Міністерство освіти і науки України
«Київський національний університет імені Тараса Шевченка»

Факультет інформаційних технологій
Кафедра кібербезпеки та захисту інформації

ПОЯСНЮВАЛЬНА ЗАПИСКА

Дипломної роботи
магістра
(назва освітньо-кваліфікаційного рівня)

галузь знань *12 Інформаційні технології*
(шифр і назва галузі знань)

спеціальність *125 Кібербезпека*
(код і назва спеціальності)

освітній ступень *магістр*
(назва освітньої програми)

освітньо-наукова програма *кібербезпека*

на тему: «Розробка моделей виявлення та ідентифікації мережевих атак»

Виконавець: студент ІІ курсу, групи КБм-21

(підпис)	**Гольшевський Андрій Денисович** (прізвище ім'я по-батькові)

	Прізвище, ініціали	**Оцінка**	**Підпис**
Науковий керівник	Бабенко Т. В.		
Рецензент			
Нормоконтроль	Даков С. Ю.		

Київ 2022

Міністерство освіти і науки України
«Київський національний університет імені Тараса Шевченка»

Факультет інформаційних технологій
Кафедра кібербезпеки та захисту інформації

ЗАТВЕРДЖЕНО:
завідувач кафедри кібербезпеки
та захисту інформації
_____ Н. В. Лукова-Чуйко
«__» _____ 2021 р.

ЗАВДАННЯ
на виконання дипломної роботи

спеціальності *125 Кібербезпека*
(код і назва спеціальності)

студента КБм-21 (група) Гольшевського Андрія Денисовича (прізвище ім'я по-батькові)

Тема дипломної роботи Розробка моделей виявлення та ідентифікації мережевих атак

1. ПІДСТАВИ ДЛЯ ПРОВЕДЕННЯ РОБОТИ

Рішення засідання кафедри кібербезпеки та захисту інформації факультету інформаційних технологій протокол № 5 від 29.10.2021

2. МЕТА ТА ВИХІДНІ ДАНІ ДЛЯ ПРОВЕДЕННЯ РОБІТ

Об'єкт досліджень	Процес виявлення та ідентифікації мережевих атак, які експлуатують вразливості логування та моніторингу
Предмет досліджень	Методи та моделі виявлення мережевих атак, які експлуатують вразливості логування та моніторингу
Мета	Розробити модель виявлення та ідентифікації мережевих атак, спрямованих на вразливості логування та моніторингу
Вихідні дані для проведення роботи	Методи виявлення мережевих атак

3. ОЧІКУВАНІ НАУКОВІ РЕЗУЛЬТАТИ

Наукова новизна вдосконалення методів виявлення та ідентифікації мережевих атак

Практична цінність можливість імплементації моделі в інформаційну систему підприємства малого та середнього бізнесу

4. ВИМОГИ ДО РЕЗУЛЬТАТІВ ВИКОНАННЯ РОБОТИ

Робота виконана у повному обсязі відповідно до теми.

5. ЕТАПИ ВИКОНАННЯ РОБОТИ

Найменування етапів робіт	**Строки виконання робіт (початок-кінець)**
Уточнення постановки задачі та аналіз літератури	29.10.2021 – 23.12.2021
Дослідження процесів логування та моніторингу	24.12.2021 – 07.02.2022
Аналіз основних моделей виявлення атак	08.02.2022 – 20.03.2022
Розробка моделі виявлення атак	21.03.2022 – 07.04.2022
Тестування та аналіз результатів моделі	08.04.2022 – 24.04.2022
Оформлення пояснювальної записки	25.04.2022 – 04.05.2022
Підготовка до захисту дипломної роботи	05.05.2022 – 19.05.2022

6. РЕАЛІЗАЦІЯ РЕЗУЛЬТАТІВ ТА ЕФЕКТИВНІСТЬ

Економічний ефект Зниження збитків через оперативне виявлення мережевих атак

Соціальний ефект Покращення технологій виявлення та протидії мережевим атакам на підприємствах.

7. ДОДАТКОВІ ВИМОГИ

Завдання видав _____ _____
(підпис) (прізвище, ініціали)

Завдання прийняв
до виконання _____ _____
(підпис) (прізвище, ініціали)

Дата видачі завдання: _____
Термін подання дипломної роботи до ЕК _____

УДК 004.056

РЕФЕРАТ

Пояснювальна записка до дипломної роботи «Розробка моделей виявлення та ідентифікації мережевих атак» складається зі вступу, основної частини, що містить 3 розділи, висновків і списку використаних джерел. Загальний обсяг роботи – 99 сторінок. Робота містить 18 рисунків та 1 таблицю. Список використаних джерел включає 36 джерел.

Об'єкт дослідження – процес виявлення та ідентифікації мережевих атак, які експлуатують вразливості логування та моніторингу.

Предмет дослідження – методи та моделі виявлення мережевих атак, які експлуатують вразливості логування та моніторингу.

Мета роботи – розробити модель виявлення та ідентифікації мережевих атак, спрямованих на вразливості логування та моніторингу.

Метод дослідження – синтез та аналіз моделей виявлення та ідентифікації мережевих атак методом імітаційного моделювання.

В роботі проведено аналіз параметрів логування та моніторингу за методологією OWASP, досліджено методи виявлення мережевих атак, визначено основні критерії щодо ідентифікації та управління подіями логування та моніторингу та рекомендації щодо впровадження моделей виявлення та ідентифікації мережевих атак для підприємств великого, середнього та малого бізнесу.

Практичне значення роботи полягає у розробці моделі виявлення мережевих атак, яка може бути імплементована в інформаційну систему підприємства малого та середнього бізнесу.

Результати здійснених у дипломній роботі досліджень можуть бути використані керівниками підприємств, спеціалістами із захисту інформації, спеціалістами із моніторингу інформаційної безпеки при виборі.

Ключові слова: логування, мережева атака, моніторинг ІБ, ризик, загроза, вразливість, інформаційна безпека, SIEM-система, контролі ІБ, метрики ІБ, інцидент, кібербезпека.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

ІБ – Інформаційна Безпека

ІС – Інформаційна Система

ІТ – Інформаційні Технології

НД – Нормативний Документ

ПЗ – Програмне Забезпечення

СУІБ – Система Управління Інформаційною Безпекою

IDS – Intrusion Detection System

SIEM – Security Information and Event Management

CVE – Common Vulnerabilities and Exposures

CVSS – Common Vulnerability Scoring System

ЗМІСТ

ВСТУП

Останнім часом тема інформаційної безпеки стала провідною майже в усіх галузях та сферах життєдіяльності людини. Кожен використовує паролі, має власну електронну скриньку, банківську карту, а тому знайомий з деякими видами ризиків та загроз інформаційної безпеки. Якщо ж збільшити масштаб до рівня організацій, компаній, державних підприємств, то ризиків та загроз стає більше і їх вплив несе набагато більше шкоди власникам активів.

В сучасному світі проблема завдання збитків та непоправної шкоди все міцніше закріплюється в кіберпросторі, компанії втрачають свої активи та репутацію, фізичні особи втрачають свої дані або стають жертвами інтернет-шантажу. Якщо ж говорити про державні органи та структури, то потенційні та реальні атаки на них є найбільш критичними за рівнем впливу. І хоча засоби кіберзахисту розвиваються такими ж кроками, як і механізми атак, проте стану абсолютної безпеки досягти неможливо.

В сьогоденні можна навіть побачити, що кіберпростір є однією з платформ ведення активних військових дій, де, на щастя, немає фізичних жертв, проте є критичні наслідки для інфраструктури та забезпечення безперервного процесу життєдіяльності окремих груп людей, тому використовувати вразливості мережі та засоби атаки можна й задля відновлення справедливості, як це демонструє своїм прикладом ІТ-армія України.

В даному дослідженні пропонується розглянути детальніше мережеві атаки, що використовують вразливості логування та моніторингу інформаційних систем (ІС) підприємств малого та середнього бізнесу. Ця проблема є не найпоширенішою чи найнебезпечнішою, проте доволі часто цю вразливість недооцінюють та не виконують заходи щодо зниження чи усунення її експлуатації, а головне – втрачають багато часу на виявлення таких загроз та відновлення безперервності роботи ІТ-систем.

РОЗДІЛ 1
СУЧАСНИЙ СТАН ПРОБЛЕМИ. ГОЛОВНІ НЕДОЛІКИ ВЕДЕННЯ ЛОГУВАННЯ ТА МОНІТОРИНГУ

1.1 Логування та моніторинг, сучасний стан

Ми живемо у складному світі з, здавалося б, безперервними заголовками про порушення, злам та інші підлі дії в Інтернеті. Програми безпеки повинні бути достатньо надійними, щоб протистояти постійним загрозам, які атакують організації сьогодні.

Спеціалісти з питань безпеки беруть до уваги безліч компонентів — ідентифікація та аутентифікація, контроль доступу, шифрування даних під час передачі й у стані спокою, цілісність даних, доступність системи та даних, керування постачальниками, реагування на інциденти, безпека персоналу, керування вразливістю, захист від шкідливих програм, безпека програм. тощо — тому вміння збирати, аналізувати й повідомляти інформацію, яка підтримує ці різні сфери, є надзвичайно важливою[1].

Майже всі серйозні інциденти з безпеки виникають через використання недостатнього рівня логування, незапланованих стратегій безпеки або недостатнього моніторингу. Підприємства, які використовують програми з недостатніми функціями реєстрації або без них, ризикують приймати атаки так довго, що вони можуть завдати значної шкоди всьому технологічному стеку.

Функції ведення журналів і моніторингу надають адміністраторам і командам безпеки необроблені дані про трафік, які допомагають виявляти потенційні загрози, виявляючи незвичайні закономірності. Ці механізми є основними опорами безпеки, які формують основу надійно керованої системи безпеки[2].

За відсутності ретельно спланованих механізмів ведення журналів організація пропускає контрольний слід для аналізу безпеки, тим самим даючи векторам атак достатньо часу для подальшого проникнення в кілька компонентів екосистеми.

Атаки, засновані на недостатньому моніторингу та вразливості журналів, зазвичай мають високі рейтинги за поширеністю, середні за можливостями та низькі за виявленням . Забезпечення реєстрації всіх подій і моніторингу подій, як наслідок, часто вважається першим кроком у виявленні вторгнень.

Якщо в роботі сервера, комп'ютера або програмного забезпечення виникла невідома помилка, перш за все дивляться логи. Лог — текстовий файл, який містить інформацію про дії програмного забезпечення або користувачів, який зберігається на комп'ютері або сервері[3]. Це хронологія подій та їх джерел, помилок та причин, через які вони відбулися. Читати та аналізувати логи можна за допомогою спеціального ПЗ.

Логуванням називають запис системних подій. Воно дозволяє відповісти на питання, що відбувалося, коли і за яких обставин. Без логів складно зрозуміти, через що з'являється помилка, якщо вона виникає періодично і лише за певних умов. Щоб полегшити завдання адміністраторам і програмістам, записується інформація не тільки про помилки, але і про причини їх виникнення.

Логи повинні записуватись під час роботи кожного IT-компонента.

Типові випадки, у яких застосовуються логи:

• Адміністратор шукає причини виникнення технічних проблем, збоїв у пристрої або операційній системі та недоступності сайту.

• Розробник проводить дебаг, тобто шукає, локалізує та усуває помилки.

• SEO-фахівці збирають статистику відвідуваності, оцінюють якість цільового трафіку.

• Адміністратор інтернет-магазину відстежує історію взаємодії з платіжними системами та дані щодо змін у замовленнях.

Типи логів

Існують різні рівні та різні подробиці логування. Коли помилку складно відтворити, використовують дуже докладні логи; якщо це не потрібно, збирають лише ключову інформацію. Для роботи з логами та пошуком інформації у великих текстових даних використовують спеціалізовані інструменти.

Для зручної роботи з логами їх поділяють на типи. Це допомагає швидше знаходити потрібні та вибирати правильні інструменти для роботи з ними. Наприклад, виділяють[3]:

• системні логи, тобто ті, що пов'язані із системними подіями;

• серверні логи, що реєструють звернення до сервера і помилки, що виникли при цьому;

• логи баз даних, що фіксують запити до баз даних;

• поштові логи, що стосуються вхідних/вихідних листів та відслідковують помилки, через які листи не були доставлені;

• логи авторизації;

• логи аутентифікації;

• логи програм, встановлених на цих операційних системах.

Також логи можна типізувати за ступенем їх важливості:

• Fatal/critical error – те, що потрібно терміново виправити.

• No critical error — помилки, які не впливають на користувача.

• Warning – попередження про те, на що потрібно звернути увагу.

• Initial information — інформація про виклики API сервісу, запити у БД, виклики інших сервісів.

Логування та моніторинг

Logging (логування) використовується як дієслово, так і іменник, посилаючись або на практику реєстрації помилок і змін, або на журнали програми, які збираються. Метою ведення журналу є створення поточного запису подій програми. Файли журналів можна використовувати для перегляду будь-яких подій у системі, включаючи збої та зміну стану. Отже, повідомлення журналу можуть надати цінну інформацію, яка допоможе визначити причину проблем з продуктивністю. Дані журналу можуть допомогти командам DevOps усунути неполадки, визначивши, які зміни призвели до повідомлення про помилки, але цінні лише стільки, скільки міститься в них інформація.

Керування журналами також служить іншим цілям, таким як створення письмових записів для цілей аудиту та відповідності, визначення тенденцій з часом і

захист конфіденційної інформації. Ведення журналів відіграє важливу роль у програмах будь-якого розміру, але має бути реалізовано продумано. Уникайте зберігання, передачі або оцінки сторонньої інформації, встановлюючи пріоритетні дії. Реєстрація занадто великої кількості даних може призвести до втрати ресурсів як з точки зору витрат, так і часу. Хороша стратегія ведення журналів зазвичай передбачає два типи даних: структуровані дані для машин і дані, які попереджають системних адміністраторів про потенційну проблему.

Моніторинг — це загальний термін, який може включати багато аспектів оцінки системи, але в цьому контексті ми маємо на увазі моніторинг продуктивності програми (APM)[4]. APM – це процес інструментування програми для збору, узагальнення та аналізу показників для кращої оцінки використання системи шляхом вимірювання доступності, часу відгуку, використання пам’яті, пропускної здатності та споживання часу ЦП.

Системи моніторингу покладаються на показники, щоб попередити IT-команди про аномалії в роботі програм і хмарних сервісів. В ідеалі команди впроваджують прилади та моніторинг на всіх системах.

Ведення журналів і моніторинг – це два різні процеси, які працюють разом, щоб забезпечити низку точок даних, які допомагають відстежувати працездатність та продуктивність вашої інфраструктури. APM використовує показники програми для вимірювання доступності та керування продуктивністю. Ведення журналу створює запис подій журналу, створений програмами, пристроями або веб-серверами, який служить детальним записом подій у системі.

Використання комбінації керування журналами для збору, упорядкування та перегляду даних та інструментів моніторингу для відстеження показників дає вичерпне уявлення про доступність вашої системи разом із детальним уявленням про будь-які проблеми, які потенційно можуть вплинути на роботу користувачів. APM розповідає вам, як поводяться програми, і реєструє дані додатків, мережевої інфраструктури та веб-серверів, що дає змогу краще зрозуміти, чому програма працює так, як вона є. Ефективна стратегія ведення журналів покращує моніторинг продуктивності програми.

У метафоричному сенсі показники моніторингу схожі на охоронну сигналізацію, яка попереджає про можливе вторгнення; файли журналів діють як кадри з камери відеоспостереження, які надають підказки, щоб розповісти, що і як сталося.

Існують деякі випадки використання, коли знадобиться лише логування чи моніторинг, але наявність обох рішень дає більшу здатність до повного розуміння вашої системи та її вразливостей.

Зрештою, мета — підтримувати працездатність додатків і користувацький досвід. Завдяки інтеграції журналів і моніторингу для досягнення цієї мети розробники та оперативні команди можуть швидше планувати та вирішувати проблеми з додатками.

Журнал безпеки — це запис інформації про те, що відбувається у мережі. Журнали безпеки включають все, від журналів брандмауера до журналів подій та журналів програм. Кожен тип журналу надає різну інформацію[2].

Наприклад, журнал брандмауера може записувати вихідну IP-адресу кожного пакету, надісланого або отриманого пристроєм. Журнали подій записують дату, час і деталі кожної події, що відбувається на комп’ютері. Журнали програми записують назву процесу, який був запущений, коли сталася подія.

Адміністратори інформаційних систем повинні вміти розпізнавати, коли відбувається напад. Це можна зробити двома способами:

1) Відстежувати свої журнали.

2) Використовувати програмні інструменти, щоб проаналізувати журнали.

Чим більше журналів відстежується, тим більше шансів виявити шкідливі дії.

Системний журнал — це набір окремих записів, які представляють конкретну діяльність, події, умови помилок, несправності або загальний стан інформаційної системи чи мережі[1]. Ці записи журналу містять важливі дані, які допомагають адміністраторам системи та безпеки зрозуміти, що відбувається в інформаційній системі.

Оскільки багато журналів містять інформацію, пов’язану з безпекою (наприклад, події автентифікації, зміни доступу, зміни в конфігурації системи

тощо), журнали допомагають адміністраторам знати про потенційну (або фактичну) шкідливу активність у системі або зміни, які можуть послабити систему. положення безпеки системи.

Якщо журнали не генеруються системою, адміністратори, персонал безпеки та команди розробників, по суті, не бачать діяльності в системі. Вони не будуть знати про потенційно зловмисні дії в системі і, звичайно, не зможуть на них реагувати. Вони не знають, як сталася компрометація і куди зловмисник повернувся з моменту першого порушення. Спроби та успішні атаки залишаться непоміченими на невизначений час, поки дані вилучаються, а їх зловмисна присутність зберігається.

Ведення журналів аудиту також важливо для підтримки будь-яких необхідних федеральних законодавчих або нормативних вимог, таких як ті, що викладені у Федеральному законі про управління інформаційною безпекою (FISMA) або в Законі про переносимість та підзвітність медичного страхування (HIPAA).

Політика реєстрації подій та моніторингу

Існує ряд факторів (наприклад, схильність до ризику, обсяг журналу, релевантність безпеки тощо), які сприяють вирішенню того, що має бути включено в конфігурації журналів безпеки організації. Існують також інші журнали, які використовуються для продуктивності, доступності, умов помилок тощо.

Те, що можливість генерувати події журналу доступна, не означає, що її потрібно ввімкнути, оскільки це лише додасть потенційного перевантаження журналу, яке є поширеним у багатьох середовищах. По суті, політика ведення журналів безпеки та моніторингу організації повинна керувати тим, що реєструється, як журнали передаються, ротація журналів, збереження, зберігання тощо.

Однією з основних причин увімкнення журналів безпеки є підтримка криміналістичних розслідувань щодо потенційних або реалізованих порушень. Тому важливо реєструвати події, які підтримають розслідування порушень, наприклад такі[1]:

• Успішні та невдалі події входу

• Дії з управління обліковим записом (наприклад, створення, зміна, видалення)

• Використання привілейованих команд в операційній системі та для програм

• Зміни в авторизації

• Доступ до даних, модифікації та видалення критичних наборів даних

Для кожної з наведених вище подій аудиту запис аудиту також повинен визначати, коли сталася подія (через позначку часу), де сталася подія (наприклад, IP-адреса хоста), джерело події, результати події (наприклад, успішний або не вдалося), а також будь-яку ідентифікаційну інформацію (наприклад, ім’я облікового запису) для осіб/процесів, що діють від імені осіб, які виконали дію.

Моніторинг журналів логування

Журнали безпеки не мають практично ніякої цінності, якщо вони не контролюються. Насправді зловмисники підстраховують свою ставку тим, що їх ціль не стежить за своїми журналами.

Моніторинг журналу, по суті, перевіряє записані дані журналу на наявність аномальних, ненормальних або підозрілих подій. Хоча моніторинг журналів можна виконувати вручну, він неефективний і має бути зарезервований для більш детального аналізу, спровокованого автоматизацією.

Важливість моніторингу подій безпеки за допомогою журналів неможливо недооцінити. Своєчасність є ключовою, без активного моніторингу журналу ймовірність того, що зловмисник зберігатиме невиявлену постійну присутність, значно зростає. Незважаючи на те, що попередження порушень є вкрай бажаним, виявлення порушення є обов’язковим, а основним механізмом виявлення порушень є виявлення аномальної активності в журналах безпеки.

Недостатній моніторинг управління журналами є основною причиною, чому компанії не можуть ефективно усувати інциденти безпеки. Це дозволяє компаніям застосовувати правильний підхід до реагування та коригувальні заходи для забезпечення безпеки систем у майбутньому. В результаті недостатнє ведення журналів і моніторинг створюють унікальний рівень вразливості, який залишається популярним аспектом експлойту зловмисника.

Це підтверджується тим фактом, що станом на 2018 рік 35% організованих зломів були безфайловими, оскільки атаки на основі файлів легше виявити за

допомогою традиційних механізмів реєстрації та моніторингу. У незмінному звіті також зазначено, що понад 93% порушень безпеки , здійснених у 2017 році, можна було запобігти за допомогою базових заходів з обліку та моніторингу[5].

Згідно даних Open Web Application Security Project (OWASP) – Відкритого проекту безпеки веб-додатків, проблема безпеки логування та моніторингу входить до переліку десяти найбільш критичних вразливостей у веб-просторі [6]:

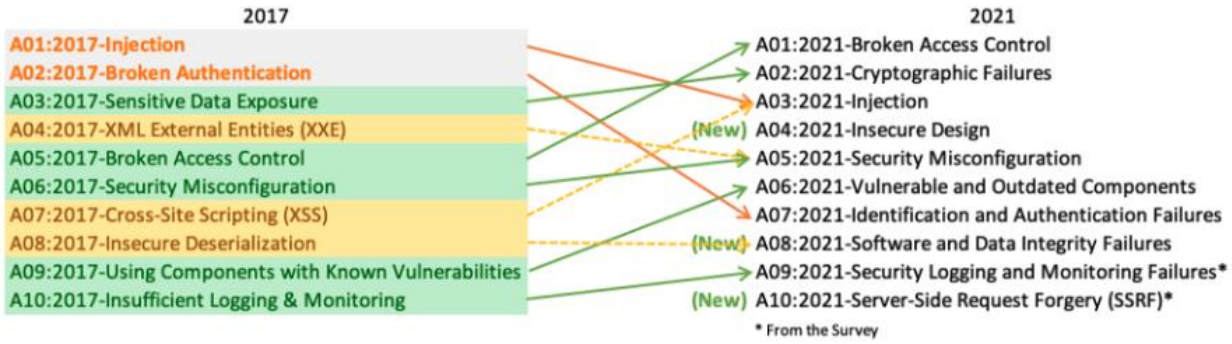

Рисунок 1.1 – Перелік вразливостей OWASP Top-10

Більш того, порівняно з дослідженням 2017-го року ця проблема піднялась в рейтингу на одну позицію вище, а саме на дев’яту.

Серед головних факторів знаходження даної вразливості в рейтингу велика кількість типів помилок в журналах логів, складність тестування логування та моніторингу, невеликий об’єм інформації щодо проблематики вразливості та безпосередній вплив на спостережність, сповіщення про інциденти та їх розслідування.

Тестування журналу логування та моніторингу може бути складним, часто передбачаючи інтерв’ю або запити, чи були атаки виявлені під час тестування проникнення. Для цієї категорії не так багато даних CVE/CVSS, але виявлення порушень і реагування на них є критичними. Тим не менш, це може дуже вплинути на підзвітність, видимість, оповіщення про інциденти та криміналістичну експертизу.

Повертаючись до топ-10 OWASP за 2021 рік, ця категорія має допомогти виявляти, ескалувати та реагувати на активні порушення. Без логування та

моніторингу порушення неможливо виявити. Проблеми логування, виявлення та моніторингу можуть виникнути будь-коли, наприклад[7]:

• Події, які підлягають аудиту, такі як вхід у систему, невдалий вхід та транзакції високої вартості, не реєструються.

• Попередження та помилки не генеруються, неадекватні або нечіткі повідомлення журналу.

• Журнали програм і API не відстежуються на предмет підозрілої активності.

• Журнали зберігаються лише локально.

• Відповідні пороги попередження та процеси ескалації реагування не діють або не ефективні.

• Тестування на проникнення та сканування за допомогою інструментів динамічного тестування безпеки додатків (DAST) не ініціюють попередження.

• Програма не може виявляти або сповіщати про активні атаки в режимі реального часу або майже в реальному часі.

Інформаційні системи вразливі до витоку інформації, роблячи журнали та події видимими для користувача або зловмисника (про що більш детально згадується в іншій категорії вразливостей OWASP Top 10, A01:2021-Контроль порушення доступу).

Приклад сценаріїв подібних атак

А) Велика індійська авіакомпанія мала порушення даних, пов’язане з персональними даними мільйонів пасажирів на суму понад десять років, включаючи дані паспортів і кредитних карток. Злом даних стався у стороннього провайдера хмарного хостингу, який через деякий час повідомив авіакомпанію про порушення.

Б) Велика європейська авіакомпанія зазнала порушення GDPR. Повідомляється, що порушення було спричинено вразливими місцями безпеки платіжних програм, якими скористалися зловмисники, які зібрали понад 400 000 записів про платежі клієнтів. Регулятор конфіденційності оштрафував авіакомпанію на 20 мільйонів фунтів стерлінгів.

Хакери використовують прогалини в реєстрації та моніторингу, покладаючись на той факт, що командам безпеки знадобиться час, щоб виявити та усунути атаку,

щоб спробувати посилити привілеї. У цьому розділі досліджуються загрози, пов’язані з недостатнім обліком і моніторингом, а також наслідки успішної атаки на бізнес.

Основна причина використання неадекватно зареєстрованої системи векторами атаки, як правило, ґрунтується на наступних недоліках, які виникають за відсутності ефективної системи реєстрації та моніторингу[2]:

• Незареєстровані події та транзакції

• Відсутні резервні копії журналів

• Неясний журнал помилок

• Відсутні плани ескалації порушень

• Погане керування аутентифікацією

• Неефективне навчання з обліку та моніторингу

• Відсутність експорту для аналізу даних журналу

• Неправильні налаштування програмного забезпечення

Приклади недостатнього протоколювання та моніторингу атак

Без належного моніторингу та реєстрації мережевого трафіку підприємствам не вдається запобігти встановленню зловмисникам шкідливого програмного забезпечення та доступу до важливих даних. Нижче наведено деякі з відомих прикладів інцидентів із безпекою, які виникають через недостатнє ведення журналів та моніторингу в новітній історії:

Атака хробака Stuxnet на ядерну програму Ірану

Хробак Stuxnet — це майстерно створене зловмисне програмне забезпечення, яке атакує системи контролю та збору даних (SCADA). У 2010 році група безпеки іранської ядерної програми виявила, що помилка використовувалася для доступу до критичних систем контролю зброї.

При більш глибокому аналізі виявилося, що помилка активна з 2005 року і поширювалася за допомогою заражених USB-накопичувачів. Хакери скористалися поганими механізмами ведення журналів і моніторингу, щоб непомітно отримати підвищений доступ.

Злом даних Verizon Communications 2017 року

Хоча жодних даних не було вкрадено, Verizon визнає, що щонайменше 14 мільйонів записів клієнтів були відкриті в Інтернеті в результаті порушення даних, виявленого в 2017 році. Ці записи включали такі дані, як номери телефонів та PIN-коди облікового запису. Ці дані не були захищені паролем, і зловмисники могли легко їх завантажити та використати.

Однак записи зберігалися в хмарному сховищі даних і були виявлені дослідником кібербезпеки до того, як зловмисники змогли скористатися лазівкою.

Порушення національних даних Dominion 2019 року

У 2019 році страхова компанія Dominion National виявила, що учасники її програми охорони здоров'я могли зазнати зламу даних, який тривав понад дев'ять років. Порушення, яке, як було встановлено, торкнулося понад 2 мільйонів осіб, оприлюднило конфіденційні дані клієнтів, зокрема:

- Номери банківських рахунків
- Маршрутні номери
- Ідентифікаційні дані платника податків
- Номери соціального страхування
- Імена та дати народження серед іншого

Після вичерпного розслідування було встановлено, що сторонні особи не отримували доступ до цієї інформації та не використовували її. Однак Dominion National було наказано покрити будь-які претензії щодо грошових збитків, які розумно простежуються до порушення.

1.2 Головні недоліки ведення логування та моніторингу та загрози, які їх експлуатують

Безсумнівно, що керування журналами є дуже важливою проблемою, враховуючи потребу в адекватних інструментах керування журналами, які можуть задовольнити вимоги безпеки та відповідності.

Повідомлення журналу – також відомі як журнали подій, записи аудиту та журнали аудиту – документують обчислювальні події, що відбуваються в IT-

середовищі[8]. Повідомлення журналу, створені або ініційовані програмним забезпеченням або користувачем, забезпечують видимість і документацію майже кожної дії в системі. Отже, маючи все це на увазі, досліджено всі найбільші проблеми керування журналами в сучасних IT та рішення цих проблем.

Основними проблемами, що стосуються ведення журналів безпеки та моніторингу, є величезний обсяг журналів, які створюються інформаційними системами та додатками, а також відсутність навченого персоналу безпеки для виявлення ненормальних подій за допомогою SIEM або інших автоматизованих методів[1].

Додаткові проблеми включають різні формати журналів на основі ОС або програми, які генерують журнал, різний вміст журналу, що ускладнює відстеження потоку на кількох платформах, і нестандартизовані позначки часу. На щастя, сучасні платформи SIEM можуть нормалізувати записи журналу в звичайний формат для аналізу, зберігаючи при цьому оригінальний запис журналу, якщо це потрібно для підтримки більш глибокого аналізу.

Найбільш поширеними проблемами ведення логування та моніторингу є наступні[9, 10]:

Високий обсяг журналу

Виходячи з визначення журналу Національного інституту стандартів і технологій (NIST), яке стверджує, що «журнал — це запис подій, що відбуваються в системах і мережах організації», ми можемо зробити висновок, що всі IT-компоненти створюють дані журналу (логи). Брандмауери, програмне забезпечення безпеки, комп’ютерні системи і навіть операційні системи створюють журнали подій системи Windows.

Щоб організація відповідала вимогам безпеки, аналізатор журналів повинен належним чином перевірити кожен журнал. Управління цим великим обсягом даних журналу, особливо у великій мережі, створює велике навантаження на IT-ресурси. Отже, інструменти моніторингу журналів старої школи не настільки ефективні, як повинні бути.

Різноманітність форматів

Кожен журнал походить з іншого джерела. Таким чином, кожне джерело має різний формат для створення та звітування журналів інструментам моніторингу журналів для аналізу. Щоб вирішити цю проблему, багато рішень моніторингу журналів подій мають загальний формат журналу, але не всі журнали можуть відповідати одному формату.

Таким чином, навіть із впровадженням загального формату журналу немає гарантії, що всі вхідні дані журналу будуть мати формат, подібний до того, що зараз аналізується. Це, у свою чергу, потребує більше зусиль і часу, щоб знайти ключову інформацію та інтерпретувати її за потреби.

Правильний аналіз інформації

Це призводить до наступної поширеної проблеми, правильного аналізу інформації в кожному журналі. При аналізі журналу системних подій необхідно робити все максимально точно. Це означає, що інформацію слід проаналізувати рядок за рядком. Існують різні рішення для цього, але для правильного аналізу інформації керування журналом потрібно адмініструвати вручну.

Крім того, якщо ключова інформація відсутня, потрібно вручну відстежити її та визначити, чому вона була відсутня в журналах спочатку. Ці проблеми роблять аналіз інформації великою проблемою для IT-адміністраторів, оскільки вони повинні встановити правила, спочатку знаючи, що їм потрібно шукати.

Швидкість

Ще однією проблемою звичайного керування журналами є швидкість. Враховуючи величезну кількість інформації, зібраної з сервера журналів для аналізу та зберігання, а також необхідність належного аналізу цієї інформації вручну, ефективне керування журналами вимагає більше часу. Він або швидкий, або добре представлений, але не обидва. Зберігати правильний баланс між обома – складно.

Кореляція журналу

Підприємства, особливо великі, щодня генерують вражаючу кількість даних. Тому, хоча більшість служб керування журналами можуть збирати ці дані, важливість керування журналами полягає у співвіднесенні даних.

Кореляція журналів – це процес встановлення зв’язків між подіями, що відбуваються в різних системах або на різних пристроях, щоб допомогти виявляти та досліджувати проблеми. За допомогою правильного інструменту аналізу журналу ви можете об’єднати різні події журналу, які інакше могли б здатися не пов’язаними.

Простіше кажучи, це різниця між «хорошою» та «поганою» активністю даних. Помилковий результат може здатися великою проблемою без відповідних протоколів і розширеної аналітики. У той же час серйозне порушення безпеки може маскуватися як нешкідлива стандартна діяльність.

У вік великих даних (які стають все більшими і більшими з кожною секундою) кореляція журналів є важливою частиною управління журналами, яка використовується для всього, починаючи від кібербезпеки, системного адміністрування та дотримання обов’язкових процедур аудиту.

Рішення з журнальною кореляцією дозволяє:

• Отримувати високоточні сповіщення, щоб виключити помилкові спрацьовування

• Розставити пріоритети попереджень на основі рівня ризику

• Використовувати розвідку про загрози, щоб допомогти виявити та дослідити ознаки компромісу

• Встановити пакети вмісту для спільного використання конфігурацій із попередньо вбудованими вводами, обробкою даних, шаблонами відображення, попередженнями та звітами

Коли і що автоматизувати

Як і більшість типів програмного забезпечення, керування журналами в значній мірі залежить від автоматизації. Дані, які практично неможливо відсортувати (людською) рукою, впорядковуються, сортуються та аналізуються програмою. Інструмент керування журналами (LMT) зробить усе можливе, щоб виконати роботу відповідно до (попередньо) встановлених параметрів, але це навряд чи є ідеальним рішенням.

Нові загрози та проблеми виникають щодня. У той час як LMT розроблено, щоб допомогти визначити та впоратися з ними, потрібен спеціальний працівник, який відповідає за його функціонування та налаштування, щоб виявити реальні переваги найкращих методів керування журналами.

Знати, що автоматизувати, а що робити самому, є набутим навиком саме по собі. І, як і будь-який інший навик, він вимагає практики, часу, навчання та людської відданості, щоб отримати від нього максимум.

Зберігання та архівування

Архівування файлів журналів зменшує обсяг даних, які потрібно зберігати на локальних серверах і жорстких дисках. Залежно від ваших потреб і вимог до відповідності, дані журналу зазвичай зберігаються локально протягом 30 днів. Однак можна налаштувати систему так, щоб дані журналу історичних даних визначали точку входу в проблему або вторгнення, яке сталося кілька місяців (або навіть років).

Хоча історичні дані використовуються для проблем безпеки та системи, вони в першу чергу для цілей аудиту. Деякі регулятивні аудити вимагають, щоб зберігались дані журналу протягом трьох-п’яти років, а інші можуть навіть вимагати їх збереження назавжди.

Ці журнали стискаються у форматі без втрат, щоб зменшити розмір журналу замість того, щоб зберігатися в необробленому вигляді. Є можливість імпортувати їх у свою програму, коли виникне бажання або необхідність. Це робить аудит набагато більш безболісною та швидкою процедурою.

Відсутність дружнього користувацього інтерфейсу

Мало що може так розчарувати користувача, як неінтуїтивно зрозумілий, погано виконаний інтерфейс.

Досвід користувача є основою того, як користувач взаємодіє з програмою. Інтерфейс користувача, який не є відразу зрозумілим і точним у своїй візуальній мові, може призвести людських та системних помилок.

Хороший досвід роботи з користувачем створюється, якщо думати як кінцевий користувач і розуміти:

• Які дані потрібні кінцевому користувачеві

• Як людина використовує інструмент

• Який робочий процес дотримується кінцевий користувач

• Потрібні різні типи параметрів перегляду, як-от графіки та кругові діаграми

Звітність та функції пошуку

Недостатньо розвинені функції пошуку та звітності є поширеними проблемами, які виникають у багатьох інструментах керування журналами. З даними файлу журналу, які вимірюються в терабайтах, необхідно мати можливість виконувати поглиблений і швидкий пошук має першорядне значення.

Аналогічно, налаштування звітів має бути інтуїтивно зрозумілим і функціональним. Незалежно від часу дня (або ночі), все може піти не так. Важливо, щоб звіти надсилалися, як тільки виникає проблема, і надходили до відповідних людей. Звітність також має бути налаштовуваною, з можливістю надсилання щоденних/щотижневих/місячних звітів електронною поштою за потреби.

Сучасні інформаційні системи складні, і для зловмисників є багато шляхів їх використання. Недостатнє ведення журналів і моніторингу дають змогу зловмисникам спочатку використовувати і покидати системи без виявлення. Тому надзвичайно важливо, щоб організації впроваджували програму реєстрації та моніторингу безпеки, керуючись своєю політикою ведення журналів безпеки та моніторингу.

Загрози, пов’язані з недостатнім логуванням та моніторингом[2]

Атаки ботнетів

Зловмисники часто використовують кілька пристроїв, підключених до Інтернету, для введення шкідливого програмного забезпечення в систему та координації кібератаки. Такими шкідливими програмами є автоматизовані боти , які маніпулюють програмою різними способами – від простих операцій розсилки спаму до виконання більш складних атак, призначених для маніпулювання програмою.

Вони також зазвичай підтримуються бот -мережами , які організовують різні атаки, включаючи атаки грубої сили, фішинг і розподілену відмову в

обслуговуванні (DDoS) . Атаки ботнетів покладаються на ланцюжок дій, що проходить через кілька етапів. За відсутності належної реєстрації даних подій ці атаки майже неможливо виявити чи проаналізувати.

Ефективна система моніторингу з такими інструментами, як Syslog, часто вважається основною першою лінією захисту, щоб зменшити ймовірність і серйозність атак через ботнет.

DNS атаки

Служба доменних імен (DNS) пропонує стандартний механізм для вказівки імен хостів машин на їхні IP-адреси. Оскільки DNS спрямовує мережевий трафік на правильні веб-сервери та цільові машини, це звичайні вразливі точки, які часто експлуатуються векторами атаки для націлювання на доступність або стабільність DNS-сервера як частину загальної стратегії атаки.

Деякі можливі атаки DNS включають:

• Отруєння кеша

• Розподілене відображення DoS-атаки

• Атаки NXDOMAIN

• Тунелювання DNS

• Випадкові атаки на субдомен

• Атака блокування домену

Якщо події на основі DNS не реєструються та не відстежуються належним чином, адміністратори не будуть знати типи машин, які зловмисники (під виглядом користувачів) запитують і з якими взаємодіють. Крім того, суб’єкти загроз можуть продовжувати зловмисні дії, такі як встановлення зловмисного програмного забезпечення, крадіжка облікових даних, комунікація з командою та керуванням, відбиток мережі та крадіжка даних за відсутності належної реєстрації та аналізу запитів .

Інсайдерські загрози

Організації, які зазвичай інвестують цілі статки в захист систем від зовнішніх атак, часто помиляються в розрахунках внутрішніх загроз. Такі внутрішні загрози продовжують викликати критичне занепокоєння організацій, оскільки їх підозріла

діяльність часто не контролюється. У таких випадках зловмисні або скомпрометовані інсайдери становлять серйозну загрозу для систем, оскільки вони мають доступ до різних заходів контролю та безпеки. Хоча подібна ситуація звучить дивовижно, пом’якшення є відносно простим і простим, що покладається на ефективний механізм реєстрації.

Недостатній моніторинг та керування журналами в таких випадках призводять до невідстежуваних моделей поведінки користувачів, що дозволяє самозванцям або зловмисним інсайдерам скомпрометувати систему на набагато більш глибокому рівні.

Деякі загальновідомі інсайдерські загрози, які виникають через недостатнє ведення журналів і моніторинг, включають:

• Трафік зловмисного програмного забезпечення

• Атаки програм-вимагачів

• Розширені постійні загрози

• Як зловмисники використовують недостатню реєстрацію та моніторинг

Без реєстрації важливої інформації про безпеку адміністратори безпеки не отримують сповіщення про будь-які незвичайні події, що перетворює кожну вразливість у потенційний злом і створює ризик подальшої ескалації привілеїв . Зазвичай це робиться в наступному порядку:

1) Після того, як зловмисник отримав доступ до системи, він намагається максимально приховати свою присутність та особистість. Для систем, які не мають комплексного керування журналами, хакери навіть намагаються стерти журнали подій, які можуть викликати тривогу.

2) Потім зловмисники намагаються використовувати ділянки веб-сервера, які були розроблені без дотримання найкращих методів безпеки. Типові активні атаки починаються з того, що хакер перевіряє систему на наявність уразливостей безпеки. Потім вони використовують переваги неефективного реагування на інциденти та усунення несправностей, щоб поглибити контроль над системою або отримати доступ до більш важливих даних. Оскільки час реагування на недостатню кількість інцидентів ведення журналу та моніторингу тривалий, як правило, 150-200 днів, ці

суб’єкти загрози мають значний час, щоб стримано перевірити наявність більш привілейованого доступу.

Хакери, як правило, використовують добре відомі передові стратегії атак, щоб охопити більше території після того, як вони отримали початковий доступ. Деякі з них включають:

Атаки паролем – різні методи спрямовані на отримання несанкціонованого доступу до облікових записів користувачів. Деякі методи атаки на паролі включають грубу силу, атаки за словниками та перебір паролів .

Розширені постійні загрози – зловмисники отримують доступ до мережі та залишаються непоміченими, зазвичай відстежуючи трафік для отримання важливих даних.

Атака «Людина в середині» (MITM) – актор загрози перехоплює та змінює повідомлення між сервером і клієнтом (або двома сторонами, що спілкуються). Такі атаки включають підслуховування Wi-Fi , зловживання сеансами та електронну пошту .

Атака «Відмова в обслуговуванні» – як тільки зловмисники отримують початковий доступ до системи, вони намагаються вимкнути мережу/машину та зменшити її здатність відповідати на запити користувачів, наповнюючи сервер величезним трафіком, створеним ботом.

Також певна кількість проблем, пов’язаних з логуванням та моніторингу, зареєстрована в CWE (Common Weakness Enumeration) – базі загального переліку дефектів (недоліків) безпеки:

• Неправильна нейтралізація виводу для журналів [11]

• Упущення інформації, що стосується безпеки [12]

• Потрапляння конфіденційної інформації у файл журнал [13]

• Недостатнє логування [14]

В якості ще однієї з можливостей експлуатації вразливостей логування OWASP виділяють Log Injection[15]. Програми зазвичай використовують файли журналів для зберігання історії подій або транзакцій для подальшого перегляду, збору статистичних даних або налагодження. Залежно від характеру програми,

завдання перегляду файлів журналів може виконуватися вручну за потреби або автоматизовано за допомогою інструмента, який автоматично відбирає журнали для важливих подій або актуальної інформації.

Запис недійсних даних користувача до файлів журналів може дозволити зловмиснику підробити записи журналу або ввести шкідливий вміст у журнали. Це називається ін’єкцією журналу.

Уразливості ін’єкції журналу виникають, коли[15]:

• Дані надходять до програми з ненадійного джерела.

• Дані записуються в файл журналу програми або системи.

Успішні атаки ін'єкції журналу можуть викликати:

• Введення нових/підроблених подій журналу (підробка журналів за допомогою ін'єкції журналу)

• Ін’єкція атак XSS, сподіваючись, що шкідлива подія журналу буде переглянута у вразливому веб-додатку

• Введення команд, які можуть виконуватися парсерами (наприклад, PHP).

Підробка логів

У найбільш безпечному випадку цієї атаки зловмисник може вставити помилкові записи до файлу журналу, надавши програмі вхідні дані, які містять відповідні символи. Якщо файл журналу обробляється автоматично, зловмисник може зробити файл непридатним для використання, пошкодивши формат файлу або ввівши неочікувані символи. Більш тонка атака може включати перекос статистики файлу журналу. Підроблені чи іншим чином пошкоджені файли журналів можуть бути використані, щоб закрити сліди зловмисника або навіть залучити іншу сторону до вчинення зловмисного дійства.

Наслідки атак недостатньої реєстрації та моніторингу на бізнес[2]

Без належних механізмів реєстрації та моніторингу організаціям значно важче виявляти та пом’якшувати порушення, що коштує підприємствам часу та грошей. Деякі наслідки недостатньої реєстрації та моніторингу атак включають:

Недоступність системи

Зловмисники, які планують здійснити атаку «Відмова в обслуговуванні» (DoS), зазвичай заповнюють цільовий сервер трафіком, доки сервер не вийде з ладу або не відповість. Ця атака грубої сили означає, що сервер перевантажений, а послуги стають недоступними для законних користувачів. Зловмисники також гарантують, що атака нагадує проблему доступності, яка не є шкідливою, що робить їх ще важче відстежити.

Порушена конфіденційність

Журнали подій зазвичай містять конфіденційну інформацію про користувачів і систему. Зловмисники, які мають доступ до системних журналів, мають необмежений доступ до цих даних, які вони можуть використовувати для інших шкідливих цілей. Неналежні механізми реєстрації та моніторингу дають зловмисникам доступ до приватної інформації, що коштує бізнесу грошей і репутації.

Знижена цілісність даних

Важко встановити належний контроль для різних фаз життєвого циклу IT-даних, якщо немає належних інструментів для реєстрації та моніторингу. Зловмисники, які отримують незаконний доступ до системи, можуть легко змінювати дані журналу, змінювати записи та вводити в систему неочікувані дані. Це також означає, що дані компанії є суперечливими, неточними або неповними, що робить їх ненадійними або недійсними для оптимальних вимог бізнесу.

Відстежуваність

Належні механізми реєстрації та моніторингу дозволяють легше ідентифікувати користувачів і процеси, які взаємодіють із системою. Без належних механізмів реєстрації важко відстежити джерело повідомлення/запиту. Це ускладнює відстеження джерела загрози, що сприяє системним атакам.

Відсутність відповідальності

Важко довіряти безпеці організації, коли немає способу відстежити безпеку користувачів і мережі. Механізми реєстрації та моніторингу служать гарантією того, що всі події, пов’язані з системою, можна відстежувати та перевіряти.

1.3 Наявні рішення проблеми логування та моніторингу

Належне ведення журналів і моніторинг є ключем до раннього виявлення та усунення більшості ризиків і загроз безпеки. Реєстрація включає відстеження та зберігання інформації, пов'язаної з подіями в системі, тоді як моніторинг складається з аналізу та візуалізації цих показників для виявлення закономірностей та аномалій.

Тому ефективні стратегії ведення журналів і моніторингу вважаються вирішальними для підтримки безпеки та продуктивності.

Найкращі методи реєстрації та моніторингу безпеки

Проект безпеки відкритих веб-додатків (OWASP) рекомендує різні найкращі методи ефективного ведення журналів і моніторингу. До них належать[2]:

• Забезпечити достатню кількість журналів для всіх помилок автентифікації, включаючи вхід, контроль доступу та перевірку на стороні сервера;

• Створити контекст і зрозуміти базовий трафік, щоб легко ідентифікувати підозрілу та шкідливу діяльність;

• Мати контрольний журнал для критичних та високоцінних транзакцій, щоб запобігти видаленню або фальсифікації;

• Резервне копіювання файлів журналу на кількох серверах для забезпечення відмовостійкості;

• Аутентифікація доступу до журналів;

• Автоматизувати моніторинг і сповіщення про події журналу;

• Створити інтегровану платформу для керування та моніторингу журналів із сповіщеннями та візуалізацією в режимі реального часу;

• Мати офіційний план реагування на інциденти та стратегію вирішення на основі ITIL, який відповідає встановленим стандартам;

• Завжди виконувати тести на проникнення, щоб виявити прогалини в моніторингу і звітності інцидентів;

• Мати в наявності план або стратегію відновлення, розроблену для чорних днів.

Згідно з запропонованою Cisco AppDynamics [16] інформацією, створити ефективну стратегію для оптимізації інтеграції рішень для реєстрації та моніторингу, можна використовуючи наступні методи:

Увімкнути спільну роботу систем логування та моніторингу

Якщо кінцева мета — оптимізувати переваги аналізу даних журналу та показників програми, спростити цей процес, налаштувавши систему на надсилання даних журналу безпосередньо до інструменту моніторингу. Зберігання повідомлень журналу на диску або надсилання їх виключно до інструменту ведення журналу створює більше витрат ресурсів, а також потенційне вузьке місце робочого процесу. Інструмент моніторингу повинен підтримувати мову програмування вашої програми, щоб забезпечити сумісність і простоту використання.

Логувати правильні дані

Дані журналу повинні розповідати стислу, але повну історію. Дані мають бути вибірковими, описовими та забезпечувати правильний контекст, щоб допомогти у вирішенні несправностей. Корисні дані журналу, як правило, включають елементи події і таку інформацію, як позначка часу, ідентифікатори користувачів, ідентифікатори сеансів та показники використання ресурсів. Збір повного спектру застосовних даних покращує інформацію, отриману з інструменту моніторингу.

Використовувати структуровані дані журналу

Спростити дані, спростивши пошук, індексацію та зберігання, переконатися, що вони структуровані. Структуровані дані надають більш повне уявлення про те, що сталося, і можуть надати інструменту моніторингу унікальні ідентифікатори, наприклад, у якому ідентифікаторі клієнта виникла помилка. Надання інформації про ідентифікатор клієнта, отриманої за допомогою журналу, дає змогу інструменту моніторингу побачити, як це вплинуло на конкретного користувача та які інші проблеми вони можуть виникнути в результаті.

Скористатися всіма перевагами даних журналу

Логування пропонує більше, ніж просте усунення несправностей і налагодження, також можна визначити тенденції програми та системи, застосовуючи статистичний аналіз до системних подій. Дані журналу містять

важливу інформацію про ваші програми та базову інфраструктуру, включаючи всі ваші бази даних. Використовуйте історичну інформацію, надану даними журналу, щоб визначити середні значення, які полегшать остаточне визначення аномалій, або згрупувати типи подій у спосіб, який дає змогу проводити точні порівняння. Ці дані також можуть бути корисними для збору, агрегації та перегляду цих даних відповідно до потреб вашого підприємства.

Популярні рішення для реєстрації та моніторингу

Є кілька корисних інструментів, які організації можуть використовувати для створення централізованої системи для реєстрації, аналізу та звітності даних про події. До них належать[2]:

OWASP AppSensor

Цей проект з відкритим вихідним кодом інтегрує доступні механізми реєстрації з найкращими методами безпеки, щоб забезпечити виявлення вторгнень рівня додатків у режимі реального часу для програм, що самовідновлюються. Проект також забезпечує основу для автоматизованого реагування на інциденти безпеки.

NLog

NLog також є гнучким рішенням з відкритим вихідним кодом для обробки подій і сповіщень, які в основному використовуються для платформ .NET. Платформа приймає дані журналу на мові .NET, а потім доповнює їх інформацією про відповідний контекст для аналізу журналів у реальному часі.

Розробники повинні впровадити деякі або всі наведені нижче засоби контролю, залежно від ризику програми[7]:

• всі помилки входу, контролю доступу та перевірки введення на стороні сервера можуть бути зареєстровані з достатнім контекстом користувача, щоб ідентифікувати підозрілі або шкідливі облікові записи, і утримуватися протягом достатнього часу, щоб дозволити відкладений криміналістичний аналіз.

• журнали створюються у форматі, який можуть легко використовувати рішення для керування журналами (наприклад SIEM-ситеми).

• дані журналу закодовані правильно, щоб запобігти ін’єкціям або атакам на системи реєстрації чи моніторингу.

• транзакції з високою вартістю мають контрольний журнал із засобами контролю цілісності, щоб запобігти підробці або видаленню, наприклад, таблиці бази даних тощо.

• команди DevSecOps повинні налагодити ефективний моніторинг та оповіщення, щоб підозрілі дії виявлялися та швидко реагували на них.

• створити або затвердити план реагування на інциденти та відновлення, наприклад Національний інститут стандартів і технологій (NIST) 800-61r2 або новішої версії.

Існують комерційні та відкриті рамки захисту додатків, такі як основний набір правил OWASP ModSecurity, і програмне забезпечення для кореляції журналів з відкритим вихідним кодом, таке як стек Elasticsearch, Logstash, Kibana (ELK), які мають спеціальні інформаційні панелі та оповіщення.

Elasticsearch, Logstash та Kibana[17]

Механізм збору логів виглядає так: Logstash збирає логи та переносить їх у сховище, Elasticsearch допомагає знайти потрібні рядки у цих логах, а Kibana візуалізує їх. Всі три компоненти розроблені на основі відкритого коду, завдяки чому їх можна модифікувати за потребами компанії.

• Logstash — програма для роботи з великими обсягами даних, збирає інформацію з різних джерел та переводить її у зручний формат.

• Elasticsearch — система пошуку інформації. Допомагає швидко знайти потрібні рядки у файлах схову.

• Kibana - плагін візуалізації даних та аналітики в Elasticsearch. Допомагає обробляти інформацію, знаходити у ній закономірності та слабкі місця.

Wazuh

Рішення з відкритим кодом для пошуку логів, що корелює з моделями загроз інформаційній безпеці. З його допомогою моніторять цілісність IT-систем та оперативно реагують на інциденти.

Wazuh допомагає:

• виявити приховані процеси програм, які використовують уразливості у ПЗ для обходу антивірусних систем;

• автоматично блокувати мережеву атаку, зупиняти шкідливі процеси та файли, заражені вірусами.

Список найкращих методів впровадження журналів безпеки[18]:

• Дотримуватись загального формату журналу та підходу в системі та між системами організації. Прикладом загальної системи ведення журналів є служба Apache Logging Services, яка допомагає забезпечити узгодженість ведення журналів між додатками Java, PHP, .NET і C++.

• Не реєструвати занадто багато або занадто мало. Наприклад, переконатися, що завжди реєструється мітку часу та ідентифікаційна інформацію, включаючи вихідну IP-адресу та ідентифікатор користувача, але бути обережним, щоб не реєструвати приватні чи конфіденційні дані.

• Звернути особливу увагу на синхронізацію часу між вузлами, щоб забезпечити узгодженість часових позначок.

Ведення журналу для виявлення вторгнень та реагування

Використовувати журнал, щоб ідентифікувати дії, які вказують на те, що користувач поводиться зловмисно. Потенційно шкідлива активність для реєстрації включає:

• Надіслані дані, які виходять за межі очікуваного числового діапазону.

• Надіслані дані, які включають зміни в дані, які не можна змінювати (виберіть список, прапорець або інший компонент обмеженого введення).

• Запити, які порушують правила контролю доступу на стороні сервера.

• Більш повний список можливих точок виявлення доступний тут.

Коли програма стикається з такою активністю, вона повинна принаймні зареєструвати цю дію та позначити її як проблему високого рівня. В ідеалі програма також повинна реагувати на можливу ідентифіковану атаку, наприклад, визнаючи недійсним сеанс користувача та блокуючи обліковий запис користувача. Механізми реагування дозволяють програмному забезпеченню реагувати в режимі реального часу на можливі виявлені атаки.

Захищений дизайн журналу

Рішення для ведення журналів мають бути створені та керовані безпечним способом. Дизайн безпечного журналу може включати наступне:

• Закодувати та перевірити будь-які небезпечні символи перед реєстрацією, щоб запобігти атакам ін’єкції журналу.

• Не реєструвати конфіденційну інформацію. Наприклад, не вводити пароль, ідентифікатор сеансу, кредитні картки чи номери соціального страхування.

• Захистити цілісність журналу. Зловмисник може спробувати підробити журнали. Тому слід розглянути дозвіл файлів журналів і аудит змін журналу.

• Пересилати журнали з розподілених систем до централізованої безпечної служби реєстрації. Це гарантує, що дані журналу не можуть бути втрачені, якщо один вузол зламаний. Це також дозволяє здійснювати централізований моніторинг.

Сьогодні системи генерують неймовірні обсяги журналів, тому автоматизація суттєво потрібна для виконання будь-якого надійного рівня моніторингу та аналізу журналів. Основним інструментом, який сьогодні використовується для моніторингу журналів безпеки, є платформа управління інформацією та подіями безпеки (SIEM).

Сьогодні на ринку існує безліч SIEM, які надають безліч різних можливостей, але основна передумова SIEM полягає в тому, щоб збирати або приймати журнали з кількох джерел, виконувати або вмикати ефективний аналіз і виконувати призначені дії, такі як попередження про події інтерес.

Нижче наведено кілька рекомендацій, які допоможуть максимально використати безпеку організації, ведення журналів і моніторинг мережі з допомогою SIEM-системи[1]:

Увімкнути реєстрацію на всіх ваших операційних системах, мережевих пристроях і програмах. Кожен компонент в системній архітектурі має бути налаштований на генерацію подій аудиту, щоб забезпечити повне охоплення та не залишати жодних сліпих зон, які можна використовувати для початкової експлуатації або опорних точок.

Налаштувати те, що зареєстровано в операційних системах, мережевих пристроях і програмах. Ознайомитись з можливостями аудиту кожного компонента в архітектурі та прийняти відкрите рішення про те, які події слід перевіряти, враховуючи політику організації журналів і моніторингу. Налаштувати важливі пристрої, такі як брандмауери та точки віддаленого доступу, для детального ведення журналу, пристосовуючи можливості аудиту інших компонентів до подій, що стосуються безпеки, чи інших подій, які цікавлять.

Встановити базовий рівень «нормальної» діяльності. Якщо ідея полягає в тому, щоб виявити ненормальну або зловмисну діяльність і належним чином попередити, організації повинні знати, що таке «нормальна» поведінка або нешкідливі, стандартні дії, які підтримують бізнес-цілі.

Налаштувати SIEM. Коли буде базова лінія діяльності, яка представляє «нормальну» поведінку, буде легше налаштувати SIEM, щоб визначити дії, які виходять за межі «звичайних» моделей поведінки. Саме ці події вимагають найбільшої концентрації та уваги з боку працівників безпеки. Крім того, налаштований SIEM вироблятиме менше помилкових спрацювань, дослідження яких забирає багато часу.

Навчити співробітників служби безпеки виявлення подій. Аналіз подій є спеціалізованою навичкою і вимагає відточеного досвіду для визначення та розуміння моделей атак.

1.4 Потенціал розвитку засобів логування та моніторингу

Майбутнє рішень для управління журналами

Управління журналами вже неодноразово бачило багато інновацій для боротьби з цими проблемами. З часом рішення вдосконалюються, а управління інформацією та подіями безпеки (SIEM) домінує в секторі завдяки високоефективному та ефективному управлінню цими проблемами, пов’язаними з керуванням журналом подій.

SIEM – це рішення, яке збирає дані журналів з кількох джерел в IT-середовищі, об’єднує, аналізує та представляє їх у спосіб, який легко спостерігати та зберігати[9]. Рішення SIEM задовольняють багато труднощів, які існують в управлінні журналами. Вони також дозволяють IT-персоналу мати чітке розуміння інформаційної безпеки.

Хоча інформаційна безпека завжди буде залежати в основному від людського фактора, рішення SIEM рухаються в правильному напрямку. Багато стандартів відповідності, наприклад ISO 27001 і стандарт безпеки даних індустрії платіжних карток (PCI DSS), вимагають включення рішення SIEM для регулювання інформаційної безпеки.

Це тому, що там, де проблеми з керуванням журналами можуть спричинити ослаблення інформаційної безпеки, рішення SIEM допоможе посилити безпеку у вашому середовищі та оптимально залучити ваші ресурси для вирішення всіх інших важливих завдань. Таким чином, легше одночасно керувати безпекою та журналами.

Вкрай важливо, що для управління повномасштабним рішенням для керування журналами потрібне значне планування. Тому необхідно ретельно складати свій план, подивитися на потенційні рішення, а потім прийняти обґрунтоване рішення.

Так як в майбутньому обчислювальні потужності будуть зростати в геометричній прогресії, то й обробка логів буде займати набагато менше часу, а отже можна буде збирати більше інформації про події, не жертвуючи при цьому часом.

Також в майбутньому за різними алгоритмами будуть навчені нейронні мережі та штучний інтелект, що зможуть виконувати роботу агрегації та кореляції логів за лічені секунди з мінімальними погрішностями.

Висновки за розділом 1

Немає прямої вразливості, яка може виникнути через ці проблеми, але загалом ведення журналів і моніторинг є критично важливими для роботи IT-систем, і їх відсутність або збої можуть безпосередньо вплинути на видимість, оповіщення про

інциденти та криміналістичну експертизу. Таким чином, дуже важливо мати функціональну систему ведення журналів і моніторингу, щоб збирати журнали, а також подавати попередження, якщо трапляються якісь несправності або помилки, інакше вони можуть залишатися непоміченими протягом тривалого часу і завдати набагато більшої шкоди.

Навіть в найбільшій інформаційній системі збору та кореляції логів закладено певні моделі виявлення та ідентифікації ризиків та загроз, а отже саме ці моделі і є найменшим елементом в протидії мережевим атакам.

В даному дослідженні пропонується розробити модель виявлення та ідентифікації мережевих атак, що використовують вразливості логування та моніторингу. Для цього буде розглянуто методи виявлення атак, побудована модель виявлення мережевих атак та проаналізовано результати її роботи.

Таким чином в даній роботі, згідно досліджуваної мети, необхідно розглянути наступні задачі:

- Аналіз параметрів логування та моніторингу;
- Збір даних про існуючі мережеві атаки: накопичення даних мережевого трафіку, журналів подій;
- Приведення даних до узагальненого вигляду: компонування їх в потоки мережевих даних (network data flows);
- Побудова моделі виявлення та ідентифікації мережевих атак на основі алгоритму дерева рішень;
- Перевірка адекватності створеної моделі.

РОЗДІЛ 2
МЕТОДИ ВИЯВЛЕННЯ ТА ІДЕНТИФІКАЦІЇ МЕРЕЖЕВИХ АТАК

2.1 Класифікація методів виявлення та ідентифікації мережевих атак

Виявлення вторгнень – це процес моніторингу подій, що відбуваються в комп'ютерній системі чи мережі, та аналізу їх на ознаки можливих інцидентів, які є порушеннями чи неминучими загрозами порушення політики комп'ютерної безпеки, політики прийнятного використання або стандартних методів безпеки[19].

Системи виявлення атак (СВА) являють собою окремий клас програмних засобів (ПЗ), під яким розуміють програми, процедури, правила, а також, якщо передбачено, супутніх їм документації та даних, що відносяться до функціонування системи обробки інформації. Повна назва СВА – це системи виявлення і запобігання атак, так як саме в можливості автоматизованої протидії атакам полягає одна з основних переваг таких систем, у порівнянні, наприклад, із засобами, заснованими на людському факторі.

Використання СВА дозволяє вирішити цілий ряд завдань, що забезпечують досягнення цілей інформаційної безпеки[20]:

• розпізнавання відомих і, по можливості, невідомих атак та попередження персоналу, що відповідає за забезпечення інформаційної безпеки (ІБ);

• статистичний аналіз шаблонів аномальних дій;

• моніторинг і аналіз користувацької, мережевої та системної активності;

• контроль цілісності файлів та інших ресурсів інформаційної системи (ІС);

• аудит системної конфігурації і виявлення вразливостей;

• інсталяція і підтримка роботи серверів-пасток для запису інформації про порушників;

• зниження навантаження на персонал (або звільнення від нього), що відповідає за ІБ, від поточних рутинних операцій з контролю за користувачами, системами і мережами, які є компонентами ІС;

• надання можливості управління функціями захисту не спеціалістам в області інформаційної безпеки.

Під виявленням атак розуміють процес оцінки подій ІС та її інформаційних потоків, який реалізується за допомогою аналізу журналів реєстрації операційних систем (ОС) і додатків або мережевого трафіку. Реалізація більшості мережевих атак здійснюються в три етапи.

Перший, підготовчий, етап полягає в пошуку передумов для здійснення тієї чи іншої атаки. На даному етапі шукають вразливості, використання яких робить можливим в принципі реалізацію атаки, яка і складає другий етап. На третьому етапі атака завершується. При цьому перший і третій етапи самі по собі можуть бути атаками. Наприклад, пошук порушником вразливостей за допомогою сканерів безпеки вже вважається атакою.

Технології виявлення атак постійно розвиваються і удосконалюються, і ця область постійно залучає нових виробників і розробників. Незважаючи на брак теоретичних основ технології виявлення атак, існують досить ефективні методи, що використовують на сьогодні.

Існує кілька способів класифікації систем виявлення атак, кожен з яких заснований на різних характеристиках. Тип слід визначати, виходячи з таких характеристик:

• Спосіб контролю за системою. За способами контролю за системою поділяються на network-based, host-based і application-based.

• Спосіб аналізу. Це частина системи визначення проникнення, яка аналізує події, отримані з джерела інформації, і приймає рішення, чи відбувається проникнення. Способами аналізу є виявлення зловживань (misuse detection) та виявлення аномалій (anomaly detection).

• Затримка в часі між отриманням інформації з джерела та її аналізом і прийняттям рішення. Залежно від затримки в часі, системи виявлення атак діляться на interval-based (або пакетний режим) і real-time.

Більшість комерційних систем виявлення атак є real-time network-based системами.

При виявленні в режимі реального часу, коли система виявлення вторгнень відстежує будь-який вид вторгнення в комп’ютерну систему, він негайно повідомляється у вигляді попередження, і вживаються відповідні дії, щоб безпека системи не порушувалась. СВА в реальному часі працює в автономному режимі, використовуючи попередні або історичні дані, зібрані під час виявлення вторгнень [21].

Класифікація методів виявлення атак[22]:

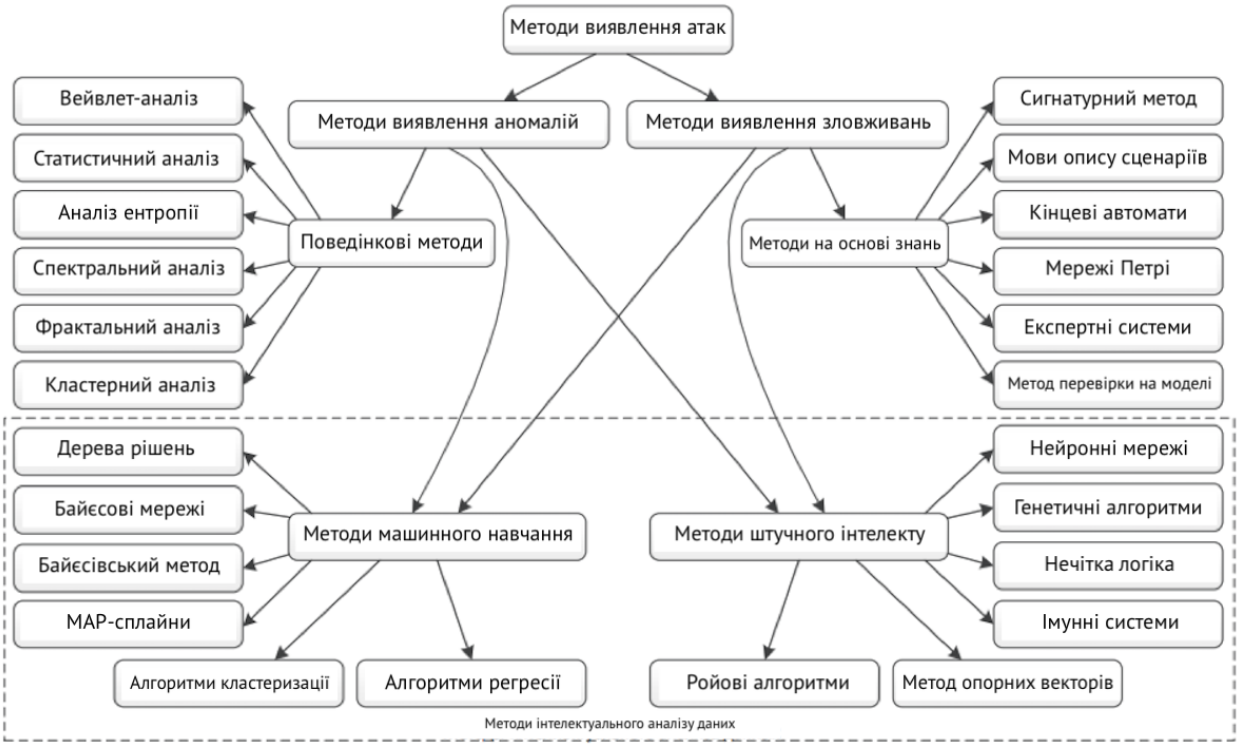

Рисунок 2.1 – Класифікація методів виявлення атак

На сьогоднішній день виділяють і рекомендують до застосування, в тому числі, і при побудові системи захисту три групи методів виявлення атак[23]:

• сигнатурні методи;

• методи виявлення аномалій;

• комбіновані методи (використовують спільно алгоритми, визначені в сигнатурних методах і методах виявлення аномалій).

Іншими словами, виявлення порушення безпеки проводиться зазвичай з використанням евристичних правил і аналізу сигнатур відомих комп'ютерних атак.

В основному існує два підходи до системи виявлення вторгнень: на основі сигнатур (також відомих як на основі неправильного використання) і на основі аномалій [24].

Зазвичай в СВА намагаються поєднувати обидві технології, щоб усунути недоліки, властиві кожній окремо. Перевага “аномальних” систем - виявлення невідомих або нових видів атак, які можуть “обійти” СВА. Реєстрація такого роду подій тягне за собою їх аналіз адміністратором, створення для них шаблону і внесення останнього до бази даних СВА. Системи, засновані на методі аномалій, вважаються досить перспективними, але ще розвиваються і перебувають у стадії дослідження.

Особливістю технології виявлення атак на основі сигнатур є процес опису атаки у вигляді шаблону або сигнатури і пошуку даного шаблону в контрольованому просторі (наприклад, мережевому трафіку або журналі реєстрації). Така СВА може виявити всі відомі атаки, але вона мало пристосована для виявлення нових, ще невідомих, атак.

При розробці СВА, заснованих на цьому підході, виникають дві основні проблеми. Перша полягає у створенні механізму опису сигнатур, тобто мови опису атак, а друга проблема виражається в наступному: як записати атаку, щоб зафіксувати всі можливі її модифікації[20].

Переваги:

• Детектори зловживань ефективно визначають атаки і дуже рідко створюють помилкові повідомлення;

• Детектори зловживань швидко й надійно діагностують використання конкретного інструментального засобу або технології атаки. Це дає змогу адміністратору скоригувати заходи для забезпечення безпеки;

• Швидкість аналізу.

Недоліки:

• Оскільки детектори зловживань виявляють лише відомі їм атаки, слід постійно оновлювати їхні бази даних для отримання сигнатур нових атак;

• Більшість детекторів зловживань розроблено так, що вони використовують лише певні сигнатури, а це не дає виявити можливі варіанти атак

Технологія виявлення атак на основі аномалій побудована на припущенні, що аномальна поведінка суб'єкта ІС (системи, програми, користувача), тобто, як правило, атака або яка-небудь ворожа дія часто проявляється як відхилення від нормальної поведінки. Зазвичай системи виявлення аномальної активності використовують як джерело даних журнали реєстрації і поточна діяльність користувача, хоча існують приклади системи виявлення аномалій в мережевому трафіку.

Традиційне використання цієї технології полягає не в чіткому виявленні атак, а у визначенні підозрілої активності, що відрізняється від нормальної. Основна проблема методу полягає в тому, щоб визначити критерій нормальної активності. Необхідно також встановити допустимі відхилення від нормального трафіку, які ще не вважатимуться атакою.

При використанні даної технології виявлення атак можливі два варіанти неправильного виявлення атаки:

• виявлення дії, яка не є атакою, і віднесення його до класу атак;

• пропуск атаки, яка не підпадає під сигнатури атак. Цей випадок більш небезпечний, ніж помилкове віднесення дозволеного дії до класу атак. Підкатегорією такого методу є аналіз на основі профілів, коли нормальна поведінка визначається для окремих суб'єктів (користувачів / систем).

Іноді елементи такого аналізу зустрічаються і в інших методах, наприклад, в розшифровці протоколу, коли виявлений елемент, що не належить наперед визначеному протоколу або порушує правила використання протоколів.

Прикладами аномальної поведінки є велика кількість з'єднань за короткий проміжок часу, високі завантаження центрального процесора і коефіцієнт мережевого навантаження або використання периферійних пристроїв, які зазвичай не використовуються.

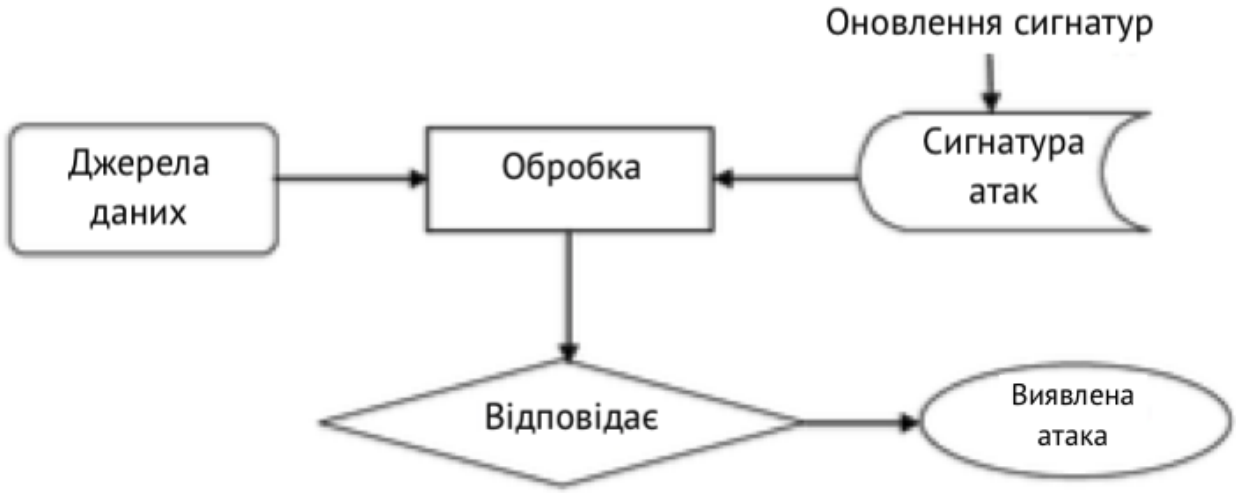

Рисунок 2.2 – Схема виявлення атак на основі сигнатур

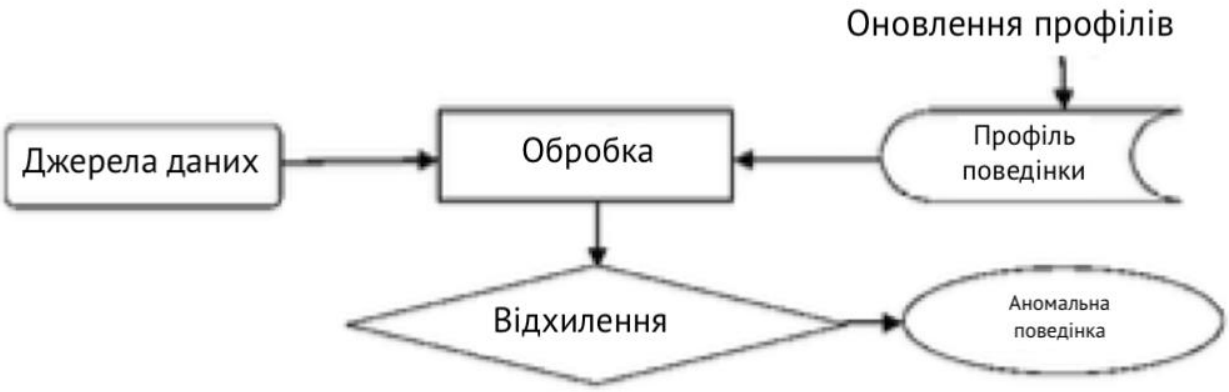

Рисунок 2.3 – Схема системи виявлення аномальної поведінки

Якщо описати профіль нормальної поведінки суб’єкта, то будь-яке відхилення від нього можна охарактеризувати як аномальна поведінка.

Переваги:

• СВА, що виявляють аномалії, фіксуючи несподівану поведінку системи, отримують можливість визначати симптоми атак, не маючи відомостей про їхні конкретні деталі;

• Детектори аномалій збирають інформацію, якою в подальшому можуть скористатися детектори зловживань для визначення сигнатур.

Недоліки:

• Під час виявлення аномалій, як правило, створюється велика кількість помилкових сигналів про атаки у разі непередбачуваної поведінки користувачів і мережної активності;

• Цей метод часто потребує певного етапу навчання системи, під час якого визначаються характеристики нормальної поведінки. Якість проведення цього навчання суттєво впливає на подальшу ефективність СВА;

• Не можна реалізувати опис атаки за елементами. Повідомляється те, що відбувається щось підозріле;

• Дана технологія значно залежить від середовища функціонування як визначального фактор анормальної поведінки;

• Відносно низька швидкість аналізу;

• Трудомістке завдання побудови профілів суб'єктів ІС.

Таблиця 2.1

Порівняння методів СВА

Характеристика	Сигнатурні методи	Методи аномалій
Множина виявлених атак	Обмежується відомими видами атак	Обмежується можливостями налаштування і методами аналізу СВА
Ймовірність пропуску атаки	Середня	Низька
Ймовірність помилкового спрацювання	Дуже низька	Висока
Вимоги до обчислювальних ресурсів ІС	Середні	Високі

2.2 Сигнатурні методи

Традиційно засновані на сигнатурах мережеві системи виявлення атак покладаються на дані експертів домену і можуть ідентифікувати атаки, лише якщо вони відбуваються як окрема подія. СВА генерує велику кількість сповіщень, і користувачам стає дуже важко переглянути кожне повідомлення. Попередні дослідження запропонували підходи на основі аналітики для аналізу моделей сповіщень на основі моделей виявлення аномалій, багатоетапних моделей або імовірнісних підходів. Однак через складність мережевих вторгнень неможливо розробити всі можливі моделі атак або уникнути помилкових спрацьовувань[25].

З розвитком технологій і популярністю мереж у нашому повсякденному житті стає все важче виявляти вторгнення в мережу. Однак, як би швидко не змінювалися технології, поведінка людей, що стоїть за кібератаками, залишається відносно постійною. Це дає можливість розробити вдосконалену систему виявлення незвичайних кібератак.

У сигнатурних методах системні події представлені у вигляді ланцюжків символів з деякого алфавіту. Суть цих методів полягає в завданні безлічі сигнатур атак у вигляді регулярних виразів (regular expressions) або правил на основі зіставлення з зразком (pattern matching) та перевірки відповідності спостережуваних подій цим виразам[26]. Типовими представниками систем, в яких реалізований такий метод, є Snort [27] та Suricata [28].

Основна перевага сигнатурного методу полягає в те, що виявлення відомих зразків аномальних подій здійснюється максимально ефективно. Але водночас використання бази сигнатур великого обсягу негативно впливає продуктивність системи виявлення [29].

Загальновідомо, що переваги СВА на основі сигнатур полягають у тому, що вони мають низький рівень помилкових тривог, а також інформацію, яку вони часто передають спеціалісту з безпеки системи про виявлену атаку. Така інформація часто закодована в правилах або шаблонах, які є центральними для функціональності таких систем. Ця інформація часто є неоціненною для початку профілактичних або

коригувальних дій. Однак СВА на основі підпису має кілька недоліків. Оскільки набір аномальних моделей базується на відомих атаках, нові атаки не можуть бути виявлені цими системами. Тому щоразу, коли виявляється нова атака, шаблони, що відповідають атаці, повинні бути створені вручну.

Більше того, не можна очікувати, що досвідчений і рішучий зловмисник покладатиметься на відомі атаки – він намагатиметься проникнути в систему за допомогою прихованих, витончених атак, використовуючи поведінку, яка не буде виявлена схемою на основі статичних шаблонів [30]. Наприклад, зловмисник може «змішати» звичайну активність із реальною атакою, щоб слід не відповідав жодному із попередньо визначених шаблонів.

На відміну від виявлення аномалії, де образ - це модель нормальної поведінки системи, при виявленні зловживання він необхідний для уявлення несанкціонованих дій зловмисника. Такий «образ» стосовно виявлення зловживань називається сигнатурою вторгнення. Формується сигнатура на основі тих самих вхідних даних, що і при виявленні аномалій, тобто на значеннях параметрів оцінки.

Сигнатури вторгнень визначають оточення, умови і спорідненість між подіями, які призводять до проникнення в систему або будь-яким іншим зловживанням. Вони корисні не тільки при виявленні вторгнень, але і при виявленні спроб здійснення незаконних дій. Частковий збіг сигнатур може означати, що в системі, що захищається мала місце спроба вторгнення. Перевага даних методів - висока точність визначення факту атаки, а очевидний недолік - неможливість виявлення атак, сигнатури яких ще не визначені[23].

Метою другого напрямку (виявлення зловживань) є пошук послідовностей подій, визначених адміністратором безпеки або експертом під час навчання СВА, як етапи реалізації вторгнення. У теперішній час виділяють лише методи з контрольованим навчанням.

Серед сигнатурних методів виявлення атак найбільш поширений метод контекстного пошуку, який полягає в виявленні у вихідній інформації певної безлічі символів. Так, для виявлення атаки на Web-сервер, що спрямована на отримання

несанкціонованого доступу до файлу паролів, проводиться пошук послідовності символів "GET* / etc / passwd" у заголовку HTTP-запиту.

Фрагмент "cwd-root" в FTP-сеанс однозначно визначає факт обходу механізму аутентифікації на FTP-сервері і спробі перейти в кореневий каталог FTP-сервера. Іншим прикладом є виявлення аплетів Java в мережевому трафіку на основі шістнадцятирічного фрагмента "CA FE BA BE". Ці ж сигнатури дозволяють виявляти троянських коней, якщо останні використовують стандартні значення портів. Наприклад, троян NetBus визначається по використанню 12345-го і 12346-го портів, а троян BackOrifice - 31337-го порту.

За допомогою контекстного пошуку ефективно виявляються атаки на основі аналізу мережевого трафіку, оскільки даний метод дозволяє найбільш точно задати параметри сигнатури, яку необхідно виявити в потоці вихідних даних.

У ряді СВА, відповідно до [29] були реалізовані ще два сигнатурних методи: метод аналізу станів і метод, який базується на експертних системах.

Метод аналізу станів або контролю частоти подій заснований на формуванні сигнатури атак у вигляді послідовності переходів інформаційної системи ІС з одного стану в інший. По суті, кожен такий перехід визначається по настанню в ІС певної події, а набір цих подій задається параметрами сигнатури атаки. Ці сигнатури описують ситуації, коли протягом деякого інтервалу часу відбуваються події, число яких перевищує задані заздалегідь показники. Прикладом такої сигнатури є виявлення сканування портів або виявлення атаки SYN Flood. У першому випадку пороговим значенням є число портів, перевірених в одиницю часу. У другому випадку - число спроб встановлення віртуального з'єднання з вузлом за одиницю часу.

Як правило, сигнатури атак, створені на основі аналізу станів, описуються математичними моделями, що базуються на теорії кінцевих автоматів або мереж Петрі.

Методи, що базуються на експертних системах, дозволяють описувати моделі атак на природній мові з високим рівнем абстракції. Експертна система, яку покладено в основу методів цього типу, складається з двох баз даних: фактів і

правил. Факти це вихідні дані про роботу ІС, а правила - алгоритми логічних рішень про факт атаки на основі набору фактів. Всі правила експертної системи записуються в форматі "якщо <...>, то <...>".

Результуюча база правил повинна описувати характерні ознаки атак, які зобов'язана виявляти СВА.

Одна з найбільш перспективних сигнатурних груп - методи, які засновані на біологічних моделях. Для їх опису можуть використовуватися генетичні або нейромережеві алгоритми.

Згідно з даними OWASP, доцільно поділити виконати групування сигнатур методу виявлення мережевих атак, що використовують вразливості логування та моніторингу, за наступними категоріями[31]:

- Authentication [AUTHN]
- Authorization [AUTHZ]
- Excessive Use [EXCESS]
- File Upload [UPLOAD]
- Input Validation [INPUT]
- Malicious Behavior [MALICIOUS]
- Privilege Changes [PRIVILEGE]
- Sensitive Data Changes [DATA]
- Sequence Errors [SEQUENCE]
- Session Management [SESSION]
- System Events [SYS]
- User Management [USER]

Автентифікація (Authentication)

Для побудови сигнатурної моделі виявлення проблем автентифікації в логах необхідно відслідковувати наступні поля:

- Datetime (дата/час)
- UID (Юзернейм)
- Source IP
- Event name – Success/Failed login (Назву події)

• AppID (Ідентифікатор додатку або ресурсу)

• TokenName (Назва токену)

• TokenExpiration: True/False (Валідність токену)

• Area (Зона підключення)

Якщо користувач з одним UID має 2 або більше Success login протягом невеликого проміжку часу, що вираховується різницею Datetime параметрів, то в такому випадку можна зробити висновок про втрату конфіденційності параметрів автентифікації і в якості рішення необхідно тимчасово заблокувати даного користувача.

Якщо користувач з одним UID має 3 або більше Failed login протягом невеликого проміжку часу, що вираховується різницею Datetime параметрів, то в такому випадку можна зробити висновок, що відбувається атака шляхом підбору паролю, тому такого користувача необхідно заблокувати, відслідковуючи далі попередню та подальшу активність з даного ip-вузлу.

Якщо користувач з одним UID виконує спробу автентифікації з двох різних Area, що вираховується за параметром Source IP протягом незначного періоду, який вираховується різницею Datetime параметрів, то в даному випадку можна зробити висновок, що дані користувача втратили конфіденційність і необхідно заблокувати його.

Якщо відбуваються 2 або більше спроб автентифікації до ресурсу з певним AppID за допомогою певного токену TokenName із TokenExpiration: Yes з конкретного Source IP, то в такому випадку можна зробити висновок, що відбувається спроба несанкціонованого доступу до ресурсу і необхідно заблокувати даний ip-вузол і відслідкувати його активність в системі.

Авторизація (Authorization)

Необхідно параметри для відслідковування атак:

• UID

• ResourceName

• newUID

Якщо користувач з певним UID намагається отримати доступ до ресурсу - ResourceName без авторизації, то можна зробити висновок, що оперує зловмисник і необхідно заблокувати даного користувача

Якщо користувач з певним UID змінює собі ідентифікатор на інший newUID, то варто видати системне попередження з подальшим відслідковуванням дій даного користувача, бо можлива мережева атака з експлуатуванням підвищення прав в системі.

*всі дії користувача з UID=admin/root необхідно логувати, незалежно від ресурсу, на якому вони відбуваються та операцій, що виконуються

Надмірне використання/Перевищення ліміту (Excessive Use)

• AppID

• ResourceName

• Name of limit

• Value

Якщо в певній системі AppID на певному ресурсі ResourceName перевищено ліміт Name of limit із значенням Value, то необхідно терміново проаналізувати проблему та вжити відповідних заходів, і в залежності від ліміту (наприклад, Кількість активних сесій) робити висновки щодо атаки.

Завантаження файлу (File Upload/Delete)

• Filename

• FileFormat

• Status Success/Failed

• AppID

• UploadValidationStatus (scan of file) Success/Failed

Якщо виконано спробу завантаження певного файлу Filename забороненого формату FileFormat, то необхідно одразу видати оповіщення щодо можливого завантаження шкідливого коду в систему. Те саме стосується й параметру UploadValidationStatus.

Валідація введених даних (Input validation)

• InputFormat

• ValidationStatus

• EnteredData

У випадку якщо ValidationStatus=False, то необхідно записати в лог-файл формат поля введення InputFormat, та вивести введені дані EnteredData, що призвели до помилки валідації, адже є ризик, що тут може відбутись спроба мережевої атаки за рахунок переповнення буферу та/або виконання шкідливого коду.

Виявлення шкідливої активності (Malicious behavior)

• UID

• IP source

• Datetime

• UserAgentName

• PossibleAttackToolName

• AppID

• Resource

• Authorization

Якщо відбувається 2 або більше спроби доступу до неіснуючого ресурсу Resource в межах певного додатку/системи AppID, то можна зробити висновок щодо мережевої атаки з застосуванням методу перебору в пошуках вразливого ресурсу і в такому випадку необхідно заблокувати користувача UID та виконати розслідування інциденту.

Якщо в параметрі UserAgentName було знайдено послідовність символів, яку використовуються у відомих інструментах мережевого сканування чи атаках, то необхідно вивести значення PossibleAttackToolName та зробити висновки щодо даного інциденту.

При спробі користувача UID отримати неавторизований доступ до ресурсу Resource зі статусом параметра Authorization=False можна припустити, що відбувається мережева атака за рахунок прямого неавторизованого доступу до ресурсу.

Зміна привілеїв (Privilege changes)

• UID

• File/Object

• FromLevel

• ToLevel

При спробі користувача UID змінити права доступу до об’єкту File/Object необхідно перевіряти параметри FromLevel та ToLevel на наявність підозрилих або різких змін типу 0400 на 0777 та ін. І за наявними матрицями доступу відстежувати атаки, що можуть бути по’язані зі зміною прав до об’єктів.

Модифікація чутливих даних (Sensitive data changes)

• UID

• File/Object

• ChangeStatus (Create/Read/Write/Delete)

Будь-яка дія над об’єктом File/Object, що містить чутливу інформацію повинна бути залогована із зазначенням параметрів UID та ChangeStatus для виявлення порушення цілісності інформації, подальшого аналізу та проведення розслідування атаки.

Порушена послідовність (Sequence errors)

• UID

• ResourceName

У випадку якщо користувач UID потрапляє на певний ресурс ResourceName, який немає прямого зв’язку з попереднім ресурсом, яка передбачена за замовчуванням, на якому був цей користувач, то робиться висновок, що відбувається мережева атака, в якій використовується прямий доступ до чутливих ресурсів.

Менеджмент сеансів (Session management)

• UID

• SessionStatus (Started/Renewed/Expired/Closed)

Необхідно вести всі записи щодо зміни стану сесії SessionStatus певного користувача UID задля виявлення неможливої послідовності зміни цих статусів та повторного відображення одного статусу підряд, адже це буде ознакою мережевої

атаки (наприклад почалось дві сесії одночасно в одного користувача, це може свідчити про дії зловмисника).

Системні події (System events)

• UID

• Datetime

• SystemName

• SystemStatus (Started/Shutdowned/Restarted/Crushed)

• Reason

• SystemMonitor (Yes/No)

Необхідно відслідковувати системні статуси SystemStatus та, за можливості, їх причини Reason в розрізі певного періоду доби Datetime на виявлення підозрілих дій, що не можуть бути виконані в даний час. Також необхідно оповіщати коли параметр SystemMonitor=No, виводячи ідентифікатор користувача UID, що прив’заний до цієї події та параметр часу Datetime.

Управління обліковими записами (User management)

• UID

• SubUID

• Status (Created/Modified/Archived/Deleted)

• NewAttributes

Необхідно виконувати запис всіх дій користувача UID щодо модифікації облікових записів інших користувачів SubUID з переліком атрибутів, що були надані NewAttributes на наявність неможливих змін, відповідно до рольової матриці, інакше це може вказувати про експлуатацію облікових записів зловмисниками.

2.3 Методи виявлення аномалії на основі машинного навчання

Існують переваги та обмеження обох методів. У СВА на основі сигнатур існує висока точність виявлення відомої атаки, але сьогодні безпека є головною проблемою в кожній галузі, і кожен день з’являється новий тип атаки. Тому СВА на основі сигнатур не може виявити, що вторгнення та система призводять до атак

нульового дня. Щоб подолати цю проблему, виявлення аномалій є найкращим методом виявлення нового типу атаки на основі їх поведінки. Виявлення аномалій також має певні обмеження, але воно захищає систему від атак нульового дня.

Виявлення аномалій складається з двох етапів[19]

а. Тренувальний етап

б. Фаза тестування

На основі цього диференціюється поведінка. На етапі навчання набір даних навчається щодо нормального та/або ненормального профілю руху. Після цього цей профіль навчання перевіряється на наборі даних тесту, щоб перевірити точність підходу виявлення.

Для виявлення аномалій можна використовувати три різні методи [32]:

Виявлення аномалій під наглядом

Методики, які навчаються в контрольованому режимі, передбачають наявність навчального набору даних, який має позначені екземпляри як для нормального, так і для аномального класу. Типовим підходом у таких випадках є побудова прогнозної моделі для класів нормальних і аномалій. Будь-який невидимий екземпляр даних порівнюється з моделлю, щоб визначити, до якого класу він належить. Під час контрольованого виявлення аномалій виникають дві основні проблеми. По-перше, аномальних екземплярів набагато менше в порівнянні зі звичайними екземплярами в даних навчання. По-друге, отримати точні та репрезентативні мітки, особливо для класу аномалій, зазвичай складно. За винятком цих двох проблем, проблема виявлення аномалій під наглядом схожа на створення прогнозних моделей.

Виявлення аномалій із напівнаглядом

Методики, які працюють у напівконтрольованому режимі, припускають, що навчальні дані мають позначені екземпляри лише для нормального класу. Оскільки вони не вимагають міток для класу аномалії, вони ширше застосовуються, ніж контрольовані методи. Наприклад, при виявленні несправностей космічного корабля сценарій аномалії буде означати аварію, яку нелегко змоделювати. Типовий підхід, який використовується в таких методах, полягає в побудові моделі для класу, що

відповідає нормальній поведінці, і використання моделі для виявлення аномалій у даних тесту.

Виявлення аномалій без нагляду

Техніки, які працюють у режимі без нагляду, не вимагають даних для навчання, і тому є найбільш широко застосовними. Методики цієї категорії роблять неявне припущення, що нормальні випадки зустрічаються набагато частіше, ніж аномалії в даних тесту. Якщо це припущення не відповідає дійсності, то такі методи страждають від високого рівня помилкових тривог.

Багато методів із напівнаглядом можна адаптувати для роботи в неконтрольованому режимі, використовуючи вибірку немаркованого набору даних як навчальні дані. Така адаптація передбачає, що дані тесту містять дуже мало аномалій, і модель, засвоєна під час навчання, стійка до цих кількох аномалій.

Виявлення аномалій у мережевій поведінці (англ. Network behavior anomaly detection, NBAD) — підхід до виявлення загроз мережевій безпеці. Є одним з підходів до побудови мережевих систем виявлення вторгнень[33]. Доповнює технологію систем що виявляють атаки на основі сигнатур пакетів. Також використовується антивірусним програмним забезпеченням та антишпигунським програмним забезпеченням.

Під час виявлення аномалій у мережевій поведінці проводиться безперервний моніторинг мережі на наявність незвичних подій або тенденцій.

Системи, які використовують виявлення аномалій у мережевій поведінці, можуть бути корисними у виявленні загроз, там де сигнатурний аналіз не може дати результатів, а саме:

• при загрозах нульового дня;

• коли трафік зашифрований, як-то передача даних на командний сервер у ботнетах.

Система відслідковує критичні характеристики мережі у реальному часі і формує сигнал тривоги при виявленні дивної події, яка може свідчити про наявність загрози. Приклади таких характеристик включають обсяг трафіку, використання смуги пропускання та використання протоколів.

Системи виявлення аномалій у мережевій поведінці також відслідковують поведінку окремих вузлів мережі. Зазвичай системи виявлення аномалій у мережевій поведінці є ефективними, якщо нормальна поведінка у мережі є сталою протягом довгого часу. Тоді деякі параметри визнаються нормальними, тоді як усі відхилення - аномалією.

Системи виявлення аномалій у мережевій поведінці повинні використовуватись разом зі звичайними мережевими екранами та застосунками для виявлення шкідливих програм. Деякі виробники визнають, що системи виявлення аномалій у мережевій поведінці є частиною їх рішень з мережевої безпеки.

Системи виявлення аномалій у мережевій поведінці використовують аналіз журналів реєстрації подій, аналіз пакетів, моніторинг мережевих потоків та аналіз маршрутів.

Переваги

До переваг підходу виявлення аномалій у мережевій поведінці у порівнянні з сигнатурним аналізом відносяться:

• можливість виявлення раніше невідомих загроз (загроз нульового дня);

• відсутність необхідності підтримувати сигнатури у актуальному стані.

Недоліки

До недоліків підходу виявлення аномалій у мережевій поведінці у порівнянні з сигнатурним аналізом відносяться:

• наявність хибно позитивних спрацьовувань при певних нестандартних, але допустимих ситуаціях (наприклад зростання трафіку у при підготовці річного звіту);

• у окремих випадках необхідність "навчання" для створення шаблонів нормальної поведінки у мережі.

Аномалія мережевого трафіку, як правило, раптова і дуже серйозно може вплинути на роботу мережі. Варто відзначити, що аномалії не завжди можуть бути викликані шкідливою активністю (наприклад, DDoS-атакою, атакою сканування портів, вірусною активністю і т.д.). Найчастіше вони можуть бути викликані зміною характеру використовуваного програмного забезпечення.

Перш ніж дати визначення аномалії необхідно розібратися, що вважати нормальним станом. Стан системи вважається нормальним тоді, коли вона виконує всі покладені на неї функції. Відповідно аномалія - такий стан, коли поведінка системи не відповідає чітко встановленим характеристикам нормальної поведінки. Аномальний стан в мережевому трафіку - стан, при якому значення функції f(t) в будь-який момент часу t відрізняється від нормального[34]. Наприклад, реєстрація якісної або кількісної зміни потоку інформації, ніяк не пов'язаного з нашою роботою в мережі. Це може свідчити про те, що здійснюється несанкціонована передача даних, ймовірно шкідлива або небажана.

Виявлення аномалій це спроба знайти якусь поведінку піднаглядного об'єкта, шаблон, який не відповідає очікуваної або нормальної поведінки. Безперечна важливість проблеми виявлення аномалій пов'язана в першу чергу з тим, що аномалії можуть серйозно впливати на те середовище, в якому сталася аномалія. Наприклад, в комп'ютерних мережах аномальна поведінка трафіку може означати, що заражений якимось вірусом комп'ютер відсилає важливу (ймовірно, цінну) інформацію «назовні», до неавторизованого комп'ютера.

В комп'ютерних мережах аномалії можуть бути викликані різними причинами. Серед них[34]:

• Несправності мережевого обладнання;

• Випадкові або навмисні дії з боку легітимних користувачів (некоректні дії користувача через низьку кваліфікацію);

• Невірна робота додатків (помилки в програмному коді);

• Дії зловмисників (вірусна атака або інша шкідлива діяльність);

• Якісні зміни складу призначеного для користувача ПО та інше.

У широкому сенсі, мережеві аномалії можна поділити на дві основні категорії: пов'язані з продуктивністю і пов'язані з безпекою.

Метод виявлення аномалій або поведінковий метод базується не на моделях інформаційних атак, а на моделях штатного функціонування (поведінки) ІС. Принцип роботи будь-якого з таких методів полягає в виявленні невідповідності між

поточним режимом роботи ІС і режимом роботи, що відповідає штатної моделі даного методу. Будь-яка невідповідність розглядається як інформаційна атака.

Наприклад, якщо система виявлення атак фіксує вхід співробітника компанії в мережу в суботу о 2.30, то це може свідчити про те, що пароль цього користувача вкрадений або підібраний і його зловмисник використовує для несанкціонованого проникнення.

Перевага методів даного типу - можливість виявлення нових атак без модифікації або поновлення параметрів моделі. На жаль, створити точну модель штатного режиму функціонування ІС дуже складно. Серед поведінкових методів найбільш поширені ті, що базуються на статистичних моделях. Такі моделі визначають статистичні показники, що характеризують параметри штатної поведінки системи. Якщо з часом спостерігається певне відхилення даних параметрів від заданих значень, то фіксується факт виявлення атаки. Як правило, в якості таких параметрів можуть виступати рівень завантаження процесора, навантаження на канали зв'язку, штатний час роботи користувачів системи, кількість звернень до мережевих ресурсів і т. д.

Методи виявлення аномалій спрямовані на виявлення невідомих атак і вторгнень. Для СВА, що захищається на основі сукупності параметрів оцінки формується «образ» нормального функціонування. В сучасних СВА виділяють кілька способів побудови «образу»[23]:

• накопичення найбільш характерної статистичної інформації для кожного параметра оцінки;

• навчання нейронних мереж значеннями параметрів оцінки;

• представлення за подіями.

Легко помітити, що у виявленні дуже значну роль відіграє безліч параметрів оцінки. Тому в виявленні аномалій одним із головних завдань є вибір оптимальної безлічі параметрів оцінки. Іншим, не менш важливим завданням є визначення загального показника аномальності. Складність полягає в тому, що ця величина повинна характеризувати загальний стан «аномальності» в системі, що захищається.

2.4 Підготовка даних до побудови моделі

Для побудови моделі виявлення мережевих атак в даному дослідженні було обрано датасет Канадського інституту кібербезпеки Intrusion Detection Evaluation Dataset(CIC-IDS2017)[35]

Набір даних CICIDS2017 містить безпечні та найсучасніші поширені атаки, що нагадує справжні дані реального світу (PCAP). Він також включає результати аналізу мережевого трафіку за допомогою CICFlowMeter з позначеними потоками на основі відмітки часу, IP-адрес джерела та призначення, портів джерела та призначення, протоколів та атаки (файли CSV).

Створення реалістичного фонового трафіку було головним пріоритетом у створенні цього набору даних. При створенні датасету було використано систему B-Profile (Benign-профіль) для профілювання абстрактної поведінки людських взаємодій і створення натуралістичного доброякісного фонового трафіку. Для цього набору даних побудувано абстрактну поведінку 25 користувачів на основі протоколів HTTP, HTTPS, FTP, SSH та електронної пошти.

Реалізовані атаки включають Brute Force, XSS та SQL Injection та відносяться до M-Profile (Malignant-профіль).

Атака грубої сили (Brute Force): одна з найпопулярніших атак, яку можна використовувати не лише для злому паролів, а також для виявлення прихованих сторінок та вмісту веб-додатку. В основному це атака "вдар та спробуй", тоді жертва досягає успіху.

Веб-атака (Web Attack): атаки цього типу зараз виявляються щодня, тому що окремі особи та організації зараз серйозно ставляться до безпеки. В даному датасеті надано такі різновиди мережевих атак як ін'єкція SQL, під час якої зловмисник може створити рядок команд SQL, і потім використати його, щоб змусити базу даних відповісти на інформацію, та міжсайтовий сценарій (XSS), який відбувається, коли розробники не тестують свій код належним чином на знаходження можливості скрипту ін'єкції та грубої сили через HTTP, який може спробувати знайти список паролів пароль адміністратора.

Повна конфігурація мережі: Повна топологія мережі включає модем, брандмауер, комутатори, маршрутизатори та наявність різноманітних операційних систем, таких як Windows, Ubuntu та Mac OS X.

Для реалізації цих сценаріїв атак, було використано Damn Vulnerable Web App (DVWA), який є вразливим веб-додатком PHP/MySQL. Для автоматизації атак у розділі XSS та Brute-force було розроблено автоматизований код із фреймворком Selenium. Зловмисником є Kali Linux (205.174.165.73), а жертвою є система Ubuntu 16.04 як веб-сервер - 205.174.165.68 (Local IP 192.168.10.50).

Повна конфігурація мережі: Повна топологія мережі включає модем, брандмауер, комутатори, маршрутизатори та наявність різноманітних операційних систем, таких як Windows, Ubuntu та Mac OS X.

Дані датасету були отримані за допомогою аналізатора мережевого трафіку CICFlowMeter та представлені у вигляді csv файлу, що містить наступні атрибути мережевого потоку:

Flow ID – ідентифікатор потоку

Source IP – IP-адресу джерела

Source Port – порт джерела потоку

Destination IP – IP-адреса цілі призначення

Destination Port – порт призначення

Protocol – номер протоколу згідно класифікації IANA (Internet Assigned Numbers Authority)

Timestamp – часова мітка потоку

Flow Duration – тривалість потоку

Total Fwd Packets – загальна кількість відправлених пакетів

Total Backward Packets – загальна кількість отриманих пакетів

Total Length of Fwd Packets – загальний розмір відправлених пакетів

Total Length of Bwd Packets – загальний розмір отриманого пакета

Fwd Packet Length Max – максимальний розмір відправленого пакета

Fwd Packet Length Min – мінімальний розмір відправленого пакета

Fwd Packet Length Mean – середній розмір відправленого пакета

Fwd Packet Length Std – стандартне відхилення розміру відправленого пакета

Bwd Packet Length Max – максимальний розмір отриманого пакета

Bwd Packet Length Min – мінімальний розмір отриманого пакета

Bwd Packet Length Mean – середній розмір отриманого пакета

Bwd Packet Length Std – стандартне відхилення розміру отриманого пакета

Flow Bytes/s – швидкість потоку Байт/с

Flow Packets/s – швидкість потоку Пакет/с

Flow IAT Mean – середній інтервал між потоками

Flow IAT Std – стандартне відхилення інтервалу потоків

Flow IAT Max – максимальний інтервал потоків

Flow IAT Min – мінімальний інтервал потоків

Fwd IAT Total – загальний інтервал вихідних потоків

Fwd IAT Mean – середній інтервал вихідних потоків

Fwd IAT Std – стандартне відхилення інтервалу вихідних потоків

Fwd IAT Max – максимальний інтервал вихідних потоків

Fwd IAT Min – мінімальний інтервал вихідних потоків

Bwd IAT Total – загальний інтервал вхідних потоків

Bwd IAT Mean – середній інтервал вхідних потоків

Bwd IAT Std – стандартне відхилення інтервалу вхідних потоків

Bwd IAT Max – максимальний інтервал вхідних потоків

Bwd IAT Min – мінімальний інтервал вхідних потоків

Fwd PSH Flags – PSH-прапорці у вихідному потоці

Bwd PSH Flags – PSH-прапорці у вхідному потоці

Fwd URG Flags – URG-прапорці у вихідному потоці

Bwd URG Flags – URG-прапорці у вхідному потоці

Fwd Header Length – довжина заголовку (в байтах) у вихідному потоці

Bwd Header Length – довжина заголовку (в байтах) у вхідному потоці

Fwd Packets/s – швидкість відправки Пакет/с

Bwd Packets/s – швидкість отримання Пакет/с

Min Packet Length – мінімальний розмір пакета

Max Packet Length – максимальний розмір пакета

Packet Length Mean – середній розмір пакета

Packet Length Std – стандартне відхилення розміру пакета

Packet Length Variance – варіативність розміру пакета

FIN Flag Count – кількість FIN-прапорців

SYN Flag Count – кількість SYN-прапорців

RST Flag Count – кількість RST-прапорців

PSH Flag Count – кількість PSH-прапорців

ACK Flag Count – кількість ACK-прапорців

URG Flag Count – кількість URG-прапорців

CWE Flag Count – кількість CWE-прапорців

ECE Flag Count – кількість ECE-прапорців

Down/Up Ratio – відношення завантаження/відправлення даних

Average Packet Size – середній розмір пакета

Avg Fwd Segment Size – середній розмір відправленого сегмента

Avg Bwd Segment Size – середній розмір отриманого сегмента

Fwd Header Len – розмір відправленого заголовку

Subflow Fwd Packets – кількість пакетів у вихідному субпотоці

Subflow Fwd Bytes – кількість байт у вихідному субпотоці

Subflow Bwd Packets – кількість пакетів у вхідному субпотоці

Subflow Bwd Bytes – кількість байт у вхідному субпотоці

Init_Win_bytes_forward – кількість байтів в початковому вікні вихідного потоку

Init_Win_bytes_backward – кількість байтів в початковому вікні вхідного потоку

act_data_pkt_fwd – кількість пакетів з активним навантаженням TCP вихідного потоку

min_seg_size_forward – мінімальний розмір сегмента вихідного потоку

Active Mean – середній період активності потоку

Active Std – стандартне відхилення періоду активності потоку

Active Max – максимальний період активності потоку

Active Min – мінімальний період активності потоку

Idle Mean – середній період бездіяльності потоку

Idle Std – стандартне відхилення періоду бездіяльності потоку

Idle Max – максимальний період бездіяльності потоку

Idle Min – мінімальний період бездіяльності потоку

В сформованому датасеті мережевого трафіку також додано атрибут Label для позначення профілю позитивного чи шкідливого трафіку (B- чи M-) та вказано тип мережевої атаки (Brute Force, XSS чи SQL-injection).

Висновки за розділом 2

В даному розділі було розглянуто методи виявлення мережевих атак, проведено аналіз сигнатурних методів та методів машинного навчання, на основі якого буде побудована модель виявлення та ідентифікації в даній роботі.

В якості даних для побудови моделі було обрано датасет з відкритого джерела, який було сформовано за допомогою ПЗ CICFlowMeter, а також було розглянуто атрибути мережевого потоку, що будуть використовуватись в якості параметрів моделювання. Даний датасет було також промарковано відповідно до атак, що відбувались в даному проміжку, тому в даному досліджені буде розроблено модель, яка ідентифікуватиме мережеві атаки типів Brute Force, XSS та SQL-injection

РОЗДІЛ 3
РОЗРОБКА МОДЕЛІ ІДЕНТИФІКАЦІЇ МЕРЕЖЕВИХ АТАК

3.1 Алгоритм дерева рішень

Дерево рішень — це техніка класифікації, яка складається з трьох компонентів: кореневий вузол, гілка (край або ланка) і листковий вузол. Корінь представляє умову тестування для різних атрибутів, гілка представляє всі можливі результати, які можуть бути в тесті, а листові вузли містять мітку класу, до якого вона належить. Кореневий вузол знаходиться на початку дерева, яке також називають вершиною дерева.

Внутрішні вузли цього дерева рішень позначають різні атрибути, а гілки між вузлами повідомляють нам можливі значення, які ці атрибути можуть мати в спостережуваних вибірках, а кінцеві вузли повідомляють нам остаточне значення (класифікацію) залежної змінної. Прогнозований атрибут є залежною змінною, а інші атрибути в дереві є незалежними змінними в наборі даних.

J48 класифікатор – це алгоритм для створення дерева рішень, що генерується алгоритмом C4.5. Він також відомий як статистичний класифікатор. Для класифікації дерева рішень потрібна база даних.

Алгоритм C4.5 — це алгоритм класифікації, який створює дерева рішень на основі теорії інформації[36]. Це розширення попереднього алгоритму ID3 Росса Квінлана, також відомого в Weka як J48, J, що означає Java. Дерева рішень, створені C4.5, використовуються для класифікації, і з цієї причини C4.5 часто називають статистичним класифікатором.

Реалізація алгоритму C4.5 J48 має багато додаткових функцій, включаючи облік відсутніх значень, обрізання дерев рішень, безперервні діапазони значень атрибутів, виведення правил тощо. У інструменті аналізу даних WEKA J48 є реалізацією Java з відкритим вихідним кодом алгоритму C4.5. J48 дозволяє класифікувати або за допомогою дерев рішень, або правил, створених з них.

Цей алгоритм будує дерева рішень на основі набору навчальних даних так само, як це робить алгоритм ID3, використовуючи концепцію інформаційної ентропії. Навчальними даними є набір S={s1, s2, …} вже класифікованих зразків. Кожен зразок si складається з p-вимірного вектора (x1, i, x2, i, …, xp, i), де xj представляє значення атрибутів або ознак відповідного зразка, а також клас, до якого входить вибірка. Щоб отримати найвищу точність класифікації, найкращим атрибутом для розділення є атрибут з найбільшою інформацією.

У кожному вузлі дерева алгоритм C4.5 вибирає атрибут даних, який найбільш ефективно розбиває його набір вибірок на підмножини, збагачені в тому чи іншому класі. Критерієм розщеплення є нормований приріст інформації, який розраховується з різниці в ентропії. Для прийняття рішення вибирається атрибут з найбільшим нормованим інформаційним посиленням. Алгоритм C4.5 потім рекурсує до розділених підсписків, використовуючи підхід «розподіляй і володарюй», і створює дерево рішень на основі алгоритму.

Під час побудови дерева J48 ігнорує відсутні значення; значення цього елемента можна передбачити за значеннями атрибутів для інших записів. Основна ідея полягає в тому, щоб розділити дані на діапазони на основі значень атрибутів, знайдених у навчальній вибірці.

Деякі основні випадки використання цього алгоритму:

• Цей алгоритм створює список для всіх зразків у класі.

• Він оцінює функції, щоб оцінити будь-який приріст інформації, якщо приріст неможливий, то цей алгоритм створює вузол вище в дереві, використовуючи очікуване значення класу.

• Якщо цей алгоритм зустрічає невідомий клас, який він досі не бачив, він створює вузол вище в дереві, використовуючи очікуване значення.

Дерева рішень використовуються для окреслення процесів прийняття рішень. Це класифікатор, який діє як блок-схема, як структура дерева, щоб відобразити моделі асоціацій. Дерева рішень використовуються для категоризації екземплярів шляхом сортування їх по дереву від початку до маленького листкового вузла. Кожен

окремий вузол визначає перевірку екземпляра, і кожен окремий поділ відповідає одній із ймовірних переваг для цього атрибута.

Він ділить набір даних на найдрібніші та менші підмножини та поступово розвивається. Кінцевим наслідком є дерево поряд із вузлами рішень і листовими вузлами. Кожен вузол рішення має два або більше відділів, а листовий вузол втілює асоціацію або рішення. Найвищий вузол прийняття рішень у дереві, який відповідає найкращому предиктору, називається вихідним вузлом.

3.2 Розробка матриці рекомендацій вибору

Для досягнення цілей дослідження було використано ПЗ Weka для приведення сирих даних до необхідного для аналізу вигляду та подальшої побудови моделі виявлення та ідентифікації мережевих атак.

WEKA — це система аналізу даних, розроблена університетом Waikato в Новій Зеландії, яка реалізує алгоритми інтелекту даних. WEKA — це найсучасніший інструмент для розробки методів машинного навчання (ML) та їх застосування до реальних проблем інтелекту даних. Це набір алгоритмів машинного навчання для завдань інтелекту. Алгоритми застосовуються безпосередньо до набору даних.

WEKA реалізує алгоритми попередньої обробки даних, класифікації, регресії, кластеризації, правила асоціації; він також включає інструменти візуалізації. Нові схеми машинного навчання також можуть бути розроблені за допомогою цього пакета. WEKA — це програмне забезпечення з відкритим вихідним кодом, яке видається під загальною суспільною ліцензією GNU. Інструменти попередньої обробки, класифікації, кластеризації, асоціації, вибору атрибутів та візуалізації WEKA Explorer.

Початковий вигляд датасету в ПЗ Weka:

Dataset.csv *

Relation: Dataset

No.	1: Flow ID Nominal	2: Source IP Nominal	3: Source Port Numeric	4: Destination IP Nominal	5: Destination Port Numeric	6: Protocol Numeric	7: Timestamp Nominal	8: Flow Duration Numeric	9: Total Fwd Packets Numeric	10: Total Backward Packets Numeric	11: Total Length of Fwd Packets Numeric	12: Total Length of Bwd Packets Numeric
20...	172.16.0....	172.16.0.1	50214.0	192.168.10.50	80.0	6.0	6/7/2017 9:27	5473398.0	3.0	1.0	0.0	0.0
20...	192.168....	192.168.10....	52684.0	192.168.10.3	53.0	17.0	6/7/2017 9:27	237.0	2.0	2.0	72.0	328.0
20...	172.16.0....	192.168.10....	80.0	172.16.0.1	50116.0	6.0	6/7/2017 9:27	765.0	1.0	1.0	0.0	0.0
20...	192.168....	192.168.10....	54771.0	192.168.10.3	53.0	17.0	6/7/2017 9:27	303.0	2.0	2.0	72.0	328.0
20...	192.168....	72.21.91.29	80.0	192.168.10.51	37530.0	6.0	6/7/2017 9:27	29.0	1.0	1.0	0.0	0.0
20...	172.217....	172.217.6.2...	443.0	192.168.10.51	53040.0	6.0	6/7/2017 9:27	59.0	1.0	1.0	0.0	0.0
20...	172.217....	172.217.10....	443.0	192.168.10.51	38236.0	6.0	6/7/2017 9:27	10.0	1.0	1.0	0.0	0.0
20...	192.168....	209.85.201....	443.0	192.168.10.51	36719.0	6.0	6/7/2017 9:27	79.0	1.0	1.0	0.0	0.0
20...	172.16.0....	192.168.10....	80.0	172.16.0.1	50136.0	6.0	6/7/2017 9:27	583.0	1.0	1.0	0.0	0.0
20...	192.168....	216.58.219....	443.0	192.168.10.51	46629.0	6.0	6/7/2017 9:27	4.0	1.0	1.0	0.0	0.0
20...	192.168....	192.168.10.5	49633.0	192.168.10.3	135.0	6.0	6/7/2017 9:27	1.1999172E7	11.0	6.0	686.0	560.0
20...	192.168....	192.168.10.5	49634.0	192.168.10.3	49666.0	6.0	6/7/2017 9:27	1.1998299E7	11.0	6.0	968.0	912.0
20...	172.16.0....	192.168.10....	80.0	172.16.0.1	50156.0	6.0	6/7/2017 9:27	549.0	1.0	1.0	0.0	0.0
20...	172.16.0....	172.16.0.1	50252.0	192.168.10.50	80.0	6.0	6/7/2017 9:27	5934163.0	3.0	1.0	0.0	0.0
20...	159.203....	192.168.10....	123.0	159.203.8.72	123.0	17.0	6/7/2017 9:27	22527.0	1.0	1.0	48.0	48.0
20...	158.69.2...	192.168.10....	123.0	158.69.247.184	123.0	17.0	6/7/2017 9:27	16424.0	1.0	1.0	48.0	48.0
20...	192.168....	192.168.10....	123.0	97.107.128.58	123.0	17.0	6/7/2017 9:27	23577.0	1.0	1.0	48.0	48.0
20...	192.168....	23.50.75.27	80.0	192.168.10.51	47382.0	6.0	6/7/2017 9:27	41.0	1.0	1.0	0.0	0.0
20...	172.16.0....	172.16.0.1	50272.0	192.168.10.50	80.0	6.0	6/7/2017 9:27	5199490.0	3.0	1.0	0.0	0.0
20...	172.16.0....	172.16.0.1	50294.0	192.168.10.50	80.0	6.0	6/7/2017 9:27	8924975.0	4.0	4.0	602.0	364.0
20...	172.16.0....	192.168.10....	80.0	172.16.0.1	50194.0	6.0	6/7/2017 9:27	734.0	1.0	1.0	0.0	0.0
20...	192.168....	192.168.10....	123.0	67.215.197.149	123.0	17.0	6/7/2017 9:27	24623.0	1.0	1.0	48.0	48.0
20...	192.168....	192.168.10....	61729.0	192.168.10.3	53.0	17.0	6/7/2017 9:27	31371.0	4.0	2.0	128.0	256.0
20...	172.16.0....	192.168.10....	80.0	172.16.0.1	50214.0	6.0	6/7/2017 9:27	778.0	1.0	1.0	0.0	0.0
20...	192.168....	192.168.10....	53100.0	192.168.10.3	53.0	17.0	6/7/2017 9:27	186.0	2.0	2.0	82.0	114.0
20...	167.114....	192.168.10....	123.0	167.114.204.238	123.0	17.0	6/7/2017 9:27	16198.0	1.0	1.0	48.0	48.0
20...	172.16.0....	172.16.0.1	50332.0	192.168.10.50	80.0	6.0	6/7/2017 9:27	3.2857904E7	203.0	104.0	43951.0	72198.0
20...	192.168....	192.168.10....	123.0	72.38.129.202	123.0	17.0	6/7/2017 9:27	32246.0	1.0	1.0	48.0	48.0
20...	192.168....	192.168.10....	52166.0	104.31.1.161	80.0	6.0	6/7/2017 9:27	15244.0	1.0	1.0	6.0	6.0
20...	172.16.0....	172.16.0.1	50352.0	192.168.10.50	80.0	6.0	6/7/2017 9:27	5470453.0	3.0	1.0	0.0	0.0
20...	172.16.0....	192.168.10....	80.0	172.16.0.1	50252.0	6.0	6/7/2017 9:27	773.0	1.0	1.0	0.0	0.0
20...	172.16.0....	192.168.10....	80.0	172.16.0.1	50272.0	6.0	6/7/2017 9:27	460.0	1.0	1.0	0.0	0.0
20...	172.16.0....	172.16.0.1	49824.0	192.168.10.50	80.0	6.0	6/7/2017 9:27	21.0	1.0	1.0	0.0	0.0
20...	192.168....	192.168.10.3	61105.0	192.168.10.1	53.0	17.0	6/7/2017 9:27	299858.0	1.0	1.0	38.0	38.0
20...	192.168....	192.168.10.3	61276.0	192.168.10.1	53.0	17.0	6/7/2017 9:27	319481.0	1.0	1.0	38.0	38.0
20...	172.16.0....	172.16.0.1	50390.0	192.168.10.50	80.0	6.0	6/7/2017 9:27	5959897.0	3.0	1.0	0.0	0.0
20...	172.16.0....	172.16.0.1	50410.0	192.168.10.50	80.0	6.0	6/7/2017 9:27	5698207.0	3.0	1.0	0.0	0.0
20...	192.168....	192.168.10.5	49633.0	192.168.10.3	135.0	6.0	6/7/2017 9:27	37.0	1.0	1.0	6.0	6.0
20...	192.168....	192.168.10.5	49633.0	192.168.10.3	135.0	6.0	6/7/2017 9:27	1.0	1.0	1.0	6.0	6.0
20...	192.168....	192.168.10.5	49634.0	192.168.10.3	49666.0	6.0	6/7/2017 9:27	59.0	1.0	1.0	6.0	6.0
20...	172.16.0....	172.16.0.1	50430.0	192.168.10.50	80.0	6.0	6/7/2017 9:27	5432004.0	3.0	1.0	0.0	0.0
20...	172.16.0....	172.16.0.1	50294.0	192.168.10.50	80.0	6.0	6/7/2017 9:27	42.0	1.0	1.0	0.0	0.0
20...	192.168....	192.168.10.3	49666.0	192.168.10.5	49634.0	6.0	6/7/2017 9:27	58.0	1.0	2.0	6.0	12.0
20...	172.16.0....	192.168.10....	80.0	172.16.0.1	50352.0	6.0	6/7/2017 9:27	742.0	1.0	1.0	0.0	0.0
20...	172.16.0....	172.16.0.1	50468.0	192.168.10.50	80.0	6.0	6/7/2017 9:28	5904686.0	3.0	1.0	0.0	0.0
20...	172.16.0....	172.16.0.1	50488.0	192.168.10.50	80.0	6.0	6/7/2017 9:28	5646956.0	3.0	1.0	0.0	0.0
20...	172.16.0....	192.168.10....	80.0	172.16.0.1	50390.0	6.0	6/7/2017 9:28	755.0	1.0	1.0	0.0	0.0
20...	172.16.0....	172.16.0.1	50508.0	192.168.10.50	80.0	6.0	6/7/2017 9:28	5385903.0	3.0	1.0	0.0	0.0

Рисунок 3.1 – Початковий вигляд датасету

Після цього було виконано збереження датасету у форматі .arff, який використовується при побудові моделі:

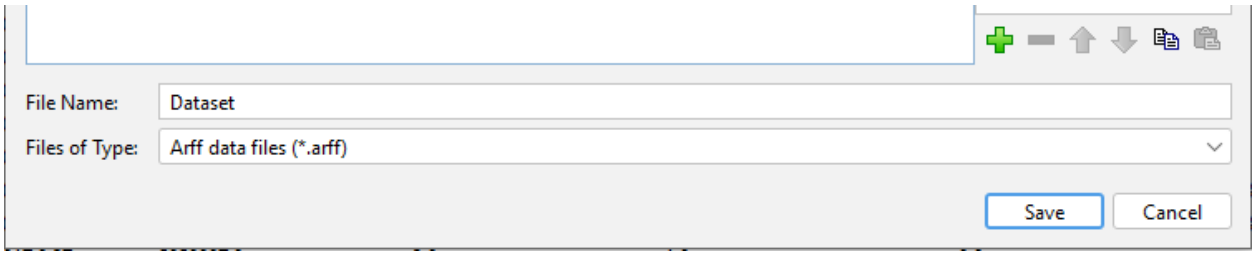

Рисунок 3.2 – Збереження датасету у форматі .arff

3.2.1 Початкова обробка даних

При відкритті датасету за допомогою Weka Explorer бачимо представлені дані в наступному вигляді:

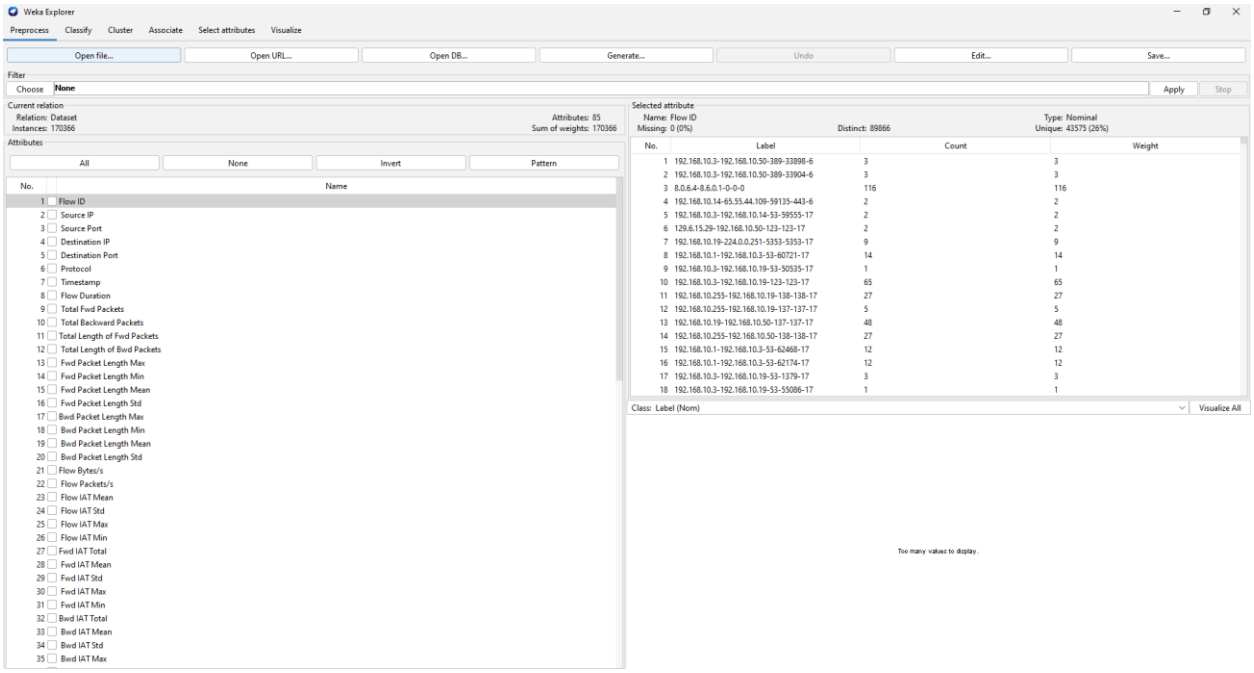

Рисунок 3.3 – Дані датасету у Weka Explorer

Необхідно виконати підготовку даних до аналізу, а саме – видалити атрибути, які мають одне значення, а також ті, що не несуть ніякої цінності для побудови моделі (Flow ID).

Внаслідок цих дій було видалено наступні атрибути: CWE Flag count, Bwd URG Flags, Fwd URG Flags, Bwd PSH Flags, Flow Packets/s, Flow Bytes/s, Flow ID.

Для деяких атрибутів необхідно виконати форматування типу з цифрового в номінальний, адже у випадку аналізу портів чи протоколу нас цікавитиме номінальне значення, а не його належність до певної групи чисел. Зробити це можна за допомогою фільтру атрибутів ПЗ Weka, що знаходиться в каталозі filters-unsupervised-attribute та має назву NumericToNominal:

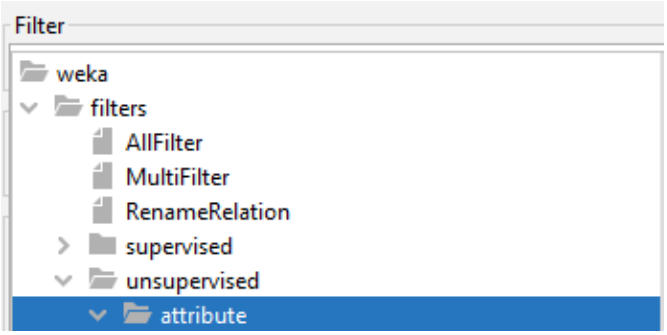

Рисунок 3.4 – Каталог фільтрів ПЗ Weka

Наступним етапом підготовки даних є їх нормалізація за допомогою фільтру Normalize:

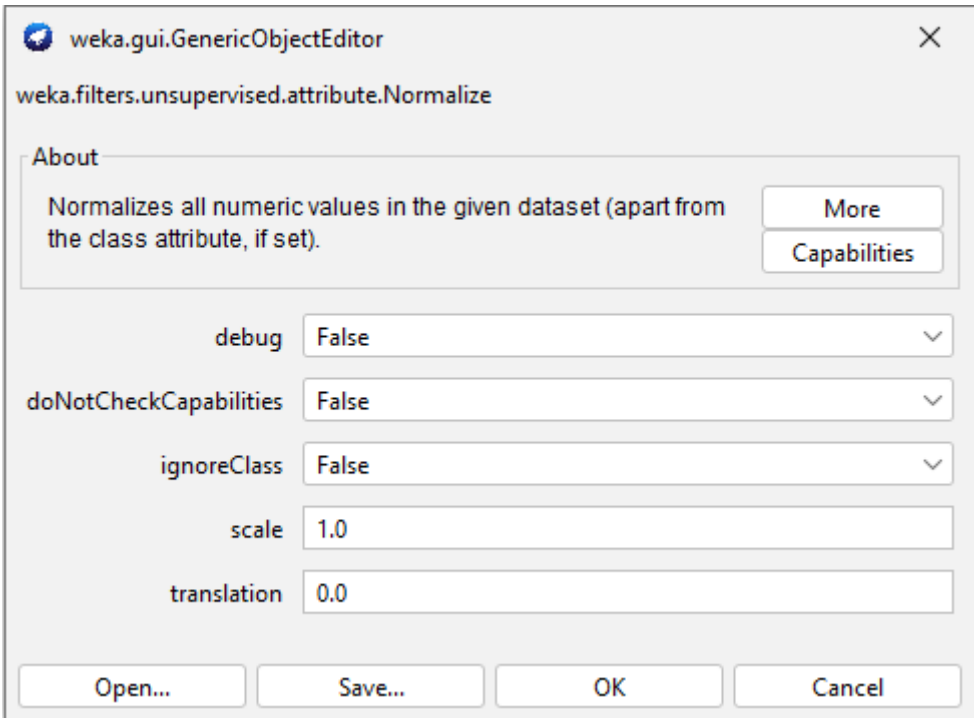

Рисунок 3.5 – Фільтр Normalize в ПЗ Weka

При застосуванні цього фільтру відбувається перетворення великих значень атрибутів в межі від 0 до 1 зі збереженням пропорції. В такий спосіб зменшується час обробки даних та збільшується швидкість навчання моделі.

3.2.2 Класифікація даних

Далі переходимо до класифікації даних за допомогою алгоритму дерева рішень J48:

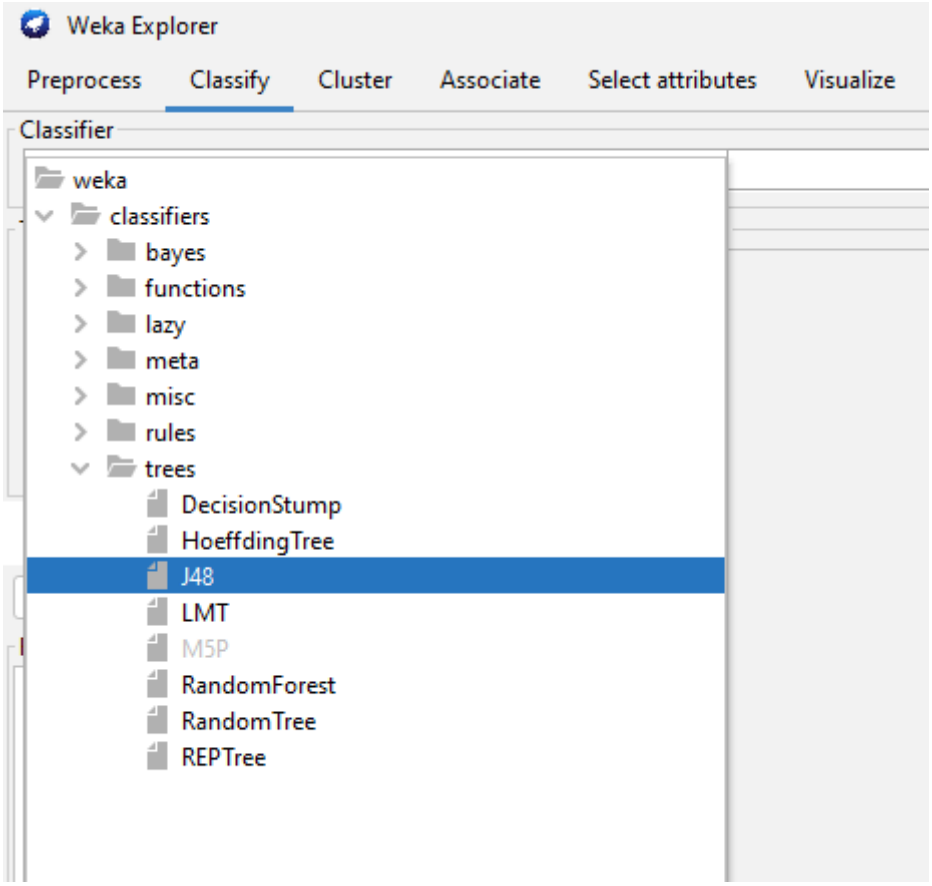

Рисунок 3.6 – Каталог алгоритмів в ПЗ Weka

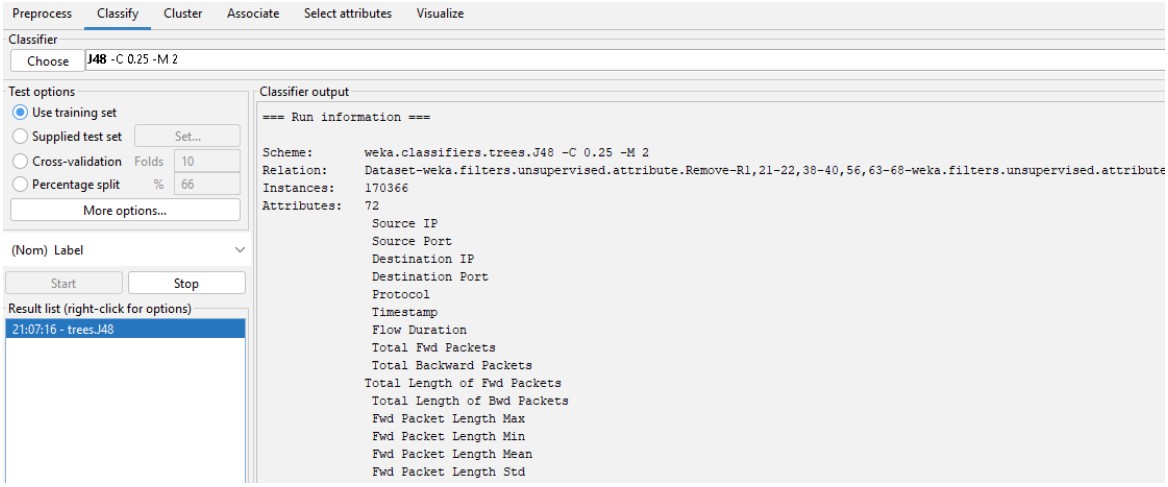

Рисунок 3.7 – Запуск алгоритму класифікації J48

Результати першого запуску тренувального сету (повний код представлений в Додатку А):

```
=== Summary ===

Correctly Classified Instances      170168               99.8838 %
Incorrectly Classified Instances       198                0.1162 %
Kappa statistic                          0.9521
Mean absolute error                      0.0012
Root mean squared error                  0.0241
Relative absolute error                  9.1418 %
Root relative squared error             30.2456 %
Total Number of Instances           170366

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1,000    0,089    0,999      1,000    0,999      0,953    0,961     0,999     BENIGN
                 0,900    0,000    0,998      0,900    0,947      0,947    0,956     0,903     Web Attack - Brute Force
                 0,956    0,000    0,998      0,956    0,976      0,977    0,982     0,960     Web Attack - XSS
                 0,333    0,000    0,875      0,333    0,483      0,540    0,805     0,294     Web Attack - Sql Injection
Weighted Avg.    0,999    0,087    0,999      0,999    0,999      0,953    0,961     0,998

=== Confusion Matrix ===

      a      b      c      d   <-- classified as
 168181      3      1      1 |      a = BENIGN
    150   1357      0      0 |      b = Web Attack - Brute Force
     29      0    623      0 |      c = Web Attack - XSS
     14      0      0      7 |      d = Web Attack - Sql Injection
```

Рисунок 3.8 – Результат першого запуску алгоритму класифікації

Після цього доцільно видалити деякі атрибути, що не впливають на результат, такі як Destination IP та Timestamp і повторити тренування. Результати другого запуску тренування (в повному вигляді результат представлений у Додатку Б):

```
=== Summary ===

Correctly Classified Instances      169565               99.5298 %
Incorrectly Classified Instances       801                0.4702 %
Kappa statistic                          0.8059
Mean absolute error                      0.0035
Root mean squared error                  0.0419
Relative absolute error                 27.735  %
Root relative squared error             52.6818 %
Total Number of Instances           170366

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1,000    0,089    0,999      1,000    0,999      0,953    0,961     0,999     BENIGN
                 0,900    0,004    0,691      0,900    0,782      0,787    0,954     0,654     Web Attack - Brute Force
                 0,028    0,000    1,000      0,028    0,054      0,166    0,978     0,344     Web Attack - XSS
                 0,333    0,000    0,875      0,333    0,483      0,540    0,805     0,294     Web Attack - Sql Injection
Weighted Avg.    0,995    0,087    0,996      0,995    0,994      0,949    0,961     0,993

=== Confusion Matrix ===

      a      b      c      d   <-- classified as
 168183      2      0      1 |      a = BENIGN
    150   1357      0      0 |      b = Web Attack - Brute Force
     29    605     18      0 |      c = Web Attack - XSS
     14      0      0      7 |      d = Web Attack - Sql Injection
```

Рисунок 3.9 – Результат другого запуску алгоритму класифікації

Тепер необхідно видалити атрибут Destination Port для більш коректної структури дерева рішень і знову виконати запуск тренування. Результат третьої спроби (повний – Додаток В):

```
=== Summary ===

Correctly Classified Instances      169566               99.5304 %
Incorrectly Classified Instances       800                0.4696 %
Kappa statistic                          0.8061
Mean absolute error                      0.0035
Root mean squared error                  0.0419
Relative absolute error                 27.699  %
Root relative squared error             52.6475 %
Total Number of Instances           170366

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1,000    0,089    0,999      1,000    0,999      0,954    0,961     0,999     BENIGN
                 0,900    0,004    0,691      0,900    0,782      0,787    0,954     0,654     Web Attack - Brute Force
                 0,028    0,000    1,000      0,028    0,054      0,166    0,978     0,345     Web Attack - XSS
                 0,333    0,000    0,875      0,333    0,483      0,540    0,805     0,294     Web Attack - Sql Injection
Weighted Avg.    0,995    0,087    0,996      0,995    0,994      0,949    0,961     0,993

=== Confusion Matrix ===

      a      b      c      d   <-- classified as
 168184      1      0      1 |      a = BENIGN
    150   1357      0      0 |      b = Web Attack - Brute Force
     29    605     18      0 |      c = Web Attack - XSS
     14      0      0      7 |      d = Web Attack - Sql Injection
```

Рисунок 3.10 – Результат третього запуску алгоритму класифікації

Аналогічно до попередніх кроків видаляємо атрибут Source IP, так як він також не впливає на процес ідентифікації атаки. Повторюємо запуск і отримуємо результат (повний результат наявний у Додатку Г):

```
=== Summary ===

Correctly Classified Instances      169565               99.5298 %
Incorrectly Classified Instances       801                0.4702 %
Kappa statistic                          0.8059
Mean absolute error                      0.0035
Root mean squared error                  0.0419
Relative absolute error                 27.7234 %
Root relative squared error             52.6707 %
Total Number of Instances           170366

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1,000    0,089    0,999      1,000    0,999      0,953    0,961     0,999     BENIGN
                 0,900    0,004    0,691      0,900    0,782      0,787    0,954     0,658     Web Attack - Brute Force
                 0,028    0,000    1,000      0,028    0,054      0,166    0,978     0,346     Web Attack - XSS
                 0,333    0,000    0,875      0,333    0,483      0,540    0,805     0,294     Web Attack - Sql Injection
Weighted Avg.    0,995    0,087    0,996      0,995    0,994      0,949    0,961     0,993

=== Confusion Matrix ===

      a      b      c      d   <-- classified as
 168183      2      0      1 |      a = BENIGN
    150   1357      0      0 |      b = Web Attack - Brute Force
     29    605     18      0 |      c = Web Attack - XSS
     14      0      0      7 |      d = Web Attack - Sql Injection
```

Рисунок 3.11 – Результат четвертого запуску алгоритму класифікації

На цьому кроці було досягнуто адекватної структури дерева рішень, про що свідчить візуалізація даних:

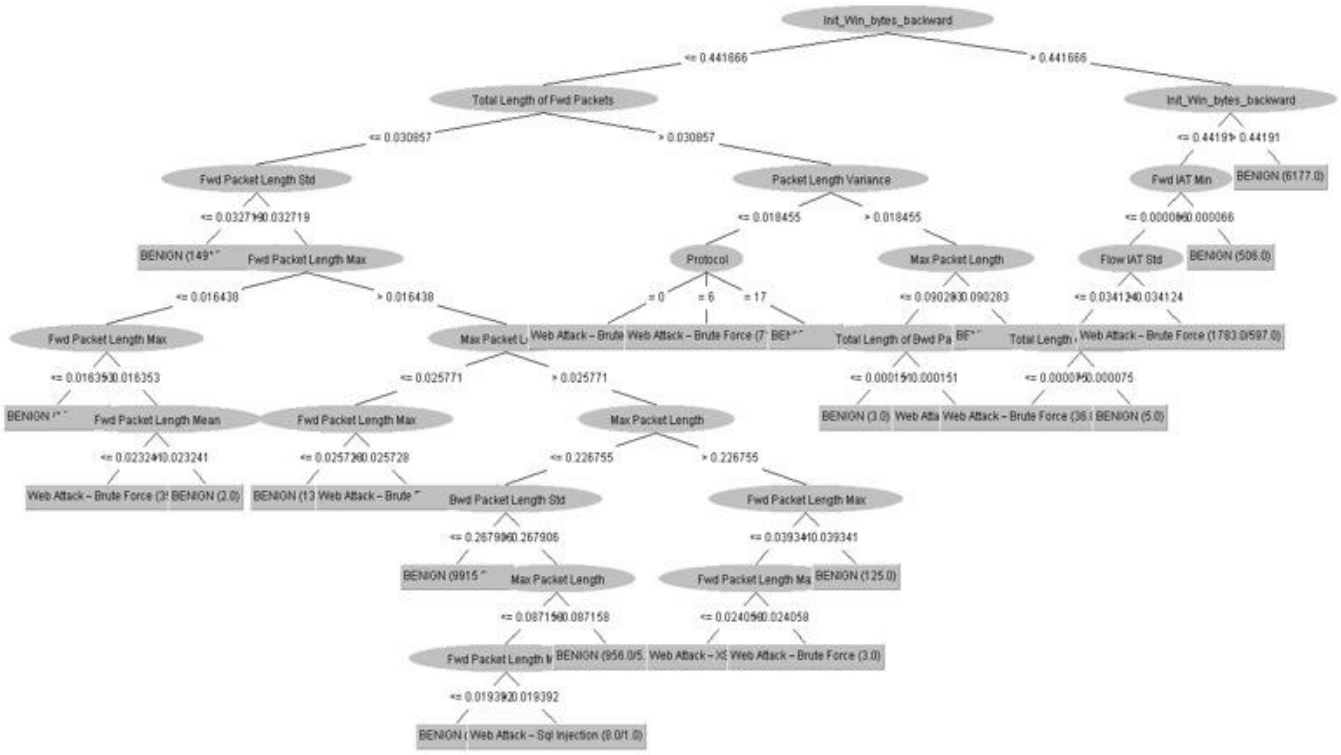

Рисунок 3.12 – Візуалізація результатів моделі класифікації

3.3 Тестування моделі

Для тестування працездатності моделі було створено тестовий датасет з вибірковими даними мережевого потоку, що відносяться до тієї чи іншої атаки, що наявні в моделі, файл з вхідними даними має розширення .arff та наступний вигляд:

@attribute ' Protocol' {6}
@attribute ' Flow Duration' numeric
@attribute ' Total Fwd Packets' numeric
@attribute ' Total Backward Packets' numeric
@attribute 'Total Length of Fwd Packets' numeric
@attribute ' Total Length of Bwd Packets' numeric
@attribute ' Fwd Packet Length Max' numeric
@attribute ' Fwd Packet Length Min' numeric
@attribute ' Fwd Packet Length Mean' numeric
@attribute ' Fwd Packet Length Std' numeric
@attribute 'Bwd Packet Length Max' numeric
@attribute ' Bwd Packet Length Min' numeric
@attribute ' Bwd Packet Length Mean' numeric
@attribute ' Bwd Packet Length Std' numeric
@attribute ' Flow IAT Mean' numeric
@attribute ' Flow IAT Std' numeric
@attribute ' Flow IAT Max' numeric
@attribute ' Flow IAT Min' numeric
@attribute 'Fwd IAT Total' numeric
@attribute ' Fwd IAT Mean' numeric
@attribute ' Fwd IAT Std' numeric
@attribute ' Fwd IAT Max' numeric
@attribute ' Fwd IAT Min' numeric
@attribute 'Bwd IAT Total' numeric
@attribute ' Bwd IAT Mean' numeric

@attribute ' Bwd IAT Std' numeric

@attribute ' Bwd IAT Max' numeric

@attribute ' Bwd IAT Min' numeric

@attribute 'Fwd PSH Flags' numeric

@attribute ' Fwd Header Length' numeric

@attribute ' Bwd Header Length' numeric

@attribute 'Fwd Packets/s' numeric

@attribute ' Bwd Packets/s' numeric

@attribute ' Min Packet Length' numeric

@attribute ' Max Packet Length' numeric

@attribute ' Packet Length Mean' numeric

@attribute ' Packet Length Std' numeric

@attribute ' Packet Length Variance' numeric

@attribute 'FIN Flag Count' numeric

@attribute ' SYN Flag Count' numeric

@attribute ' RST Flag Count' numeric

@attribute ' PSH Flag Count' numeric

@attribute ' ACK Flag Count' numeric

@attribute ' URG Flag Count' numeric

@attribute ' ECE Flag Count' numeric

@attribute ' Down/Up Ratio' numeric

@attribute ' Average Packet Size' numeric

@attribute ' Avg Fwd Segment Size' numeric

@attribute ' Avg Bwd Segment Size' numeric

@attribute ' Fwd Header Len' numeric

@attribute 'Subflow Fwd Packets' numeric

@attribute ' Subflow Fwd Bytes' numeric

@attribute ' Subflow Bwd Packets' numeric

@attribute ' Subflow Bwd Bytes' numeric

@attribute Init_Win_bytes_forward numeric

@attribute ' Init_Win_bytes_backward' numeric

@attribute ' act_data_pkt_fwd' numeric

@attribute ' min_seg_size_forward' numeric

@attribute 'Active Mean' numeric

@attribute ' Active Std' numeric

@attribute ' Active Max' numeric

@attribute ' Active Min' numeric

@attribute 'Idle Mean' numeric

@attribute ' Idle Std' numeric

@attribute ' Idle Max' numeric

@attribute ' Idle Min' numeric

@attribute ' Label' {'Web Attack – Brute Force','Web Attack – XSS','Web Attack – Sql Injection', BENIGN}

@data

6,0.043209,0.00003,0.000026,0.000854,0.000004,0.015925,0,0.034903,0.033693,0.07968,0,0.094872,0.12805,0.003324,0.016181,0.041366,0,0.001843,0.000307,0.000675,0.001179,0,0.043208,0.007201,0.024341,0.04168,0.000007,0,0.000056,0.000043,0,0.006,0,0,0.103393,0.087269,0.007623,0,0,0,1,0,0,0,0.1,0.085771,0.034903,0.094872,0.000056,0.00003,0.000854,0.000026,0.000004,0.00245,0.00361,0.000016,0.533333,0,0,0,0,0,0,0,0,'Web Attack – Brute Force'

6,0.000001,0,0.000004,0,0,0,0,0,0,0,0,0,0,0.000001,0,0.000001,0.000001,0,0,0,0,0,0,0,0,0,0,0,0,0.000008,0.000006,0.004115,0.006173,0,0,0,0,0,0,0,0,0,1,1,0,0.1,0,0,0,0.000008,0,0,0.000004,0,0.003154,0.00358,0,0.533333,0,0,0,0,0,0,0,0,'Web Attack – Brute Force'

6,0.045619,0.00001,0.000004,0,0,0,0,0,0,0,0,0,0,0,0,0.015206,0.037262,0.045612,0.000001,0.045619,0.02281,0.046511,0.045612,0.000007,0,0,0,0,0,0,0.000025,0.000007,0,0,0.007,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0.000025,0.00001,0,0.000004,0,0.001548,0.003597,0,0.533333,0,0,0,0,0,0,0,0,'Web Attack – Brute Force'

6,0.046987,0.00001,0.000004,0.0312,0.0002,0,0,0,0,0,0,0,0,0.015662,0.038382,0.046981,0.000001,0.046987,0.023493,0.047909,0.046981,0.000006,0,0,0,0,0,0,0.000025,0.000007,0,0,0,0,0,0,0.02487,0,0,0,1,0,0,0,0,0,0,0,0.000025,0.00001,0,0.000004,0,0.445572,0.43191,0,0.533333,0,0,0,0,0,0,0,0,'Web Attack – XSS'

6,0.046988,0.00001,0.000004,0.0416,0.0002,0,0,0,0,0,0,0,0,0.015663,0.038382,0.046981,0.000001,0.046988,0.023494,0.047908,0.046981,0.000006,0,0,0,0,0,0,0.000025,0.000007,0,0,0,0,0,0,0.021546,0,0,0,1,0,0,0,0,0,0,0,0.000025,0.00001,0,0.000004,0,0.445572,0.43191,0,0.533333,0,0,0,0,0,0,0,0,'Web Attack – XSS'

6,0.049548,0.00001,0.000004,0.0354,0.0002,0,0,0,0,0,0,0,0,0.016516,0.040474,0.049542,0,0.049548,0.024774,0.050521,0.049542,0.000005,0,0,0,0,0,0,0.000025,0.000007,0,0,0,0,0,0,0.023454,0,0,0,1,0,0,0,0,0,0,0,0.000025,0.00001,0,0.000004,0,0.445572,0.43191,0,0.533333,0,0,0,0,0,0,0,0,'Web Attack – XSS'

6,0.041718,0.000015,0.000015,0.000373,0.000001,0.019435,0,0.026715,0.040908,0.040335,0,0.13162,0.267948,0.006036,0.02224,0.041674,0,0.000575,0.000192,0.000469,0.000568,0,0.042248,0.014083,0.034433,0.04168,0.000007,0,0.000033,0.000025,0,0,0,0.078767,0.118558,0.161549,0.026123,0,0,0,1,0,0,0,0.1,0.103269,0.027492,0.13162,0.000033,0.000015,0.000384,0.000015,0.000003,0.445572,0.003601,0.000005,0.533333,0,0,0,0,0,0,0,0,'Web Attack – Sql Injection'

6,0.042249,0.000015,0.000015,0.000384,0.000003,0.019692,0,0.027492,0.042098,0.14003,0,0.13162,0.267948,0.006036,0.02224,0.041674,0,0.000575,0.000192,0.000469,0.000568,0,0.042248,0.014083,0.034433,0.04168,0.000007,0,0.000033,0.000025,0,0,0,0.078767,0.118558,0.161549,0.026123,0,0,0,1,0,0,0,0.1,0.103269,0.027492,0.13162,0.000033,0.000015,0.000384,0.000015,0.000003,0.445572,0.003601,0.000005,0.533333,0,0,0,0,0,0,0,0,'Web Attack – Sql Injection'

6,0.003022,0.000025,0.000026,0.001189,0.000002,0.036601,0,0.056697,0.066672,0.073135,0,0.046762,0.103698,0.000252,0.000595,0.000948,0,0.001891,0.000378,0.000743,0.000946,0.000002,0.002083,0.000347,0.000679,0.000952,0,0,0.000048,0.000043,0.000006,0.00001,0,0.041139,0.085064,0.088807,0.007894,0,0,0,1,0,0,0,0.1,0.070927,0.056697,0.046762,0.000048,0.000025,0.001189,0.000026,0.000002,0.445572,0.003799,0.000016,0.533333,0,0,0,0,0,0,0,0,BENIGN

6,0.000044,0.000005,0,0,0,0,0,0,0,0,0,0,0.000044,0,0.000044,0.000044,0.000044,0.000044,0,0.000044,0.000044,0,0,0,0,0,0.000015,0,0.000128,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0.000015,0.000005,0,0,0,0.00386,0,0,0.533333,0,0,0,0,0,0,0,0,BENIGN

В опціях тестування обираємо параметр Supplied test set та відкриваємо наш тестовий файл:

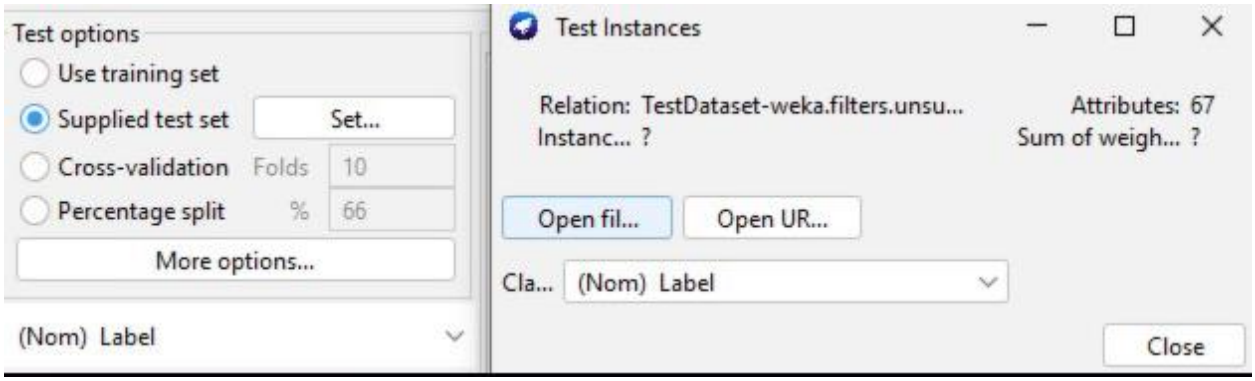

Рисунок 3.13 – Відкриття файлу для тестування моделі

Після чого запускаємо тестування та отримуємо наступні результати класифікації:

```
=== Summary ===

Correctly Classified Instances          10               100      %
Incorrectly Classified Instances         0                 0      %
Kappa statistic                          1
K&B Relative Info Score                 99.367  %
K&B Information Score                   69.971  bits      6.9971 bits/instance
Class complexity | order 0              70.4168 bits      7.0417 bits/instance
Class complexity | scheme                0.4458 bits      0.0446 bits/instance
Complexity improvement     (Sf)         69.971  bits      6.9971 bits/instance
Mean absolute error                      0.0146
Root mean squared error                  0.0399
Relative absolute error                  3.6513 %
Root relative squared error              6.3618 %
Total Number of Instances               10

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               1,000    0,000    1,000      1,000    1,000      1,000    1,000     1,000     BENIGN
               1,000    0,000    1,000      1,000    1,000      1,000    1,000     1,000     Web Attack - Brute Force
               1,000    0,000    1,000      1,000    1,000      1,000    1,000     1,000     Web Attack - XSS
               1,000    0,000    1,000      1,000    1,000      1,000    1,000     1,000     Web Attack - Sql Injection
Weighted Avg.  1,000    0,000    1,000      1,000    1,000      1,000    1,000     1,000

=== Confusion Matrix ===

 a b c d   <-- classified as
 2 0 0 0 | a = BENIGN
 0 3 0 0 | b = Web Attack - Brute Force
 0 0 3 0 | c = Web Attack - XSS
 0 0 0 2 | d = Web Attack - Sql Injection
```

Рисунок 3.14 – Результати тестування моделі класифікації

Як бачимо, всі дані було класифіковано вірно, а отже модель можна вважати робочою.

3.4 Аналіз результатів

Порівняно з результатами навчання моделі (98% вірно ідентифікованих та класифікованих атак) маємо результат 100% під час тестування моделі, бо тестові дані не мали записів, що містять відхилення від правил ідентифікації. У випадку дослідження таких типів мережевих атак як Brute Force, XSS та SQL-injection визначальними атрибутами ідентифікації є:

• Init_Win_bytes_backward – кількість байтів в початковому вікні вхідного потоку,

• Total Length of Fwd Packets – загальний розмір відправлених пакетів,

• Max Packet Length – максимальний розмір пакета,

• Bwd Packets/s – швидкість отримання Пакет/с,

• Init_Win_bytes_forward – кількість байтів в початковому вікні вихідного потоку,

• Protocol – номер протоколу,

• Flow IAT Min – мінімальний інтервал потоків,

• Fwd IAT Std – стандартне відхилення інтервалу вихідних потоків.

Висновки за розділом 3

В даному розділі було описано та розроблено модель виявлення та ідентифікації мережевих атак на базі машинного навчання за допомогою ПЗ Weka. Результати дослідження дали змогу отримати дані щодо найважливіших атрибутів для класифікації атаки, та значення цих атрибутів, за допомогою яких відбувається процес ідентифікації шляхом побудови дерева рішень.

ВИСНОВКИ

Під час виконання роботи було проаналізовано проблематику логування та моніторингу сучасних інформаційних систем, розглянуто методи виявлення мережевих атак та розроблено модель виявлення та ідентифікації на основі машинного навчання.

При аналізі параметрів логування та моніторингу було виокремлено наступні параметри, що безпосередньо впливають на виявлення та ідентифікацію мережевих атак: розмір журналу, різноманітність форматів, швидкість обробки журналів, правила кореляції та зберігання журналів.

Було виконано збір даних про існуючі мережеві атаки шляхом накопичення мережевого трафіку подіями, що були класифіковані за профілями B (Benign), тобто відображали звичайну поведінку користувача в мережі (доброякісний трафік), та M (Malignant), які несли в собі інформацію щодо певних мережевих атак (Brute Force, XSS та SQL Injection).

Всі зібрані дані було приведено до узагальненого вигляду за рахунок видалення атрибутів, які були зі сталими значеннями, форматування типів деяких атрибутів в номінальний та нормалізації великих значень атрибутів в межі від 0 до 1 зі збереженням пропорції задля оптимізації швидкості обробки даних.

Побудовано модель виявлення та ідентифікації мережевих атак на основі алгоритму дерева рішень J48 методом класифікації даних та отримано візуалізацію дерева рішень, яке дає змогу ідентифікувати ту чи іншу атаку.

Під час перевірки адекватності створеної моделі було отримано точність результатів ідентифікації мережевих атак 98%, що вказує на можливість застосування результатів дослідження в інформаційних системах малого та середнього бізнесу в якості складової систем моніторингу та виявлення вторгнень з подальшим навчанням та вдосконаленням.

Перспективою розвитку даних моделей є впровадження основ штучного інтелекту для обробки та підготовки даних, а також для вибору найкращого алгоритму з метою отримання найбільш точних результатів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Logging and Monitoring – An Essential Part of Every Security Program [Електронний ресурс] – Режим доступу до ресурсу: https://linfordco.com/blog/logging-and-monitoring/.

2. Comprehensive Guide to Insufficient Logging & Monitoring and How to Prevent It [Електронний ресурс] – Режим доступу до ресурсу: https://crashtest-security.com/insufficient-logging-monitoring-guide/.

3. Logging and Monitoring – An Essential Part of Every Security Program [Електронний ресурс] – Режим доступу до ресурсу: https://itglobal.com/ru-ru/company/blog/logirovanie/.

4. What’s the Difference? Logging vs Monitoring [Електронний ресурс] – Режим доступу до ресурсу: https://www.appdynamics.com/product/how-it-works/application-analytics/log-analytics/monitoring-vs-logging-best-practices/.

5. ImmuniWeb AI Platform [Електронний ресурс] – Режим доступу до ресурсу: www.immuniweb.com/.

6. OWASP Top 10 - 2021 [Електронний ресурс] – Режим доступу до ресурсу: https://owasp.org/Top10/.

7. A09:2021 – Security Logging and Monitoring Failures [Електронний ресурс] – Режим доступу до ресурсу: https://owasp.org/Top10/A09_2021-Security_Logging_and_Monitoring_Failures/.

8. Understanding log management: issues and challenges [Електронний ресурс] – Режим доступу до ресурсу: https://www.graylog.org/post/understanding-log-management-issues-and-challenges.

9. Log Management Challenges in Modern IT Environments [Електронний ресурс] – Режим доступу до ресурсу: https://www.virtualmetric.com/blog/log-management-challenges-in-modern-it-environments/.

10. LOG MANAGEMENT ISSUES - LOG CORRELATION [Електронний ресурс] – Режим доступу до ресурсу: https://www.graylog.org/post/understanding-log-management-issues-and-challenges/.

11. CWE-117: Improper Output Neutralization for Logs [Електронний ресурс] – Режим доступу до ресурсу: https://cwe.mitre.org/data/definitions/117.html.

12. CWE-223: Omission of Security-relevant Information [Електронний ресурс] – Режим доступу до ресурсу: https://cwe.mitre.org/data/definitions/223.html.

13. CWE-532: Insertion of Sensitive Information into Log File [Електронний ресурс] – Режим доступу до ресурсу: https://cwe.mitre.org/data/definitions/532.html.

14. CWE-778: Insufficient Logging [Електронний ресурс] – Режим доступу до ресурсу: https://cwe.mitre.org/data/definitions/778.html.

15. Log Injection [Електронний ресурс] – Режим доступу до ресурсу: https://owasp.org/www-community/attacks/Log_Injection.

16. Why You Need Both Logging and Monitoring [Електронний ресурс] – Режим доступу до ресурсу: https://www.appdynamics.com/product/how-it-works/application-analytics/log-analytics/monitoring-vs-logging-best-practices#~cloudops-vs-devops.

17. Логирование: что это и в чем его польза [Електронний ресурс] – Режим доступу до ресурсу: https://itglobal.com/ru-ru/company/blog/logirovanie/.

18. C9: Implement Security Logging and Monitoring [Електронний ресурс] – Режим доступу до ресурсу: https://owasp.org/www-project-proactive-controls/v3/en/c9-security-logging.html.

19. Pokhrel R., Pokharel P., Kumar Timalsina A. Anomaly-Based – Intrusion Detection System using User Profile Generated from System Logs. International Journal of Scientific and Research Publications (IJSRP). 2019. Vol. 9, no. 2. P. p8631. URL: https://doi.org/10.29322/ijsrp.9.02.2019.p8631.

20. Носенко К. М. Огляд систем виявлення атак в мережевому трафіку. Адаптивні системи автоматичного управління. 2014. Т. 1, № 24. С. 67–75. URL: https://doi.org/10.20535/1560-8956.24.2014.38193.

21. S. Kumar. Classification and Detection of Computer Intrusions. Department of Computer Science, Purdue University, -, PhD Dissertation -, 1995. URL: https://www.cerias.purdue.edu/assets/pdf/bibtex_archive/95-08.pdf.

22. Kabiri P., Ghorbani A.A. Research on Intrusion Detection and Response: A Survey //International Journal of Network Security. 2005. vol. 1, no. 2. pp. 84–102.

23. Мостовий М. Е. Аналіз ефективності використання систем виявлення вторгнень для аудиту безпеки. 2020. URL: https://openarchive.nure.ua/bitstream/document/15226/1/2020_M_BIT_Mostovy_ME.pdf.

24. V.V.R. Prasad, K.M. Prasad V. Jyothsna, "A Review of Anomaly based Intrusion Detection Systems," International Journal of Computer Applications (0975 – 8887), vol. 28, no. 7, pp. 26-35, August 2011.

25. Human perspective to anomaly detection for cybersecurity [Електронний ресурс] – Режим доступу до ресурсу: https://link.springer.com/article/10.1007/s10844-013-0266-3.

26. Браницкий А. А., Котенко И. В. Анализ и классификация методов обнаружения сетевых атак. Тр. СПИИРАН, 2016. № 45. С. 207–244. URL: https://doi.org/10.15622/sp.45.13

27. Snort. Open Source Intrusion Detection System // URL: https://www.snort.org/.

28. Suricata. Open Source IDS/IPS/NSM engine // URL: http://suricata-ids.org/

29. Лукацький А.В. Виявлення атак// СПб.: БХВ-Петербург. 2003. с.608

30. N. A. Durgin and P. Zhang, "Profile-Based Adaptive Anomaly Detection for Network Security," Sandia National Laboratories, Livermore, California, SANDIA REPORT SAND2005-7293, 2005.

31. Application Logging Vocabulary Cheat Sheet [Електронний ресурс] – Режим доступу до ресурсу: https://cheatsheetseries.owasp.org/cheatsheets/Logging_Vocabulary_Cheat_Sheet.html.

32. A. Banerjee, V. Kumar V. Chandola, "Anomaly Detection: A Survey," ACM Computing Surveys, vol. 41, no. 3, p. 58, July 2009.

33. Network behavior anomaly detection [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Network_behavior_anomaly_detection.

34. Труш В. Є. Методи виявлення аномалій трафіку за допомогою нейронних мереж. 2019. URL: https://openarchive.nure.ua/bitstream/document/10928/1/2019_M_BIT_Trush.pdf

35. Intrusion Detection Evaluation Dataset (CIC-IDS2017) [Електронний ресурс] – Режим доступу до ресурсу: https://www.unb.ca/cic/datasets/ids-2017.html.

36. J48 Classification (C4.5 Algorithm) in a Nutshell [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/@nilimakhanna1/j48-classification-c4-5-algorithm-in-a-nutshell-24c50d20658e.

ДОДАТОК А

Результат першого етапу класифікації:

=== Classifier model (full training set) ===

J48 pruned tree

```
 Init_Win_bytes_backward <= 0.441666
|   Total Length of Fwd Packets <= 0.030857
|   |   Fwd Packet Length Std <= 0.032719: BENIGN (149109.0/177.0)
|   |   Fwd Packet Length Std > 0.032719
|   |   |   Fwd Packet Length Max <= 0.016438
|   |   |   |   Fwd Packet Length Max <= 0.016353: BENIGN (101.0/4.0)
|   |   |   |   Fwd Packet Length Max > 0.016353
|   |   |   |   |   Fwd Packet Length Mean <= 0.023241: Web Attack – Brute Force (35.0)
|   |   |   |   |   Fwd Packet Length Mean > 0.023241: BENIGN (2.0)
|   |   |   Fwd Packet Length Max > 0.016438
|   |   |   |   Max Packet Length <= 0.025771
|   |   |   |   |   Fwd Packet Length Max <= 0.025728: BENIGN (1333.0/2.0)
|   |   |   |   |   Fwd Packet Length Max > 0.025728: Web Attack – Brute Force (36.0/1.0)
|   |   |   |   Max Packet Length > 0.025771
|   |   |   |   |   Max Packet Length <= 0.226755
|   |   |   |   |   |   Bwd Packet Length Std <= 0.267906: BENIGN (9915.0/4.0)
|   |   |   |   |   |   Bwd Packet Length Std > 0.267906
|   |   |   |   |   |   |   Max Packet Length <= 0.087158
|   |   |   |   |   |   |   |   Fwd Packet Length Max <= 0.019392: BENIGN (29.0)
|   |   |   |   |   |   |   |   Fwd Packet Length Max > 0.019392: Web Attack – Sql Injection
(8.0/1.0)
|   |   |   |   |   |   |   Max Packet Length > 0.087158: BENIGN (956.0/5.0)
|   |   |   |   |   Max Packet Length > 0.226755
|   |   |   |   |   |   Fwd Packet Length Max <= 0.039341
|   |   |   |   |   |   |   Fwd Packet Length Max <= 0.024058: Web Attack – XSS (3.0)
|   |   |   |   |   |   |   Fwd Packet Length Max > 0.024058: Web Attack – Brute Force (3.0)
|   |   |   |   |   |   Fwd Packet Length Max > 0.039341: BENIGN (125.0)
|   Total Length of Fwd Packets > 0.030857
|   |   Packet Length Variance <= 0.018455
|   |   |   Destination IP = 192.168.10.3: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 8.0.6.4: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 65.55.44.109: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 129.6.15.29: Web Attack – Brute Force (0.0)
```

```
|   |   |   Destination IP = 192.168.10.14: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 224.0.0.251: BENIGN (2.0)
|   |   |   Destination IP = 192.168.10.1: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 192.168.10.255: Web Attack – Brute Force (0.0)
…
|   |   |   Destination IP = 23.194.109.196: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 31.13.69.245: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 194.246.78.103: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 72.247.140.159: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 52.51.29.111: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 74.217.63.33: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 52.17.172.46: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 52.5.189.119: Web Attack – Brute Force (0.0)
|   |    Packet Length Variance > 0.018455
|   |   |   Max Packet Length <= 0.090283
|   |   |   |   Total Length of Bwd Packets <= 0.000151: BENIGN (3.0)
|   |   |   |   Total Length of Bwd Packets > 0.000151: Web Attack – XSS (15.0)
|   |   |   Max Packet Length > 0.090283: BENIGN (113.0/1.0)
 Init_Win_bytes_backward > 0.441666
|   Init_Win_bytes_backward <= 0.44191
|   |   Fwd IAT Min <= 0.000066
|   |   |   Timestamp = 6/7/2017 8:59: Web Attack – Brute Force (0.0)
|   |   |   Timestamp = 6/7/2017 9:00: Web Attack – Brute Force (0.0)
|   |   |   Timestamp = 6/7/2017 9:01: Web Attack – Brute Force (0.0)
|   |   |   Timestamp = 6/7/2017 9:02: Web Attack – Brute Force (0.0)
|   |   |   Timestamp = 6/7/2017 9:03: Web Attack – Brute Force (0.0)
…
|   |   |   Timestamp = 6/7/2017 12:56: Web Attack – Brute Force (0.0)
|   |   |   Timestamp = 6/7/2017 12:57: Web Attack – Brute Force (0.0)
|   |   |   Timestamp = 6/7/2017 12:58: Web Attack – Brute Force (0.0)
|   |   |   Timestamp = 6/7/2017 12:59: Web Attack – Brute Force (0.0)
|   |   Fwd IAT Min > 0.000066: BENIGN (506.0)
|   Init_Win_bytes_backward > 0.44191: BENIGN (6177.0)

Number of Leaves  :       5473

Size of the tree :    5494


Time taken to build model: 32.6 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0.47 seconds
```

=== Summary ===

Correctly Classified Instances	170168	99.8838 %
Incorrectly Classified Instances	198	0.1162 %
Kappa statistic	0.9521	
Mean absolute error	0.0012	
Root mean squared error	0.0241	
Relative absolute error	9.1418 %	
Root relative squared error	30.2456 %	
Total Number of Instances	170366	

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
1,000	0,089	0,999	1,000	0,999	0,953	0,961	0,999	BENIGN
0,900	0,000	0,998	0,900	0,947	0,947	0,956	0,903	Web Attack – Brute Force
0,956	0,000	0,998	0,956	0,976	0,977	0,982	0,960	Web Attack – XSS
0,333	0,000	0,875	0,333	0,483	0,540	0,805	0,294	Web Attack – Sql Injection
Weighted Avg. 0,999	0,087	0,999	0,999	0,999	0,953	0,961	0,998	

=== Confusion Matrix ===

a	b	c	d	<-- classified as
168181	3	1	1	a = BENIGN
150	1357	0	0	b = Web Attack – Brute Force
29	0	623	0	c = Web Attack – XSS
14	0	0	7	d = Web Attack – Sql Injection

ДОДАТОК Б

Результат другого етапу класифікації:

=== Classifier model (full training set) ===

J48 pruned tree

```
 Init_Win_bytes_backward <= 0.441666
|   Total Length of Fwd Packets <= 0.030857
|   |   Fwd Packet Length Std <= 0.032719: BENIGN (149109.0/177.0)
|   |   Fwd Packet Length Std > 0.032719
|   |   |   Fwd Packet Length Max <= 0.016438
|   |   |   |   Fwd Packet Length Max <= 0.016353: BENIGN (101.0/4.0)
|   |   |   |   Fwd Packet Length Max > 0.016353
|   |   |   |   |   Fwd Packet Length Mean <= 0.023241: Web Attack – Brute Force (35.0)
|   |   |   |   |   Fwd Packet Length Mean > 0.023241: BENIGN (2.0)
|   |   |   Fwd Packet Length Max > 0.016438
|   |   |   |   Max Packet Length <= 0.025771
|   |   |   |   |   Fwd Packet Length Max <= 0.025728: BENIGN (1333.0/2.0)
|   |   |   |   |   Fwd Packet Length Max > 0.025728: Web Attack – Brute Force (36.0/1.0)
|   |   |   |   Max Packet Length > 0.025771
|   |   |   |   |   Max Packet Length <= 0.226755
|   |   |   |   |   |   Bwd Packet Length Std <= 0.267906: BENIGN (9915.0/4.0)
|   |   |   |   |   |   Bwd Packet Length Std > 0.267906
|   |   |   |   |   |   |   Max Packet Length <= 0.087158
|   |   |   |   |   |   |   |   Fwd Packet Length Max <= 0.019392: BENIGN (29.0)
|   |   |   |   |   |   |   |   Fwd Packet Length Max > 0.019392: Web Attack – Sql Injection
(8.0/1.0)
|   |   |   |   |   |   |   Max Packet Length > 0.087158: BENIGN (956.0/5.0)
|   |   |   |   |   Max Packet Length > 0.226755
|   |   |   |   |   |   Fwd Packet Length Max <= 0.039341
|   |   |   |   |   |   |   Fwd Packet Length Max <= 0.024058: Web Attack – XSS (3.0)
|   |   |   |   |   |   |   Fwd Packet Length Max > 0.024058: Web Attack – Brute Force (3.0)
|   |   |   |   |   |   Fwd Packet Length Max > 0.039341: BENIGN (125.0)
|   Total Length of Fwd Packets > 0.030857
|   |   Packet Length Variance <= 0.018455
|   |   |   Destination IP = 192.168.10.3: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 8.0.6.4: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 65.55.44.109: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 129.6.15.29: Web Attack – Brute Force (0.0)
```

```
|   |   |   Destination IP = 192.168.10.14: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 224.0.0.251: BENIGN (2.0)
|   |   |   Destination IP = 192.168.10.1: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 192.168.10.255: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 192.168.10.19: Web Attack – Brute Force (0.0)
…
|   |   |   Destination IP = 31.13.69.245: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 194.246.78.103: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 72.247.140.159: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 52.51.29.111: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 74.217.63.33: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 52.17.172.46: Web Attack – Brute Force (0.0)
|   |   |   Destination IP = 52.5.189.119: Web Attack – Brute Force (0.0)
|   |    Packet Length Variance > 0.018455
|   |   |   Max Packet Length <= 0.090283
|   |   |   |   Total Length of Bwd Packets <= 0.000151: BENIGN (3.0)
|   |   |   |   Total Length of Bwd Packets > 0.000151: Web Attack – XSS (15.0)
|   |   |   Max Packet Length > 0.090283: BENIGN (113.0/1.0)
 Init_Win_bytes_backward > 0.441666
|   Init_Win_bytes_backward <= 0.44191
|   |   Fwd IAT Min <= 0.000066
|   |   |   Destination Port = 0: Web Attack – Brute Force (0.0)
|   |   |   Destination Port = 21: Web Attack – Brute Force (0.0)
|   |   |   Destination Port = 22: Web Attack – Brute Force (0.0)
|   |   |   Destination Port = 42: Web Attack – Brute Force (0.0)
|   |   |   Destination Port = 53: Web Attack – Brute Force (0.0)
|   |   |   Destination Port = 80: Web Attack – Brute Force (1819.0/606.0)
|   |   |   Destination Port = 88: Web Attack – Brute Force (0.0)
|   |   |   Destination Port = 123: Web Attack – Brute Force (0.0)
…
|   |   |   Destination Port = 64295: Web Attack – Brute Force (0.0)
|   |   |   Destination Port = 64335: Web Attack – Brute Force (0.0)
|   |   |   Destination Port = 64347: Web Attack – Brute Force (0.0)
|   |   |   Destination Port = 64375: Web Attack – Brute Force (0.0)
|   |   |   Destination Port = 64771: Web Attack – Brute Force (0.0)
|   |   |   Destination Port = 64854: Web Attack – Brute Force (0.0)
|   |   |   Destination Port = 64916: Web Attack – Brute Force (0.0)
|   |   |   Destination Port = 64934: Web Attack – Brute Force (0.0)
|   |   |   Destination Port = 65529: Web Attack – Brute Force (0.0)
|   |   Fwd IAT Min > 0.000066: BENIGN (506.0)
|   Init_Win_bytes_backward > 0.44191: BENIGN (6177.0)

Number of Leaves  :       23019

Size of the tree :    23040
```

Time taken to build model: 30.29 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0.18 seconds

=== Summary ===

Correctly Classified Instances	169565	99.5298 %
Incorrectly Classified Instances	801	0.4702 %
Kappa statistic	0.8059	
Mean absolute error	0.0035	
Root mean squared error	0.0419	
Relative absolute error	27.735 %	
Root relative squared error	52.6818 %	
Total Number of Instances	170366	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	1,000	0,089	0,999	1,000	0,999	0,953	0,961	0,999	BENIGN
	0,900	0,004	0,691	0,900	0,782	0,787	0,954	0,654	Web Attack – Brute Force
	0,028	0,000	1,000	0,028	0,054	0,166	0,978	0,344	Web Attack – XSS
	0,333	0,000	0,875	0,333	0,483	0,540	0,805	0,294	Web Attack – Sql Injection
Weighted Avg.	0,995	0,087	0,996	0,995	0,994	0,949	0,961	0,993	

=== Confusion Matrix ===

a	b	c	d	<-- classified as
168183	2	0	1	a = BENIGN
150	1357	0	0	b = Web Attack – Brute Force
29	5	18	0	c = Web Attack – XSS
14	0	0	7	d = Web Attack – Sql Injection

ДОДАТОК В

Результат третього етапу класифікації:

=== Classifier model (full training set) ===

J48 pruned tree

```
 Init_Win_bytes_backward <= 0.441666
|   Total Length of Fwd Packets <= 0.030857
|   |   Fwd Packet Length Std <= 0.032719: BENIGN (149109.0/177.0)
|   |   Fwd Packet Length Std > 0.032719
|   |   |   Fwd Packet Length Max <= 0.016438
|   |   |   |   Fwd Packet Length Max <= 0.016353: BENIGN (101.0/4.0)
|   |   |   |   Fwd Packet Length Max > 0.016353
|   |   |   |   |   Fwd Packet Length Mean <= 0.023241: Web Attack – Brute Force (35.0)
|   |   |   |   |   Fwd Packet Length Mean > 0.023241: BENIGN (2.0)
|   |   |   Fwd Packet Length Max > 0.016438
|   |   |   |   Max Packet Length <= 0.025771
|   |   |   |   |   Fwd Packet Length Max <= 0.025728: BENIGN (1333.0/2.0)
|   |   |   |   |   Fwd Packet Length Max > 0.025728: Web Attack – Brute Force (36.0/1.0)
|   |   |   |   Max Packet Length > 0.025771
|   |   |   |   |   Max Packet Length <= 0.226755
|   |   |   |   |   |   Bwd Packet Length Std <= 0.267906: BENIGN (9915.0/4.0)
|   |   |   |   |   |   Bwd Packet Length Std > 0.267906
|   |   |   |   |   |   |   Max Packet Length <= 0.087158
|   |   |   |   |   |   |   |   Fwd Packet Length Max <= 0.019392: BENIGN (29.0)
|   |   |   |   |   |   |   |   Fwd Packet Length Max > 0.019392: Web Attack – Sql Injection
(8.0/1.0)
|   |   |   |   |   |   |   Max Packet Length > 0.087158: BENIGN (956.0/5.0)
|   |   |   |   |   Max Packet Length > 0.226755
|   |   |   |   |   |   Fwd Packet Length Max <= 0.039341
|   |   |   |   |   |   |   Fwd Packet Length Max <= 0.024058: Web Attack – XSS (3.0)
|   |   |   |   |   |   |   Fwd Packet Length Max > 0.024058: Web Attack – Brute Force (3.0)
|   |   |   |   |   |   Fwd Packet Length Max > 0.039341: BENIGN (125.0)
|   Total Length of Fwd Packets > 0.030857
|   |   Packet Length Variance <= 0.018455
|   |   |   Protocol = 0: Web Attack – Brute Force (0.0)
|   |   |   Protocol = 6: Web Attack – Brute Force (71.0)
|   |   |   Protocol = 17: BENIGN (2.0)
|   |   Packet Length Variance > 0.018455
```

```
|   |   |   Max Packet Length <= 0.090283
|   |   |   |   Total Length of Bwd Packets <= 0.000151: BENIGN (3.0)
|   |   |   |   Total Length of Bwd Packets > 0.000151: Web Attack – XSS (15.0)
|   |   |   Max Packet Length > 0.090283: BENIGN (113.0/1.0)
 Init_Win_bytes_backward > 0.441666
|   Init_Win_bytes_backward <= 0.44191
|   |   Fwd IAT Min <= 0.000066
|   |   |   Source IP = 192.168.10.50: Web Attack – Brute Force (0.0)
|   |   |   Source IP = 8.6.0.1: Web Attack – Brute Force (0.0)
|   |   |   Source IP = 192.168.10.14: Web Attack – Brute Force (0.0)
|   |   |   Source IP = 65.55.44.109: Web Attack – Brute Force (0.0)
|   |   |   Source IP = 192.168.10.19: BENIGN (4.0)
|   |   |   Source IP = 192.168.10.3: Web Attack – Brute Force (0.0)
|   |   |   Source IP = 192.168.10.12: Web Attack – Brute Force (0.0)
|   |   |   Source IP = 192.168.10.15: Web Attack – Brute Force (0.0)
|   |   |   Source IP = 192.168.10.8: Web Attack – Brute Force (0.0)
|   |   |   Source IP = 192.168.10.9: Web Attack – Brute Force (0.0)
|   |   |   Source IP = 192.168.10.5: Web Attack – Brute Force (0.0)
|   |   |   Source IP = 23.15.4.17: Web Attack – Brute Force (0.0)
…
|   |   |   Source IP = 52.17.172.46: Web Attack – Brute Force (0.0)
|   |   |   Source IP = 54.165.89.238: Web Attack – Brute Force (0.0)
|   |   |   Source IP = 104.23.129.81: Web Attack – Brute Force (0.0)
|   |   |   Source IP = 23.52.154.88: Web Attack – Brute Force (0.0)
|   |   |   Source IP = 72.247.142.76: Web Attack – Brute Force (0.0)
|   |   |   Source IP = 23.194.109.88: Web Attack – Brute Force (0.0)
|   |   |   Source IP = 23.194.109.196: Web Attack – Brute Force (0.0)
|   |   |   Source IP = 31.13.69.245: Web Attack – Brute Force (0.0)
|   |   Fwd IAT Min > 0.000066: BENIGN (506.0)
|   Init_Win_bytes_backward > 0.44191: BENIGN (6177.0)

Number of Leaves  :       4224

Size of the tree :    4245


Time taken to build model: 29.51 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0.24 seconds

=== Summary ===

Correctly Classified Instances      169566               99.5304 %
```

Incorrectly Classified Instances 800 0.4696 %
Kappa statistic 0.8061
Mean absolute error 0.0035
Root mean squared error 0.0419
Relative absolute error 27.699 %
Root relative squared error 52.6475 %
Total Number of Instances 170366

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
1,000	0,089	0,999	1,000	0,999	0,954	0,961	0,999	BENIGN
0,900	0,004	0,691	0,900	0,782	0,787	0,954	0,654	Web Attack – Brute Force
0,028	0,000	1,000	0,028	0,054	0,166	0,978	0,345	Web Attack – XSS
0,333	0,000	0,875	0,333	0,483	0,540	0,805	0,294	Web Attack – Sql Injection
Weighted Avg. 0,995	0,087	0,996	0,995	0,994	0,949	0,961	0,993	

=== Confusion Matrix ===

a	b	c	d	<-- classified as
168184	1	0	1	a = BENIGN
150	1357	0	0	b = Web Attack – Brute Force
29	5	18	0	c = Web Attack – XSS
14	0	0	7	d = Web Attack – Sql Injection

ДОДАТОК Д

Результат четвертого етапу класифікації:

=== Classifier model (full training set) ===

J48 pruned tree

```
 Init_Win_bytes_backward <= 0.441666
|   Total Length of Fwd Packets <= 0.030857
|   |   Fwd Packet Length Std <= 0.032719: BENIGN (149109.0/177.0)
|   |   Fwd Packet Length Std > 0.032719
|   |   |   Fwd Packet Length Max <= 0.016438
|   |   |   |   Fwd Packet Length Max <= 0.016353: BENIGN (101.0/4.0)
|   |   |   |   Fwd Packet Length Max > 0.016353
|   |   |   |   |   Fwd Packet Length Mean <= 0.023241: Web Attack – Brute Force (35.0)
|   |   |   |   |   Fwd Packet Length Mean > 0.023241: BENIGN (2.0)
|   |   |   Fwd Packet Length Max > 0.016438
|   |   |   |   Max Packet Length <= 0.025771
|   |   |   |   |   Fwd Packet Length Max <= 0.025728: BENIGN (1333.0/2.0)
|   |   |   |   |   Fwd Packet Length Max > 0.025728: Web Attack – Brute Force (36.0/1.0)
|   |   |   |   Max Packet Length > 0.025771
|   |   |   |   |   Max Packet Length <= 0.226755
|   |   |   |   |   |   Bwd Packet Length Std <= 0.267906: BENIGN (9915.0/4.0)
|   |   |   |   |   |   Bwd Packet Length Std > 0.267906
|   |   |   |   |   |   |   Max Packet Length <= 0.087158
|   |   |   |   |   |   |   |   Fwd Packet Length Max <= 0.019392: BENIGN (29.0)
|   |   |   |   |   |   |   |   Fwd Packet Length Max > 0.019392: Web Attack – Sql Injection
(8.0/1.0)
|   |   |   |   |   |   |   Max Packet Length > 0.087158: BENIGN (956.0/5.0)
|   |   |   |   |   Max Packet Length > 0.226755
|   |   |   |   |   |   Fwd Packet Length Max <= 0.039341
|   |   |   |   |   |   |   Fwd Packet Length Max <= 0.024058: Web Attack – XSS (3.0)
|   |   |   |   |   |   |   Fwd Packet Length Max > 0.024058: Web Attack – Brute Force (3.0)
|   |   |   |   |   |   Fwd Packet Length Max > 0.039341: BENIGN (125.0)
|   Total Length of Fwd Packets > 0.030857
|   |   Packet Length Variance <= 0.018455
|   |   |   Protocol = 0: Web Attack – Brute Force (0.0)
|   |   |   Protocol = 6: Web Attack – Brute Force (71.0)
|   |   |   Protocol = 17: BENIGN (2.0)
|   |   Packet Length Variance > 0.018455
```

```
|   |   |   Max Packet Length <= 0.090283
|   |   |   |   Total Length of Bwd Packets <= 0.000151: BENIGN (3.0)
|   |   |   |   Total Length of Bwd Packets > 0.000151: Web Attack – XSS (15.0)
|   |   |   Max Packet Length > 0.090283: BENIGN (113.0/1.0)
 Init_Win_bytes_backward > 0.441666
|   Init_Win_bytes_backward <= 0.44191
|   |   Fwd IAT Min <= 0.000066
|   |   |   Flow IAT Std <= 0.034124
|   |   |   |   Total Length of Fwd Packets <= 0.000075: Web Attack – Brute Force (36.0/9.0)
|   |   |   |   Total Length of Fwd Packets > 0.000075: BENIGN (5.0)
|   |   |   Flow IAT Std > 0.034124: Web Attack – Brute Force (1783.0/597.0)
|   |   Fwd IAT Min > 0.000066: BENIGN (506.0)
|   Init_Win_bytes_backward > 0.44191: BENIGN (6177.0)

Number of Leaves  :       24

Size of the tree :    46


Time taken to build model: 30.27 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0.12 seconds

=== Summary ===

Correctly Classified Instances      169565               99.5298 %
Incorrectly Classified Instances       801               0.4702 %
Kappa statistic                          0.8059
Mean absolute error                      0.0035
Root mean squared error                  0.0419
Relative absolute error                 27.7234 %
Root relative squared error             52.6707 %
Total Number of Instances           170366

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
                 1,000    0,089    0,999      1,000    0,999      0,953    0,961     0,999     BENIGN
                 0,900    0,004    0,691      0,900    0,782      0,787    0,954     0,658     Web Attack – Brute Force
                 0,028    0,000    1,000      0,028    0,054      0,166    0,978     0,346     Web Attack – XSS
```

0,333 0,000 0,875 0,333 0,483 0,540 0,805 0,294 Web Attack – Sql Injection
Weighted Avg. 0,995 0,087 0,996 0,995 0,994 0,949 0,961 0,993

=== Confusion Matrix ===

a	b	c	d	<-- classified as
168183	2	0	1	a = BENIGN
150	1357	0	0	b = Web Attack – Brute Force
29	5	18	0	c = Web Attack – XSS
14	0	0	7	d = Web Attack – Sql Injection