інструментів, спрямованих для створення поштового серверу та системи, тому для виконання процедур тестування необхідно надсилати повідомлення через вказаний агент передачі пошти. Для реалізації даної процедури необхідно застосувати наступні функції:

- Хост сервера SMTP;
- Порт сервера SMTP;
- Адреса відправника;
- Адреса одержувача;
- Комбінація імені користувача та пароля;
- Тема і текст повідомлення.

Як приклад роботи, буде розглянуто систему використання SMTP-серверу у сервісі Google Mail. Для початку роботи необхідно застосувати інструмент SMTP Sampler і внести наступні параметри:

- Сервер: smtp.googlemail.com;
- Порт: 587;
- Gmail пошта: blazemeter.test@gmail.com.

Після вибору вхідних параметрів необхідно виконати автентифікацію, згідно правил Google Mail SMTP Server. Далі необхідно передати наступні дані:

- Ім’я користувача: blazemeter.test@gmail.com;
- Пароль: some_pwd (Пароль можна взяти із файлу Jmeter.jmx).

Наступним етапом необхідно активувати StartTLS параметр згідно правил Google Mail SMTP Server. Для тестування необхідно заповнити тему та текст повідомлення. Як приклад, у тему повідомлення було занесено повідомлення «Привіт від Jmeter SMTP Sampler». Для вибору покращених механік, можна

застосувати функцію System.currentTimeMillis() до теми повідомлення, обравши час його відправлення.

Далі необхідно заповнити тіло повідомлення. У даному середовищі вводиться будь-яка інформація, котра буде надсилатися до пошти (як приклад, було надіслано повідомлення «Повідомлення з потоку Jmeter #${__threadNum}». Параметр #${__threadNum} надає номер потоку (у даній ситуації, номер потоку позначає номер повідомлення по порядку). Детальну інформацію про процес заповнення полей можна розглянути на рисунку 2.46

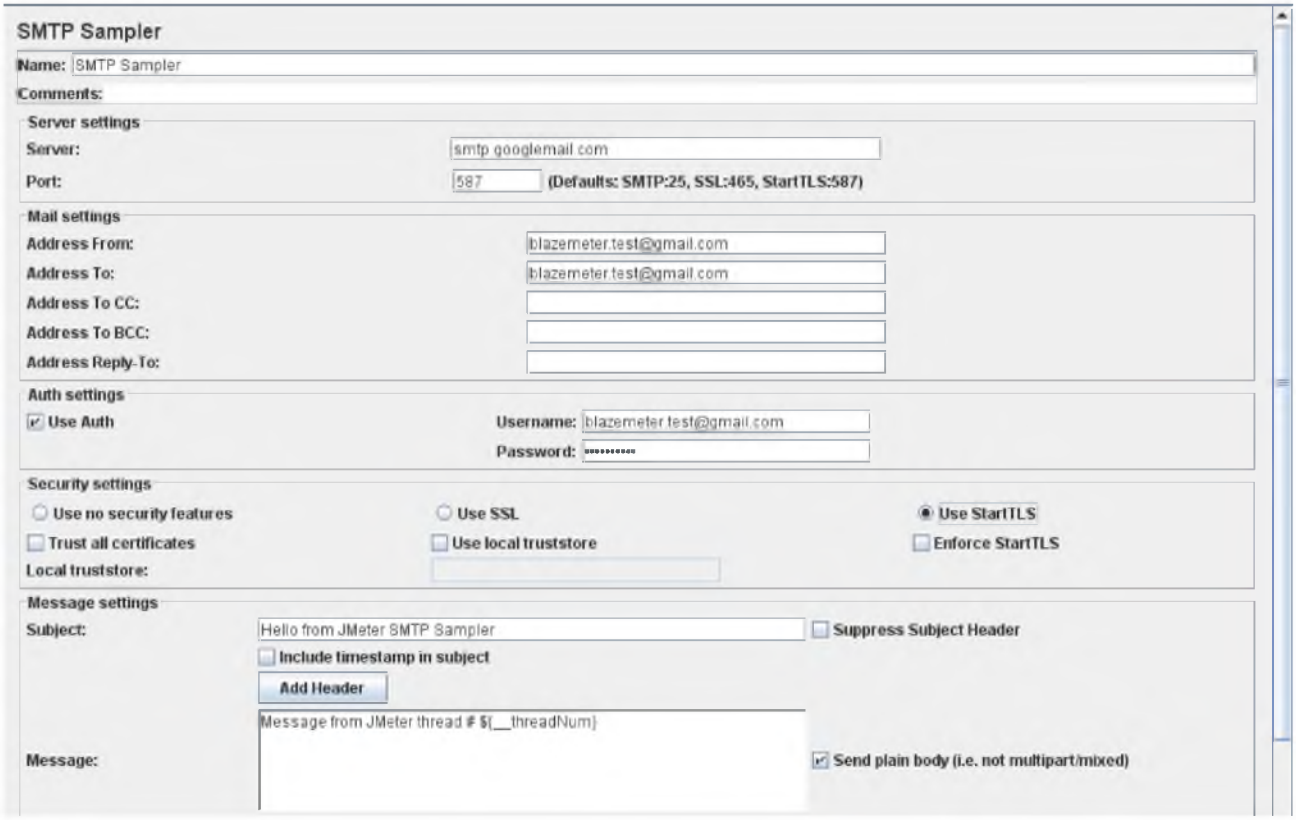


Рисунок 2.46 – Функція SMTP Sampler

Після запуску тестів, отримані результати можна розглянути на рисунку 2.47.

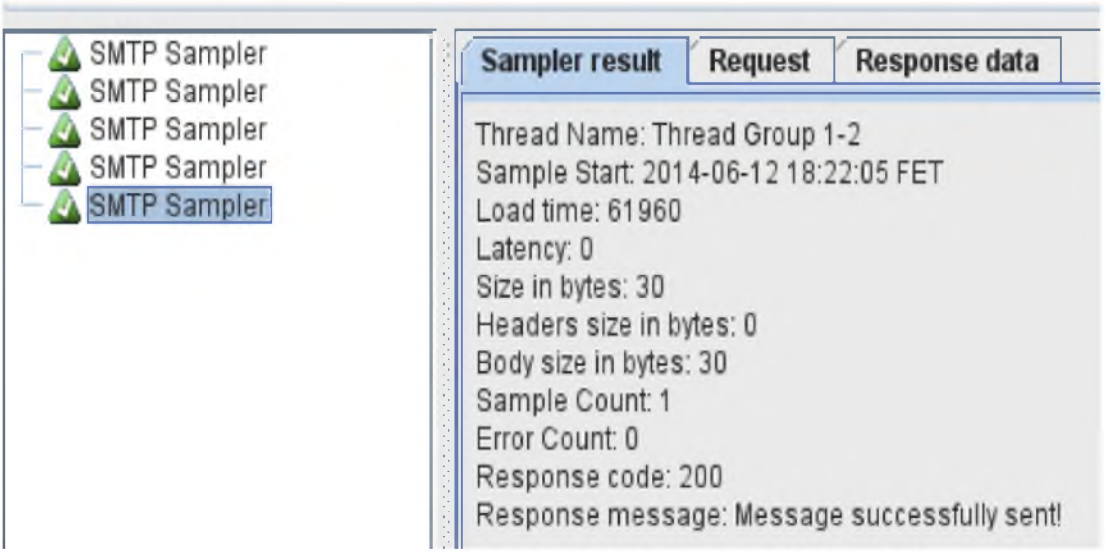


Рисунок 2.47 – Результат роботи SMTP Sampler

Для підтвердження результату, було додатково залогінено в систему Google Mail. Отримані результати можна розглянути на рисунку 2.48.

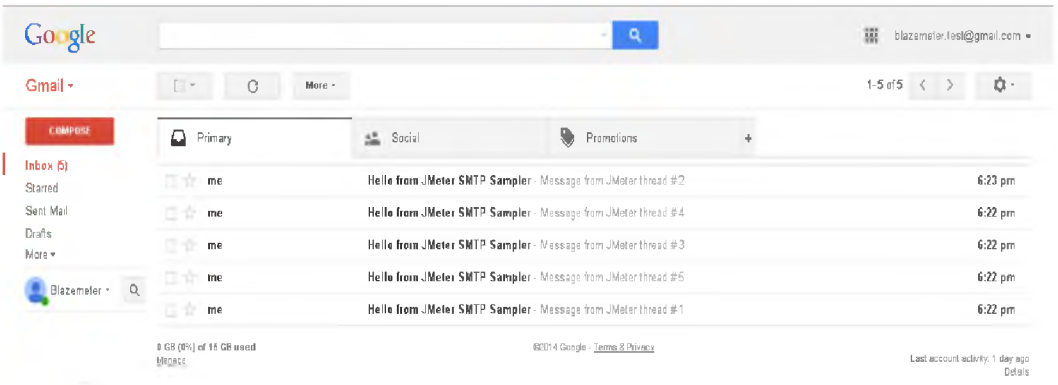


Рисунок 2.48 – Google Mail (Gmail). Надіслані повідомлення.

Для прочитання надісланих повідомлень, необхідно застосувати спеціальну функцію, яка використовує протоколи POP3 і IMAP, а саме – Mail Reader Sampler. Після вибору даного параметру, необхідно заповнити наступні поля:

- Протокол: POP3 або IMAP;
- Хост серверу: DNS-ім’я або IP-адреса поштового серверу;
- Порт серверу: Спеціальний порт, на якому знаходиться поштовий сервер. Стандартні порти можна оглянути у протоколах Jmeter;

- Ім’я користувача: Ім’я користувача POP3/IMAP серверу;
- Пароль: Пароль, який необхідний для отримання доступу до серверу;
- Папка: Спеціальне сховище поштового серверу (шлях до папки);
- Кількість повідомлень для отримання;
- Функція видалення повідомлень: Дана функція необхідна для видалення відправлених повідомлень до поштового серверу після завершення його роботи;
- Функція збереження MIME: Дана функція необхідна для збереження теми та тіла повідомлення у даних відповіді;
- Функція налаштування систем безпеки: Дана функція містить опції, які спрямовані на захист повідомлення під час його передачі до серверу.

Заповнені дані можна розглянути на рисунку 2.49.

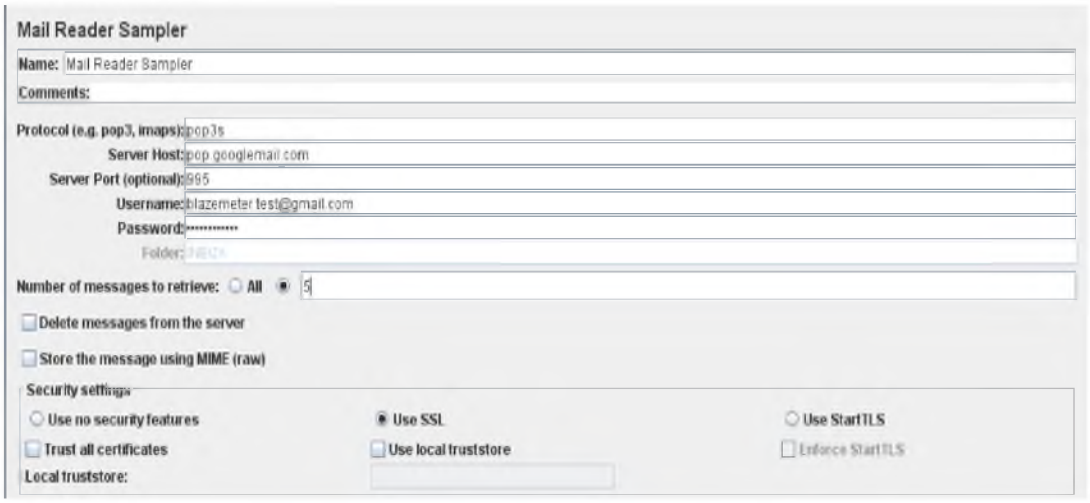


Рисунок 2.49 – Mail Reader Sampler

Після запуску процесу зчитування SMTP запитів в одному потоку можна розглянути наступні результати, описані на рисунку 2.50. Інструментарій Jmeter дозволяє створити отримати результати виконаних тестів у «базовій» (у вигляді набіру даних і результату виконання тесту) та у «поліпшеній» формах (у табличному вигляді формату CSV або будь-якій іншій доступній формі).

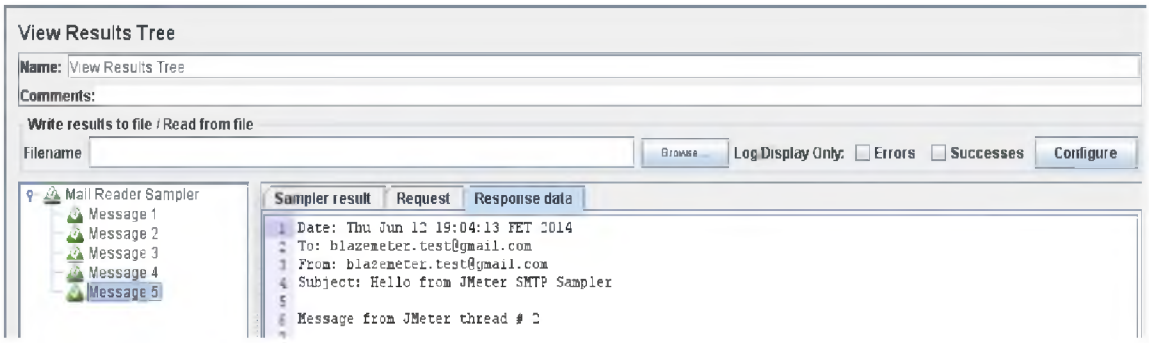


Рисунок 2.50 – Результат роботи Mail Reader Sampler

Конфігурація інструменту Jmeter і системи штучного навантаження мережі Blazemeter дозволяє перевірити витривалість серверних та технічних ресурсів. Отриману інформацію можна використовувати як параметри, необхідні для налаштування алгоритмів нечіткої логіки. Наприклад, на рисунку 2.51 було розглянуто графік співвідношення активних користувачів та час відповіді на запити.

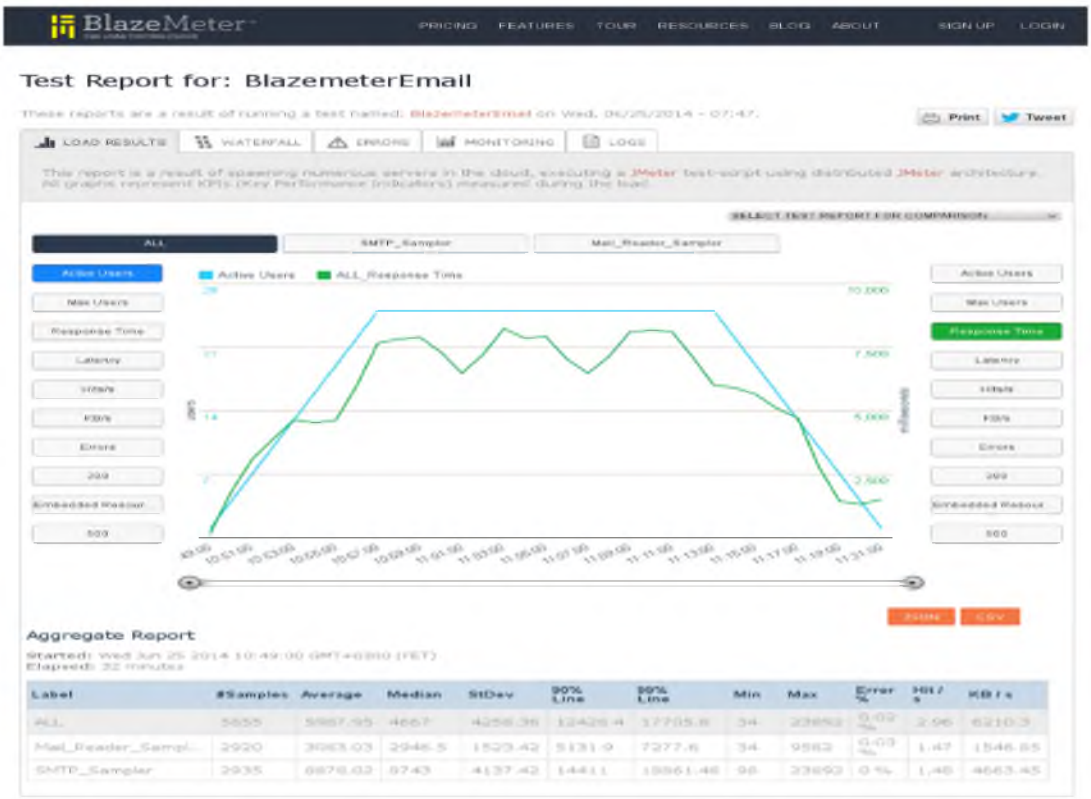


Рисунок 2.51 – Графік співвідношення кількості користувачів і часу відповіді

На даному графіку можна побачити, що у процесі навантажувального тестування приймало участь 5855 користувачів, які робили запит на поштовий сервер. Швидкість обробки повідомлень знижувалась при надсиланні великої кількості запитів. Загальний результат можна отримати, враховуючи піки навантаження і стабілізації трафіку, використовуючи формулу 2.29.

$$\frac{RT_{UP1} + RT_{UP2} + \cdots + RT_{UPn}}{RT_{DOWN1} + RT_{DOWN2} + \cdots + RT_{DOWNn}} \quad (2.29)$$

Де RT_{UP} – час відповіді при максимальних навантаженнях, RT_{DOWN} – час відповіді при стабілізації серверних ресурсів під час максимального навантаження. [15]

2.5 Висновки до другого розділу

Підсумовуючи, можна сказати, що Ddos-атаки представляють собою набір запитів, які у великій кількості надсилаються до системи та її ресурсів. Задля зниження реалізації атак, які знешкоджують працездатність систем, необхідно застосувати спеціалізовані механізми, які спрямовані на фільтрацію трафіку.

Основні проблеми закладаються у відсутності фільтруючого механізму, технічних ресурсів та способів знешкодження і мінімізації даного типу атаки. Проте з іншої сторони, дані атаки залишають велику кількість інформації, яку можна використати для аналізу і розробці алгоритмів, головний напрямок яких концентрується на фільтрації трафіку, встановлення точок реалізації атак і блокування даного типу атаки в цілому.

Для реалізації даного механізму необхідно застосовувати методи машинного навчання, які базуються на збору інформації (або статистичної вибірки), виходячи із спонтанних даних (запитів), які надходять до системи, тобто на методах навчання «без вчителя».

Основна проблема методів машинного навчання із вчителем залежить від зовнішніх чинників, тобто дій осіб, які займаються навчанням системи. Із систем

навчання без даних факторів підходять методи кластеризації даних та їх механізми.

Одним із ключових механізмів розробки даної системи вважається система методу кластеризації на основі k-means, оскільки даний алгоритм дозволяє використовувати шаблони даних, прорахованих з математичної точки зору, як константу для фільтрації трафіку. Структура, яка побудована на даній системі, в контексті виявлення DdoS-атак дозволяє налаштувати межу фільтрації надійного і шкідливого трафіку, встановити точку початку атаки і як наслідок – прорахувати технічні можливості зловмисника та його активність.

Проте, дана система має купу недоліків, пов'язаних із продуктивністю алгоритму, а саме – низька працездатність та відсутність мобільності. У будь-якому підприємстві швидкість обробки запитів являється ключовим фактором для ефективності роботи системи. У результаті реалізації атаки, швидкість обробки інформації трафіку може бути критично низькою. Задля підвищення ефективності роботи алгоритму, було проаналізовано алгоритм k-means ++, який частково знищує описані проблеми.

Не виключним фактором являється відсутність тестової вибірки, необхідної для встановлення початкових параметрів, завдяки яким, система зможе функціонувати якісніше. Для вибіру і аналізу якості систем, було запропоновано і описано інструмент Jmeter, який спрямований на перевірку якості систем, які оброблюють інформацію. Завдяки даному інструменту можна проаналізувати множину елементів, спрямованих на встановлення ефективності роботи систем. Проаналізовані дані можна використовувати для навчання алгоритмів нечіткої логіки, оскільки Jmeter надає отримані результати у вигляді констант (наприклад, максимальна кількість користувачів в системі, швидкість обробки запитів тощо). [17]

РОЗДІЛ 3. ЕКОНОМІЧНИЙ РОЗДІЛ.

3.1 Капітальні витрати

Головною метою даного розділу можна вважати реалізацію проекту з питань безпеки інформації і фільтрації трафіку в мережі із урахуванням економічних можливостей підприємства, таких як розрахунок витрат на створення і підтримку проекту та визначення його окупності.

Для реалізації проекту необхідні фінансові ресурси. Головною метою даного розділу можна вважати прорахунок витрат на розробку та підтримку проекту і кількість часу, необхідного для окуплення описаних витрат.

Задля коректного прорахунку, необхідно враховувати множину зовнішніх та внутрішніх економічних чинників, які будуть впливати на кінцевий результат. До даних пунктів можна віднести наступні етапи:

- Капітал;
- Час;

Капітал являє основну складову, необхідну для початку процесу розробки проекту. Даний параметр розраховується на підставі витрат, які являються ключовими при розробці якісного проекту.

На таблиці 3.1 відображено сукупність програмно-апаратних технологій необхідних для реалізації проекту.

Таблиця 3.1 – Витрати на реалізацію проекту

№	**Засоби**	**Кількість**	**Вартість**
Апаратні засоби			
1	Сервер ArtLine Business T25	3	26178 * 3 = 78534 грн
2	Комп'ютер 2E Complex Gaming	7	24000 * 7 = 168000 грн

Продовження таблиці 3.1				
3		Монітор ASUS 249QB	7	6000 * 7 = 42000 грн
Програмні засоби				
4	Matlab Standart Version		1	63450 грн
Комунальні послуги				
5	Електроенергія		-	250 кВт/год * 1.68 грн = 420 грн

Час, необхідний для розробки проекту залежить від якості робочої сили, тобто кількості працівників, їх досвід тощо. Для реалізації даного проекту було застосовану мінімальну кількість робочої сили. Рівень заробітньої платні розраховується на підставі досвіду, освіченості кваліфікованого спеціаліста. На таблиці 3.2 буде запропоновано інформацію, про кількість осіб, які приймають участь у розробці проекту.

Таблиця 3.2 – Таблиця співробітників, необхідних для створення проекту

№	**Робоча позиція**	**Заробітня платня**	**Годинна оплата**	**Кількість годин (розробка проекту)**	**Всього**
1	Спеціаліст з обробки та аналізу даних	69390 грн	361 грн/год	78 год	28158 грн
2	Тестувальник програмного забезпечення	45900 грн	239 грн/год	105 год	25095 грн
3	Розробник програмного забезпечення	54000 грн	281 грн/год	248 год	69688 грн
4	Менеджер проекту	49140 грн	256 грн/год	104 год	26624 грн
5	Системний адміністратор	32400 грн	169 грн/год	28 год	4732 грн
6	Архітектор проекту	148500 грн	773 грн/год	118 год	91214 грн
7	Спеціаліст експлуатації і розробки проекту	78300 грн	408 грн/год	56 год	22848 грн
Всього: 28158 + 25095 + 69688 * 2 + 26624 + 4732 + 91214 + 22848 = 338047 грн					

Виходячи із даних параметрів, необхідно розрахувати витрати, необхідні на закупівлю програмно-апаратних засобів, згідно формули 3.1,

$$S = E_1 + E_2 + \cdots + E_n \quad (3.1)$$

де S – сума витрат, E – витрати на елемент. При розрахунках, було отримано, що сума витрат S окремо на ПЗ, апаратне забезпечення і інші витрати складають,

$$S_{ar} = 78354 + 168000 + 42000 = 288534 \text{ грн}$$

$$S_{pr} = 63450 \text{ грн}$$

$$S_{en} = 420 \text{ грн}$$

де S_{ar} – капітальні витрати на апаратне забезпечення, S_{pr} – капітальні витрати на ПЗ, S_{en} – витрати на комунальні послуги.

Команда розробки включає наступних спеціалістів:

- 2 розробники ПЗ;
- 1 спеціаліст з обробки та аналізу даних;
- 1 системний адміністратор;
- 1 спеціаліст експлуатації та розробки проекту;
- 1 менеджер;
- 1 архітектор проекту;
- 1 тестувальник ПЗ;

Далі необхідно розрахувати амортизаційні витрати, необхідні для підтримки проекту. Процес розрахунку на підтримку систем буде відштовхуватися від половини заробітньої платні спеціалістів, які підтримують і оновлюють проект, тобто від технічного персоналу проекту. Даний параметр можна враховувати, як вартість проекту.

Коефіцієнт амортизації можна розрахувати згідно формули 3.2,

$$A = \frac{S}{Y} \quad (3.2)$$

де А – коефіцієнт амортизації, Y – срок корисного використання у роках. Оскільки перевірка проводиться раз на 5 років для апаратного забпезпечення і раз на 3 роки, то далі буде розраховано амортизацію на ПЗ, апаратне забезпечення і інші ресурси, необхідні для розробки проекту згідно формули 3.2.

$$A_{ar} = \frac{288534}{5} = 57707 \text{ грн.}$$

$$A_{pr} = \frac{63450}{3} = 21150 \text{ грн.}$$

$$A = 57707 \text{ грн} + 21150 \text{ грн} = 138379 \text{ грн.}$$

Далі необхідно прорахувати заробітню платню виконавця, застосовуючи формули 3.3 і 3.4.

$$Ззп = t \times Зіб \quad (3.3)$$

$$Змч = t \times Смч \quad (3.4)$$

де, t - час, витрачений на процес розробки, Зіб - середня погодинна заробітна платня, Смч – вартість 1 години машинного часу ПК. Для даної задачі, машинний час дорівнює часу роботи спеціаліста (Змч = t).

Далі необхідно прорахувати час, необхідний для рорзобки проекту та спеціалісту для підтримки систем захисту, згідно формули 3.5,

$$t = t_{тз} + t_{р} + t_{т} + t_{нал} + t_{пд}, год \quad (3.5)$$

де $t_{тз}$ – час, необхідний для створення технічного завдання, $t_{р}$ – час, необхідний для розробки плану перевірки, $t_{т}$ – час, необхідний для проведення тестування початкових процесів, $t_{нал}$ – час, необхідний для переналаштування механізмів у результаті некоректної роботи, $t_{пд}$ – час, необхідний для підтримки програмно-апаратних ресурсів.

Більш детальну інформацію можна розглянути на таблиці 3.3.

Таблиця 3.3 – Час, витрачений на розробку проекту

Позначення	**Пояснення і витрачений час**
ттз	Оформлення технічного завдання – 60 год, обговорення деталей проекту – 18 год, затвердження деталей проекту – 10 год, редагування технічного завдання до технічного документу – 16 год (Всього: 104 год)
тр	Розробка архітектури – 112 год, вибір технологій – 14 год, розробка проекту – 200 год (Всього: 326 год)
тт	Тестування проекту – 70 год, виправлення дефектів проекту – 48 год, повторне тестування проекту – 35 год (Всього: 143 год)
тнал	Налаштування систем інтеграції – 60 год, налаштування технічних ресурсів – 60 год, інтеграція проекту і системи – 14 год (Всього: 134 год)
тпд	Підтримка проекту – 50 год, оформлення звіту про статус проекту – 8 год, розробка документації – 14 год (Всього: 72 год)
Сумарно витрачений час (Формула 3.5): t1 = 104 + 326 + 143 + 134 + 72 = 779 год	

Далі необхідно прорахувати витрати, які необхідні для розробки проекту, використовуючи формули 3.3 – 3.5. Так як розробка проекту займає приблизно місяць і систему необхідно постійно оновлювати, відштовхуючись від шляху розробки проекту, то,

$$\text{Ззп} = \text{Крп} = S_{per}$$

де S_{per} – витрати на заробітню платню спеціалістів.

Отже, сума капітальних витрат буде розрахована шляхом додавання всіх витрат, описаних вище. (закупівля обладнання, зарплати спеціалістів тощо)

$$\text{Капітальні витрати} = 420 \text{ грн} + 338047 \text{ грн} + 288534 \text{ грн} = 627001 \text{ грн}$$

Можна сказати, що сума разових витрат на розробку проекту систем безпеки складає 627001 грн із урахуванням закупівлі обладнання, програмних

ресурсів, виплати спеціалістам заробітньої платні та витрат на комунальні послуги.

3.2 Експлуатаційні витрати

Далі необхідно прорахувати час, який працює спеціаліст. В середньому, спеціаліст працює 5 днів по 8 годин кожен тиждень. Як середню константу роботи за місяць, спеціаліст працює 24 дні, тобто:

$$24 \text{ дні} * 8 \text{ годин} = 192 \text{ год.}$$

Оскільки участь у процесі підтримки проекту вимагає весь технічний персонал проекту, то у вирішенні даного питання необхідно відокремити спеціаліста, який являється найбільш активним у підтримці системи. У даному випадку, зазначену роль виконує адміністратор системи.

Отже, можна розрахувати зарплатню за одиницю часу, як:

$$\frac{16200 \text{ грн}}{192 \text{ год}} = 84 \text{ грн/год}$$

Виходячи із аналізу часу, необхідного для підтримки проекту, можна розрахувати точну зарплатню спеціаліста, за підтримку систем. Капітал, який витрачається на спеціаліста можна розрахувати використовуючи формулу 3.6,

$$\text{Крп} = \text{Ззп} * \text{Змч} \tag{3.6}$$

де Крп – витрати на підтримку системи, Ззп – заробітня платня спеціаліста, Змч – час, витрачений на підтримку системи. Точна заробітня платня спеціаліста спеціаліста (виконавця) розрахувується наступним чином (формула 3.3):

$$84 \ {}^{\text{грн}}/_{\text{год}} * 50 \text{ год} = 4200 \text{ грн}$$

Далі буде прораховано час, необхідний на підтримку систем. Детальну інформацію можна розглянути на таблиці 3.4.

Таблиця 3.4 – Час, витрачений на підтримку проекту

Позначення	**Пояснення і витрачений час**
ттз	Огляд ПЗ, згідно стандартів компанії - 4 год, оформлення технічного завдання - 2 год (Всього: 6 год)
тр	Розробка плану перевірки – 6 год (Всього: 6 год)
тт	Проведення тестування механізмів перевірки – 10 год, введення механізмів перевірки у віртуальну ОС – 4 год (Всього: 14 год)
тнал	Виведення механізмів перевірки із віртуальної ОС – 4 год, введення механізмів перевірки у проект – 4 год, налаштування серверних ресурсів – 2 год (Всього: 10 год)
тпд	Підтримка механізмів перевірки – 8 год, виведення результату про кінцеву роботу – 4 год, оформлення звіту – 2 год (Всього: 14 год)
Сумарно витрачений час (Формула 3.5): t1 = 6 + 6 + 14 + 10 + 14 = 50 год	

Тоді, згідно формули 3.6 можна знайти витрати на підтримку системи (щомісячно):

$$Крп = 4200 \text{ грн} * 50 \text{ год} + 420 \text{ грн} = 210420 \text{ грн}$$

Наступним етапом розрахунків витрат підприємства буде прорахування поточних витрат на підтримку проекту. Оскільки у даній організації присутні 8 осіб, які забезпечують якість та адміністрування проекту і заробітна платня працівників статична (без врахування премій), то витрати на підтримку проекту буде розраховано за формулою 3.7,

$$С = Сн + Са + Сз + Сел + Стос \quad (3.7)$$

де Са - річний фонд амортизаційних витрат, Сз - річний фонд заробітної платні інженерно-технічного персоналу, Сн - витрати на навчання адміністративного персоналу і користувачів, Сел - вартість електроенергії, Стос

- витрати на технічне й організаційне адміністрування. Оскільки половина заробітньої платні спеціалістів витрачається на розробку проекту, то необхідно виключити осіб нетехнічного персоналу. Згідно контракту, спеціаліст отримує половину заробітньої платні за підтримку систем. Умовно, можна сказати, що:

$$Сз = \frac{S}{2}$$

Далі буде прораховано витрати інженерно-технічного персоналу:

$$Сз = \frac{60390\ грн + 45900\ грн * 2 + 54000\ грн + 78300\ грн}{2} = 142245\ грн$$

Витрати адміністративного персоналу і користувачів визначаються виключно із документації, яка описує принципи роботи і структуру проекту. У даному випдаку, описану роль відіграє процес створення документації, тобто, згідно даних із таблиці 3.4 виділимо час, необхідний для створення документації. Також не виключним фактором необхідно враховувати заробітню платню за годину праці спеціалістів, які займаються розробкою документації. Посилаючись на таблицю 3.2 можемо розрахувати погодинну оплату праці спеціалістів, які приймають участь у розробці документації, а саме:

- Спеціаліст з обробки та аналізу даних;
- Розробник програмного забезпечення;
- Тестувальник програмного забезпечення;
- Менеджер проекту.

Відштовхуючись від вищеописаних даних і кількості годин, необхідних для розробки документації, можна прорахувати параметр Сн наступним чином.

$$Сн = 14\ год * 1137\ грн/год = 15918\ грн$$

Вартість електроенергії можна прорахувати, використовуючи формулу 3.8,

$$Сел = Р * Fp * Це, грн \qquad (3.8)$$

де Р - встановлена потужність апаратури (кВт), Fp - річний фонд робочого часу системи ІБ (год), Це - тариф на електроенергію (грн\Квт * год).

Це дорівнює 1.87 грн\Квт * год. Даний показник встановлений із стандартами договору умовного підприємства. За розрахунками даного року встановлена потужність апаратури складає 4.78 кВт. Враховуючи святкові та вихідні дні, річний фонд робочого часу складає приблизно 216 днів. Отже, згідно формули 3.10:

$$Сел = 1.87 \text{ грн\кВт} * 4.78 \text{ кВт} * 216 \text{ днів} * 8 \text{ годин} = 15446 \text{ грн}$$

Згідно із таблиці вагової частки витрат:

Таблиця 3.5 - Вагова частка витрат

Фіксовані (капітальні) вкладення	21%
Поточні витрати, у т.ч.	79%
Керування системою	12%
Технічна підтримка й відновлення	21%
Активність користувача	46%

Річний амортизаційний фонд дорівнює відсотку від суми капітальних інвестицій. У нашому випадку, згідно із табличкою вище, капітальні вкладення складають 21%, тобто Са буде складати приблизно 10%.

Витрати на технічне і адміністративне керування визначаються у відсотках від вартості капітальних витрат (Як правило 1-3%. Для даної задачі візьмемо 2%). Повертаючись до формули 3.7 можемо розрахувати експлуатаційні витрати.

$$С = 0.1 + 0.02 + 15446 \text{ грн} + 15918 \text{ грн} + 142245 \text{ грн} = 173609{,}12 \text{ грн}$$

Отже, поточні витрати на рік складають 173609,12 грн враховуючи витрати на підтримку проекту та заробітню платню спеціалістів.

3.3 Розрахунок збитків

Далі буде прораховано можливі збитки у результаті реалізації загроз. Дані показники обираються умовним чином і проводяться в межах структури компанії. Доступ до проекту і систем мають наступні спеціалісти:

- Розробник ПЗ;

- Тестувальник ПЗ;
- Архітектор проекту;
- Системний адміністратор;
- Спеціаліст з обробки і аналізу даних;
- Спеціаліст з обробки і експлуатації систем.

Можливі збитки і упущену вигоду можна прорахувати, використовуючи формули 3.9 – 3.11,

$$U = \text{Пп} + \text{Пв} + V \quad (3.9)$$

$$\text{Пп} = \frac{\sum \text{Зс}}{F} \times t\text{п} \quad (3.10)$$

$$\text{Пв} = \text{Пви} + \text{Пвч} + \text{Пзч} \quad (3.11)$$

де Пп – оплачувані втрати робочого часу, Пв – вартість відновлення працездатності вузла корпоративної мережі, Пви – витрати на повторне уведення інформації, Ппв – витрати на відновлення вузла або сегмента корпоративної мережі, Пзч – вартість заміни устаткування або запасних частин, tп – час простою, F – місячний фонд робочого часу, V – втрати від зниження обсягу продажів.

Необхідно знайти параметр втрат робочого часу. Умовно, час простою tп можна взяти приблизно 6 годин. Місячний фонд робочого часу умовно являється кількістю робочих часів за тиждень. Погодинну заробітню платню спеціалістів технічного персоналу можна розрахувати, відштовхуючись від таблиці 3.2. Отже, згідно формули 3.10:

$$\text{Пп} = \frac{1739 \text{ грн/год}}{40 \text{ год}} * 6 \text{ год} = 260{,}85 \text{ грн}$$

Витрати на відновлення інформації і вузла можна знайти за формулами 3.13 і 3.14:

$$\text{Пви} = \frac{\sum \text{Зви}}{F} \times t\text{ви} \quad (3.13)$$

$$\text{Ппв} = \frac{\sum \text{Зпв}}{F} \times t\text{пв} \qquad (3.14)$$

Кількість часу на відновлення програмно-апаратної частини приблизно однакова, тобто:

$$\text{Ппв} = \text{Пви} = \text{Пзч} = \text{Пп} = 260{,}85 \text{ грн}$$

Витрати на зниження обсягу можемо найти за формулою 3.16,

$$V = \frac{O}{F\text{г} \times \text{час відновлення}} \qquad (3.15)$$

де Fг – річний фонд робочого часу роботи організації, О - обсяг продажів атакованого вузла. Умовно було зазначено, що О = 900000 грн\рік. Отже, згідно формули 3.15 можемо прорахувати даний параметр:

$$V = \frac{900000 \text{ грн\рік}}{2040 \text{ год} \times 6 \text{ год}} = 73{,}52 \text{ грн}$$

Далі, використовуючи формулу 3.9 необхідно знайти невідомий параметр:

$$U = 260{,}85 \text{ грн} \times 4 + 73{,}52 \text{ грн} = 1116{,}92 \text{ грн}$$

3.4 Розрахунок економічної ефективності

Враховуючи вищеописані збитки, логічним шляхом буде прорахування ефективності систем інформаційної безпеки. Прорахувати дані аспекти можна за формулою 3.16:

$$E = B \times R - C \qquad (3.16)$$

де В – загальний збиток від атаки, R – ймовірність реалізації атаки, С – поточні витрати на підтримку систем інформаційної безпеки. Знайти загальний збиток атаки необхідно за формулою 3.17:

$$B = \sum i \times \sum n \times U \qquad (3.17)$$

де і – кількість атакованих вузлів, n – середнє число атак на рік.

Слід враховувати, що термін окупності залежить від діяльності підприємства та його капіталу. Якщо підприємство являє собою невелике за

обсягом компанію, то кількість вузлів та атак буде значно нижче, ніж у великих організаціях. Тобто, термін окупності проекту буде вище.

Для даного прикладу було обрано невелике за обсягом підприємство, яке займається ремонтом технічного приладдя різного призначення. За нещодавніми показниками, в ІТС за 2 попередніх роки загалом було атаковано 6 вузлів корпоративної мережі. Можна припустити, що за рік було проведено атаку на 3 вузла з метою викрадення інформації про клієнтів. В середньому за рік було проведено 12 атак (одна атака на місяць). Отже, можемо прорахувати загальні збитки від атаки:

$$B = 3 \text{ вузла} \times 12 \text{ атак} \times 1116{,}92 \text{ грн} = 40\,209{,}12 \text{ грн}$$

Ймовірність реалізації загрози умовно було взято як 50%. Можемо прорахувати ефект від впровадження систем інформаційної безпеки.

$$E = 40\,209{,}12 \text{ грн} \times 0.5 - 173609{,}12 \text{ грн} = 153504{,}56 \text{ грн}$$

Можна сказати, що ефект реалізованої системи достатньо максимально знизить можливі збитки від реалізації атак на вузли та сегменти мережі. Наступним шляхом буде прорахування ефективності системи. По-перше, треба визначити коефіцієнт повернення інвестицій за формулою 3.18:

$$ROSI = \frac{E}{K} \tag{3.18}$$

де E – загальний ефект від впровадження систем інформаційної безпеки, K – капітальні інвестиції за варіантами, що забезпечили цей ефект. Далі можна прорахувати даний коефіцієнт за умовою, що S = K, де S представляє собою всі капітальні витрати. Отже, згідно описаної формули, можемо розрахувати даний параметр:

$$ROSI = \frac{153504{,}56 \text{ грн}}{627001 \text{ грн}} = 0{,}244$$

Останнім пунктом розрахування витрат на реалізацію проекту буде прорахунок терміну окупності, який представляє собою час, необхідний для окуплення встановленних систем інформаційної безпеки. Даний елемент, можна розрахувати за формулою 3.19,

$$To = \frac{1}{ROSI} \quad (3.19)$$

Підставляючи параметри, можна отримати наступні результати:

$$To = \frac{1}{0{,}244} = 4 \text{ роки}$$

3.5 Висновки до третього розділу

Підводячи підсумки, можна сказати, що для будь-якого підприємства, котре турбує питання стосовно необхідності для розробки і реалізації проекту необхідно враховувати фактори економічного, технічного та іншого характеру, які можуть вплинути на подальший економічний статус організації, а саме:

- Критичність інформації, яка циркулює в системі організації;
- Капітал, необхідний для розробки, модернізації і підтримки проекту;
- Можливі витрати в результаті реалізації загроз і їх мінімізація з боку реалізованого проекту;
- Особливості ІТС компанії;
- Варіативність вибору механізмів захисту, згідно із потребами актів, пов'язаних із безпекою інформації в ІТС;

Для реалізації проекту підприємству необхідно 627001 грн. Організація щорічно повинна витрачати 173609,12 грн на підтримку проекту, політик безпеки та механізмів захисту. Амортизаційні відшкодування складають 138379 грн.

У результаті реалізації даної системи інформаційної безпеки ефект від її реалізації буде складати 153504,56 грн. Збитки від реалізації атак на сегменти мережі та вузли складають 40209,12 грн. Порівнюючи ефект упущених витрат від

реалізації атак на ІТС можна сказати, що результат буде достатньо високим у результаті відтворення описаної системи. Для окупності проекту компанії необхідно 4 роки.

ВИСНОВКИ

У роботі було запропоновано метод виявлення DDoS-атак за допомогою алгоритму нечіткої логіки на основі k-means та введення проаналізованого методу системи виявлення вторгнень.

- Проаналізовано різні варіанти DDoS-атак та їх потенційний вплив на систему;
- Досліджено методи захисту у існуючих системах виявлення вторгнень;
- Запропоновано алгоритм нечіткої логіки на основі k-means, як систему фільтрації надійного і шкідливого трафіку.
- Модифіковано описаний алгоритм нечіткої логіки з метою оптимізації та покращення процесу аналізу трафіку;
- Запропоновано інструмент Jmeter, як приклад вибору статистичної вибірки для навчання алгоритму і проведення початкового аналізу системи.

В економічному розділі роботи було розраховано потенціальні втрати на реалізацію проекту та терміни окупності дослідженої технології.

ПЕРЕЛІК ПОСИЛАНЬ

1. Корниенко А. А. Системы и методы обнаружения вторжений: современное состояние и направления совершенствования [Електронний ресурс] / А. А. Корниенко, И. М. Слюсаренко // InfoSoftCom & ПГУПС. – 2009. – Режим доступу до ресурсу: http://citforum.ru/security/internet/ids_overview/.

2. Бобров А. О. Системы обнаружения вторжений [Електронний ресурс] / Артем Олегович Бобров // ICMM. – 2012. – Режим доступу до ресурсу: http://portal1.icmm.ru/~masich/win/lexion/ids/ids.html.

3. Системы обнаружения вторженй. Оборона и безопасность [Електронний ресурс] // Центр стратегических оценок и прогнозов. – 2007. – Режим доступу до ресурсу: http://csef.ru/ru/oborona-i-bezopasnost/272/sistemy-obnaruzheniya-vtorzhenij-520.

4. Shacklett M. E. What is attack vector? [Електронний ресурс] / Mary Edward Shacklett // TechTarget. – 2014. – Режим доступу до ресурсу: https://searchsecurity.techtarget.com/definition/attack-vector#:~:text=An%20attack%20vector%20is%20a,a%20payload%20or%20malicious%20outcome.&text=Common%20cyber%20attack%20vectors%20include,IMs)%2C%20chatrooms%20and%20deception..

5. Защита сервера от DDOS-атак [Електронний ресурс] // ITELON. – 2015. – Режим доступу до ресурсу: https://itelon.ru/blog/zashchita-servera-ot-ddos-atak/.

6. DDoS-атаки: типы атак и уровни модели OSI [Електронний ресурс] // ST-VDS. – 2014. – Режим доступу до ресурсу: https://firstvds.ru/technology/types-of-ddos.

7. What is an Intrusion Detection System (IDS)? [Електронний ресурс] // Check Point. – 2017. – Режим доступу до ресурсу:

https://www.checkpoint.com/cyber-hub/network-security/what-is-an-intrusion-detection-system-ids/#.

8. МЕТОД ТА АЛГОРИТМИ ВИЯВЛЕННЯ ПОРУШЕННЯ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ ПРИ DDOS-АТАКАХ [Електронний ресурс] // Хмельницький Національний Університет. – 2019. – Режим доступу до ресурсу: http://konkurs.khnu.km.ua/wp-content/uploads/sites/25/2019/04/DP2_Master-1.pdf.

9. Голяндина Н. Э. Метод "Гусеница" - SSA: анализ временных рядов [Електронний ресурс] / Нина Эдгаровна Голяндина // Санкт-Петербургский государственный университет. – 2004. – Режим доступу до ресурсу: https://www.gistatgroup.com/gus/ssa_an.pdf.

10. Francq C. RECENT RESULTS FOR LINEAR TIME SERIES MODELS WITH NON INDEPENDENT INNOVATIONS [Електронний ресурс] / C. Francq, J. Zakoıan // Springer. – 2005. – Режим доступу до ресурсу: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.721.1754&rep=rep1&type=pdf.

11. Айвазян С. А. Прикладная статистика. Основы эконометрики [Електронний ресурс] / Самвел Араратович Айвазян // Economix. – 2001. – Режим доступу до ресурсу: http://ecsocman.hse.ru/data/2010/11/09/1214795494/%D0%9A%D0%BD%D0%B8%D0%B3%D0%B0.pdf.

12. У чому полягає метод змінного середнього? [Електронний ресурс] // Stud. – 2007. – Режим доступу до ресурсу: https://stud.com.ua/23521/ekonomika/chomu_polyagaye_metod_zminnogo_serednogo

13. Apache Jmeter Guideline [Електронний ресурс] // Apache. – 2021. – Режим доступу до ресурсу: https://jmeter.apache.org/.

14. Учебное пособие по Jmeter [Електронний ресурс] // CoderLessons. – 2019. – Режим доступу до ресурсу: https://coderlessons.com/tutorials/java-tekhnologii/vyuchi-jmeter/uchebnoe-posobie-po-jmeter.

15. Load Testing Your Email Server: How to Send and Receive E-mails with JMeter [Електронний ресурс] // Blazemeter. – 2015. – Режим доступу до ресурсу: https://www.blazemeter.com/blog/load-testing-your-email-server-how-send-and-receive-e-mails-jmeter.

16. JMeter-тестирование динамической нагрузки Restful API [Електронний ресурс] // Nuances of Programming. – 2020. – Режим доступу до ресурсу: https://medium.com/nuances-of-programming/jmeter-restful-api-dbd4f29bc9d6.

17. How to Load Test TCP Protocol Services with JMeter [Електронний ресурс] // Blazemeter. – 2017. – Режим доступу до ресурсу: https://www.blazemeter.com/blog/how-load-test-tcp-protocol-services-jmeter.

18. Arthur D. k-means++: The Advantages of Careful Seeding [Електронний ресурс] / D. Arthur, S. Vassilvitski // Stanford University. – 2006. – Режим доступу до ресурсу: http://ilpubs.stanford.edu:8090/778/1/2006-13.pdf.

ДОДАТОК А. Відомість матеріалів кваліфікаційної роботи

№	Формат	Найменування	Кількість листів	Примітки
Документація				
1	А4	Реферат	1	
2	А4	Список умовних скорочень	2	
3	А4	Зміст	1	
4	А4	Вступ	4	
5	А4	Стан питання. Постановка задачі	16	
6	А4	Спеціальна частина	63	
7	А4	Економічний розділ	13	
8	А4	Висновки		
9	А4	Перелік посилань	3	
10	А4	Додаток А. Відомість матеріалів дипломної роботи.	1	
11	А4	Додаток Б. Перелік документів на оптичному носії.	1	
12	А4	Додаток В. Відгуки керівників розділів.	1	
13	А4	Додаток Г. Відгук керівника кваліфікаційної роботи	2	

ДОДАТОК Б. Перелік документів на оптичному носії

1 Презентація Хухрянський.pptx

2 Диплом Хухрянський.pdf

ДОДАТОК В. Відгуки керівників розділів

Відгук керівника економічного розділу:

Керівник розділу _____ _____

(підпис) (ініціали, прізвище)

ДОДАТОК Г. Відгук керівника кваліфікаційної роботи

Відгук

на кваліфікаційну роботу магістра на тему:

«Вдосконалення засобів раннього виявлення та протидії загрозам порушення інформаційної безпеки при DDoS-атаках»

студента групи 125м-20-1

Хухрянського Андрія Олександровича

Мета кваліфікаційної роботи – підвищення ефективності виявлення DdoS-атак і мінімізація втрат від реалізованої атаки шляхом фільтрації трафіку.

Тема кваліфікаційної роботи безпосередньо пов'язана з об'єктом діяльності фахівця за спеціальністю 125 Кібербезпека – розвиток методів виявлення атак.

Задачі кваліфікаційної роботи (дослідження різних варіантів DdoS-атак, аналіз існуючих систем виявлення вторгнень та їх методів знаходження атак, огляд існуючих механізмів протидій DdoS-атакам, запропонування алгоритму нечіткої логіки на основі k-means для визначення характеристик трафіку та його фільтрації, визначення статистичної вибірки і збір інформації про технічні ресурси системи завдяки інструменту Jmeter) віднесені в освітньо-кваліфікаційній характеристиці магістра до класу евристичних, вирішення яких основане на знаково-розумових уміннях фахівця.

Оригінальність технічних рішень полягає у запропонованих показниках рівня виявлення атаки.

Практичне значення результатів проектування полягає у можливості реалізації запропонованих рішень на базі існуючих програмно-апаратних засобах.

До недоліків кваліфікаційної роботи відносяться:

– недостатньо обґрунтовано показники рівня виявлення атак;

– не в повному обсязі проведено випробування запропонованої системи;

– не в повному обсязі було проведено дослідження алгоритму;

Оформлення пояснювальної записки до кваліфікаційного проекту виконано з деякими відхиленнями від стандартів.

Рівень запозичень у кваліфікаційній роботі не перевищує вимог положення про систему виявлення та запобіганню плагіату.

Ступінь самостійності виконання кваліфікаційної роботи висока.

За час дипломування Хухрянського А.О. виявив себе фахівцем, здатним самостійно, на високому рівні вирішувати поставлені задачі.

В цілому кваліфікаційна робота виконана у відповідності до вимог, що ставляться до кваліфікаційної роботи магістра, заслуговує оцінки “_____”, а Хухрянський А.О. присвоєння їй кваліфікації магістр з кібербезпеки, освітньо-професійна програма «Кібербезпека».

Керівник спеціальної частини
кваліфікаційної роботи магістра,
д.т.н, професор _____ В.І. Корнієнко

Керівник кваліфікаційної
роботи магістра,
д.т.н, професор _____ В.І. Корнієнко