Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет навчально-науковий центр заочного навчання
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

АТЕСТАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Методи виявлення атак на комп'ютерну мережу

(тема)

Виконав:
студент II курсу, групи КСМзм-19-1
Жукова І.Ю.
(прізвище, ініціали)

Спеціальність
123 – Комп'ютерна інженерія
(код і повна назва спеціальності)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма
Комп'ютерні системи та мережі
(повна назва освітньої програми)

Керівник: доц. Голубничий Д.Ю.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ЕОМ _____ Коваленко А.А.
(підпис) (прізвище, ініціали)

2020 р.

Харківський національний університет радіоелектроніки

Факультет навчально-науковий центр заочного навчання

Кафедра електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

Спеціальність 123 – Комп'ютерна інженерія
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Комп'ютерні системи та мережі
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
"____" _____ 20___ р.

ЗАВДАННЯ

НА АТЕСТАЦІЙНУ РОБОТУ

студентові Жуковій Ірині Юріївні
(прізвище, ім'я, по батькові)

1. Тема роботи Методи виявлення атак на комп'ютерну мережу

затверджена наказом по університету від " 3 " листопада 2020 р. № 183 Стз

2. Термін подання студентом роботи до екзаменаційної комісії 14 грудня 2020 р.

3. Вхідні дані до роботи 1) метод виявлення атак: нейромережевий; 2) рівень нагляду системи: трафік; 3) набір даних: CICIDS2017

4. Перелік питань, що потрібно опрацювати в роботі
1) огляд та аналіз сучасних методів виявлення атак в комп'ютерних мережах;
2) вибір та обгрунтування методики та засобів дослідження;
3) програмна реалізація моделей СВА;
4) проведення експериментальних досліджень;
5) висновки;

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) 12 слайдів

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз сучасних підходів до виявлення атак	05.11.20-09.11.20	
2	Вибір та обгрунтування методики дослідження	10.11.20-17.11.20	
3	Вибір інструментальних засобів	18.11.20-23.11.20	
4	Проведення експериментів	24.11.20-01.12.20	
5	Оформлення матеріалів атестаційної роботи	02.12.20-07.12.20	
6	Подання атестаційної роботи керівникові та	08.12.20-09.12.20	
	її попередній захист		
7	Подання атестаційної роботи на рецензування	10.12.20-11.12.20	

Дата видачі завдання 04 листопада 2020 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Голубничий Д.Ю.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка атестаційної роботи: 88 с., 22 рис., 9 табл., 1 дод., 20 джерел.

INTERNET, КОМП'ЮТЕРНА МЕРЕЖА, ТРАФІК, АТАКА, НЕЙРОННА МЕРЕЖА, АНОМАЛІЯ, PYTHON.

Метою атестаційної роботи є дослідження методів виявлення атак в комп'ютерній мережі та розробка системи виявлення атак на мережеві ресурси з використанням нейромережевих технологій.

У ході виконання атестаційної роботи розглянуто та проаналізовано типи й характеристики мережевих атак, методів та систем їх виявлення. Розроблено систему виявлення атак у мережевому трафіку із використанням технологій навчання нейронних мереж та наступним використанням їх у детектуванні зловмисних втручань. Спроектовано дві моделі нейронних мереж, сховище даних отриманих результатів та додаток з візуалізацією для користувача.

ABSTRACT

Master’s thesis: 88 pages, 22 figures, 9 tables, 1 appendice, 20 sources.

INTERNET, COMPUTER NETWORK, TRAFFIC, ATTACK, NEURAL NETWORK, ANOMALY, PYTHON.

The major goal of this thesis is to study the methods of detecting attacks in a computer network and to develop a system for detecting attacks on network resources using neural network technologies.

In order to execute of attestation work the types and characteristics of network attacks, methods and systems of their detection are considered and analyzed. A system for detecting attacks in network traffic using neural network learning technologies and their subsequent use in detecting malicious interference has been developed. Two models of neural networks, a data warehouse of the obtained results and an application with visualization for the user have been designed.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

БД – база даних

МПС – мікропроцесорна система

ОС – операційна система

ПЗ – програмне забезпечення

СВА – система виявлення атак

ШНМ – штучні нейронні мережі

API – інтерфейс прикладного програмування (англ., Application Programming Interface)

CSV – значення, розділені комами (англ., Comma-Separated Values)

DoS – відмова в обслуговуванні (англ., Denial of Service)

IP – протокол Інтернету (англ., Internet Protocol)

Json – позначення об’єкта JavaScript (англ., JavaScript Object Notation)

ВСТУП

Із-за високої складності і вартості розробки захищених систем by design, на початку XXI століття з'явилося і почало активно розвиватися напрям інформаційної безпеки, пов'язаний з виявленням (і, можливо, подальшим реагуванням) порушень безпеки інформаційних систем, як ефективне тимчасове рішення, що дозволяє закривати "проломи" в безпеці систем до їх виправлення. Даний напрям отримав назву "Виявлення атак" (intrusion detection). За минулі роки в рамках академічних розробок були створені сотні систем виявлення атак для різних платформ: від систем класу mainframe до сучасних операційних систем загального призначення, систем управління базами даних і поширених застосувань [6, 7, 12, 14].

Створення ефективних систем захисту інформаційних систем зіштовхується також з браком обчислювальної потужності. З самого початку розвитку комп'ютерів і комп'ютерних мереж спостерігаються дві тенденції відповідно закону Мура і закону Гілдера. Закон Мура говорить про щорічне подвоєння продуктивності обчислювачів, доступних за одну і ту ж вартість, а закон Гілдера – про потроєння пропускної спроможності каналів зв'язку за той же період. Таким чином, зростання обчислювальної потужності вузлів мережі відстає від зростання об'ємів переданої по мережі інформації, що з кожним роком посилює вимоги до обчислювальної складності алгоритмів систем захисту інформації.

Методи виявлення атак в сучасних системах виявлення атак (СВА) недостатньо опрацьовані в частині формальної моделі атаки, і, отже, для них достатньо складно строго оцінити такі властивості як обчислювальна складність, коректність, тощо [5, 14]. Прийнято розділяти методи виявлення атак на методи виявлення аномалій і методи виявлення зловживань [5]. До другого типу методів відносяться більшість сучасних комерційних систем – вони використовують сигнатурні (експертні) методи виявлення [5, 14].

Актуальність дослідження в атестаційній роботі підтверджується тим, що існує множина академічних розробок в області виявлення аномалій, але в промислових системах вони використовуються рідко і з великою обережністю, оскільки такі системи породжують велику кількість помилкових спрацьовувань. Для експертних же систем основною проблемою є низька, близька до нуля, ефективність виявлення невідомих атак (адаптивність) [6, 7, 17]. Низька адаптивність до цих пір залишається проблемою, хоча такі достоїнства як низька обчислювальна складність і мала вартість розгортання визначають домінування таких систем в даній області.

Метою атестаційної роботи є дослідження методів виявлення атак в комп’ютерній мережі та розробка системи виявлення атак на мережеві ресурси із застосуванням нейромережевих технологій.

Об'єктом дослідження є процеси протидії мережевим атакам.

Предметом дослідження виступає мережевий трафік.

Завдання атестаційної роботи:

- аналіз існуючих атак, методів та систем їх виявлення;
- дослідження сучасних технологій і програмного забезпечення що використовують для розробки компонентів системи;
- розробка збирача даних;
- розробка аналізуючого модуля;
- розробка веб-додатку для перегляду інформації про знайдену підозрілу активність;
- розробка двох моделей нейронних мереж для виявлення різних атак.

1 АНАЛІЗ ІСНУЮЧИХ ПІДХОДІВ ДЛЯ ВИЯВЛЕННЯ АТАК

1.1 Класифікація атак

На сьогоднішній день відома велика кількість різних типів класифікаційних ознак. Всі можливі мережеві атаки можна розділити на наступні типи:

- віддалене проникнення – це тип атак, які дозволяють реалізувати дистанційне керування комп'ютером через мережу (наприклад, атаки з використанням програм NetBus або BackOrifice);

- локальне проникнення (англ. Local penetration) – це тип атак, які призводять до отримання несанкціонованого доступу до вузла, на який вони спрямовані (прикладом такої атаки є атака з використанням програми GetAdmin);

- віддалена відмова в обслуговуванні – тип атак, які дозволяють порушити функціонування системи в рамках глобальної мережі (приклад такої атаки – Teardrop або trinOO);

- локальна відмова в обслуговуванні (англ. Local denial of service) – тип атак, що дозволяють порушити функціонування системи в рамках локальної мережі або хоста, до якого хакер має доступ (наприклад, впровадження і запуск ворожої програми, яка навантажує центральний процесор нескінченим циклом, що призводить до неможливості обробки запитів);

- атаки з використанням мережевих сканерів – програм, які аналізують топологію мережі і виявляють сервіси, доступні для атаки (наприклад, атака з використанням утиліти nmap);

- атаки з використанням сканерів вразливостей – програм, які здійснюють пошук вразливостей на вузлах мережі, які в подальшому можуть бути застосовані для реалізації мережевих атак (прикладами мережевих

сканерів можуть служити системи SATAN і Shadow Security Scanner);

- атаки з використанням зломщиків паролів – програм, що підбирають паролі користувачів (наприклад, програма LOphtCrack для ОС Windows або програма Crack для ОС Unix);

- атаки з використанням аналізаторів протоколів (sniffers) – програм, прослуховуючих мережевий трафік. За їх допомогою можна автоматизувати пошук в мережевому трафіку такої інформації, як ідентифікатори і паролі користувачів, інформацію про кредитні картки і т.д.

Інша класифікація, запропонована компанією Internet Security Systems, Inc., містить п'ять типів атак:

- збір інформації;
- спроби несанкціонованого доступу;
- відмова в обслуговуванні;
- із підозрілою активністю;
- системні атаки.

У своїх продуктах, призначених для захисту мереж, серверів і робочих станцій (таких як, наприклад, Real Secure, System scanner та ін.) Компанія Internet Security Systems використовує кілька інших класифікаційних ознак можливих мережевих атак.

За ступенем ризику – має велике практичне значення та дозволяє ранжувати небезпеку атак по класах:

- високий (High) – атаки, успішна реалізація яких дозволяє атакуючому негайно отримати доступ до машини, отримати права адміністратора або обійти міжмережеві екрани;

- середній (Medium) – атаки, успішна реалізація яких потенційно може дати атакуючому доступ до машини (наприклад, помилки в сервері NIS, що дозволяють атакуючому отримати файл з гостьовим паролем);

- низький (Low) – атаки, при успішній реалізації яких атакуючий може отримати відомості, що полегшать йому завдання злому даної машини (наприклад, використовуючи сервіс finger, атакуючий може визначити

список користувачів сервера і, використовуючи атаку по словнику, спробувати отримати доступ до машини).

За типом атаки – дозволяє судити про те, чи може атака бути здійснена віддалено або тільки локально:

- локальні (Host Based);
- віддалені (Network Based).

За схильністю до даної атаки програмного забезпечення (англ. Platforms Affected).

Крім того, існує класифікація за характером дій, використовуваних в атаці:

- "чорні ходи" (Backdoors) – атаки, засновані на використанні недокументованих розробниками можливостей програмного забезпечення, які можуть привести до виконання користувачем несанкціонованих операцій на сервері що атакується;
- атаки типу "відмова в обслуговуванні" (Denial of Service, DoS) – атаки, засновані на використанні помилок, що дозволяють атакуючому зробити будь-якій сервер недоступним для легітимних користувачів;
- розподілені атаки типу "відмова в обслуговуванні" (Distributed Denial of Service) – кілька користувачів (або програм) посилають велику кількість фіктивних запитів на сервер, приводячи останній в неробочий стан;
- потенційно незахищена операційна система (OS Sensor);
- неавторизований доступ (Unauthorized Access Attempts).

Недолік даного типу класифікації полягає в неможливості опису мети атаки, а також її наслідки. Наприклад, класифікаційна ознака "за характером дій" містить два класи атак типу "відмова в обслуговуванні", але в той же час не містить класів, що описують атаки, спрямовані на перехоплення трафіку.

Інший підхід був застосований в класифікації, використаній в програмному продукті Nessus, призначеному для аналізу безпеки серверів. Тут використовується класифікація за характером уразливості, використовуваної для реалізації атаки:

- "чорні ходи" (Backdoors);
- помилки в CGI-скриптах;
- атаки типу "відмова в обслуговуванні";
- помилки в FTP-серверах;
- наявність на комп'ютері сервісу Finger або помилки в програмах, що реалізують цей сервіс;
- помилки в реалізації міжмережевих екранів;
- помилки, що дозволяють користувачеві, що має термінальний вхід на даний сервер, отримати права адміністратора;
- помилки, що дозволяють атакуючому віддалено отримати права адміністратора;
- інші помилки, які не ввійшли в інші категорії;
- помилки в NIS-серверах;
- помилки в RPC-серверах;
- вразливості, що дозволяють атакуючому віддалено отримати будь-який файл з сервера;
- помилки в SMTP-серверах;
- невикористовувані сервіси (Useless services).

Крім того, за типом програмного середовища вони поділяються на вразливості в операційній системі, вразливості в певному сервісі і вразливості в певному програмному забезпеченні. Для визначення вразливості в операційній системі використовується параметр Host/OS, вразливості в конкретних сервісах і в певному програмному забезпеченні класифікуються за групами.

У цій класифікації більш детально, в порівнянні з попередніми, опрацьовані атаки, що використовують вразливості в системному, прикладному і мережевому програмному забезпеченні. Однак дана класифікація не охоплює всіх існуючих мережевих атак. За межами розгляду залишаються такі небезпечні атаки, як атаки типу "відмова в обслуговуванні", перехоплення даних і атаки, спрямовані на мережеве

обладнання. Позитивною рисою даної класифікації є наявність класу "інші помилки, які не ввійшли в категорії", так як формально до будь-якої атаки, в тому числі нової, завдяки цьому класу буде застосовна дана класифікація. Однак, з іншого боку, цей клас не потрібен, оскільки не містить ніякої додаткової інформації.

Виходячи з описаних класифікацій, далеко не всі вони є повними. У деяких випадках під виглядом єдиної класифікації робиться спроба об'єднати кілька класифікацій, проведених за різними характеристиками та параметрами. Поява нових атак призводить до зниження ефективності застосування існуючих класифікацій, тому їх використання без внесення змін не виявляється можливим. Дана ситуація пояснюється величезною кількістю різних мережевих атак і постійною появою нових атак, деякі з яких не підкоряються критеріям існуючих класифікацій.

Таким чином, всі типи атак, розглянуті в рамках наведених класифікацій, можна звести до двох категорій: аномалії і зловживання.

При виявленні аномалій ідентифікується діяльність, яка відрізняється від шаблонів, встановлених для користувачів або груп користувачів. Виявлення аномалій, як правило, включає створення бази даних, яка містить профілі контрольованої діяльності.

Виявлення зловживань включає порівняння діяльності користувача з відомими шаблонами поведінки хакера, який намагається проникнути в систему.

У той час як при виявленні аномалій, як правило, використовується моніторинг порогової величини для визначення досягнення певної встановленої межі, методи виявлення зловживань часто використовують підхід на основі правил.

Механізм виявлення ідентифікує потенційні атаки, в разі, якщо дії користувача збігаються з встановленими правилами. Наявність вичерпних баз даних таких правил є найбільш важливим аспектом для експертних систем виявлення атак.

1.2 Сучасні підходи до виявлення атак

Більшість сучасних підходів до процесу виявлення атак використовують деяку форму аналізу на основі правил. Даний аналіз спирається на набір заздалегідь визначених правил, які надаються адміністратором, автоматично створюються системою або використовуються обидва варіанти.

Методи виявлення аномалій спрямовані на виявлення невідомих атак і вторгнень. Для СВА на основі сукупності параметрів оцінки формується «уявлення» нормального функціонування. В сучасних СВА виділяють кілька способів побудови «уявлення»:

- накопичення найбільш характерної статистичної інформації для кожного параметра оцінки;
- навчання нейронних мереж значеннями параметрів оцінки;
- уявлення події.

Легко помітити, що у виявленні дуже значну роль відіграє безліч параметрів оцінки. Тому у 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 «аномальність» в захищається системі.

1.2.1 Метод сигнатурного аналізу

Сигнатурний аналіз – зіставлення реальної сигнатури (в конкретній точці пристрою), відображеної на дисплеї сигнатурного аналізатора, з еталонною сигнатурою цієї точки (зазначеної на схемі або в таблиці).

Подібно логічним аналізаторам, сигнатурні аналізатори реєструють потоки цифрової двійковій інформації. Але на відміну від логічних аналізаторів, які тільки наочно представляють інформаційні канали,

сигнатурні аналізатори обробляють «довгі» потоки двійковій інформації, «стискаючи» їх з високою достовірністю.

Отримані «короткі» (зазвичай складаються з чотирьох шістнадцяткових чисел) формати – кодові еталони – називаються сигнатурами. Зміст сигнатур носить формальний характер, і наявність певної сигнатури в деякій точці схеми свідчить про конкретний розподіл бітів інформації в потоці даних, що реєструється протягом заданого інтервалу часу. Таким чином, сигнатурний аналіз засновано на перетворенні довгих послідовностей двійкових сигналів в двійкове число, зване сігнатурой.

Вимірювані виконавчі послідовності збуджуються в контрольних точках МПС під дією спеціальної тестової програми. Сигнатури контрольних точок вимірюються на свідомо працездатній системі і вказуються на принциповій схемі МПС подібно до того, як на схемах аналогових пристроїв вказуються осцилограми і деякі параметри аналогових сигналів.

Під час пошуку несправності в МПС оператору досить встановити режим виконання тестової програми і потім, простежуючи сигнатури в контрольних точках схеми від виходів до входів, знайти елемент, у якого вхідні сигнатури вірні, а вихідних немає. У цьому елементі або його вихідному ланцюгу і прихована несправність.

Для зручності двійкова сигнатура представляється, як правило, у вигляді декількох шістнадцяткових цифр за допомогою семисегментного індикатора.

1.2.2 Статистичний метод

Основні переваги статистичного підходу – використання вже розробленого і перевіреного апарату математичної статистики і адаптація до поведінки суб'єкта.

Спочатку для всіх суб'єктів аналізованої системи визначаються профілі. Будь-яке відхилення використовуваного профілю від еталонного вважається

несанкціонованою діяльністю. Статистичні методи універсальні, оскільки для проведення аналізу не потрібно знання про можливі атаки і використані ними вразливості. Однак при впровадженні цих методик виникають і проблеми:

- «статистичні» системи не чутливі до порядку проходження подій; в деяких випадках одні й ті ж події в залежності від порядку їх слідування можуть характеризувати аномальну або нормальну діяльність;
- важко поставити граничні (порогові) значення, що відслідковуються системою виявлення атак характеристик, щоб адекватно ідентифікувати аномальну діяльність;
- «статистичні» системи можуть бути з плином часу «навчені» порушниками так, щоб атакуючі дії розглядалися як нормальні.

Слід також враховувати, що статистичні методи не можуть застосовуватися в тих випадках, коли для користувача відсутні шаблони типової поведінки або коли для користувача типові несанкціоновані дії.

1.2.3 Нейромережеві методи

Більшість сучасних методів виявлення атак використовують деяку форму аналізу контрольованого простору на основі правил або статистичного підходу. Як контрольований простір можуть виступати журнали реєстрації або мережевий трафік. Аналіз спирається на набір заздалегідь визначених правил, які створюються адміністратором або самою системою виявлення атак.

Будь-який поділ атаки в часі або серед кількох зловмисників є важким для виявлення за допомогою експертних систем. Через велике розмаїття атак і хакерів навіть спеціальні постійні оновлення БД правил експертної системи ніколи не дадуть гарантії точної ідентифікації всього діапазону атак.

Використання нейронних мереж є одним із способів подолання зазначених проблем експертних систем. На відміну від експертних систем,

які можуть дати користувачеві певну відповідь про відповідність розглянутих характеристик закладених в БД правилами, нейронна мережа проводить аналіз інформації та надає можливість оцінити, чи узгоджуються дані з характеристиками, які вона навчена розпізнавати. У той час як ступінь відповідності нейромережевого уявлення може досягати 100%, достовірність вибору повністю залежить від якості системи в аналізі прикладів поставленого завдання.

Спочатку нейромережу навчають правильної ідентифікації на попередньо підібраній вибірці прикладів предметної області. Реакція нейромережі аналізується і система налаштовується таким чином, щоб досягти задовільних результатів. До початкового періоду навчання, нейромережа набирається досвіду в міру того, як вона проводить аналіз.

Важливою перевагою нейронних мереж при виявленні зловживань є їх здатність «вивчати» характеристики навмисних атак і ідентифікувати елементи, які не схожі на ті, що спостерігалися в мережі раніше.

1.2.4 Штучні імунні системи

Штучні імунні системи будуються за аналогією з імунною системою живого організму.

Штучні імунні сістеми представляють собою складну адаптивну структуру, що ефективно використовує різноманітні механізми навчання, пам'яті і асоціативного пошуку для вирішення завдань розпізнаванія і класифікації.

Ідея створення штучних імунних систем з'явилася в результаті вивчення процесів біологічного імунітету, який захищає організм від хвороботворних бактерій і вірусів, виявляючи і знищуючи їх.

При побудові штучної імунної системи для виявлення і класифікації мережевих атак на комп'ютерні системи використовуються базові принципи і механізми біологічної імунної сістеми.

Штучна імунна система моделює основні процеси біологічної імунної системи, а також їх взаємодію. Відмінність полягає в способі передставлення інформації та структурі імунного детектора.

Імунні детектори генеруються за випадковим алгоритмом, що дає можливість створення великої кількості різноманітних за своєю структурою детекторів, які здатні реагувати на будь-яку аномалію. Далі детектори проходять стадію навчання, на якій вони набувають здатність коректно реагувати на чужорідні об'єкти або явища. Для того щоб детектори не генерували помилкові спрацьовування, вони ретельно відбираються. Ті з них, які не навчилися коректно класифікувати об'єкти, видаляються. Відібрані детектори допускаються до виконання функцій класифікації об'єктів.

Кожному детектору виділяється деяка лімітована кількість часу (час життя), на протязі якого він може існувати.

Якщо протягом цього часу детектор не може виявити аномалій, то він видаляется, а на його місце приходить новий, структурований відмінний детектор. Якщо детектор виявив аномалію, відбувається так звана стадія активації. На цій стадії відбувається інформування про виявлену аномалію та її знищення. Детектор, який знайшов аномалію, трансформується в детектор імунної пам'яті. Детектори імунної пам'яті характеризуються великим часом життя і рівнем довіри.

1.2.5 Графові моделі атак

Для аналізу стійкості роботи мережі часто потрібно досліджувати її вразливість, тобто знайти "вузькі" місця, послідовне використання яких може призвести до порушення функціонування мережі. Для вирішення таких завдань використовують різні підходи, одним з яких є побудова для вихідної мережі так званого графа атак.

Метод полягає в побудові графа, який містить всі відомі сценарії атак, на основі певного характеру коректності системи. Для побудови графа

сценаріїв атак створюється формальний опис системи що потребує захисту і виявляється властивість коректності системи. На основі властивості коректності можлива поведінку системи ділиться на допустиме і неприпустиме. Неприпустима поведінка системи трактується як можлива атака.

В даному методі будується повний набір варіантів неприпустимої поведінки для конкретної системи, що дає на виході опис можливих шляхів атаки. Даний метод може бути використаний для пошуку вразливостей при проектуванні систем, однак для завдання виявлення атак непридатний через високу обчислювальну складність.

1.2.6 Експертні системи

Експертні системи складаються з набору правил, які охоплюють знання людини-експерта. Використання експертних систем являє собою поширений метод виявлення атак, при якому інформація про атаки формулюється у вигляді правил. Ці правила можуть бути записані, наприклад, у вигляді послідовності дій або у вигляді сигнатури. При виконанні будь-якого з цих правил приймається рішення про наявність несанкціонованої діяльності. Важливою перевагою такого підходу є практично повна відсутність помилкових тривог.

БД експертної системи повинна містити сценарії більшості відомих на сьогоднішній день атак. Для того щоб залишатися постійно актуальними, експертні системи вимагають постійного оновлення БД. Хоча експертні системи пропонують гарну можливість для перегляду даних в журналах реєстрації, необхідні оновлення можуть або ігноруватися, або виконуватися адміністратором вручну. Як мінімум, це призводить до формування експертної системі з ослабленими можливостями. У гіршому випадку відсутність належного супроводу знижує ступінь захищеності всієї мережі, вводячи її користувачів в оману щодо дійсного рівня захищеності.

Основним недоліком є неможливість відображення невідомих атак. При цьому навіть невелика зміна вже відомої атаки може стати серйозною перешкодою для функціонування системи виявлення атак.

1.2.7 Кластерний аналіз

Термін кластерний аналіз, що вперше ввів Тріона (Tryon) в 1939 році, включає в себе безліч різних алгоритмів (більше 100). Кластерний аналіз являє собою один з методів інтелектуального аналізу даних, що дозволяє групувати об'єкти в кластери на підставі обраної міри схожості між об'єктами.

Загальна ідея алгоритмів заснована на визначенні ступеня подібності об'єктів на підставі відстані між об'єктами. Відстань між об'єктами обчислюється на основі чисельних параметрів об'єктів. Ступінь схожості об'єктів обернено пропорційна відстані між об'єктами.

На відміну від завдань класифікації, кластерний аналіз не вимагає апріорних припущень про набір даних, що не накладає обмеженнь на подання досліджуваних об'єктів, дозволяє аналізувати показники різних типів даних.

У контексті вирішення завдання виявлення атак на інформаційні системи, кластерний аналіз може застосовуватися як один з методів, що реалізує виявлення аномалій. В якості вхідних даних методи кластерного аналізу використовують вектора, що містять характеристики системи в процесі роботи, наприклад, такі як параметри мережевих з'єднань системи, ступінь завантаженості ресурсів системи або активність процесів, що виконуються на вузлі системи. На підставі аналізу векторів навчальної вибірки проводиться побудова кластерів, що описують нормальну поведінку системи, після чого побудована сукупність кластерів використовується для виявлення аномалій.

1.2.8 Порівняння характеристик розглянутих методів виявлення атак

Ефективність системи виявлення атак багато в чому залежить від застосовуваних методів аналізу отриманої інформації. У перших системах виявлення атак використовувалися статистичні методи виявлення атак. Сьогодні до статистичного аналізу додався ряд нових методик, починаючи з експертних систем і нечіткої логіки і закінчуючи використанням нейронних мереж. (таблиця 1.1).

Таблиця 1.1 – Основні характеристики методів виявлення атак

Методи виявлення атак	Рівень нагляду за системою	Верифікуємість	Адаптивність	Стійкість	Обчислювана складність
Аналіз сигнатур	Host; Network; Application	+	–	глобальна	O(logn)
Статистичний контроль	Host; Network	–	+	локальна	O(n)
Нейромережеві методи	Host; Network; Application	+	+	локальна	O(n)
Штучні імунні системи	Host; Network	–	+	локальна	O(n)
Графові моделі атак	Host; Network; Application	+	+	локальна	NP
Методи кластерного аналізу	Host; Network; Application	–	+	локальна	O(n)
Експертні системи	Host; Network	+	+	глобальна	NP

У зв'язку тим, що число нових, раніше невідомих атак росте з кожним роком, найбільш переважними для використання є адаптивні методи виявлення вторгнень.

З результатів аналізу, представлених в таблиці 1.1, виходить, що до адаптивних методів виявлення вторгнень серед розглянутих в роботі відносяться: статистичний аналіз, графи сценаріїв атак, експертні системи, нейронні мережі, імунні мережі та кластерний аналіз.

Методи, засновані на побудові графа сценаріїв атак і експертних системах, практично не використовуються в існуючих системах виявлення та запобігання вторгнень в силу високої обчислювальної складності їх реалізації.

Методи виявлення вторгнень, засновані на використанні імунних мереж і поведінкової біометрії, також не використовуються в готових рішеннях через складність реалізації.

З решти адаптивних методів виявлення вторгнень метод статистичного аналізу і аналогічний йому по суті метод кластерного аналізу мають високу ймовірність помилкового спрацьовування.

Також неможливість зміни характеристик об'єкта в ході функціонування методів може привести як до помилкових спрацьовувань, так і до пропущених атак.

В результаті проведеного аналізу було встановлено, що для забезпечення захисту систем розподіленої обробки даних найбільш переважними є методи, що функціонують на принципах нейромережевого аналізу.

Основними перевагами даних методів є: адаптивність, невисока обчислювальна складність, а також при використанні належної навчальної вибірки можливість забезпечувати надійне функціонування системи виявлення та запобігання вторгнень, виявляючи як відомі, так і невідомі атаки різних класів.

1.3 Огляд та аналіз існуючих систем виявлення мережевих атак

1.3.1 Система виявлення атак Bro

Система Bro є мережевою системою виявлення атак. Вона є набором модулів декомпозиції даних різних мережевих протоколів (від мережевого до прикладного рівня) і набором сигнатур над подіями відповідних протоколів. Сигнатури Bro фактично є регулярними виразами в алфавітах протоколів.

Дана система виявляє атаки наступних класів (L,R,A,D):

- L = $\{li \cup le\}$ (внутрішні і зовнішні атаки);
- R = $\{rn\} \cap \{ru \cup rs\}$ (атаки на мережеві призначені для користувача ресурси і системні ресурси);
- A = $\{as \cup au \cup ar \cup ad\}$ (збір інформації про систему, спроби отримання прав користувача, спроби отримання прав адміністратора і порушення працездатності ресурсу);
- D = $\{dn \cup dd\}$ (нерозподілені і розподілені).

1.3.2 Система виявлення атак OSSEC

Система OSSEC, єдина з розглянутих в даній роботі, котра є спочатку орієнтованою на виявлення атак рівня системи (вузлових). Вона найбільш нова з розглянутих систем; її остання версія призначена, зокрема, для аналізу журналів реєстрації UNIX, типових застосувань (ftpd, apache, mail, etc), а також журналів міжмережних екранів і мережних СВА. OSSEC включає набір аналізаторів для різних джерел даних, контроль цілісності файлової системи, сигнатури відомих троянських закладок (rootkits) та ін.

Виявляються атаки наступних класів:

- L = $\{li\}$ (атакуючі об'єкти знаходяться усередині системи);
- R = $\{rl\} \cap \{ru \cap rs\}$ (вузлові призначені для користувача і системні ресурси);

- A = $\{au \cup ar \cup ai\}$ (спроби отримання прав користувача, спроби отримання прав адміністратора, порушення цілісності ресурсу);
- D = $\{dn \cup dd\}$ (нерозподілені і розподілені).

1.3.3 Система виявлення атак STAT

Система STAT є експериментальною університетською розробкою, і найбільш "старою" з даних систем – перші публікації по STAT датуються 1992 роком. Система включає набір компонентів виявлення атак різних рівнів – мережний (NETSTAT), вузловий (USTAT, WINSTAT), додатків (WEBSTAT), тобто є класичною гібридною системою.

СВА виявляє атаки наступних класів: (L,R,A,D):

- L = $\{li \cup le\}$ (внутрішні і зовнішні атаки);
- R = $\{rl \cup rn\} \cap \{ru \cup rs\}$ (атаки на вузлові або мережеві призначені для користувача ресурси і системні ресурси);
- A = $\{as \cup au \cup ar \cup ad\}$ (збір інформації про систему, спроби отримання прав користувача, спроби отримання прав адміністратора і порушення працездатності ресурсу);
- D = $\{dn\}$ (нерозподілені).

1.3.4 Система виявлення атак Prelude

Система Prelude, як і NETSTAT, є гібридною, тобто здатна виявити атаки як на рівні системи, так і на рівні мережі. Дана система спочатку розроблялася як самостійної СВА, але в даний час є високорівневою надбудовою над відкритими СВА і системами контролю цілісності (AIDE, Osiris і тому подібне). Вузлова частина Prelude має достатньо широкий набір описів атак і, як джерело інформації, використовує різні журнали реєстрації:

- журнали реєстрації міжмережного екрану IPFW;
- журнали реєстрації NetFilter ОС Linux;

- журнали реєстрації маршрутизаторів Cisco and Zyxel;
- журнали реєстрації GRSecurity
- журнали реєстрації типових сервісів ОС UNIX та інші.

СВА виявляє атаки наступних класів: (L, R, A, D):

- L = $\{li \cup le\}$ (внутрішні і зовнішні атаки);
- R = $\{rl \cup rn\} \cap \{ru \cup rs \cup rp\}$ (атаки на локальні або мережеві призначені для користувача ресурси, системні ресурси і ресурси захисту);
- A = $\{as \cup au \cup ar \cup ad\}$ (збір інформації про систему, спроби отримання прав користувача, спроби отримання прав адміністратора і порушення працездатності ресурсу);
- D = $\{dn\}$ (нерозподілені).

1.3.5 Система виявлення атак Snort

Система Snort це найбільш популярна на сьогоднішній день некомерційна СВА. Вона активно і динамічно розвивається, оновлення бази відомих атак відбуваються з частотою, порівнянною з комерційними аналогами (зазвичай оновлення Snort випереджають комерційні). Snort є чисто мережною СВА і, окрім основної бази описів атак, має набір підключаємих модулів для виявлення специфічних атак або таких, що реалізовують альтернативні методи виявлення.

Система здатна виявити атаки наступних класів (L, R, A, D):

- L = $\{li \cup le\}$ (внутрішні і зовнішні атаки);
- R = $\{rl \cup rn\} \cap \{ru \cup rs \cup rp\}$ (атаки на локальні або мережеві призначені для користувача ресурси, системні ресурси і ресурси захисту);
- A = $\{as \cup au \cup ar \cup ad\}$ (збір інформації про систему, спроби отримання прав користувача, спроби отримання прав адміністратора і порушення працездатності ресурсу);
- D = $\{dn \cup dd\}$ (нерозподілені і розподілені).

1.3.6 Порівняння характеристик розглянутих систем виявлення мережевих атак

Таким чином, жодна з розглянутих систем не покриває всю множину класів атак. Слід також відзначити, що ці системи використовують неадаптивні методи виявлення атак. звідні результати порівняння розглянутих систем по вибраних критеріях наведені в таблиці 1.2.

Таблиця 1.2 – Результати порівняння відкритих СВА

	Класи атак	Рівень спостереження за системою	Метод виявлення	Адап-тивність	Масшта бова-ність	Реак-ція	Захист
Bro	L={$li \cup le$} R={rn} ∩ {$ru \cup rs$} A={$as \cup au \cup ar \cup ad$} D={$dn \cup dd$}	Системний	Сигнатурний	–	–	–	–
OSSEC	L={li} R={rl} ∩ {$ru \cap rs$} A={$au \cup ar \cup ai$} D={$dn \cup dd$}	Системний	Сигнатурний	+/–	+	–	–
STAT	L={$li \cup le$} R={$rl \cup rn$} ∩ {$ru \cup rs$} A={$as \cup au \cup ar \cup ad$} D={$dn$}	Системний, мережевий	Сигнатурний, аналіз переходів станів	–	+	+	SSL
Prelude	L={$li \cup le$} R={$rl \cup rn$} ∩ {$ru \cup rs \cup rp$} A={$as \cup au \cup ar \cup ad$} D={$dn$}	Системний, мережевий	Сигнатурний	–	+	+	SSL, Libsafe
Snort	L={$li \cup le$} R={$rl \cup rn$} ∩ {$ru \cup rs \cup rp$} A={$as \cup au \cup ar \cup ad$} D={$dn \cup dd$}	Мережевий	Сигнатурний	–	–	+	–

1.4 Висновки за розділом 1

На підставі проведеного порівняльного аналізу можна зробити висновок, що жодна з розглянутих вище відкритих СВА, не відповідає повною мірою критеріям "ідеальної" СВА. Основним недоліком є відсутність адаптивності до невідомих атак і неможливість аналізувати поведінку об'єктів КМ на всіх рівнях одночасно.

2 ДОСЛІДЖЕННЯ ТЕХНОЛОГІЙ І ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ, ВИКОРИСТОВУВАНИХ ДЛЯ РОЗРОБКИ СИСТЕМИ ВИЯВЛЕННЯ АТАК НА МЕРЕЖЕВІ РЕСУРСИ ІЗ ЗАСТОСУВАННЯМ НЕЙРОМЕРЕЖНИХ ТЕХНОЛОГІЙ

Нейронні мережі – це один з напрямків досліджень в області штучного інтелекту, заснований на спробах відтворити нервову систему людини, а саме: здатність нервової системи навчатися та виправляти помилки, що має дозволити змоделювати, хоча і досить грубо, роботу людського мозку.

До завдань, що вирішують нейронні мережі, можна віднести:

- розпізнавання образів і класифікація. На вхід нейронної мережі подається зображення, мережа визначає об'єкт, що знаходиться на цьому зображенні. Під час навчання нейронної мережі на вхід подається вектор значень і ознак способу із зазначенням приналежності до певного класу;

- завдання кластеризації. Завдання кластеризації, відомі так само, як класифікація образів «без вчителя», не мають навчальної вибірки з мітками класів. Їх принцип роботи заснований на виявленні закономірностей і подібностей між образами і розміщенні цих образів в один кластер або категорію;

- апроксимація функцій. Завдання апроксимації, полягає в знаходженні оцінки невідомої функції по її значенням. точність апроксимації залежить від вибору структури нейронної мережі;

- оптимізація. Способи оптимізації з використанням нейронних мереж, мають на увазі знаходження такого рішення, яке задовольняє системі умов і мінімізує цільову функцію. В якості прикладу можна розглянути алгоритм, який реалізує якусь послідовність дій, який можна повністю замінити функціонуванням нейронної мережі, використовуючи правила, за якими він складений.

Нейронна мережа складається з нейронів. Кожен нейрон є елементарною структурною одиницею нейронної мережі (рисунок 2.1). Процес навчання мережі зводиться до зміни вагових коефіцієнтів Wn. NET в даному випадку і є результат обчислень нейрона. Результати обчислення передаються на вихід через функцію активації.

Функція активації нейрона – це функція, яка обчислює вихідний сигнал нейрона. На вхід цієї функції подається сума всіх добутків сигналів і ваг цих сигналів (рисунок 2.1).

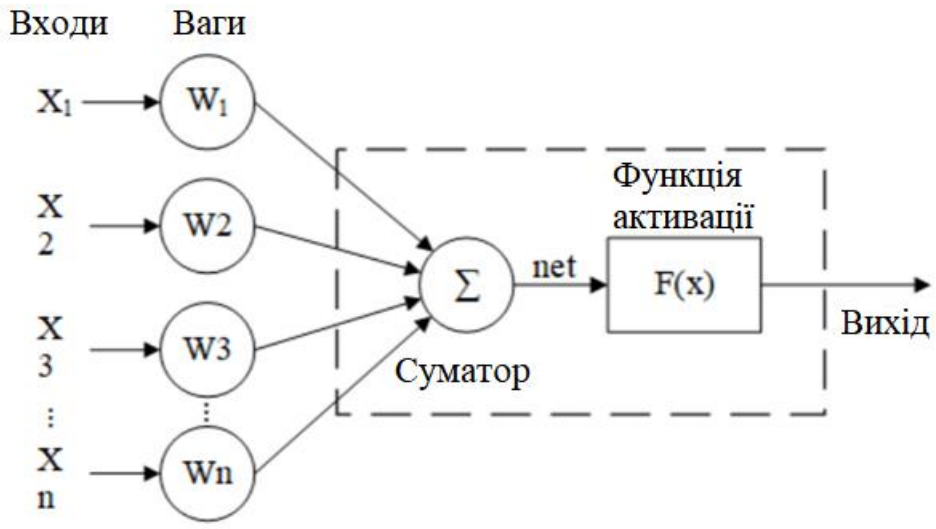

Рисунок 2.1 – Структурна схема нейрона

Можна виділити наступні функції активації.

Одиничний стрибок або жорстка порогова функція. Функція описується наступною формулою :

$$\text{Out} = \begin{cases} 0, \text{NET} < \theta \\ 0, \text{NET} \geq \theta \end{cases} \quad (2.1)$$

Поки середньозважена сума менше певного значення, функція активації повертає нуль, а коли стає більшою – одиницю.

Сигмоїдальна функція або сигмоїд. Формула що описує сигмоїд:

$$\text{Out} = \frac{1}{1+ e^{-NET}} \tag{2.2}$$

Часто застосовується в багатошарових нейронних та інших мережах з безперервними сигналами. Гладкість і безперервність функції – важливі позитивні якості.

Гіперболічний тангенс. Функція описується наступними формулами:

$$\text{Out} = \text{th}(\text{NET}) \tag{2.3}$$

або

$$\text{Out} = \frac{e^{NET} - e^{-NET}}{e^{NET} + e^{-NET}} \tag{2.4}$$

Також часто застосовується в мережах з безперервними сигналами. Її особливість в тому, що вона може повертати негативні значення результату.

2.1 Багатошарові мережі

У багатозв'язних (або багатошарових) мережах нейрони об'єднуються в шари (рисунок 2.2). Шар містить сукупність нейронів з єдиними вхідними сигналами. Число нейронів в кожному шарі може бути будь-яким і ніяк заздалегідь не пов'язане з кількістю нейронів в інших шарах. В загальному випадку мережа складається з Q шарів, пронумерованих зліва направо. Зовнішні вхідні сигнали подаються на входи нейронів першого шару (вхідний шар часто нумерують як нульовий), а виходами мережі є вихідні сигнали останнього шару. Вхід нейронної мережі можна розглядати як вихід «нульового шару» вироджених нейронів, які служать лише в якості

розподільних точок, підсумовування і перетворення сигналів тут не проводиться. Крім вхідного і вихідного шарів в багатошаровій нейронної мережі є один або кілька проміжних (прихованих) шарів. Зв'язки від виходів нейронів деякого шару q до входів нейронів наступного шару (q + 1).

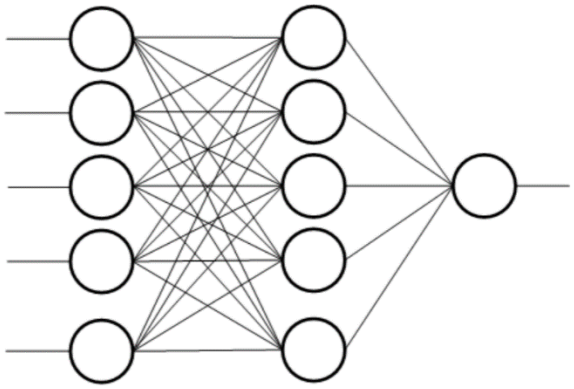

Рисунок 2.2 – Структурна схема багатошарової мережі

2.2 Повнозв’язні мережі

Повнозв'язні мережі являють собою ШНМ, кожен нейрон якої передає свій вихідний сигнал іншим нейронам, в тому числі і самому собі. Всі вхідні сигнали подаються всім нейронам. Вихідними сигналами мережі можуть бути всі або деякі вихідні сигнали нейронів після кількох тактів функціонування мережі (рисунок 2.3).

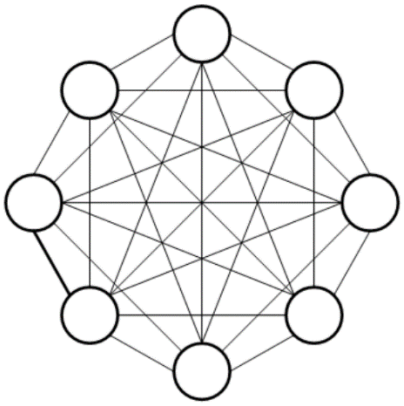

Рисунок 2.3 – Структурна схема повнозв’язної мережі

2.3 Слабозв'язані мережі

Слабозв'язані мережі (нейронні мережі з локальними зв'язками) представляють собою шаруваті мережі з невеликою кількістю зв'язків.

Найважливішим властивістю нейронних мереж є їх здатність навчатися на основі даних навколишнього середовища і в результаті навчання підвищувати свою продуктивність. Підвищення продуктивності відбувається з часом відповідно до певних правил. Навчання нейронної мережі відбувається за допомогою інтерактивного процесу коригування синаптичних ваг і порогів. В ідеальному випадку нейронна мережа отримує знання про навколишнє середовище на кожній ітерації процесу навчання (рисунок 2.4).

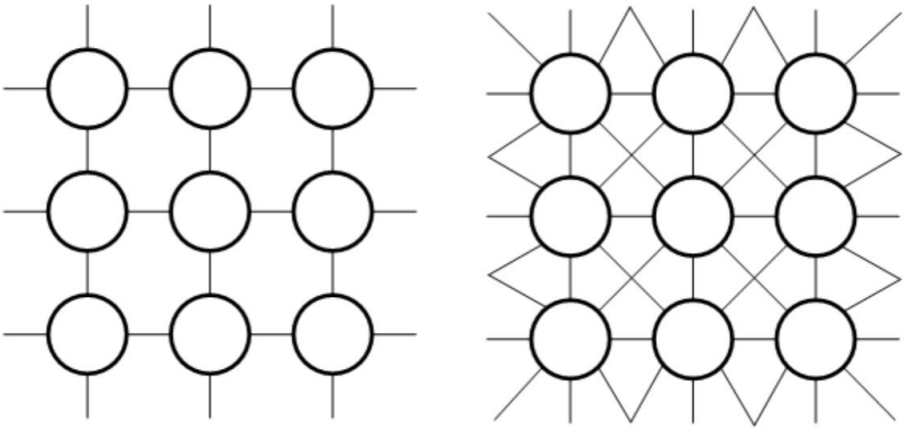

Рисунок 2.4 – Структурна схема слабозв'язаних мереж

2.4 Навчання нейронної мережі

Існують два концептуальних підходи до навчання нейронних мереж: навчання з вчителем і навчання без вчителя.

Навчання нейронної мережі з вчителем передбачає, що для кожного вхідного вектора з навчальної множини існує необхідне значення вихідного вектора. Ці вектора утворюють навчальну пару і ваги в мережі змінюються

до тих пір, поки прийнятний рівень відхилення між векторами не буде досягнутий (рисунок 2.5).

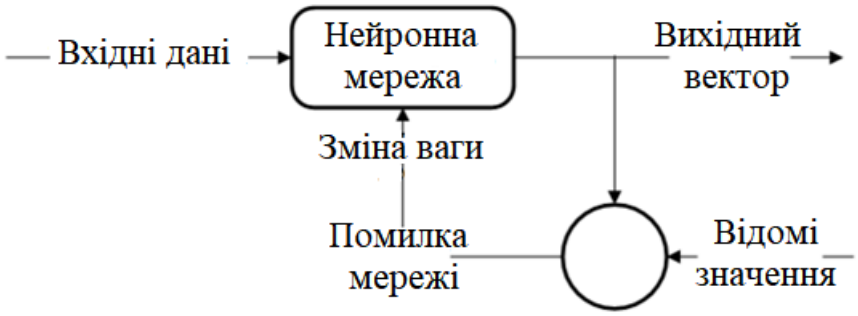

Рисунок 2.5 – Алгоритм навчання нейронної мережі з учителем

Навчання нейронної мережі без вчителя передбачає те, що навчальна безліч складається тільки з вхідних векторів. Алгоритм навчання мережі підлаштовує ваги так, щоб виходили узгоджені вихідні вектори.

При цьому дотримується наступна послідовність подій:

- в нейронну мережу надходять зовнішні сигнали (вхідні параметри);
- вільні параметри мережі змінюються;
- після змін нейронна мережа приймає вхідні сигнали вже іншим чином.

Рисунок 2.6 – Алгоритм навчання нейронної мережі без вчителя

Для навчання моделі нейронної мережі для системи виявлення атак на мережеві ресурси буде використовуватися підхід з вчителем , Навчання буде проводитися на наборі даних CICIDS.

Набір даних CICIDS містить доброякісні і найсучасніші поширені атаки, які нагадують реальні дані (PCAPs). Він також включає в себе результати аналізу мережевого трафіку з використанням CICFlowMeter з маркованими потоками на основі тимчасової мітки, IP – адреси джерела і призначення, портів джерела і призначення, протоколів і атак. Опис параметрів, що містяться в датасетах приведено в таблиці 2.1.

Таблиця 2.1 – Опис параметрів датасета CICIDS

№	Назва параметру	Опис
1	2	3
1	Destination Port	Порт призначення
2	Flow Duration	Тривалість потоку
3	Total Fwd Packets	Всього пакетів в прямому напрямку
4	Total Backward Packets	Всього пакетів в зворотньому напрямку
5	Total Length of Fwd Packets	Загальний розмір пакетів в прямому напрямку
6	Total Length of Bwd Packets	Загальний розмір пакетів в зворотньому напрямку
7	Fwd Packet Length Max	Максимальний розмір пакета в прямому напрямку
8	Fwd Packet Length Min	Мінімальний розмір пакета в прямому напрямку
9	Fwd Packet Length Mean	Середній розмір пакета в прямому напрямку
10	Fwd Packet Length Std	Розмір стандартного відхилення пакета в прямому напрямку
11	Bwd Packet Length Max	Максимальний розмір пакета в зворотньому напрямку

Продовження таблиці 2.1

1	2	3
12	Bwd Packet Length Min	Мінімальний розмір пакета в зворотньому напрямку
13	Bwd Packet Length Mean	Середній розмір пакета в зворотньому напрямку
14	Bwd Packet Length Std	Розмір стандартного відхилення пакета в зворотньому напрямку
15	Flow Bytes/s	Швидкість потоку в байтах, тобто, кількість байтів в секунду
16	Flow Packets/s	Швидкість потоку в пакетах, тобто, кількість пакетів в секунду
17	Flow IAT Mean	Середній час між двома потоками
18	FlowIAT Std	Стандартне відхилення часу двох потоків
19	Flow IAT Max	Максимальний час між двома потоками
20	Flow IAT Min	Мінімальний час між двома потоками
21	Fwd IAT Total	Загальний час між двома пакетами, що відправлені в прямому напрямку
22	Fwd IAT Mean	Середній час між двома пакетами, що відправлені в прямому напрямку
23	Fwd IAT Std	Час стандартного відхилення між двома пакетами, що відправлені в прямому напрямку
24	Fwd IAT Max	Максимальний час між двома пакетами, що відправлені в прямому напрямку
25	Fwd IAT Min	Мінімальний час між двома пакетами, що відправлені в прямому напрямку
26	Bwd IAT Total	Загальний час між двома пакетами, що відправлені в зворотньому напрямку

Продовження таблиці 2.1

1	2	3
27	Bwd IAT Mean	Середній час між двома пакетами, що відправлені в зворотньому напрямку
28	Bwd IAT Std	Час стандартного відхилення між двома пакетами, що відправлені в зворотньому напрямку
29	Bwd IAT Max	Максимальний час між двома пакетами, що відправлені в зворотньому напрямку
30	Bwd IAT Min	Мінімальний час між двома пакетами, що відправлені в зворотньому напрямку
31	Fwd PSH Flags	Кількість разів, коли флаг PSH був встановлен в пакетах, що відправлені в прямому напрямку (0 для UDP)
32	Bwd PSH Flags	Кількість разів, коли флаг PSH був встановлен в пакетах, що відправлені в зворотньому напрямку (0 для UDP)
33	Fwd URG Flags	Кількість разів, коли флаг URG був встановлен в пакетах, що відправлені в прямому напрямку (0 для UDP)
34	Bwd URG Flags	Кількість разів, коли флаг URG був встановлен в пакетах, що відправлені в зворотньому напрямку (0 для UDP)
35	Fwd Header Length	Всього байт використаних для заголовків в прямому напрямку
36	Bwd Header Length	Всього байт використаних для заголовків в зворотньому напрямку
37	Fwd Packets/s	Кількість прямих пакетів в секунду
38	Bwd Packets/s	Кількість зворотніх пакетів в секунду

Продовження таблиці 2.1

1	2	3
39	Min Packet Length	Мінімальна довжина пакету
40	Max Packet Length	Максимальна довжина пакету
41	Packet Length Mean	Середня довжина пакету
42	Packet Length Std	Стандартне відхилення довжини пакету
43	Packet Length Variance	Мінімальний час прибуття пакету
44	FIN Flag Count	Кількість пакетів з FIN
45	SYN Flag Count	Кількість пакетів з SYN
46	RST Flag Count	Кількість пакетів з RST
47	PSH Flag Count	Кількість пакетів з PSH
48	ACK Flag Count	Кількість пакетів з ACK
49	URG Flag Count	Кількість пакетів з URG
50	CWE Flag Count	Кількість пакетів з CWE
51	ECE Flag Count	Кількість пакетів з ECE
52	Down/Up Ratio	Коефіцієнт завантаження/вивантаження
53	Average Packet Size	Середній розмір пакету
54	Avg Fwd Segment Size	Середній розмір сегменту в прямому напрямку
55	Avg Bwd Segment Size	Середній розмір сегменту в зворотньому напрямку
56	Fwd Header Length	Розмір заголовку в прямому напрямку
57	Fwd Avg Bytes/Bulk	Середній обсяг байтів в прямому напрямку
58	Fwd Avg Packets/Bulk	Середній обсяг пакетів в прямому напрямку
59	Fwd Avg Bulk Rate	Рівень середнього обсягу пакетів прямого напрямку
60	Bwd Avg Bytes/Bulk	Середній обсяг байтів в зворотньому напрямку

Продовження таблиці 2.1

1	2	3
61	Bwd Avg Packets/Bulk	Середній обсяг пакетів в зворотньому напрямку
62	Bwd Avg Bulk Rate	Рівень середнього обсягу пакетів зворотнього напрямку
63	Subflow Fwd Packets	Середня кількість пакетів подпотоку прямого напрямку
64	Subflow Fwd Bytes	Середня кількість байтів подпотоку прямого напрямку
65	Subflow Bwd Packets	Середня кількість пакетів подпотоку зворотнього напрямку
66	Subflow Bwd Bytes	Середня кількість байтів подпотоку зворотнього напрямку
67	Init_Win_bytes_forward	Кількість байтів, що відправлено в початковому вікні в прямому напрямку
68	Init_Win_bytes_backward	Кількість байтів, що відправлено в початковому вікні в зворотньому напрямку
69	act_data_pkt_fwd	Пакетів з не менш ніж 1 байт корисного завантаження даних TCP в прямому напрямку
70	min_seg_size_forward	Мінімальний розмір сегменту, що спостерігається в прямому напрямку
71	Active Mean	Середній час коли поток був активним перш ніж стати вільним
72	Active Std	Стандартне відхилення часу протягом якого поток був активним до застою
73	Active Max	Максимальний час протягом якого поток був активним до застою

Продовження таблиці 2.1

1	2	3
74	Active Min	Мінімальний час протягом якого поток був активним до застою
75	Idle Mean	Середній час коли поток був у застої пер ніж стати активним
76	Idle Std	Стандартне відхилення часу впродовж якого поток був у застої пер ніж стати активним
77	Idle Max	Максимальний час застою потоку до активації
78	Idle Min	Мінімальний час застою потоку до активації
79	Label	Маркування пакету

2.5 Вибір інструментів реалізації

2.5.1 Програма захоплення трафіку

В якості програми для захоплення трафіку обраний CICFlowMeter. CICFlowMeter – генерує двонаправлені потоки (Biflow), де перший пакет визначає прямий (джерело до місця призначення) і зворотній (місце призначення до джерела) напрямки, звідси виходять статистичні характеристики, такі як тривалість, кількість пакетів, кількість байтів, довжина пакетів, і т.д. також розраховуються окремо в прямому і зворотному напрямку. Вихідні дані додатка є формат файлу CSV з шістьма стовпчиками, поміченими для кожного потоку, а саме FlowID, SourceIP, DestinationIP, SourcePort, DestinationPort і Protocol з більш ніж 80 функціями мережевого трафіку.

2.5.2 Мова програмування

Основною мовою програмування для всіх створюваних додатків в атестаційній роботі був обраний Python.

Python – високорівнева мова програмування загального призначення, орієнтована на підвищення продуктивності коду. Вибір Python в якості мови програмування не випадковий. Ця мова має низький поріг входження, можливість, без особливих проблем, запуститися на операційній системі (ОС) і на цій мові написані бібліотеки для побудови нейронних мереж.

Python Asyncio – asyncio надає інфраструктуру для написання однопотокового паралельного коду з використанням співпрограми, мультиплексування доступу введення-виведення через сокети і інші ресурси, запуску мережевих клієнтів і серверів та інших пов'язаних примітивів. Ось більш докладний список вмісту пакета:

- підключається цикл подій з різними системними реалізаціями;
- абстракції транспорту та протоколу (аналогічні тим, що є в Twisted);
- конкретна підтримка TCP, UDP, SSL, каналів підпроцесу, відкладених викликів та інших (деякі можуть залежати від системи);
- клас Future, який імітує клас в модулі concurrent.futures, але адаптований для використання з циклом подій;
- співпрограми і завдання, засновані на виході з (PEP 380), щоб допомогти написати паралельний код в послідовному порядку;
- підтримка скасування ф'ючерсів і співпрограм;
- примітиви синхронізації для використання між співпрограмами в одному потоці, імітуючи їх в потоковому модулі;
- інтерфейс для передачі роботи в пул потоків, коли вам абсолютно необхідно використовувати бібліотеку, яка блокує виклики введення/виведення.

Для реалізації веб серверів і клієнта буде використовуватися бібліотека aiohttp.

Aiohttp – це HTTP Web сервер і клієнт для asyncio (PEP-3156).

Так само для реалізації веб додатка буде необхідна бібліотека Aiohttp Jinja2.

Aiohttp Jinja2 – це сучасна і зручна для розробників мова шаблонів для Python, створена за зразком шаблонів Django. Вона швидка, широко використовувана і безпечна завдяки додатковому середовищу виконання шаблонів в пісочниці. Це текстовий шаблонизатор, Тому він може бути використаний для створення будь-якого виду розмітки, а також вихідного коду. Ліцензований під BSD ліцензією. Шаблонизатор Jinja дозволяє налаштовувати:

- теги;
- фільтри;
- тести;
- глобальні змінні.

Також, на відміну від шаблонізатора Django, Jinja дозволяє конструктору шаблонів викликати функції з аргументами на об'єктах. Jinja, як і Smarty, також поставляється з простою у використанні системою фільтрів, схожою на конвеєр Unix.

2.5.3 Робота з даними

Як сховище даних буде використовуватися база даних PostgreSQL.

PostgreSQL – вільна об'єктно-реляційна система управління базами даних (СУБД).

Для роботи з PostgreSQL буде використана бібліотека Asyncpg. Asyncpg – це бібліотека інтерфейсу бази даних, розроблена спеціально для PostgreSQL і Python/asyncio. Asyncpg – це ефективна, чиста реалізація двійкового протоколу сервера PostgreSQL для використання з платформою Python asyncio. Однією з особливостей є те, що asyncpg спочатку реалізує

серверний протокол PostgreSQL і надає його можливості безпосередньо, а не приховує їх за загальним фасадом, таким як DB-API.

Для роботи з датасетом будуть використовуватися бібліотеки pandas і NumPy.

Pandas – програмна бібліотека мовою Python для обробки і аналізу даних. Робота pandas з даними будується поверх бібліотеки NumPy, що є інструментом нижчого рівня. Надає спеціальні структури даних і операції для маніпулювання числовими таблицями і тимчасовими рядами.

NumPy – це бібліотека мови Python, що додає підтримку великих багатовимірних масивів і матриць, разом з великою бібліотекою високорівневих (і дуже швидких) математичних функцій для операцій з цими масивами.

2.5.4 Бібліотеки навчання нейронних мереж

В якості бібліотек що використовуються для навчання нейронних мереж будуть виступати TensorFlow та Keras.

TensorFlow – відкрита програмна бібліотека для машинного навчання, розроблена компанією Google для вирішення завдань побудови і тренування нейронної мережі з метою автоматичного знаходження та класифікації образів, досягаючи якості людського сприйняття. Дана бібліотека дозволяє без особливих проблем створювати нейронні мережі різної складності, готувати дані для їх навчання і тестування. Так само TensorFlow надає інструменти для візуалізації процесів, що відбуваються під час навчання. Візуалізація є досить важливою частиною для розуміння якості навчання нейронної мережі. Але робота з TensorFlow вимагає досить глибоких пізнань в нейронних мережах. Для полегшення цього завдання існує бібліотека Keras.

Keras – відкрита нейромережева бібліотека, написана на мові Python. Вона являє собою надбудову над фреймворками Deeplearning, TensorFlow і Theano. Keras є високорівневим API для нейронних мереж, може працювати

поверх TensorFlow, CNTK або Theano. Інтерфейс був розроблений з метою забезпечення швидкого експериментування завдяки здатності переходити від ідеї до результату з найменшими тимчасовими затратамі.

2.6 Висновки за розділом 2

У розділі проведено дослідження технологій і програмного забезпечення, що використовуються для розробки системи виявлення атак на мережеві ресурси із застосуванням нейромережевих технологій. Дано визначення нейронних мереж і визначені завдання, які вони здатні вирішувати. Зроблено аналіз базової одиниці нейронної мережі – нейрона. Розглянуто різні структури нейронних мереж, такі як багатошарові, повнозв'язні і слабозв'язаних мережі.

Так само розглянуті підходи до навчання і встановлено, що для реалізації моделі нейронної мережі буде використовуватися навчання з учителем.

Зроблено розбір датасета CICIDS2017, який буде використовуватися для навчання та тестування моделі нейронної мережі. Наведені описи кожного з параметрів.

Окреслено основні інструменти, які використано для розробки системи виявлення атак на мережеві ресурси із застосуванням нейромережевих технологій. Цими інструментами будуть високорівнева мова програмування Python і спільні бібліотеки Aiohttp, Aiohttp Jinja2, Asyncpg, Pandas, NumPy, TensorFlow та Keras.

3 РОЗРОБКА СИСТЕМИ ВИЯВЛЕННЯ АТАК НА МЕРЕЖЕВІ РЕСУРСИ ІЗ ЗАСТОСУВАННЯМ НЕЙРОМЕРЕЖЕВИХ ТЕХНОЛОГІЙ

3.1 Визначення компонентів системи

Так як для збору трафіку буде використовуватися CICFlowMeter, а його єдиним можливим вихідним форматом даних є постійно оновлюваний csv файл, то існує необхідність в створенні інтерфейсу для його читання і видачі нових значень в зручному вигляді. Цим інтерфейсом буде служити веб-сервер, який в тлі буде зчитувати csv файл, переформатовувати вилучені дані в формат json і видавати їх у вигляді веб-сторінки.

Далі дані з веб-сторінки будуть зчитуватися веб-клієнтом і передаватися на аналіз в нейронну мережу. За результатами аналізу вони будуть позначені як нормальний або аномальний трафік. Аномальний трафік буде записаний в базу даних.

Дані з бази даних будуть формуватися і виводитися у вигляді невеликих звітів в спеціальному веб-додатку.

Така структура системи виявлення атак дозволить створювати необмежену кількість аналізуючих модулів, що дозволить обробляти трафік в реальному часі. Аналізуючи модулі можна розташовувати на декількох серверах організації. Так само при виході з ладу одного або декількох аналізуючих модулів система продовжує свою роботу.

3.2 Розробка колектора даних

Для початку розробки необхідно визначитися з форматом вхідних даних. Для цього запустимо програму CICFlowMeter і зробимо захоплення трафіку (рисунок 3.1).

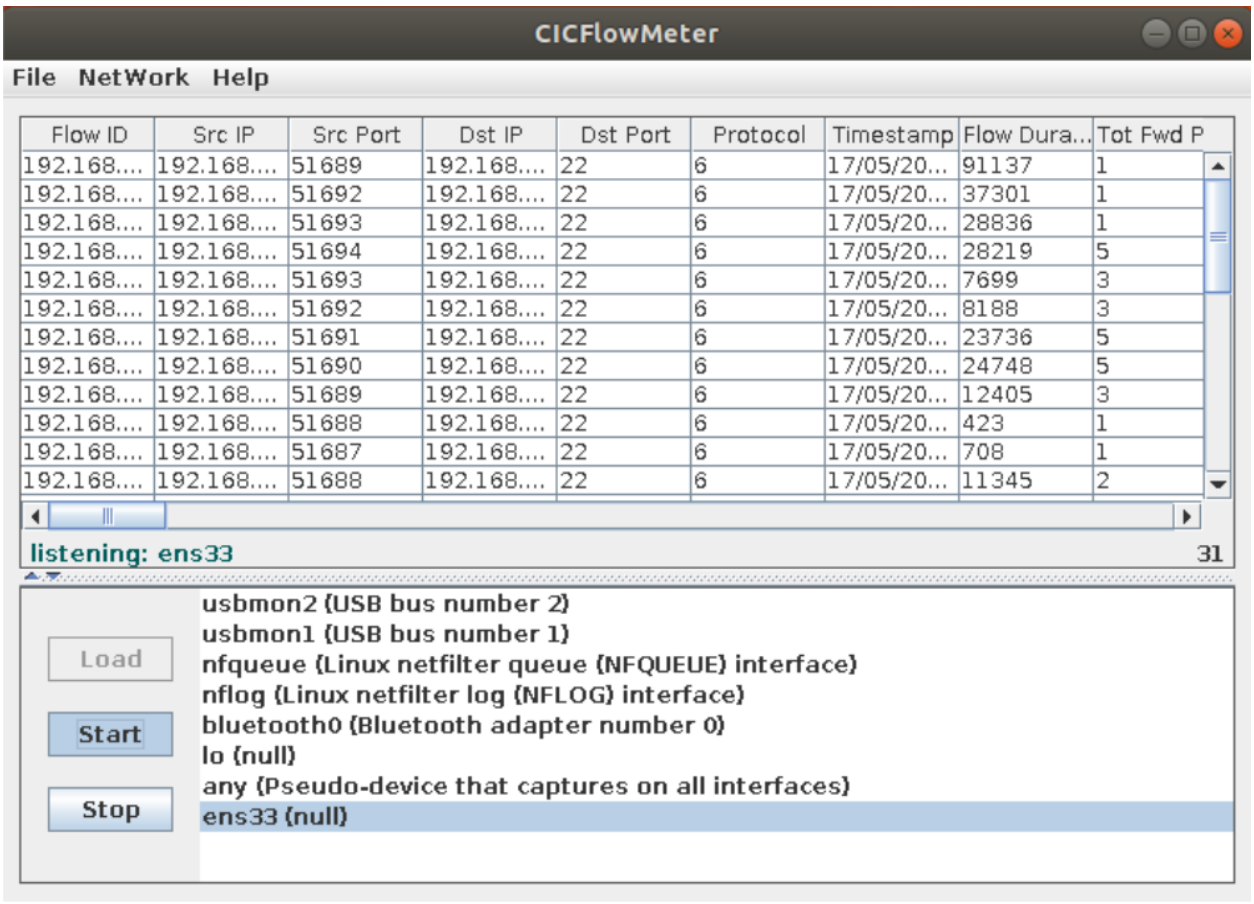

Рисунок 3.1 – Інтерфейс програми CICFlowMeter

Після початку захоплення трафіку програма створила файл 2020-05-17_Flow.csv. Приклад полів у файлі представлений в таблиці 3.1.

Таблиця 3.1 – Приклад формату даних наданих програмою

№	Назва параметру	Приклад даних
1	Flow ID	152.199.19.161–192.168.1.2–443–51472–6
2	Src IP	192.168.1.2
3	Src Port	51472
4	Dst IP	152.199.19.161
5	Dst Port	443
6	Protocol	6
7	Timestamp	17/05/2020 02:47:24 AM

Продовження таблиці 3.1

№	Назва параметру	Приклад даних
8	Flow Duration	299482
9	Tot Fwd Pkts	1
10	Tot Bwd Pkts	1
11	TotLen Fwd Pkts	0
12	TotLen Bwd Pkts	0
13	Fwd Pkt Len Max	0
14	Fwd Pkt Len Min	0
15	Fwd Pkt Len Mean	0
16	Fwd Pkt Len Std	0
17	Bwd Pkt Len Max	0
18	Bwd Pkt Len Min	0
19	Bwd Pkt Len Mean	0
20	Bwd Pkt Len Std	0
21	Flow Byts/s	0
22	Flow Pkts/s	6,678198
23	Flow IAT Mean	299482
24	Flow IAT Std	0
25	Flow IAT Max	299482
26	Flow IAT Min	299482
27	Fwd IAT Tot	0
28	Fwd IAT Mean	0
29	Fwd IAT Std	0
30	Fwd IAT Max	0
31	Fwd IAT Min	0
32	Bwd IAT Tot	0
33	Bwd IAT Mean	0

Продовження таблиці 3.1

№	Назва параметру	Приклад даних
34	Bwd IAT Std	0
35	Bwd IAT Max	0
36	Bwd IAT Min	0
37	Fwd PSH Flags	0
38	Bwd PSH Flags	0
39	Fwd URG Flags	0
40	Bwd URG Flags	0
41	Fwd Header Len	20
42	Bwd Header Len	20
43	Fwd Pkts/s	3,339099
44	Bwd Pkts/s	3,339099
45	Pkt Len Min	0
46	Pkt Len Max	0
47	Pkt Len Mean	0
48	Pkt Len Std	0
49	Pkt Len Var	0
50	FIN Flag Cnt	1
51	SYN Flag Cnt	0
52	RST Flag Cnt	0
53	PSH Flag Cnt	0
54	ACK Flag Cnt	1
55	URG Flag Cnt	0
56	CWE Flag Count	0
57	ECE Flag Cnt	0
58	Down/Up Ratio	1
59	Pkt Size Avg	0

Продовження таблиці 3.1

№	Назва параметру	Приклад даних
60	Fwd Seg Size Avg	0
61	Bwd Seg Size Avg	0
62	Fwd Byts/b Avg	0
63	Fwd Pkts/b Avg	0
64	Fwd Blk Rate Avg	0
65	Bwd Byts/b Avg	0
66	Bwd Pkts/b Avg	0
67	Bwd Blk Rate Avg	0
68	Subflow Fwd Pkts	1
69	Subflow Fwd Byts	0
70	Subflow Bwd Pkts	1
71	Subflow Bwd Byts	0
72	Init Fwd Win Byts	–1
73	Init Bwd Win Byts	1024
74	Fwd Act Data Pkts	0
75	Fwd Seg Size Min	0
76	Active Mean	0
77	Active Std	0
78	Active Max	0
79	Active Min	0
80	Idle Mean	0
81	Idle Std	0
82	Idle Max	0
83	Idle Min	0
84	Label	No Label

Після того як формат вхідних даних визначено приступимо до розробки програми. Для його реалізації вибрана мова python, також буде задіяна бібліотека aiohttp.

Логіка роботи програми представлена на діаграмах активності (рисунки 3.2 – 3.4).

Процес веб-сервера і процес читання і перетворення даних з файлу запускаються паралельно і виконуються до тих пір, поки не буде отримано сигнал виходу з програми. Після запуску програма очистить всі файли в директорії і чекатиме новий, який повинен створити CICFlowMeter. Далі буде відбуватися построкове читання файлу, форматування і видача даних у вигляді веб сторінки.

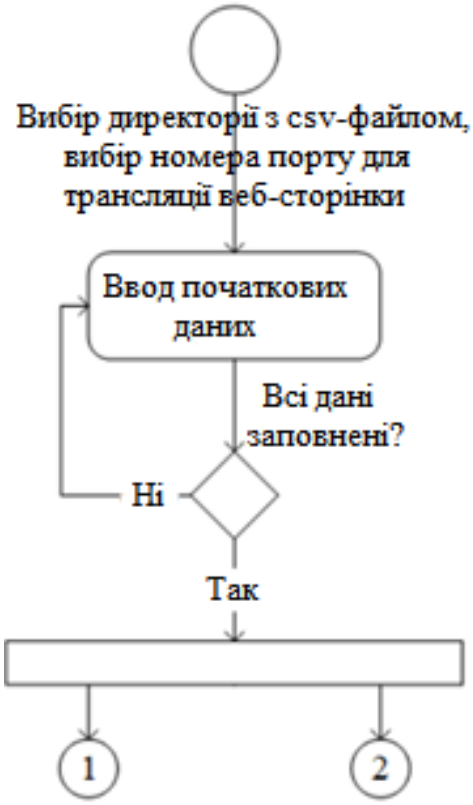

Рисунок 3.2 – Діаграма активності програми збирача даних

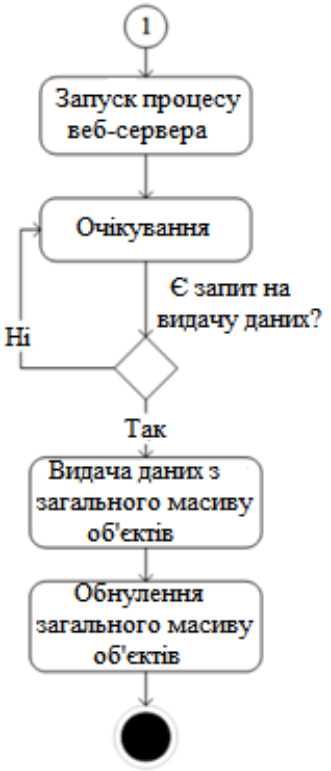

Рисунок 3.3 – Діаграма активності збирача, веб-сервер

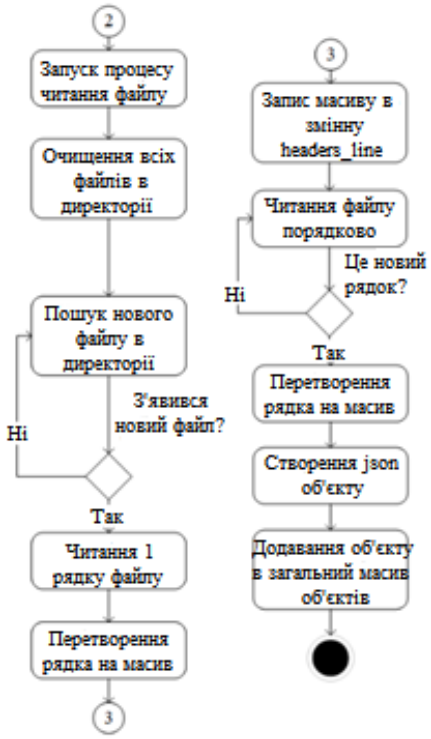

Рисунок 3.4 – Діаграма активності програми збирача даних, модуль читання і перетворення даних з файлу

Рисунок 3.5 – Вихідні дані програми в форматі json

3.3 Розробка аналізуючого модуля

Для реалізації аналізуючого модуля так само вибрана мова python з використанням бібліотек aiohttp, asyncpg, pandas, keras і numpy. Логіка роботи програми представлена на діаграмі активності (рисунок 3.6).

Процес аналізу запускається в нескінченному циклі і працює до тих пір, поки не буде отримано сигнал виходу з програми. Дані, які будуть отримані від збирача будуть надмірними для аналізу їх нейронною мережею, але так як нейронні мережі, налаштовані під різні атаки використовують різні поля з цих даних, від цієї надмірності можна відмовитися.

Варто відзначити, що вміст файлу nn_analyse.py може змінюватися в залежності від моделі нейронної мережі. Так як на вхід нейронної мережі можуть подаватися різні дані. У даній програмній реалізації можливе підключення декількох нейронних мереж для аналізу даних.

Так само була створена база даних, її даталогічна модель (рисунок 3.7).

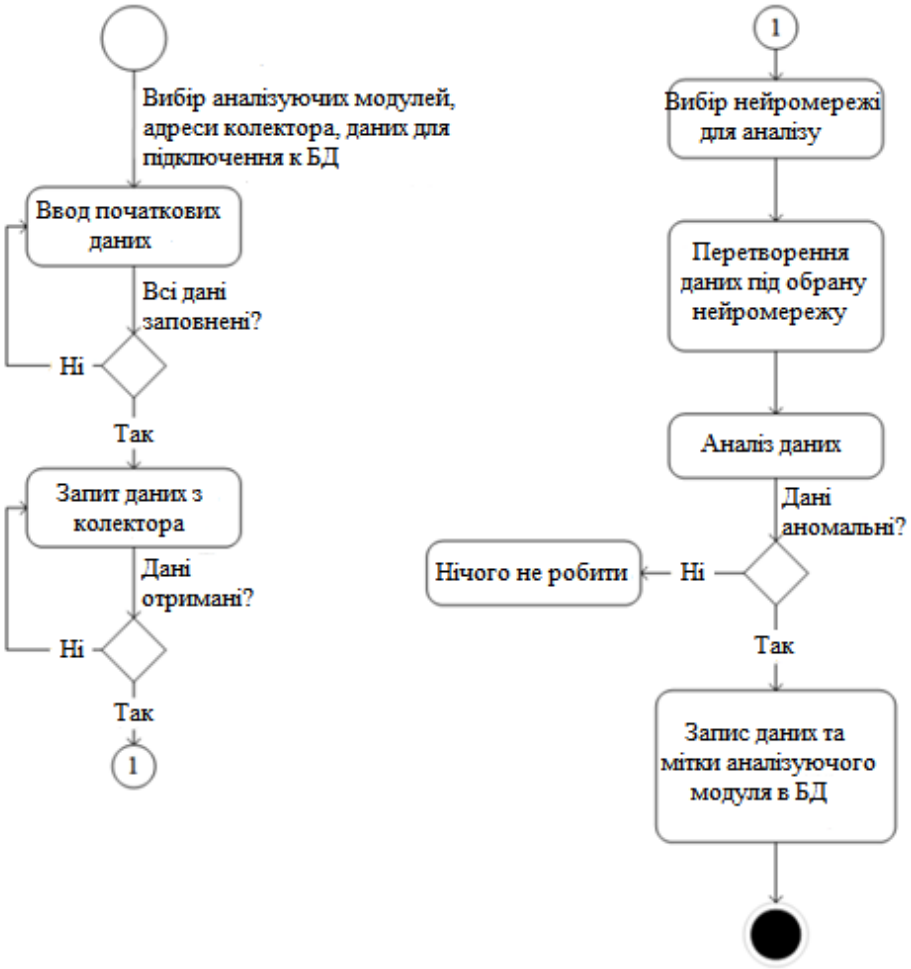

Рисунок 3.6 – Діаграма активності програми аналізатора

3.4 Розробка веб-додатку для перегляду підозрілої активності

Веб-додаток не матиме моделі авторизації. Так як управління доступом буде забезпечено на рівні веб-сервера, який обмежить список ip адрес з яких може бути отриманий доступ до веб-додатку. Як сховище даних буде використаний PostgreSQL сервер.

У відповідності до потреб були розроблені такі екрани:

- головний екран зі списком подій;
- екран з подброною інформацією про подію.

Головна сторінка додатка (рисунок 3.8). На ній відображені інциденти що відбулися, їх номер, ім'я модуля що виявив, ip:порт адреси джерела, ip:порт адреси призначення і датафіксації інциденту.

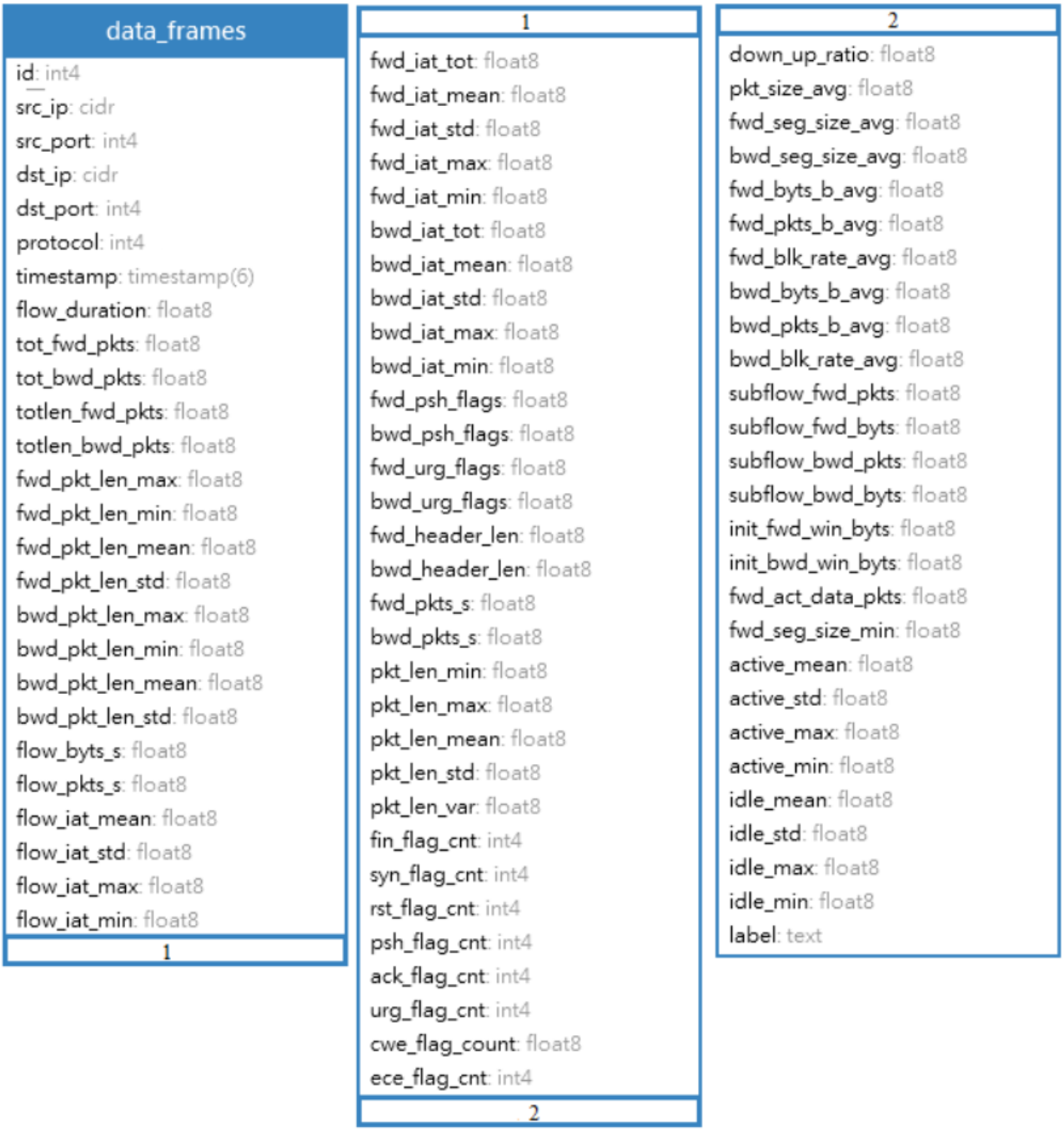

Рисунок 3.7– Даталогіческая модель бази даних

Всі інциденти відсортовані за датою. Так само є можливість побачити більш детальну інформацію про інцидент, натиснувши на кнопку «Детальніше».

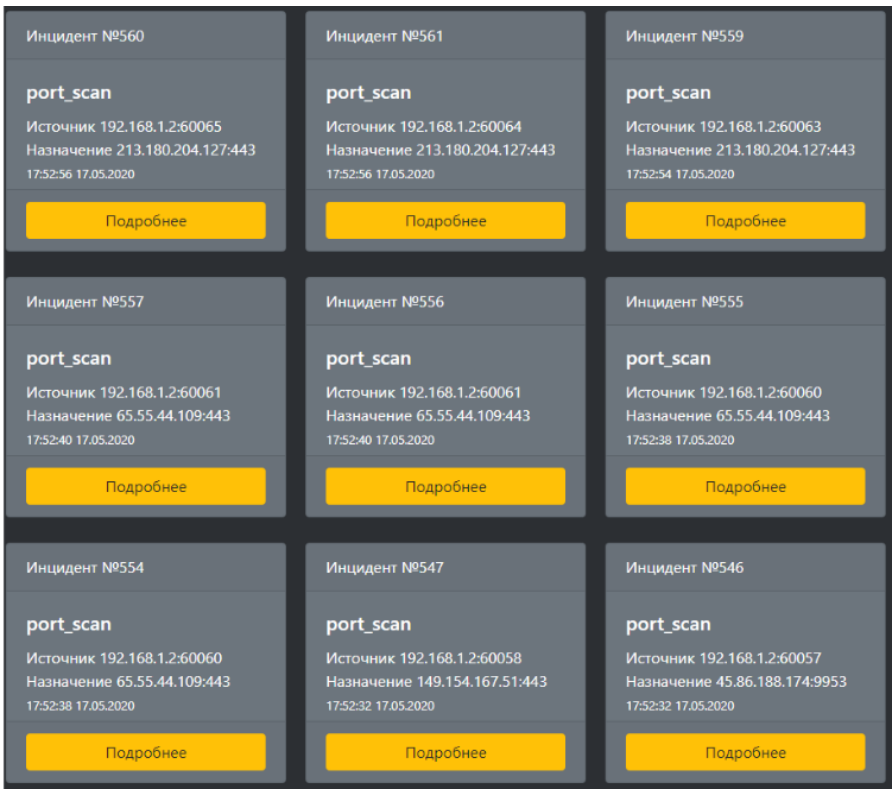

Рисунок 3.8 – Головна сторінка веб-додатку

Натиснувши на кнопку «більше» користувач потрапляє на сторінку з докладним описом інциденту (рисунок 3.9). Тут у вигляді таблиці відбито повний список даних щодо цього інциденту.

Рисунок 3.9 – Детальний опис інциденту

3.5 Висновки за розділом 3

У розділі виконано визначення компонентів системи в ході якого було встановлено які компоненти необхідно розробити для системи виявлення атак на мережеві ресурси із застосуванням нейромережних технологій щоб вона могла працювати зі збирачем трафіку CICFlowMeter і при цьому забезпечувала роботу в реальному часі.

Проведена розробка збирача даних. В ході розробки було встановлено які дані надає програма CICFlowMeter і в якому форматі. Далі був розроблений веб сервер на мові python, який зчитує csv файл, що генерується програмою CICFlowMeter, перетворює їх в формат json і надає їх на спеціальній веб сторінці.

Проведена розробка аналізуючого модуля. Аналізуючий модуль збирає дані що генеруються збирачем даних і «проганяє» їх через моделі нейронних мереж. Якщо дані позначаються як аномальні, вони записуються в базу даних. В ході розробки аналізуючого модуля була розроблена модель бази даних для зберігання даних помічених як аномальні.

Проведена розробка веб-додатку для перегляду даних, що були позначені як аномальні. Додаток має всього 2 екрани, але при цьому надає повну інформацію про аномальні дані.

4 РОЗРОБКА ТА ТЕСТУВАННЯ МОДЕЛІ НЕЙРОННОЇ МЕРЕЖІ

4.1 Розробка моделі для виявлення DoS атак

Для початку розробки моделі потрібно визначитися що являє із себе модель нейронної мережі. Модель нейронної мережі описує її архітектуру і конфігурацію, а також використовувані алгоритми навчання.

Архітектура нейронної мережі визначає загальні принципи її побудови (плоскошарова, повнозв’язна, слабозв'язана, прямого поширення, рекурентна і т.д.).

Конфігурація конкретизує структуру мережі в рамках заданої архітектури: число нейронів, число входів і виходів мережі, які використовують активаційні функції.

Виходячи з цього необхідно вибрати архітектуру нейронної мережі. Перед нейронною мережею стоїть завдання класифікації трафіку. Для задач класифікації підходять наступні архітектури нейронних мереж: перцептрон, згорткові нейронні мережі, мережі адаптивного резонансу, мережа радіально-базисних функцій.

Виходячи з того, що дані, які будуть використовуватися для навчання вже заздалегідь розмічені, то розумно буде обирати між згортковими нейронними мережами і перцептроном, бо саме вони підходять для навчання з учителем. Так як згорткові нейронні мережі це спеціальна архітектура штучних нейронних мереж, запропонована Яном Лекуном в 1988 році і націлена на ефективне розпізнавання образів, то для реалізації моделі нейронної мережі для системи виявлення атак на мережеві ресурси буде обрана архітектура перцептрону (рисунок 4.1).

Далі визначаємо конфігурації моделі. Число входів і виходів нейронної мережі буде залежати від даних, які використовуються для навчання. Для атестаційній роботи буде використана база даних атак CICIDS2017.

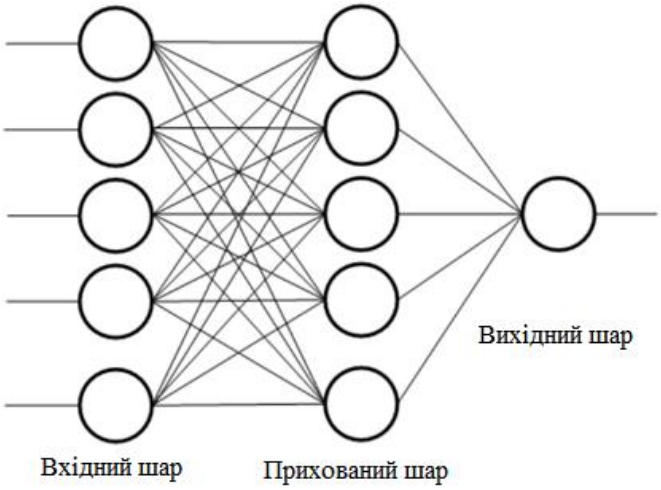

Рисунок 4.1 – Структурна схема перцептрону

Дані являють собою csv файл в якому міститься 77 параметрів і підсумкова оцінка до цих параметрів чи є трафік аномальним або нормальним. Спочатку дані мають вигляд, представлений в таблиці 4.1 в стовпці «Приклад даних до перетворення». Такий формат для нейронної мережі не підходить. Всі дані повинні мати числовий формат. Тому вони були приведені до вигляду, представленому в таблиці 4.1 в стовпці «Приклад даних після перетворення».

Таблиця 4.1 – Параметри датасета до і після перетворення

№	Назва параметру	Приклади даних до перетворення	Приклади даних після перетворення
1	2	3	4
1	Flow Duration	3	3
2	Total Fwd Packets	2	2
3	Total Backward Packets	0	0
4	Total Length of Fwd Packets	12	12
5	Total Length of Bwd Packets	0	0
6	Fwd Packet Length Max	6	6

Продовження таблиці 4.1

1	2	3	4
7	Fwd Packet Length Min	6	6
8	Fwd Packet Length Mean	6	6
9	Fwd Packet Length Std	0	0
10	Bwd Packet Length Max	0	0
11	Bwd Packet Length Min	0	0
12	Bwd Packet Length Mean	0	0
13	Bwd Packet Length Std	0	0
14	Flow Bytes/s	4000000	4000000
15	Flow Packets/s	666666,7	666666,7
16	Flow IAT Mean	3	3
17	FlowIAT Std	0	0
18	Flow IAT Max	3	3
19	Flow IAT Min	3	3
20	Fwd IAT Total	3	3
21	Fwd IAT Mean	3	3
22	Fwd IAT Std	0	0
23	Fwd IAT Max	3	3
24	Fwd IAT Min	3	3
25	Bwd IAT Total	0	0
26	Bwd IAT Mean	0	0
27	Bwd IAT Std	0	0
28	Bwd IAT Max	0	0
29	Bwd IAT Min	0	0
30	Fwd PSH Flags	0	0
31	Bwd PSH Flags	0	0
32	Fwd URG Flags	0	0

Продовження таблиці 4.1

1	2	3	4
33	Bwd URG Flags	0	0
34	Fwd Header Length	40	40
35	Bwd Header Length	0	0
36	Fwd Packets/s	666666,7	666666,7
37	Bwd Packets/s	0	0
38	Min Packet Length	6	6
39	Max Packet Length	6	6
40	Packet Length Mean	6	6
41	Packet Length Std	0	0
42	Packet Length Variance	0	0
43	FIN Flag Count	0	0
44	SYN Flag Count	0	0
45	RST Flag Count	0	0
46	PSH Flag Count	0	0
47	ACK Flag Count	1	1
48	URG Flag Count	0	0
49	CWE Flag Count	0	0
50	ECE Flag Count	0	0
51	Down/Up Ratio	0	0
52	Average Packet Size	9	9
53	Avg Fwd Segment Size	6	6
54	Avg Bwd Segment Size	0	0
55	Fwd Header Length	40	40
56	Fwd Avg Bytes/Bulk	0	0
57	Fwd Avg Packets/Bulk	0	0
58	Fwd Avg Bulk Rate	0	0

Продовження таблиці 4.1

1	2	3	4
59	Bwd Avg Bytes/Bulk	0	0
60	Bwd Avg Packets/Bulk	0	0
61	Bwd Avg Bulk Rate	0	0
62	Subflow Fwd Packets	2	2
63	Subflow Fwd Bytes	12	12
64	Subflow Bwd Packets	0	0
65	Subflow Bwd Bytes	0	0
66	Init_Win_bytes_forward	33	33
67	Init_Win_bytes_backward	–1	–1
68	act_data_pkt_fwd	1	1
69	min_seg_size_forward	20	20
70	Active Mean	0	0
71	Active Std	0	0
72	Active Max	0	0
73	Active Min	0	0
74	Idle Mean	0	0
75	Idle Std	0	0
76	Idle Max	0	0
77	Idle Min	0	0
78	Label	BENIGN	0

Параметр оцінки трафіку отримав номер, де 1 – аномальний трафік, 0 – нормальний. Всі числа були приведені до матеріального типу з плаваючою комою. Оскільки для навчання необхідно 2 набори даних, тренувальний і тестовий, то датасет буде поділений так, що 20% даних буде віддано під тести, а 80% для навчання.

Для початку розробки моделі необхідно визначитися що таке DoS атака і чим вона характеризується. Датасет містить 78 параметрів їх повне використання є недоцільним, бо деякі параметри під час атак залишаються незмінними. А велика кількість вхідних параметрів істотно ускладнить створення моделі.

DoS (Denial of Service «відмова в обслуговуванні») – хакерська атака на обчислювальну систему з метою довести її до відмови, тобто створення таких умов, при яких сумлінні користувачі системи не зможуть отримати доступ до наданих системних ресурсів (серверів), або цей доступ буде ускладнений. Виходячи з характеру впливу DoS атаки можна віднести до активних атак.

Для DoS атак характерна велика кількість пакетів, які необхідно обробити, але через те, що обчислювальні потужності обмежені виникають затримки, а згодом і відмови в обслуговуванні. Виходячи з цього з таблиці 4.1 було обрано такі параметри: Total Fwd Packets, Total Backward Packets, Fwd Packet Length Min, FwdPacket Length Mean, Fwd Packet Length Std, Bwd Packet Length Min, Fwd Header Length, Bwd Header Length, Min Packet Length, Packet Length Mean, PSH Flag Count, ACK Flag Count, URG Flag Count, Down/Up Ratio, Average Packet Size, Avg Fwd Segment Size, Subflow Fwd Packets, Subflow Bwd Packets, act_data_pkt_fwd, min_seg_size_forward, Label.

Таким чином модель матиме 20 нейронів на вході і 2 на виході. Вихідні нейрони будуть визначені як Normal і DoS, а їх вихідні значення будуть показувати в якій мірі нейронна мережа схиляється до тої чи іншої відповіді.

В якості активаційної функції у всіх шарах крім вихідного виступатиме функція relu, на вихідному шарі буде функція softmax. Вибір relu обумовлений тим, що вона є найбільш популярною і найбільш використовуваною в створенні нейронних мереж. Вибір функції softmax на вихідному шарі обумовлений видом вихідних даних.

Для того щоб уникнути перенавчання нейронної мережі в бібліотеці Keras передбачена функцій ранньої зупинки. Рання зупинка – це метод, який

дозволяє вказати довільно велику кількість епох навчання і припинити навчання, як тільки продуктивність моделі перестає поліпшуватися.

Так як кількість епох навчання буде контролюватися ранньою зупинкою, то залишається підібрати кількість прихованих шарів і кількість нейронів в прихованих шарах.

Знайдемо оптимальну кількість прихованих шарів. Кількість нейронів у кожному прихованому шарі буде дорівнювати 20. Оцінка якості навчання буде проводитися за двома параметрами: точність і втрати. Точність повинна прагнути до 1, а втрати – до 0. Результати підбору представлені в таблиці 4.2.

Таблиця 4.2 – Підбір оптимальної кількості прихованих шарів

№	Кількість прихованих шарів	Втрати	Точність
1	2	3	4
1	1	5,874060	0,634876
2	2	0,009109	0,998627
3	3	0,008066	0,998760
4	4	0,008952	0,998582
5	5	0,004545	0,998782
6	6	0,004774	0,998826
7	7	0,008644	0,998671
8	8	0,012150	0,998516
9	9	0,004999	0,998870
10	10	0,005463	0,998760
11	11	0,012202	0,998538
12	12	0,015628	0,997807
13	13	0,009046	0,998671
14	14	0,005136	0,998893
15	15	0,400654	0,998893
16	16	0,549039	0,634743
17	17	0,683352	0,569935
18	18	0,683342	0,569935

Продовження таблиці 4.2

1	2	3	4
19	19	0,683556	0,569935
20	20	0,683431	0,569935
21	21	0,683355	0,569935
22	22	0,683346	0,569935
23	23	0,683348	0,569935
24	24	0,683530	0,569935
25	25	0,683574	0,569935

За підсумками тестування оптимальною кількістю прихованих шарів є 6, адже в такому випадку виходить найбільша точність і найменші втрати.

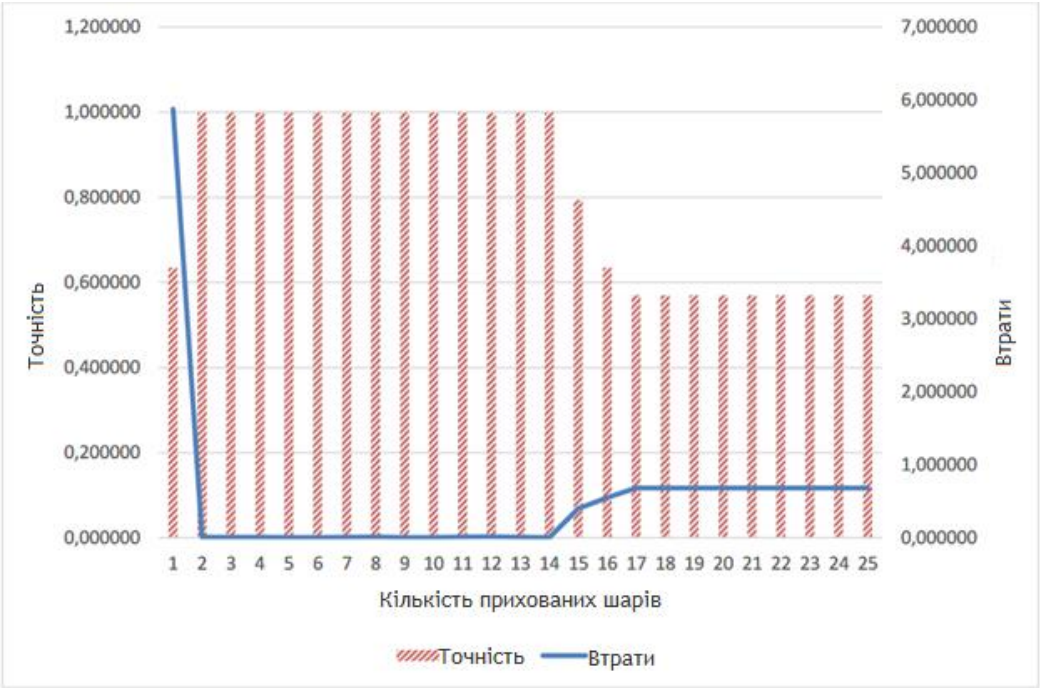

Рисунок 4.2 – Результати тестування оптимальної кількості шарів

Далі знайдемо оптимальну кількість нейронів в прихованих шарах. Кількість прихованих шарів буде дорівнювати 6.

Таблиця 4.3 – Підбір оптимальної кількості нейронів в прихованих шарах

№	Кількість прихованих шарів	Втрати	Точність
1	2	3	4
1	1	0,68335	0,56994
2	2	0,68346	0,56994
3	3	0,02298	0,99772
4	4	0,00729	0,99863
5	5	0,01075	0,99849
6	6	0,00679	0,99883
7	7	0,00722	0,99854
8	8	0,00806	0,99865
9	9	0,00736	0,99867
10	10	0,01158	0,99858
11	11	0,01019	0,99860
12	12	0,00678	0,99865
13	13	0,01083	0,99856
14	14	0,00699	0,99865
15	15	0,00533	0,99865
16	16	0,01168	0,99847
17	17	0,00521	0,99876
18	18	0,00540	0,99883
19	19	0,00709	0,99874
20	20	0,00946	0,99825
21	21	0,00693	0,99883
22	22	0,00572	0,99874
23	23	0,00922	0,99807
24	24	0,00543	0,99894

Продовження таблиці 4.3

1	2	3	4
25	25	0,00979	0,99852
26	26	0,00684	0,99874
27	27	0,00492	0,99883
28	28	0,01066	0,99816
29	29	0,00844	0,99867
30	30	0,00946	0,99872
31	31	0,00526	0,99887
32	32	0,00453	0,99885
33	33	0,00762	0,99856
34	34	0,01094	0,99856
35	35	0,00687	0,99874
36	36	0,00903	0,99872
37	37	0,00975	0,99863
38	38	0,00845	0,99863
39	39	0,00879	0,99869
40	40	0,00870	0,99869
41	41	5,87296	0,63499
42	42	0,00456	0,99887
43	43	0,00834	0,99876
44	44	0,00911	0,99869
45	45	0,00856	0,99872
46	46	0,00472	0,99885
47	47	0,00528	0,99869
48	48	0,00892	0,99867
49	49	0,01034	0,99858
50	50	5,87522	0,63470

За підсумками тестування оптимальною кількістю нейронів в прихованих шарах є 42, так як в такому випадку отримано найбільшу точність і найменші втрати.

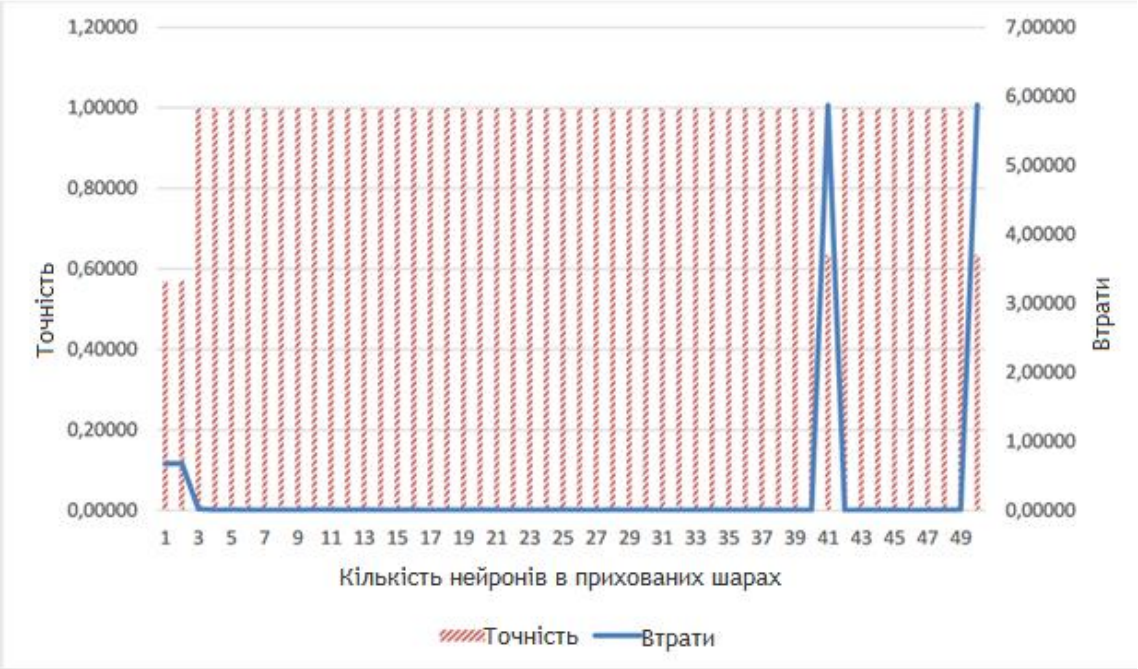

Рисунок 4.3 – Результати тестування оптимальної кількості нейронів в прихованих шарах

Таким чином модель матиме 6 прихованих шарів і в кожному прихованому шарі буде знаходитися по 42 нейрона. Точність моделі досягла 99,88%. Розроблена модель збережена в файл dos.h5 і може бути підключена до раніше розробленого аналізуючого модуля.

4.2 Розробка моделі для виявлення PortScan атак

Так само як і в попередньому розділі визначимо що таке PortScan атака і чим вона характеризується. PortScan це процес, який відправляє запити клієнта до діапазону портів сервера адрес на хості, з метою знаходження активного порту. Виходячи з характеру впливу PortScan атаки можна

віднести до пасивних атак. Для PortScan атак характерне мала кількість невеликих пакетів. Так само з'єднання, встановлене сканером з пристрієм що атакується триває дуже короткий час. Таким чином модель матиме 38 нейронів на вході і 2 на виході. Вихідні нейрони будуть визначені як Normal та PortScan, а їх вихідні значення будуть показувати в якій мірі нейронна мережа схиляється до тієї чи іншої відповіді. В якості активаційної функції у всіх шарах крім вихідного виступатиме функція relu, на вихідному шарі буде функція softmax. Вибір relu обумовлен тим, що вона є найбільш популярною і найбільш використовуваною в створенні нейронних мереж. Вибір функції softmax на вихідному шарі обумовлений видом вихідних даних.

Для того щоб уникнути перенавчання нейронної мережі в бібліотеці Keras передбачена функцій ранньої зупинки. Так як кількість епох навчання буде контролюватися ранньої зупинкою, то залишається підібрати кількість прихованих шарів і кількість нейронів в прихованих шарах. Знайдемо оптимальну кількість прихованих шарів. Кількість нейронів у кожному прихованому шарі буде дорівнювати 38. Оцінка якості навчання буде проводитися за двома параметрами: точність і втрати. Точність повинна прагнути до 1, а втрати – до 0. Результати підбору представлені в таблиці 4.4.

Таблиця 4.4 – Підбір оптимальної кількості прихованих шарів

№	Кількість прихованих шарів	Втрати	Точність
1	2	3	4
1	1	0,77454	0,951914686
2	2	0,66013	0,959000943
3	3	8,93537	0,445631305
4	4	0,08174	0,993786435
5	5	0,08399	0,992861382
6	6	0,05737	0,995200195
7	7	0,0618	0,993297728

Продовження таблиці 4.4

1	2	3	4
8	8	0,02638	0,997521555
9	9	0,05061	0,990138583
10	10	0,03054	0,99581108
11	11	0,02781	0,997853178
12	12	0,0296	0,996736133
13	13	0,04935	0,993210458
14	14	0,02609	0,997818271
15	15	0,03108	0,997050302
16	16	0,03206	0,994798757
17	17	0,02788	0,99745174
18	18	0,03445	0,994886026
19	19	0,02999	0,996317241
20	20	0,03727	0,994973296
21	21	0,03036	0,99640451
22	22	0,03096	0,996212518
23	23	0,03043	0,996666318
24	24	0,68726	0,554368695
25	25	0,68733	0,554368695
26	26	0,68726	0,554368695
27	27	0,68722	0,554368695
28	28	0,68729	0,554368695
29	29	0,68723	0,554368695
30	30	0,68724	0,554368695

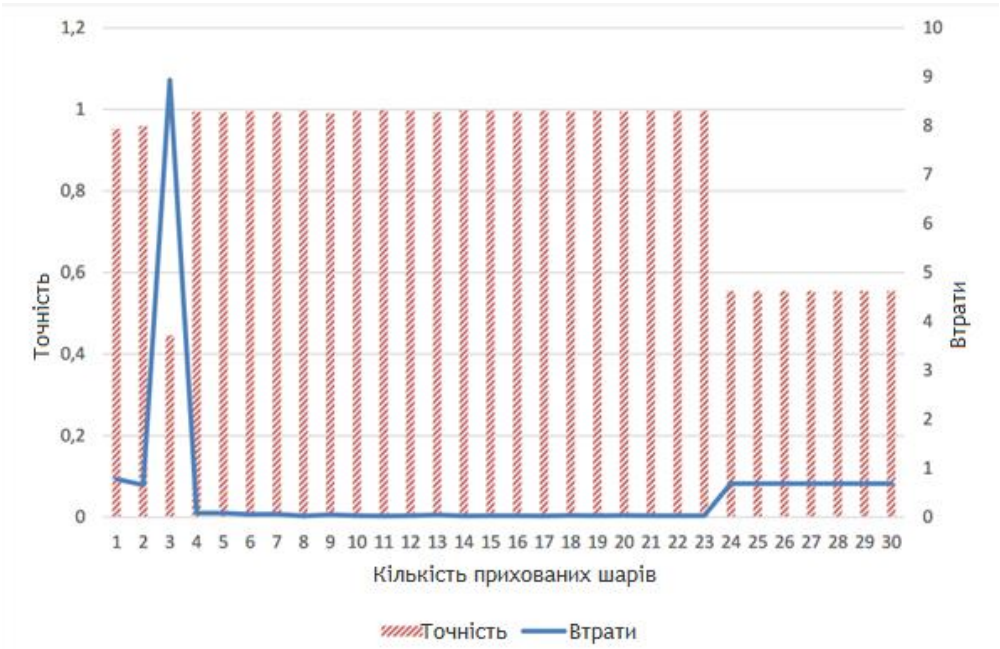

Рисунок 4.4 – Результати тестування оптимальної кількості шарів

За підсумками тестування оптимальною кількістю прихованих шарів є 14, так як в такому випадку виходить найбільша точність і найменші втрати.

Далі знайдемо оптимальну кількість нейронів в прихованих шарах. Кількість прихованих шарів буде дорівнювати 14

Таблиця 4.5 – Підбір оптимальної кількості нейронів в прихованих шарах

№	Кількість прихованих шарів	Втрати	Точність
1	2	3	4
1	1	0,68724	0,554368695
2	2	0,68722	0,554368695
3	3	0,68725	0,554368695
4	4	0,68723	0,554368695
5	5	0,68722	0,554368695
6	6	0,68726	0,554368695
7	7	0,68722	0,554368695

Продовження таблиці 4.5

1	2	3	4
8	8	0,03793	0,992250497
9	9	0,68723	0,554368695
10	10	0,02732	0,99708521
11	11	0,03046	0,996055433
12	12	0,02655	0,997032848
13	13	0,02669	0,996910671
14	14	0,04446	0,992669389
15	15	0,03115	0,996352149
16	16	0,03293	0,996020526
17	17	0,03476	0,996334695
18	18	0,03157	0,996910671
19	19	0,03471	0,995601634
20	20	0,05284	0,988218662
21	21	0,06192	0,99408315
22	22	0,02651	0,996928125
23	23	0,03173	0,995741264
24	24	0,02888	0,996648864
25	25	0,03809	0,994065696
26	26	0,0799	0,993402451
27	27	0,05415	0,996090341
28	28	0,02688	0,997591371
29	29	0,05823	0,984972248
30	30	0,03932	0,99513038
31	31	0,02987	0,996247426
32	32	0,02874	0,997347017

Продовження таблиці 4.5

1	2	3	4
33	33	0,02867	0,996928125
34	34	0,05028	0,990958914
35	35	0,05611	0,994938388
36	36	0,02964	0,99663141
37	37	0,0324	0,99581108
38	38	0,03413	0,99535728
39	39	0,03951	0,996160156
40	40	0,05177	0,996299787
41	41	0,0818	0,993001012
42	42	0,02605	0,997469194
43	43	0,04115	0,99431005
44	44	0,06058	0,989841868
45	45	0,02857	0,997032848
46	46	0,05755	0,994920934
47	47	0,05755	0,997626278
48	48	0,05754	0,989684784
49	49	0,03143	0,997259748
50	50	0,03291	0,996090341
51	51	0,04797	0,996195064
52	52	0,08009	0,993524627
53	53	0,02998	0,997155025
54	54	0,05547	0,995549272
55	55	0,07225	0,992529759
56	56	7,18273	0,554368695
57	57	0,03767	0,993908612

Продовження таблиці 4.5

1	2	3	4
58	58	0,05888	0,995235103
59	59	7,18266	0,554368695
60	60	0,03585	0,995723811
61	61	0,03558	0,993018466
62	62	0,04722	0,993053374
63	63	0,06523	0,992896289
64	64	0,03649	0,994275142
65	65	0,02938	0,997294656
66	66	0,08681	0,993594443
67	67	0,08031	0,993873704
68	68	0,03433	0,995514365
69	69	0,08241	0,993891158
70	70	0,04659	0,992093413
71	71	0,17195	0,968809998
72	72	0,04732	0,99221559
73	73	0,13207	0,970572835
74	74	0,07791	0,993891158
75	75	0,06514	0,993786435
76	76	0,10259	0,991447621
77	77	0,06179	0,989283346
78	78	0,08095	0,99371662
79	79	0,02548	0,99790554
80	80	0,05432	0,995688903
81	81	0,062	0,995741264
82	82	0,0796	0,994048242

Продовження таблиці 4.5

1	2	3	4
83	83	0,05671	0,995392188
84	84	0,03129	0,995915803
85	85	0,05911	0,994624219
86	86	0,03058	0,996613956
87	87	0,05476	0,996299787
88	88	0,13811	0,989265892
89	89	7,18273	0,554368695
90	90	7,18273	0,554368695
91	91	0,03707	0,995741264
92	92	7,18273	0,554368695
93	93	0,08534	0,992651936
94	94	0,05691	0,995304918
95	95	0,0513	0,99603798
96	96	0,03068	0,996020526
97	97	0,05749	0,994990749
98	98	0,03293	0,996701225
99	99	0,03494	0,995619088
100	100	0,03573	0,99581108

За підсумками тестування оптимальною кількістю нейронів в прихованих шарах є 79, так як в такому випадку виходить найбільша точність і найменші втрати.

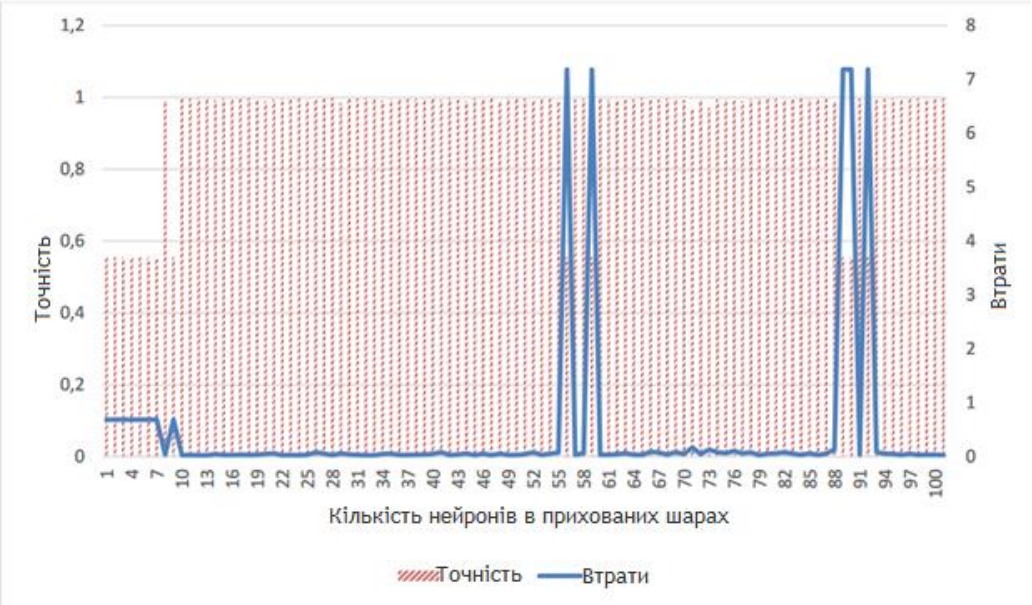

Рисунок 4.5 – Результати тестування оптимальної кількості нейронів в прихованих шарах

Таким чином модель матиме 14 прихованих шарів і в кожному прихованому шарі буде знаходитися по 79 нейронів. Точність моделі досягла 99,79%. Розроблена модель збережена в файл portscan.h5 і може бути підключена до раніше розробленого аналізуючого модулю.

4.3 Тестування розроблених моделей

Після того як були розроблені всі компоненти системи їх можна об'єднати разом і протестувати. Тестування буде проведено на записі трафіку, прокладеному до датасета. До цього запису докладено опис, о котрій годині і які атаки проводилися. Завдяки цьому опису можна буде оцінити працездатність системи. Відповідно до опису проводилися такі атаки:

Port Scan.

З увімкненим файрволлом атаки проводилися о: 13:55 – 13:57, 13:58 – 14:00, 14:01 – 14: 04, 14:05 – 14: 07, 14:08 – 14:10, 14:11 – 14:13, 14:14 – 14:16, 14:17 – 14:19, 14:20 – 14:21, 14:22 – 14:24, 14:33 – 14:33, 14:35 – 14:35.

З вимкненим файрволлом атаки проводились о: 14:51 – 14:53, 14:54 – 14:56, 14:57 – 14:59, 15:00 – 15:02, 15:03 – 15:05, 15:06 – 15:07, 15:08 – 15:10, 15:11 – 15:12, 15:13 – 15:15, 15:16 – 15:18, 15:19–15:21, 15:22 – 15:24, 15:25 – 15:25, 15:26 – 15:27, 15:28 – 15:29. Атака відбувалася з IP 205.174.165.73 на IP 172.16.0.1, між атакуючим і тим що був атакований був фаєрвол з IP 205.174.165.80.

DDoS LOIT атака проводилася з 15:56 до 16:16. Атака відбувалася з IP 205.174.165.69, 70, 71 на IP 172.16.0.1, між атакуючим і тим що був атакований був фаєрвол з IP 205.174.165.80.

Перед початком тестування необхідно трохи змінити модуль тестування. Потрібно змінити дані, які будуть подаватися на вхід моделі нейронної мережі, на такі що використовувалися при навчанні.

Запустимо модулі системи:

- база даних;
- веб-додаток;
- модуль збору. Після запуску він буде очікувати появи в зазначеній папці csv файлу. Після його появи він почне його зчитувати, а зчитані дані будуть поставлені в чергу для аналізу;
- CICFlowMeter. Після запуску необхідно вибрати розташування файлу з записаним трафіком, шлях збереження вихідного csv файлу і натиснути кнопку «ОК». Шлях збереження повинен збігатися зі шляхом зазначеним в модулі збору. Інтерфейс CICFlowMeter із запущеним читанням трафіку (рисунок 4.6);
- аналізуючий модуль. Після запуску почнеться аналіз даних, вся знайдена підозріла активність буде записана в базу даних і її можна буде переглянути через веб-додаток.

В ході читання трафіку з файлу система виявила 26 інцидентів. Відображення помічених атак в веб-додатку (рисунок 4.7).

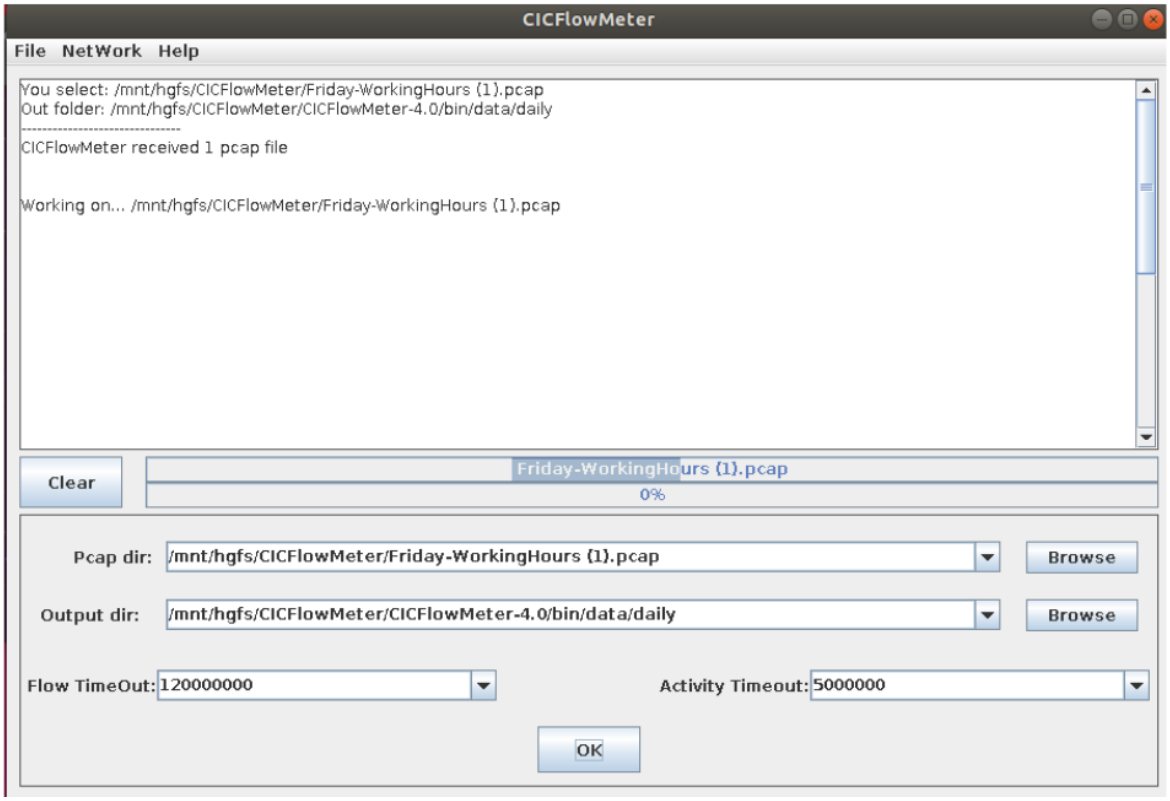

Рисунок 4.6 – Інтерфейс програми CICFlowMeter з запущеним читанням трафіку з файлу

З 26 інцидентів 13 були помічені як DDoS, а 13 як PortScan. DDoS інциденти позначалися на всьому часовому проміжку, зазначеному в описі до запису трафіку, при увімкнутому файрволлі не було розпізнано жодної PortScan атаки, при вимкненому файрволлі була пропущена 1 PortScan атака, інші були помічені як інциденти і збережені в базі даних.

4.4 Висновки за розділом 4

У розділі проведена розробка двох моделей нейронних мереж. Утворені моделі отримали наступні параметри:

- модель для виявлення DoS атак – 20 нейронів на вході, 2 на виході, 6 прихованих шарів, в кожному прихованому шарі по 42 нейрони, точність моделі 99,88%;

- модель для виявлення PortScan атак – 38 нейронів на вході, 2 на виході, 14 прихованих шарів, в кожному шарі по 79 нейронів, точність моделі досягла 99,79%.

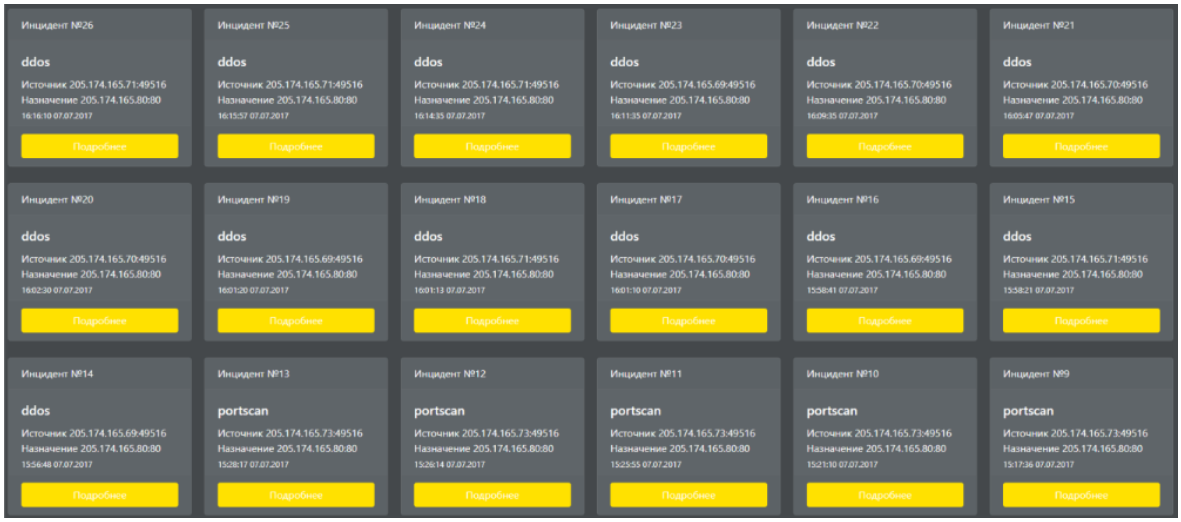

Рисунок 4.7 – Інтерфейс веб-додатки з позначеними атаками

Також в розділі проведено загальне тестування системи в ході якого система виявила 26 інцидентів, 13 з яких були позначені як DDoS, а 13 як PortScan. DDoS інциденти позначалися на всьому часовому проміжку, зазначеному в описі до запису трафіку, при увімкненому файрволлі не було розпізнано жодної PortScan атаки, при вимкненому файрволлі була пропущена 1 PortScan атака, інші були помічені як інциденти і збережені в базі даних.

ВИСНОВКИ

В результаті виконання атестаційної роботи розроблена система виявлення атак на мережеві ресурси із застосуванням нейромережевих технологій.

У процесі досягнення поставленої мети вирішені завдання:

- проведено аналіз існуючих атак, методів та систем їх виявлення;
- досліджено технології і програмне забезпечення використане для розробки компонентів системи;
- спроектована і розроблена програма збирача даних;
- спроектована і розроблена програма аналізатор;
- спроектован і розроблен веб додаток для перегляду підозрілої активності;
- спроектовані і розроблені дві моделі нейронних мереж.

Моделі здатні аналізувати дані з мережі, навіть якщо дані неповні або перекручені. Кожна модель може проводити аналіз в нелінійній формі. Обидві ці характеристики важливі для використання в мережевих технологіях, де інформація часто піддається випадковим помилкам обладнання.

Обидві моделі нейронних мереж показали точність близьку до 100%, що підтвердилося в ході тестування системи.

Атестаційна робота враховує сучасні тенденції в інформаційних технологіях і є економічно вигідним рішенням.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. J. Allen, A. Christie, W. Fithen, J. McHuge, J. Pickel, E. Stoner,State of Practice of intrusion detection technologies// Technical Report CMU/SEI-99-TR-028. Carnegie Mellon Software Engineering Institute. 2000.

2. Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani, A Detailed Analysis of the KDD CUP 99 Data Set. CISDA 2009.

3. Hee-su Chae, Byung-oh Jo, Sang-Hyun Choi, Twae-kyung Park. Feature Selection for Intrusion Detection using NSL-KDD

4. Downtime, Outages and Failures – Understanding Their True Costs [Електроний ресурс]. – Режим доступу: https://www.evolven.com/-blog/downtime-outages-and-failures-understanding-their-true-costs.html

5. Bustamante, Fabián, et al. «A methodological proposal concerning to the management of information security in Industrial Control Systems.» 2016 IEEE Ecuador Technical Chapters Meeting (ETCM). IEEE, 2016.

6. Y. Zhang, S. Singh, S. Sen, N. Duffiend, and C. Lund “Online identification of hierarchical heavy hitters: algorithms, evaluation, and applications”[text] in Proc. of the 4th ACM SIGCOMM conference on Internet measurement (IMC), 2004, pp. 101-114.

7. Мерков А.Б. Распознавание образов. Построение и обучение вероятностных моделей. 2014. – 238 с.

8. Mohiuddin Ahmed, Abdun Naser Mahmood, Jiankun Hu // A survey of network anomaly detection techniques [text] -Journal of Network and Computer Applications Volume 60, January 2016. – P. 21.

9. Analysis of the Cyber Attack on the Ukrainian Power Grid: March 2016, SANS (ICS) & E-ISAC. – Pp. 11–19.

10. V. Kumar, —“Parallel and distributed computing for cybersecurity”, [text] IEEE Distributed Systems Online, vol. 6, no. 10, 2005.

11. Kurt, Mehmet Necip, Yasin Yılmaz, and Xiaodong Wang. «Real-time

detection of hybrid and stealthy cyber-attacks in smart grid.» IEEE Transactions on Information Forensics and Security 14.2 (2018): 498-513.

12. Varun Chandola, Arindam Banerjee, Vipin Kumar // Anomaly Detection: A Survey [text] -ACM Computing Surveys, September 2009. p. 7

13. Сайт системи виявлення атак IDS“Snort” [Електроний ресурс].– Режим доступу: https://www.snort.org/

14. Eckmann S.T., Vigna G., Kemmerer R.A. STATL: An Attack Language for State-ased Inrusion Detection. –Dept. of Computer Science, University of California, Santa Barbara, 2000.

15. Ozturk A. OSSEC-HIDS Capabilities, Architecture and plans // Presentation at the 5thLinux and Free Software Festival, Ankara, Turkey, 2006.

16. Paxson V. Bro: A System for Detecting Network Intuders in Realime // Computer Networks. 1999. 31 (23-24). P. 2435-2463

17. Roesch M. Snort Users Manual, Snort Release: 2.4, 2007. –snort.org.

18. Balasubramaniyan J., Garcia-Fernandez J.O., Spafford E.H., Zamboni D. An Architecture for Intrusion Detection using Autonomous Agents, Departament of Computer Sciences, Purdue University; Coast TR 98-05; 1998.

19. Yaroshkin F. SnortNet –A Ditributed Intrusion Detection System // IVT-1/95, Kyrgyz Russian Slavic University, Bishkek, Kyrgystan, June.

20. Голубничий Д.Ю. Оцінка складності методів виявлення атак / А.В. Власов, В.Ф. Третяк, Д.М. Запара, І.Ю. Жукова // Scientific Collection «InterConf», (37): with the Proceedings of the 1st International Scientific and Practical Conference «Recent Scientific Investigation» (December 6-8, 2020). – Oslo, Norway: Dagens naeringsliv forlag, 2020. – Pp. 1061 – 1070.

21. Голубничий Д.Ю. Технології аудиту кібербезпеки інформаційних систем / Д.Ю. Голубничий, О.В. Коломійцев, В.Ф. Третяк, С.Г. Рязанін // Scientific Collection «InterConf», (36): with the Proceedings of the 7th International Scientific and Practical Conference «Challenges in Science of Nowadays» (November 26 - 28, 2020) in Washington, USA: EnDeavours Publisher, 2020. – Pp. 333 – 342.