Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
(повна назва)

Кафедра Безпеки інформаційних технологій
(повна назва)

АТЕСТАЦІЙНА РОБОТА

Пояснювальна записка

рівень вищої освіти другий (магістерський)

Методи виявлення аномалій трафіку за допомогою нейронних мереж
(тема)

Виконав: Труш В.Є.
(прізвище, ініціали)

студент 2 курсу, групи БІКСм-18-1

Спеціальність 125 Кібербезпека
(код і повна назва спеціальності)

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

Освітня програма «Безпека інформаційних і комунікаційних систем»
(повна назва освітньої програми)

Керівник доцент Федюшин О.І.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Халімов Г.З.
(підпис) (прізвище, ініціали)

2019 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
(повна назва)
Кафедра Безпеки інформаційних технологій
(повна назва)
Рівень вищої освіти другий (магістерський)
Спеціальність 125 Кібербезпека
(код і повна назва)
Тип програми освітньо-професійна
(освітньо-професійна, або освітньо-наукова)
Освітня програма «Безпека інформаційних і комунікаційних систем»
(повна назва)

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

ЗАВДАННЯ
НА АТЕСТАЦІЙНУ РОБОТУ

студентові *Трушу Владиславу Євгенійовичу*
(прізвище, ім'я, по батькові)

1. Тема роботи *Методи виявлення аномалій трафіку за допомогою нейронних мереж* затверджена наказом по університету від "04" листопада 2019 р. № 1649Ст
2. Термін подання студентом роботи до екзаменаційної комісії _____
3. Вихідні дані до роботи
1. *База даних NSL-KDD.*
2. Bernd Fritzke. A Growing Neural Gas Network Learns Topologies.
4. Перелік питань, що потрібно опрацювати в роботі
1. Розглянути та проаналізувати сучасні методи моделювання атак в ком'ютерних мережах
2. Розглянути та проаналізувати методи виявлення аномалій трафуку в локальних обчислювальних мере
3. Програмна реалізація нейронного детектору виявлення аномалій мережевого трафіку на основі зростаючого нейронного газу GNG
4. Навчання детектору
5. Тестування детектору
6. Результати досліджень
5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій *Презентаційний матеріал у вигляді слайдів*

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Отримання завдання на атестаційну роботу	04.09.2019	виконано
2.	Аналіз літературних джерел за темою атестаційної роботи	04.09.2019-05.10.2019	виконано
3.	Огляд існуючих методів виявлення аномалій трафіку	06.10.2019-23.10.2019	виконано
4.	Аналіз програмних засобів та їх можливостей щодо виявлення вторгнень та аномалій трафіку	24.10.2019-01.11.2019	виконано
5.	Розробка методу виявлення аномалій трафіку на основі моделі зростаючого нейронного газу (GNG)	01.11.2019-15.11.2019	виконано
6.	Розробка програмного додатку для виявлення аномалій трафіку на основі моделі зростаючого нейронного газу (GNG)	16.11.2019-25.11.2019	виконано
7.	Оформлення роботи	26.11.2019-07.12.2019	виконано
8.	Представлення роботи на здачу	16.12.2019	виконано

Студент _____

(підпис)

Керівник роботи (проекту) _____ _доц. Федюшин О. І.____

(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Атестаційна робота містить: 86 сторінок, 10 таблиць, 14 рисунків, 19 джерел, 1 додаток.

НЕЙРОННА МЕРЕЖА, ТРАФІК, АНОМАЛІЯ, GNG, СВВ.

Об’єкт дослідження – процес виявлення аномалій трафіку в локальних обчислювальних мережах.

Предмет дослідження – методи виявлення аномалій трафіку на основі нейронних мереж.

Мета роботи – побудувати модель нейронного детектору аномалій трафіку та провести її експериментальне дослідження.

Основним завданням роботи є виявлення аномалій трафіку за допомогою нейронного детектору, який може бути включений до складу системи виявлення вторгнень.

В роботі удосконалена модель виявлення аномалій трафіку на основі зростаючого нейронного газу GNG. Запроновано програмну реалізацію детектора та проведена експериментальна оцінка його властивостей.

РЕФЕРАТ

Аттестационная работа содержит: 86 страниц, 10 таблиц, 14 рисунков, 19 источников, 1 приложение.

НЕЙРОННАЯ СЕТЬ, ТРАФИК, АНОМАЛИЯ, GNG, СОВ.

Объект исследования – процес выявления аномалий трафика в локальных вычислительных сетях.

Предмет исследования – методы выявления аномалий трафика на основе нейронных сетей.

Цель работы – построить модель нейронного детектора аномалий трафика.

Основной задачей работы является определение аномалий трафика с помощью нейронного детектора, который может быть внедрен в состав системы обнаружения вторжений

В роботе усовершенствована модель определения аномалий трафика на основе растущего нейронного газа GNG. Предложена программная реализация детектора и проведена экспериментальная оценка его свойств.

ABSTRACT

The attestation work contains 86 pages, 19 bibliographic titles, 10 tables, 14 pictures, 1 appendix.

NEURAL NETWORK, TRAFFIC, ANOMALIES, GNG, IDS.

The object of the research is the process of detecting traffic anomalies in local area networks.

The subject of the research are methods of detecting traffic anomalies based on neural networks.

The main task of the work is to detect traffic anomalies with a neural detector, which can be included in the intrusion detection system.

In attestation work have improved the model of traffic anomaly detection based on the growing neural gas GNG. The software implementation of the detector is offered and the experimental evaluation of its properties is carried out.

ЗМІСТ

УМОВНІ ПОЗНАЧЕННЯ, СИМВОЛИ, ОДИНИЦІ СКОРОЧЕННЯ І ТЕРМІНИ

СВВ	–	Система виявлення вторгнень;
IDS	–	Intrusion Detection System
GNG	–	Growing neural gas
ЛОМ	–	Локальна обчислювальна мережа

ВСТУП

Найважливішим атрибутом нашого часу є глобальна інформаційна інтеграція, заснована на побудові комп'ютерних мереж масштабу підприємства і їх об'єднання за допомогою Інтернету.

Складність логічної і фізичної організації сучасних мереж призводить до об'єктивних труднощів при вирішенні питань управління та захисту мереж. В процесі експлуатації комп'ютерних мереж адміністраторам доводиться вирішувати два головні завдання: діагностувати роботу мережі і підключених до неї серверів, робочих станцій і відповідного програмного забезпечення; захищати інформаційні ресурси мережі від несанкціонованої діяльності, впливів вірусів, тобто забезпечувати їх конфіденційність, цілісність і доступність.

При вирішенні завдань, пов'язаних з діагностикою та захистом мережевих ресурсів, центральним питанням є оперативне виявлення станів мережі, що призводять до втрати повної або часткової її працездатності, знищення, спотворення чи витоку інформації, що є наслідком відмов, збоїв випадкового характеру або результатом отримання зловмисником несанкціонованого доступу до мережевих ресурсів, проникнення мережевих черв'яків, вірусів і інших погроз інформаційної безпеки. Своєчасне виявлення таких станів дозволить вчасно усунути їх причину, а також попередити можливі катастрофічні наслідки.

Для їх виявлення використовується великий спектр спеціалізованих систем. Так, при вирішенні проблем діагностики мереж застосовуються засоби систем управління, аналізатори мережевих протоколів, системи тестування навантаження, системи моніторингу мережі. Проблеми захисту інформаційних ресурсів мереж вирішуються за допомогою міжмережевих екранів (firewall), антивірусів, систем виявлення атак (вторгнень) (СВВ) (Intrusion Detection System, IDS), систем контролю цілісності, криптографічних засобів захисту.

Таким чином, на сьогоднішній день дуже актуальним завданням є пошук більш ефективних методів виявлення неприпустимих подій (аномалій) в роботі мережі, які є наслідком технічних збоїв або несанкціонованих дій.

1.АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аномалії мережевого трафіку

Стабільна і безперебійна робота мережі підприємства є запорукою успішного функціонування бізнес процесу, в той час як навіть мінімальний простій означає втрату прибутку і репутаційні витрати [5]. Для вирішення цих завдань адміністратор повинен мати в своєму розпорядженні інструменти для виявлення аномалій в мережевому трафіку.

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

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

Виявлення аномалій це спроба знайти якусь поведінку піднаглядного об'єкта, шаблон, який не відповідає очікуваної або нормальної поведінки. Безперечна важливість проблеми виявлення аномалій пов'язана в першу чергу з тим, що аномалії можуть серйозно впливати на те середовище, в якому сталася

аномалія. Наприклад, в комп'ютерних мережах аномальна поведінка трафіку може означати, що заражений якимось вірусом комп'ютер відсилає важливу (ймовірно, цінну) інформацію «назовні», до неавторизованого комп'ютера.

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

- Несправності мережевого обладнання;
- Випадкові або навмисні дії з боку легітимних користувачів (некоректні дії користувача через низьку кваліфікацію);
- Невірна робота додатків (помилки в програмному коді);
- Дії зловмисників (вірусна атака або інша шкідлива діяльність);
- Якісні зміни складу призначеного для користувача ПО та інше.

У широкому сенсі, мережеві аномалії можна поділити на дві основні категорії: пов'язані з продуктивністю і пов'язані з безпекою. У наступному розділі буде представлена більш повна класифікація аномалій, технік і методів їх виявлень.

1.2 Типи аномалій і техніки виявлень

Як зазначено в попередньому підрозділі, мережеві аномалії можна поділити на пов'язані з продуктивністю і пов'язані з безпекою.

Існують три основні типи аномалій, пов'язаних з безпекою:

- Точкові аномалії (Point Anomalies). Точковою може називатися аномалія однієї одиниці даних щодо всього об'єму даних;
- Аномалія в контексті (Contextual Anomalies). Контекстна аномалія та, яка є аномалією тільки в певних умовах і тільки в них;
- Групова аномалія (Collective Anomalies). Аномалія на певній групі одиниць інформації. При цьому кожна одиниця може не бути аномальної сама по собі.

Нижче перераховані три широко використовуваних категорії технік виявлення аномалій:

- Непідконтрольні методи виявлення аномалії (Unsupervised anomaly detection techniques). Системи, що працюють з використанням подібних технік, не вимагають заздалегідь підготовлених даних, а значить, найбільш широко застосовуються. Передбачається, що нормальні дані в наборі даних зустрічаються набагато частіше, ніж аномальні. У випадках якщо припущення невірно системи, побудовані з використанням подібних технік, страждають від частих помилкових спрацьовувань;
- Керовані методи виявлення аномалії (Supervised anomaly detection techniques). У цій техніці передбачається, що є два класи сутностей для нормальної і аномальної поведінки. Як правило, будується модель для нормального і аномального класів, після чого дані, до того не вивчені порівнюються з обома класами, щоб з'ясувати до якого вони належать;
- Напівконтрольні методи виявлення аномалії (semi-supervised anomaly detection techniques). Техніка має на увазі, що вивчені дані є тільки для класу "норма". З огляду на, що набагато простіше побудувати тільки модель нормального поведінки (з огляду на те, що не можна передбачити всі можливі аномалії) ця техніка більш широко застосовується, ніж контрольований метод.

У свою чергу різноманітність методів виявлення найбільш наочно представляється можливим представити у вигляді схеми, рисунку 1.1.

Слід зауважити, що багато сучасних рішення не обмежуються суворим використанням якогось одного методу. Замість цього використовуються гібридні методи (наприклад, оповіщення адміністратора і застосування до трафіку якогось дії).

1.3 Система виявлення вторгнень

Важливе місце серед інструментів мережевого адміністратора для виявлення мережевих аномалій займають Системи виявлення вторгнень. Система виявлення вторгнень (СВВ, IDS – Intrusion Detection System) – програмний або апаратний засіб, призначений для виявлення фактів несанкціонованого доступу (вторгнення або мережевої атаки) в комп'ютерну систему або мережу [2].

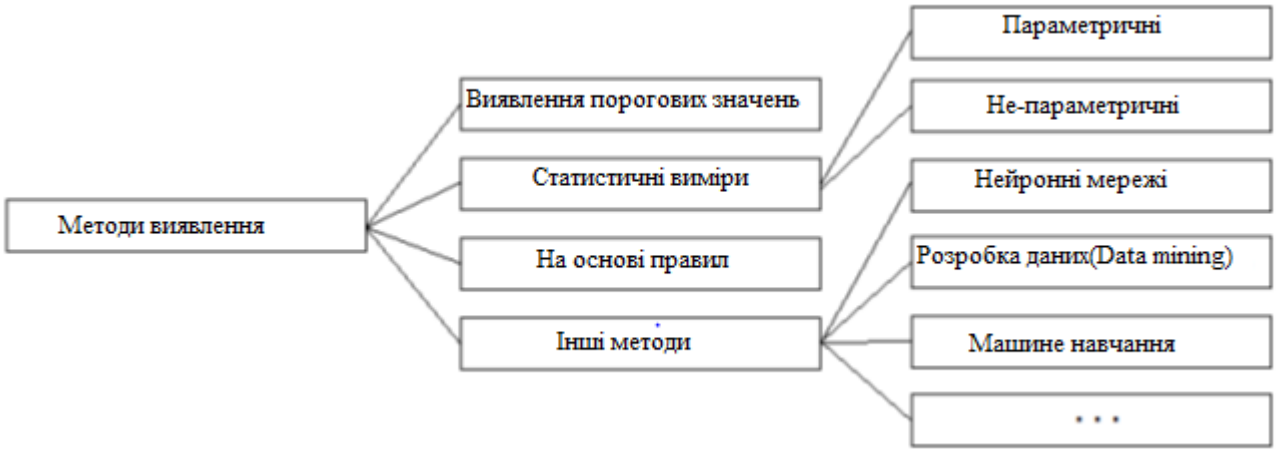

Рисунок 1.1 – Класифікація методів виявлення аномалій

Системи виявлення вторгнень забезпечують виявлення:

- мережевих атак проти вразливих сервісів;
- атак, спрямованих на підвищення прав користувачів;
- неавторизованого доступу до важливих файлів;
- дій шкідливого ПЗ.

Використання СВВ допомагає досягнути наступної мети:

- виявити вторгнення або мережеві атаки;
- забезпечити належний контроль якості адміністрування, особливо у великих і складних мережах;

- спрогнозувати можливі майбутні атаки і виявити вразливості для запобігання їх подальшого розвитку;
- отримати корисну інформацію про проникнення, для відновлення і налаштування конфігурації мережі;
- визначити розташування джерела атаки по відношенню до локальної мережі (зовнішні або внутрішні атаки).

Схема типової СВВ зображена на рисунку 1.2.

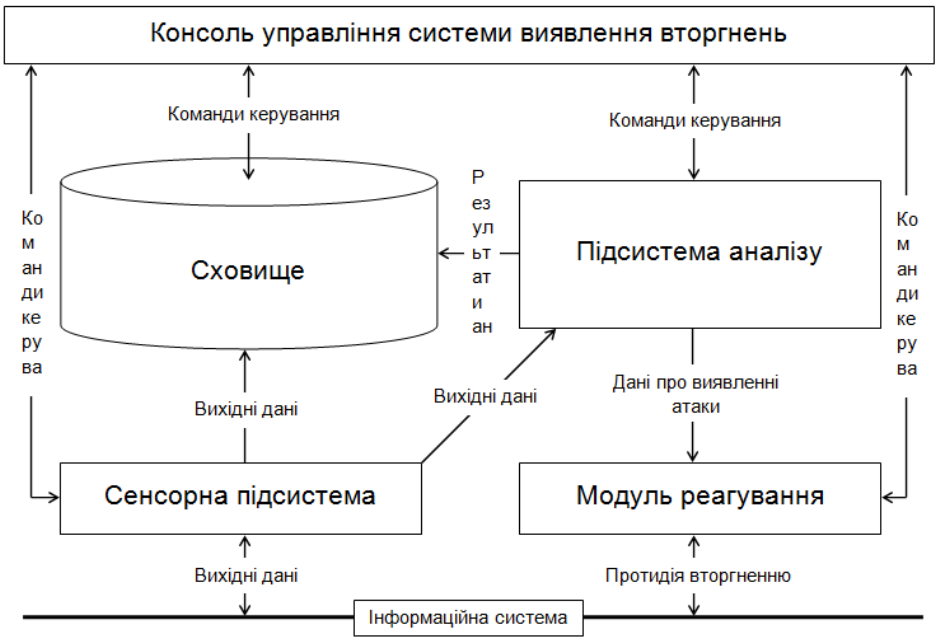

Рисунок 1.2 – Схема типової системи виявлення вторгнень

Основними компонентами систем виявлення вторгнень є сенсорна підсистема, підсистема аналізу, сховище, консоль управління та модуль реагування.

Сенсорна підсистема, призначена для збору подій, пов'язаних з безпекою мережі або системи, що захищається.

Підсистема аналізу, призначена для виявлення мережевих атак і підозрілих дій.

Сховище, в якому накопичується база первинних подій і результати аналізу.

Консоль управління, що дозволяє конфігурувати СВВ, спостерігати за станом мережі або інформаційної системи та СВВ, переглядати виявлені підсистемою аналізу інциденти несанкціонованих вторгнень.

Модуль реагування, який встановлений в системах активної протидії відповідає за виконання інструкцій по протидії несанкціонованому вторгненню в мережу або систему.

1.4 Методи виявлення вторгнень

Серед методів, використовуваних в підсистемі аналізу сучасних СВВ, можна виділити два напрямки: перший спрямований на виявлення аномалій в системі, що захищається, а інший - на пошук зловживань [1]. Кожний з цих напрямків має свої переваги і недоліки, тому в більшості існуючих СВВ застосовуються комбіновані рішення, засновані на синтезі відповідних методів. Ідея методів, що використовуються для виявлення аномалій, полягає в тому, щоб розпізнати, чи є процес, що викликав зміни в роботі системи, діями зловмисника. Методи пошуку аномалій наведені в таблицях 1.1 і 1.2.

Виділяються дві групи методів: з контрольованим навчанням («навчання з учителем»), і з неконтрольованим навчанням («навчання без учителя»). Основна відмінність між ними полягає в тому, що методи контрольованого навчання використовують фіксований набір параметрів оцінки і якісь апріорні відомості про значення параметрів оцінки. Час навчання фіксований. У неконтрольованому ж навчанні безліч параметрів оцінки може змінюватися з плином часу, а процес навчання відбувається постійно.

Таблиця 1.1 - Виявлення аномалії - контрольоване навчання («навчання з учителем»)

Методи виявлення	Опис методу
Моделювання правил	Система виявлення протягом процесу навчання формує набір правил, що описують нормальну поведінку системи. На стадії пошуку несанкціонованих дій система застосовує отримані правила і в разі недостатньої відповідності сигналізує про виявлення аномалії.
Описова статистика	Навчання полягає в зборі простої описової статистики безлічі показників системи, що захищається в спеціальну структуру. Для виявлення аномалій обчислюється «відстань» між двома векторами показників - поточними і збереженими значеннями. Стан в системі вважається аномальним, якщо отримана відстань досить велика.
Нейронні мережі	Структура застосовуваних нейронних мереж різна. Але у всіх випадках навчання виконується даними, що представляють нормальну поведінку системи. Отримана навчена нейронна мережа потім використовується для оцінки аномальність системи. Вихід нейронної мережі говорить про наявність аномалії.

Мета другого напрямку (виявлення зловживань) – пошук послідовностей подій, визначених (адміністратором безпеки або експертом під час навчання СВВ) як етапи реалізації вторгнення. Методи пошуку зловживань наведені в

таблиці 1.3. У теперішній час виділяються лише методи з контрольованим навчанням.

Реалізовані в даний час в СВВ методи засновані на загальних уявленнях теорії розпізнавання образів. Відповідно до них для виявлення аномалії на основі експертної оцінки формується образ нормального функціонування інформаційної системи. Цей образ виступає як сукупність значень параметрів оцінки. Його зміна вважається проявом аномального функціонування системи. Після виявлення аномалії і оцінки її ступеня формується судження про природу змін: чи є вони наслідком вторгнення або допустимим відхиленням. Для виявлення зловживань також використовується образ (сигнатура), однак тут він відображає заздалегідь відомі дії атакуючого.

Таблиця 1.2 - Виявлення аномалії - неконтрольоване навчання («навчання без учителя»)

Методи виявлення	Опис методу
Моделювання множини станів	Нормальна поведінка системи описується в вигляді набору фіксованих станів і переходів між ними. Тут стан є не що інше як вектор певних значень параметрів вимірювань системи.
Описова статистика	Аналогічний відповідному методу в контрольованому навчанні.

1.4.1Аналіз методу виявлення аномалій

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

1) накопичення найбільш характерною статистичної інформації для кожного параметра оцінки;

2) навчання нейронних мереж значеннями параметрів оцінки;

Таблиця 1.3 - Виявлення зловживань - контрольоване навчання («навчання з учителем»)

Методи виявлення	Опис методу
Моделювання станів	Вторгнення представляється як послідовність станів, де стан - вектор значення параметрів оцінки системи, що захищається. Необхідна і достатня умова наявності вторгнення - присутність цієї послідовності. Виділяють два основних способи подання сценарію вторгнень: 1) у вигляді простого ланцюжка подій; 2) з використанням мереж Петрі, де вузли - події.
Моделювання правил	Простий варіант експертних систем.
Синтаксичний аналіз	Системою виявлення виконується синтаксичний розбір з метою виявлення певної комбінації символів, що передаються між підсистемами і системами захищається комплексу.

Легко помітити, що у виявленні дуже значну роль відіграє безліч параметрів оцінки. Тому в виявленні аномалій одним із головних завдань є вибір оптимального безлічі параметрів оцінки.

Іншим, не менш важливим завданням є визначення загального показника аномальності. Складність полягає в тому, що ця величина повинна характеризувати загальний стан «аномальності» в системі, що захищається.

1.4.2 Вибір оптимальної сукупності ознак оцінки системи, що підлягає захисту

У теперішній час використовується евристичне визначення (вибір) безлічі параметрів вимірювань системи, що захищається, використання якої має дати найбільш ефективне і точне розпізнавання вторгнень. Складність вибору безлічі можна пояснити тим, що складові цієї підмножини залежать від типів вторгнень, що виявляються. Тому одна і та ж сукупність не може буде адекватною для всіх типів вторгнень.

Будь-яку систему, що складається зі звичних апаратних і програмних засобів, можна розглядати як унікальний комплекс зі своїми особливостями. Це є поясненням можливості пропуску специфічних для системи, що захищається , вторгнень тими СВВ, які використовують один і той же набір параметрів оцінки. Більш вдале рішення - визначення необхідних параметрів оцінки в процесі роботи. Труднощі ефективного динамічного формування параметрів оцінки полягає в тому, що розмір області пошуку експоненціально залежить від потужності початкової безлічі. Якщо є початковий список з N параметрів, актуальних вторгнень, що передбачаються, то кількість підмножин цього списку становить 2^N. Тому не представляється можливим використання алгоритмів перебору для знаходження оптимальної множини. Одне з можливих рішень - використання генетичного алгоритму

1.4.3Нейронні мережі

Інший спосіб представлення «образу» нормальної поведінки системи - навчання нейронної мережі значенням параметрів оцінки.

Навчання нейронної мережі здійснюється послідовністю інформаційних одиниць (далі команд), кожна з яких може перебувати на більш абстрактному рівні в порівнянні з використовуваними параметрами оцінки. Вхідні дані мережі складаються з поточних команд і минулих W команд, які обробляються нейронною мережею з метою передбачення наступних команд; W також називають розміром вікна. Після того як нейронна мережа навчена безліччю

послідовних команд системи, що захищається або однієї з її підсистем, мережа являє собою «образ» нормальної поведінки.

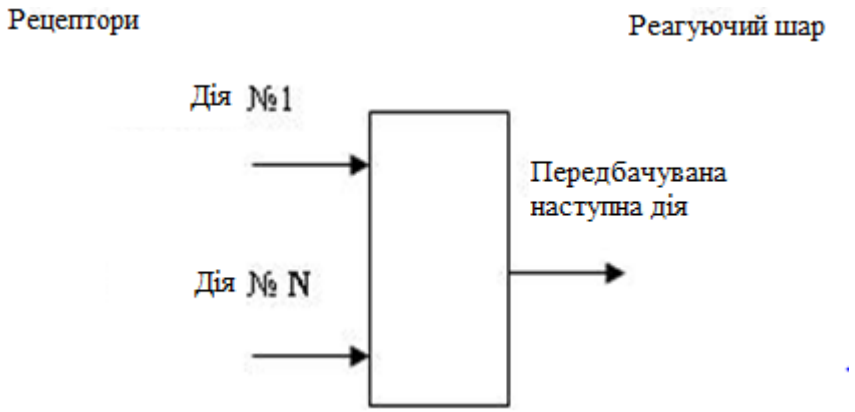

Рисунок. 1.3 - Концептуальна схема нейронних мереж СВВ

Процес виявлення аномалій є визначенням показника неправильно передбачених команд, тобто фактично виявляється відмінність у поведінці об'єкта. На рівні рецептора (рис. 1.3) стрілки показують вхідні дані останніх **W** команд, виконаних користувачем. Вхідний параметр задає кілька значень або рівнів, кожен з яких унікально визначає команду. Вихідний шар, що реагує, складається з одного багаторівневого, який передбачає наступну можливу команду користувача.

Недоліки:

1) топологія мережі і ваги вузлів визначаються тільки після величезного числа проб і помилок;

2) розмір вікна - ще одна величина, яка має величезне значення при розробці; якщо зробити вікно маленьким то мережу буде не досить продуктивною, надто великим - буде страждати від недоречних даних.

Переваги:

1) успіх даного підходу не залежить від природи вихідних даних;

2) нейронні мережі легко справляються з зашумленими даними;

3) автоматично враховуються зв'язки між різними вимірами, які, без сумніву, впливають на результат оцінки.

1.4.4 Генерація шаблонів

Представлення «образу» в даному випадку ґрунтується на припущенні про те, що поточні значення параметрів оцінки можна пов'язати з поточним станом системи. Після цього функціонування представляється у вигляді послідовності подій або станів.

Тимчасові правила, які характеризують сукупності значень параметрів оцінки (далі шаблон) нормальної (не аномальної) роботи. Ці правила формуються індуктивно і замінюються більш «хорошими» правилами динамічно під час навчання. Під «хорошими правилами» розуміються правила з більшою ймовірністю їх появи і з великим рівнем унікальності для системи , що захищається. Для прикладу розглянемо наступне правило:

$$E1 - E2 - E3 \rightarrow (E4 = 95\%, E5 = 5\%), \quad (1.1)$$

де *E1… E5* – події безпеки.

Це твердження, засноване даних, що раніше спостерігалися, говорить про те, що для послідовності шаблонів встановилася наступна залежність: якщо має місце *E1* і далі *E2* та *E3*, то після цього ймовірність появи *E4* 95% та *E5* – 5%.

Саме безліч правил, створюваних індуктивно під час спостереження роботи користувача, становить «образ». Аномалія реєструється в тому випадку, якщо послідовність подій, що спостерігається відповідає лівій частині правила виведеного раніше, а події, які мали місце в системі після цього, значно відрізняються від тих, які мають наступити за правилом.

Основний недолік даного підходу полягає в тому, що невпізнанні шаблони поведінки можуть бути не прийняті за аномальні через те, що вони не відповідають ні однієї з лівих частин всіх правил.

Даний метод досить ефективно визначає вторгнення, так як приймаються до уваги:

1) залежності між подіями;

2) послідовність появи подій.

Переваги методу:

1) найкраща обробка користувачів з великим коливанням поведінки, але з чіткою послідовністю шаблонів;

2) можливість звернути увагу на деякі важливі події безпеки, а не на всю сесію, яка позначена як підозріла;

3) найкраща чутливість до виявлення порушень: правила містять у собі семантику процесів, що дозволяє набагато простіше помітити зловмисників, які намагаються навчити систему в своїх цілях.

1.4.5 Аналіз методу виявлення зловживань

Використання тільки методів виявлення аномалій не гарантує виявлення всіх порушень безпеки, тому в більшості СВВ існує технології розпізнавання зловживань. Виявлення вторгнень-зловживань ґрунтується на прогностичному визначенні атак і подальшим спостереженням за їх появою [2]. На відміну від виявлення аномалії, де образ - це модель нормального поведінки системи, при виявленні зловживання він необхідний для подання несанкціонованих дій зловмисника. Такий «образ» стосовно виявлення зловживань називається сигнатурою вторгнення. Формується сигнатура на основі тих самих вхідних даних, що і при виявленні аномалій, тобто на значеннях параметрів оцінки. Сигнатури вторгнень визначають оточення, умови і спорідненість між подіями, які призводять до проникнення в систему або будь-яким іншим зловживанням. Вони корисні не тільки при виявленні вторгнень, але і при виявленні спроб здійснення незаконних дій. Частковий збіг сигнатур може означати, що в системі, що захищається мала місце спроба вторгнення.

1.4.6. Продукційні / Експертні системи

Головна перевага використання продукційних систем полягає в можливості поділу причин і рішень виникаючих проблем.

Приклади використання таких систем в СВВ описані досить широко. Така система кодує інформацію про вторгнення в правилах виду if (якщо) причина

then (то) рішення, причому при додаванні правил причина відповідає події (ям), що реєструються підсистемою збору інформації СВВ. У частині (if) правила кодуються умови (причини), необхідні для атаки. Коли всі умови в лівій частині правила задоволені, виконується дія (рішення), заданий в правій його частині.

Основні проблеми додатків, що використовують даний метод, які зазвичай виникають при їх практичному застосуванні:

1) недостатня ефективність при роботі з великими обсягами даних;

2) важко врахувати залежну природу даних параметрів оцінки.

При використанні продукційних систем для виявлення вторгнень можна встановити символічне прояв вторгнення за допомогою наявних даних.

Труднощі:

1) відсутність вбудованої або природної обробки порядку послідовностей в аналізованих даних. База фактів, відповідна лівій частині «продукції», використовується для визначення правій частині. У лівій частині продукційного правила всі елементи об'єднуються за допомогою зв'язку «і»;

2) вбудована експертиза може бути вдалою лише в тому випадку, якщо модельовані навички адміністратора безпеки не суперечливі. Це практичне міркування, можливо, стосується браку централізованності зусиль експертів безпеки в напрямку створення вичерпних множин правил;

3) виявляються лише відомі уразливості;

4) існує певний програмний інжиніринг, пов'язаний з установкою (підтримкою) баз знань. При додаванні або видаленні будь-якого з правил повинна змінюватися інша безліч правил;

5) об'єднання різних вимірів вторгнень і створення пов'язаної картини вторгнення призводить до того, що приватні причини стають невизначеними. Обмеження продукційних систем, в яких використовується невизначена причина, досить добре відомі.

1.4.7 Спостереження за натисканням клавіш

Для виявлення атак в даній технології використовується моніторинг за натисканням користувача на клавіші клавіатури. Основна ідея - послідовність натиснень користувача задає патерн атаки. Недоліком цього підходу є відсутність досить надійного механізму перехоплення роботи з клавіатурою без підтримки операційної системи, а також велика кількість можливих варіантів представлення однієї і тієї ж атаки. Крім того, без семантичного аналізатора натискань різного роду псевдоніми команд можуть легко зруйнувати цю технологію. Оскільки вона спрямована на аналіз натискань клавіш, автоматизовані атаки, які є результатом виконання програм зловмисника, також можуть бути не виявлені.

1.4.8 Методи, що базуються на моделюванні поведінки зловмисника

Одним з варіантів виявлення зловживання є метод об'єднання моделі зловживання з очевидними причинами. Його суть полягає в наступному: є база даних сценаріїв атак, кожна з яких об'єднує послідовність поводжень, що становлять атаку. У будь-який момент часу існує можливість того, що в системі має місце одне з цих підмножин сценаріїв атак. Робиться спроба перевірки припущення про їхню наявність шляхом пошуку інформації в записах аудиту. Результатом пошуку є якась кількість фактів, достатню для підтвердження або спростування гіпотези. Перевірка виконується в одному процесі, який отримав назву антісіпатор. Антісіпатор, ґрунтуючись на поточній активній моделі, формує наступну можливу множину поводжень, яке необхідно перевірити в записах аудиту, і передає їх планувальникам. Планувальник визначає, як передбачувана поведінка відбивається в записах аудиту і трансформує їх у системно-аудітозалежний вираз. Ці вирази повинні складатися з таких структур, які можна було б просто знайти в записах аудиту, і для яких була б досить висока ймовірність появи в записах аудиту.

У міру того як підстави для підозр деяких сценаріїв накопичуються, а для інших - знижуються, список моделей активностей зменшується. Обчислення

причин вбудовано в систему і дозволяє оновлювати ймовірність появи сценаріїв атак в списку моделей активності.

Переваги:

1) з'являється можливість зменшити кількість істотних обробок, необхідних для одного запису аудиту; спочатку спостерігаються більш «грубі» події в пасивному режимі, і далі, як тільки одне з них виявлено, спостерігаються більш точні події;

2) планувальник забезпечує незалежність подання від форми даних аудиту.

Недоліки:

1) при застосуванні даного підходу у особи, відповідальної за створення моделі виявлення вторгнення, з'являється додаткове навантаження, пов'язане з призначенням змістовних і точних кількісних характеристик для різних частин графічного представлення моделі;

2) ефективність цього підходу була продемонстрована створенням програмного прототипу; з опису моделі не ясно, як поведінки можуть бути ефективно складені в планувальнику, і який ефект це матиме на систему під час роботи;

3) цей підхід доповнює, але не замінює підсистему виявлення аномалій.

У зв'язку тим, що число нових, раніше невідомих атак зростає з кожним роком, кращим варіантом для практичного використання є адаптивні методи виявлення вторгнень. З результатів аналізу, представлених у розділі 1, випливає, що до адаптивних методів виявлення вторгнень серед розглянутих у роботі відносяться: статистичний аналіз, графи сценаріїв атак, експертні системи, нейронні мережі, імунні мережі, кластерний аналіз і методи, засновані на вивченні поведінки. Методи, засновані на побудові графа сценаріїв атак і експертних системах, практично не використовуються в існуючих системах виявлення й запобігання вторгнень у 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 адаптивних методів виявлення вторгнень, що залишилися, метод статистичного аналізу й аналогічний йому по суті метод кластерного аналізу мають високу ймовірність неправильного спрацьовування. Також неможливість зміни характеристик об'єкта в ході функціонування методів може призвести як до неправильних спрацьовувань, так і до пропущених атак.

У результаті проведеного аналізу було встановлено, що для забезпечення захисту систем розподіленої обробки даних найбільш кращими є методи, що функціонують на принципах нейромережного аналізу. Основними перевагами даних методів є: адаптивність, невисока обчислювальна складність, а також при використанні належної навчальної вибірки можливість забезпечувати надійне функціонування системи виявлення й запобігання вторгнень, виявляючи як відомі, так і невідомі атаки різних класів.

1.5 Розробка моделей загроз і порушника

Визначення моделей порушника й загроз є головним завданням при проектуванні системи захисту, тому що саме це дозволяє визначити, що повинна забезпечувати система захисту інформації.

1.5.1 Розробка моделі порушника.

Модель порушника необхідна для уточнення типів зловмисників, здатних провести несанкціонований доступ (НСД) до розглянутої автоматизованої системи (АС). Порушники класифікуються по рівнях можливостей, що надаються ним штатними засобами автоматизованих систем і засобами обчислювальної техніки, а також способам фізичного доступу до АС. Визначається чотири основні рівні по можливостях і два по способу доступу.

Класи порушників по рівнях можливостей:

- перший рівень: запуск завдань (програм) фіксованого набору, що реалізують заздалегідь передбачені функції по обробці інформації;

- другий рівень: створення й запуск власних програм з новими функціями по обробці інформації;

- третій рівень: можливість керування функціонуванням АС, тобто вплив на базове програмне забезпечення системи та на склад і конфігурацію її устаткування;

– четвертий рівень: увесь обсяг можливостей осіб, що здійснюють проектування, реалізацію та ремонт технічних засобів АС, аж до включення до складу засобів обчислювальної техніки власних технічних засобів з новими функціями по обробці інформації.

У кожному своєму рівні порушник є фахівцем високої кваліфікації, знає все про АС і про систему та засоби її захисту.

Класи порушників по способах фізичного доступу до ЛОМ:

- зовнішні, тобто, що не мають доступу в контрольовану зону АС, що реалізують загрози із зовнішніх мереж зв'язку загального користування та (або) мереж міжнародного інформаційного обміну;

- внутрішні, тобто, що мають доступ до АС, включаючи користувачів АС, що реалізують загрози безпосередньо в АС.

Будемо вважати, що для неприпустимості діяльності внутрішніх порушників в АС реалізується необхідний комплекс організаційних і режимно-технічних заходів.

Порушник може використовувати наступні засоби атак на АС:

- штатні засоби АС;

- засобу перехоплення побічних електромагнітних випромінювань та наводок (ПЕМВН), що супроводжують функціонування технічних засобів АС;

- засоби перехоплення й обробки інформації в безпровідних каналах зв'язку, що перебувають за межами контрольованої зони розміщення АС;

- електронні пристрої негласного одержання інформації, можливо впроваджені в технічні засоби АС.

У загальному випадку під загрозою для системи захисту інформації або об'єкта розуміється потенційно можлива дія або подія, реалізація якої може

принести збиток інтересам об'єкта, що захищається, або його власникові. Визначення списку можливих загроз захисту інформації складається з метою формування повного опису вимог по захисту інформації до формованої системи захисту. Також перелік загроз для об'єкта із вказівкою ймовірності їх реалізації необхідний для аналізу можливих ризиків для системи, що захищається, і формулювання вимог до використовуваних методів і способів захисту інформації в системі забезпечення інформаційної безпеки автоматизованої системи. Даний список повинен бути складений з урахуванням класифікації загроз і уразливостей, а також ризиків по ряду ознак. Кожна з таких ознак може відображати одну з виділених вимог до систем захисту інформації. При цьому для кожної ознаки можуть бути деталізовані загрози.

1.5.2 Розробка моделі загроз

Модель загроз пропонується розглядати виходячи з базової моделі загроз при обробці персональних даних у ЛОМ, що має підключення до мереж загального користування типу Інтернет. Можливі наступні загрози:

- загрози від витоку інформації по технічних каналах;
- загрози від НСД до даних, оброблюваних в АС.

Загрози першого роду в явному виді пов'язані з використанням зовнішніми порушниками можливостей технічної розвідки. Дані загрози, пов'язані з наявністю в обладнанні технічних закладок, фактично не зустрічаються, тому що вбудовування закладок у масове вільно розпосюджуване обладнання не допускається фірмами виробниками через можливу втрату іміджу.

Загрози другого роду можливі до реалізації як у провідних системах зв'язку, що мають підключення до загальнодоступних мереж, так і в безпровідних ЛОМ і містять у собі:

1) загрози сканування й аналізу мережного трафіка, призначені для одержання паролів, мережних адрес робочих станцій, відкритих портів і служб, відкритих з'єднань і т.п.;

2) загрози сканування й аналізу мережного трафіка з перехопленням обміну усередині бездротової ЛОМ організації, а також переданої/прийнятої інформації в зовнішню мережу або з неї;

3) загрози НСД за рахунок підміни довіреного користувача, впровадження неправильного об'єкта мережі або нав'язування неправильного маршруту мережі;

4) загрози перевантаження мережі й появи не штатних ситуацій типу відмови в обслуговуванні або доступності мережі;

5) загрози вилученого запуску не санкціонованих додатків;

6) загрози впровадження в мережі вірусів і/або шкідливих програм.

У цьому випадку довіреним об'єктом вважається об'єкт мережі − міжмережний екран, маршрутизатор, комп'ютер і т.п. − легально підключений усередині мережі.

Враховуючи даний факт можна відзначити, що з перерахованих вище загроз 2,3 і 4 є визначальними, а 1, 5 і 6, що фактично забезпечують їх реалізацію.

При визначенні класів захищеності інформаційних систем особливу увагу слід звертати на три показники:

- конфіденційність (неправомірний доступ, копіювання, поширення);
- цілісність (неправомірне знищення або модифікування);
- доступність (неправомірне блокування).

Таким чином, основними загрозами для мережі, що порушують конфіденційність, цілісність і доступність інформації є:

- загрози сканування й аналізу мережного трафіка;
- загрози НСД;
- загрози перевантаження мережі й появи не штатних ситуацій.

Відповідно, при побудові системи захисту необхідно в першу чергу виключити можливість здійснення даних загроз.

Ця обставина частково виконується за рахунок впровадження в мережі ефективних політик контролю трафіку та використання систем виявлення та попередження вторгнень.

2. ПОРІВНЯННЯ ІСНУЮЧИХ СВВ

2.1 Оляд та аналіз існуючих СВВ

Наведемо короткий опис найбільш поширинних в даний час СВВ. Для кожної системи описується її архітектура, яка використовується платформа і деякі індивідуальні особливості

2.1.1 Bro

Система Bro [14,15] є розробкою Національної лабораторії Лоуренса Берклі Каліфорнійського університету, Берклі, США. Система є відкритою і поширюється за власною відкритою ліцензією у вихідних текстах і бінарних пакетах для ряду UNIX-платформ. Система призначена для пасивного відстеження трафіка і пошуку підозрілої активності. Виявлення атак виконується на декількох рівнях: вхідний мережевий трафік розбирається для виявлення семантики рівня додатків, після чого отримана траса подій прикладного рівня аналізується набором подієво-орієнтованих аналізаторів і порівнюється з шаблонами атак.

Bro використовує спеціалізовану мову управління політиками, що дозволяє підлаштовувати поведінку системи під захисну систему відповідно до поточних зовнішніх умов. При виявленні атаки система може виконати різні дії - записати повідомлення в журнал, оповістити оператора, виконати команди операційної системи.

Призначенням системи є високошвидкісне виявлення атак на мережевих каналах з високою пропускною здатністю (1 Гбіт / с). Архітектурно Bro можна розділити на три основних компонента: бібліотека libpcap для захоплення пакетів, модуль генерації подій і інтерпретатор сценаріїв.

Інтерпретатор сценаріїв виконує аналізатори подій, написані на мові Bro. Даний набір сценаріїв є політикою безпеки мережі, який визначає реакцію

системи на різні події. Сценарій може генерувати повідомлення, а також виконувати довільні команди операційної системи, тобто реагувати на атаки. До складу системи входить утиліта snort2bro, яка транслює сигнатури Snort в сценарії Bro. Крім трансляції, дана утиліта виконує оптимізацію сигнатур під аналізатор Bro.

2.1.2 OSSEC

OSSEC HIDS є відкритою вузловий системою виявлення атак [13]. В її завдання входить: аналіз журналів, контроль цілісності, виявлення закладок, повідомлення про атаки і активна реакція на атаки.

Система може бути встановлена як в одиночній конфігурації на одному вузлі, так і в розподіленої конфігурації на декількох вузлах - в такому випадку одна з інсталяцій стає сервером, а інші – агентами системи. При цьому управління агентами виконується централізовано з сервера.

До складу системи входять декілька різних аналізаторів. Аналізатор журналів використовує файли журналів типових додатків для UNIX-систем, системні журнали Windows і деяких додатків (Internet Information Server, IIS). Модуль виявлення закладок сканує файлову систему вузла і шукає відомі закладки по сигнатурам, а також невідомі закладки та закладки на рівні ядра на основі виявлення аномалій.

Модуль контролю цілісності виконує перевірку найбільш критичних системних файлів (виконувані, конфігураційні, файли бібліотек і т. п.). При першому запуску даний модуль створює базу даних критичних файлів і зберігає в неї, крім самих файлів, інформацію цілісності: параметри доступу, розмір, інформацію про власників, контрольні суми MD5 і SHA1. Потім модуль періодично робить повне сканування системи і порівнює системні файли з копіями в базі даних. У тому випадку, якщо який-небудь файл змінився, генерується повідомлення адміністратору.

Система OSSEC також включає в себе модуль кореляції повідомлень про атаки, який розширює можливості по аналізу повідомлень системи Snort,

видаляє помилкові повідомлення і дозволяє ініціювати реакцію на складні події.

2.1.3 STAT

Система STAT (State Transition Analysis Tool - засіб аналізу систем переходів) є результатом проекту Каліфорнійського університету, Санта-Барбара, США [12]. Перші публікації про систему датовані 1992 р останні - 2003 г. Основою використовуваного системою методу є опис вихідної інформації, що захищається системи у вигляді набору станів компонентів і подальший аналіз переходів зі стану в стан в результаті активних зовнішніх впливів. Стан захисної системи визначаються при налаштуванні і конфігурації системи виявлення атак. Для кожного стану визначається характеристика захищеності.

Визначаються переходи - зміни стану захисної системи.

Атаки описуються у вигляді послідовності переходів. Даний підхід також є наближеним і споріднений підходу, використовуваному в експертних системах, що базуються на сигнатурах атак. Опис атак у вигляді послідовності переходів термінів станів покликане уникнути традиційних обмежень методів, заснованих на сигнатурах і дати можливість описати шаблони для цілих класів типових атак.

Основою системи є мова STATL - розширювана мова, призначений для опису шаблонів атак в термінах STAT. Базова мова оперує найбільш абстрактними поняттями, що не залежать від конкретної системи і її конфігурації.

Мова дозволяє добудовувати себе, додаючи специфічні для конкретної системи події. Для кожного нового події описується його предикат. Наприклад, для розширення мови з метою визначення подій, характерних для веб-сервера Apache, необхідно визначити події, що описують з'являються в журналах даного додатка записи. Тобто подія матиме поля host, ident, authuser, request, status та інші, визначені в Apache's Common Log Format. Після цього потрібно

описати предикати подій, що цікавлять. Наприклад, предикат isCGIrequest () буде повертати true, якщо мав місце виклик CGI-сценарію. Описи подій і предикатів групуються в Language Extension Module (модуль розширення мови) і надалі їх можна використовувати в описі сценаріїв атак для STAT. Потік реальних подій порівнюється зі сценаріями ядром STAT. Всі конструкції мови STATL і його розширення транслюються в мову C ++. Основною функціональною частиною системи є ядро STAT. Цей компонент оперує абстрактними об'єктами і подіями, незалежними від конкретної системи. Він порівнює вхідний потік подій з наявними сценаріями атак і виконує безпосередньо функцію виявлення. Для генерації потоку подій використовується джерело подій - програмний компонент системи виявлення, який здійснює перетворення інформації з системних джерел, таких як журнали реєстрації, в формат, придатний для функціонування ядра STAT.

В системі також можуть використовуватися модулі реакції - пов'язані з ядром компоненти, які здійснюють реагування на виявлену атаку.

Дана система є відкритою і дозволяє будувати масштабовані системи виявлення. На основі STAT будуються агенти або сенсори системи виявлення, все інше - питання архітектури і взаємодії агентів в розподіленій системі. Цьому і присвячені всі роботи xSTAT, які застосовують цю технологію для конструювання систем виявлення атак різних типів - вузлові, мережеві.

Основні компоненти:

- STAT sensor - модуль виявлення атак на вузлі на основі ядра STAT.
- STAT proxy - модуль, що зв'язує сенсори з центральним модулем MetaSTAT.
- MetaSTAT - модуль збору інформації про атаки, повідомлення адміністратора, зберігання інформації про атаки.

Мовою реалізації системи NetSTAT є мова C ++. Система працює під управлінням ОС Linux і Solaris.

2.1.4 Prelude

Система Prelude є системою з відкритими початковими текстами. Початок розробки - 1998 г. Вона спочатку замислювалася як гібридна СОВ, яка могла б допомогти адміністратору мережі відстежувати активність як на рівні мережі, так і на рівні окремих вузлів. Система розподілена і складається з наступних компонентів [17]:

- мережеві сенсори - різні сенсори, що аналізують дані на рівні мережі на основі сигнатурного аналізу. Сенсори генерують повідомлення про виявлення атак і відправляють їх модулів управління. Система Prelude використовує в якості мережевого сенсора систему Snort;
- вузлові сенсори - різні сенсори рівня системи, що аналізують журнали реєстрації ОС, додатків;
- модулі управління - процеси, які отримують і обробляють повідомлення сенсорів;
- агенти реагування - реалізують сгенерированную менеджером реакцію на атаку.

Сенсори генерують повідомлення про виявлення аномалій і відправляють їх до модулів управління. Існуючий набір сенсорів дозволяє аналізувати дані журналів реєстрації таких систем і додатків, як міжмережевий екран IPFW, що входить до складу ОС FreeBSD, NetFilter ОС Linux 2.4.x, маршрутизатори Cisco і Zyxel, GRSecurity і типові сервіси ОС UNIX.

Відрізняються такі види модулів управління:

- модулі журналізації - відповідають за реєстрацію повідомлень в журналах реєстрації або базах даних. В даний час реалізовані модулі для MySQL, PostgreSQL;
- модулі реагування - аналізують повідомлення і генерують можливу реакцію СОВ на атаку. можливі такі види реакції, як блокування порушника на межсетевом екрані (NetFilter, IPFilter). Надалі можливі такі типи реакції, як ізоляція порушника і звуження пропускної здатності каналу порушника.

Інтерфейс, що базується на протоколі http, надає можливість отримувати статистику і управляти системою за допомогою web-браузера.

У системи Prelude є кілька особливостей, які відрізняють її від інших сучасних відкритих СОВ. Система всюди, де можливо, побудована на використанні відкритих стандартів. Так, для обміну повідомленнями використовується формат IDMEF (Intrusion Detection Message Exchange Format), оптимізований для високошвидкісної обробки. Це дозволяє в подальшому інтегрувати компоненти в системи сторонніх виробників і навпаки.

При розробці системи особлива увага була приділена питанням безпеки. Канали передачі даних шифруються по протоколу SSL, крім того, використовується спеціалізована бібліотека, яка запобігає класичні помилки виходу за межі масивів і переповнення буферів.

Додаткові модулі аналізу мережевих даних роблять систему стійкою до некоректних мережевим пакетам на різних рівнях стека і виходу її компонентів з ладу. Такі атаки, як відправка пакетів з неправильними контрольними сумами, обнуленими прапорами TCP, ресинхронізація сесій, помилкової відправки і «обрізання» сегментів системою, ігноруються і не призводять до відмови компонентів СОВ.

2.1.5 Snort

Snort є відкритою Network Intrusion Detection / Prevention System (NIDS / NIPS) системою, що дозволяє проводити аналіз трафіку в реальному часі, а також логінг пакетів в IP мережах [11, 16]. Вона дозволяє аналізувати протоколи верхніх рівнів на предмет пошуку і відповідності потрібного вмісту і може використовуватися для виявлення різних атак, таких як buffer overflows, stealth port scans, CGI attacks, SMB probes, OS fingerprinting attempts та інших. Snort використовує гнучку мову написання правил, яка дозволяє охарактеризувати цікавий трафік для збору або аналізу, а також має детектор атак, що має модульну архітектуру. Виявляє атаки виключно на основі аналізу мережевого трафіку. Основним методом виявлення атак, що використовуються

в системі, є виявлення зловживань на основі опису сигнатур атак. В системі використовується проста мова опису сигнатур атак, яка повністю описана в документації і дозволяє адміністраторам системи доповнювати базу сигнатур своїми сигнатурами. Кожне правило на цій мові складається з двох частин: умова застосування і дію.

Приклад правила системи Snort зображений на рисунку 2.1.

$$alerttcpanyany- > 10.1.1.0/2480$$
$$(content : "/cgi - bin/phf"; msg : "PHFprobe!";)$$

Рисунок 2.1 – Приклад правил системи Snort

Це правило визначає, що будь-який сегмент TCP, спрямований на порт 80 на будь-яку адресу в мережі 10.1.1.0/24, і при цьому має в полі даних рядок «/ cgibin / phf», є підозрілим і необхідно надіслати повідомлення адміністратору.

Крім того, в останніх версіях системи з'явилася спеціальна конструкція мови сигнатур, що дозволяє класифікувати мережевий трафік за ступенем потенційної небезпеки. Ступінь небезпеки визначається експертом, який формує сигнатуру атаки.

Архітектура системи Snort цілком розроблялася з міркувань ефективності та швидкості роботи. Тому вона дуже проста і складається з наступних підсистем: декодер пакетів, ядро виявлення і підсистеми оповіщення та реагування. Декодер пакетів реалізує набір процедур для послідовної декомпозиції пакетів відповідно до рівнів мережевого стека, тобто прийнятий кадр послідовно перетворюється в пакет, сегмент і блок даних із застосуванням специфічних для даного рівня сигнатур атак. В даний час підтримуються протоколи канального рівня Ethernet, SLIP, PPP. Ядро вибудовує наявні правила в так званій ланцюга правил - двовимірної послідовності правил, де правила із загальною частиною умов застосування об'єднуються в одну ланку ланцюга, а не співпадаючі компоненти правил будуються ланцюгом у другому вимірі від

отриманого ланки. Це зроблено для прискорення аналізу мережевого трафіку. Кожен пакет проходить по ланцюжку від кореня, перше правило, що підходить виконує свій блок дій і прохід завершується.

Крім модуля аналізу трафіку на основі правил, до ядра виявлення можуть підключатися модулі сторонніх розробників (препроцесори) і проводити аналіз на одному з рівнів декомпозиції пакета. За допомогою таких модулів можна додавати функціональність ядра виявлення атак і реалізовувати різні методи виявлення. Препроцесори виконують декомпозицію мережевого трафіку і перевірку відповідності специфікаціям протоколів, дефрагментацію і т. П. В поставку системи входить кілька модулів, наприклад модуль виявлення сканування портів, модуль виявлення Unicode-атаки на веб-сервер компанії Microsoft та інші.

Крім того, в одній з версій в складі Snort був модуль статистичного аналізу, який призначений для виявлення аномалій в мережевому трафіку. Підсистема повідомлення і реагування відповідає за збереження результатів аналізу трафіку в журнали реєстрації самої системи Snort або висновок цієї інформації через системні служби реєстрації подій ОС. Наприклад, в UNIX-подібної ОС це може бути сервіс реєстрації подій syslog. Система Snort реалізована під безліч UNIX платформ.

Snort можна використовуватися в трьох різних варіантах як:

- пакетного сніфферу (packet sniffer) за типом утиліти tcpdump;
- реєстратору пакетів (packet logger) для вивчення мережевого трафіку;
- NIDS або NIPS.

Для роботи IDS / IPS Snort необхідний якомога більш потужний сервер з великим об'ємом дискового простору для зберігання бази подій. Для розгортання Snort необхідно встановити і налаштувати ряд програм:

- операційна система FreeBSD або MS Windows;
- Snort - сам сенсор з детекторами для виявлення атак;
- Libpcap - сниффер для захоплення пакетів;

- Система управління базами даних MySQL - для зберігання бази даних подій;
- PHP - мова розробки для Web;
- Apache - web-сервер;
- Basic Analysis and Security Engine (BASE) - консоль управління і перегляду подій (alerts);
- Oinkmaster - утиліта для оновлення сигнатур.

2.1.6 SnortNet

SnortNet це розподілене розширення системи Snort, метою якого є надання їй додаткових можливостей по масштабованості і розширюваності [19]. Система складається з декількох програмних модулів: сенсорів, модулів пересилання повідомлень і станції моніторингу. Система дозволяє проводити моніторинг мережевого трафіку і здійснювати інформування станції моніторингу про всі виявлені аномалії в поведінці мережевих об'єктів. Система використовує Snort в якості мережного сенсора. Як протокол обміну даними система використовує протокол Internet Alert Protocol (протокол передачі сигналів тривоги - IAP). Для шифрування каналів передачі даних, аутентифікації і контролю доступу система використовує бібліотеки SSL і TCP wrappers.

Виявлення атак проводиться сенсорами Snort, після чого повідомлення відправляються за протоколом IAP через модулі пересилання повідомлень (proxy) на станцію моніторингу. Модулі пересилання повідомлень і станцію моніторингу рекомендується розташовувати в демілітаризованій зоні мережі(рис 1.5).

Система SnortNet протестована під операційними системами Linux, FreeBSD, OpenBSD.

2.1.7 AAFID

Система AAFID (Autonomous Agents for Intrusion Detection) розроблена в університеті Purdue, West Lafayette, IN, USA. Перші публікації по системі датовані 1998 р останні - 1999 г. AAFID це одночасно назва розподіленої архітектури систем виявлення атак і власне системи виявлення атак [18]. Основою системи є автономні агенти виявлення. Найбільш цікава в системі

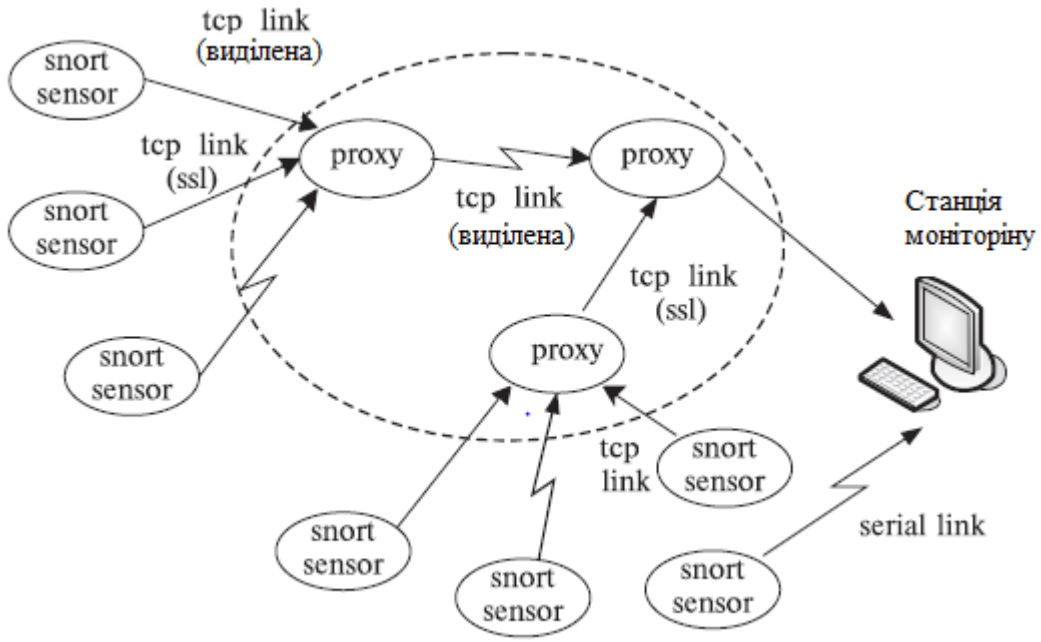

Рисунок 2.2 – Конфігурація системи сенсорів SnortNet

саме її архітектура. Дана система базується на роботах Crosbie і Spafford, які запропонували використання автономних агентів, що працюють на основі генетичних алгоритмів і адаптуються до поведінки користувачів. Ідея використання генетичних алгоритмів реалізовано не було, але архітектурні ідеї були втілені в системі AAFID.

Основними компонентами системи є агенти, трансивери, монітори, фільтри. Система AAFID є повністю розподіленою. На будь-якому вузлі в ЛОМ може бути розміщено будь-яке число агентів, що спостерігають за цікавими з їх точки зору подіями на даному вузлі.

Агент – автономний програмний компонент системи виявлення атак, функціонування якої залежить лише від ОС, під управлінням якої вона працює. Тобто функціонування агента не залежить від інших компонентів системи виявлення атак. Агенти можуть використовувати фільтри для збору інформації про поведінку об'єктів в архітектурно-незалежному поданні. Всі агенти на одному вузлі передають зібрану інформацію одному трансивер.

Трансівер – компонент системи виявлення, який управляє запуском і зупинкою агентів на даному вузлі. Кожен вузол, на якому є агенти, має один трансивер. Крім того, трансивери можуть здійснювати деяку редукцію одержуваної від агентів інформації в більш узагальнене уявлення. Трансивери передають отриману інформацію одному або декільком моніторів. Кожен монітор спостерігає і взаємодіє з декількома трансиверами.

Монітори функціонують на рівні захищається системи в цілому, тому вони можуть аналізувати отриману інформацію з урахуванням кореляції подій в різних областях захищається системи. Монітори можуть розташовуватися в ієрархічному порядку. Монітор також є сполучною компонентом між системою виявлення і призначеним для користувача інтерфейсом - він отримує керуючі команди і віддає отриману і проаналізовану інформацію.

Зупинимося детальніше на описі компонентів системи AAFID. Агент спостерігає за конкретними аспектами функціонування вузла і повідомляє відповідний трансивер про аномальні події на вузлі. Агенти не можуть безпосередньо генерувати повідомлення для користувача. Таким чином, трансивери отримують повну інформацію про функціонування вузла, а монітори - про мережу в цілому. Функціональність агента нічим не обмежена, він може реалізовувати будь-який метод виявлення атак. Агенти можуть бути призначені для виявлення аномалій і зловживань будь-якого типу.

Фільтри призначені для збору однотипної інформації з різних джерел (різних системних журналів або від спостерігачів). Фільтри також виконують функції рівня представлення моделі OSI (Open System Interconnect):

перетворять формати даних, що збираються з аналогічних джерел, але на різних архітектурах, до єдиного вигляду.

Один фільтр може надавати дані декільком агентам. Типовий приклад застосування фільтра - використання його для перетворення журналів реєстрації різних версій ОС UNIX в формат, який розуміється агентами. Це знімає необхідність реалізовувати різні агенти для різних платформ.

На кожне джерело даних існує тільки один фільтр. Агенти можуть підписуватися на отримання даних від фільтра.

Трансівер - інтерфейс взаємодії між вузловими компонентами системи виявлення та мережевими компонентами системи виявлення. Основними функціями трансивера є управління агентами на даному вузлі (запуск, зупинка), збір і аналіз даних від агентів, передача даних і результатів аналізу моніторів або іншим агентам.

Монітори - найвищі в ієрархії компоненти системи. Функціонально вони схожі на трансивери з тією відмінністю, що збір і аналіз інформації відбувається не на одному вузлі, а на частині мережі або на всій мережі. Монітори можуть управляти трансиверами і іншими моніторами. Монітори мають механізми взаємодії з призначеним для користувача інтерфейсом і є точками доступу до системи виявлення атак.

В системі AAFID практично не реалізовані агенти, необхідні для ефективного виявлення атак різних класів, і основною цінністю даної системи є її архітектура. У документації добре описані і стандартизовані інтерфейси між компонентами системи виявлення атак. Особлива увага приділяється питанням безпечної взаємодії компонентів розподіленої системи виявлення. Наскільки можна судити за описом системи, в даний час всі робочі агенти реалізовані за принципом експертної системи, вони виявляють заздалегідь описані ненормальні ситуації і події на вузлі і в мережі.

Система все ще знаходиться на стадії прототипу. Останнє оновлення мало місце у вересні 1999 року, і дана версія була реалізована і протестована під

операційними системами Solaris і Linux. Мовою реалізації є алгоритмічний мову Perl. На ньому реалізовані всі компоненти системи.

2.1.8 Ефективність СВВ.

За принципом побудови СВВ можна класифікувати як монолітні та компонентні. У першому випадку система представлена як єдиний бінарний файл, запуск якого засобами ОС, як правило, породжує не більш одного процесу. До них з розглянутих СВВ належать Snort і Bro. Компонентний підхід має на увазі розбивку системи на кілька функціональних блоків, кожний з яких запускається в окремому просторі пулу адрес пам'яті, виконує конкретне завдання й спілкується з іншими на основі механізмів міжпроцесної взаємодії. До ряду таких систем належать OSSEC і Prelude. Як уже було відзначено раніше, основними елементами для створення додаткових функцій у роботі систем Snort є декодери й препроцесори, основа функціонування яких полягає у виклику певних функцій з динамічно завантажуваних бібліотек (so/dll). Найбільш просунутий спосіб створення нових модулів, що розширюють функціональні можливості системи, має Bro. Уся рутинна робота з написання шаблонових файлів майбутнього додатка зведена до мінімуму. Разом з вихідними кодами даної системи поставляється bash-скрипт init-plugin, призначений для автоматичної генерації початкового кістяка майбутнього модуля, правил його складання й установки. Після компіляції модуль являє собою готову so-бібліотеку, прототипи функцій з якої обгорнені у відповідний bro-скрипт. Наступне завантаження, ініціалізація покажчиків на функції та деалокація модулів системою зводяться до виклику функцій dlopen, dlsym і dlclose відповідно.

В роботі проведений порівняльний аналіз характеристик сучасних відкритих систем виявлення аномалій.

Серед розглянутих систем найбільш повні характеристики має Prelude. Будучи спроектованою з самого початку розподіленою системою, вона підтримує гібридний моніторинг контрольованих вузлів, здійснюючи аналіз як

на рівні мережі, так і на рівні хоста. Крім того, ця система є масштабованою, що дозволяє їй використовувати безліч різнорідних джерел для збору й обробки даних. Модульний принцип установки цієї системи також дозволяє добитися більш гнучкого настроювання кожного з її компонентів окремо. Можливість підключення кожної із представлених СВА в якості сенсорів для Prelude (для Bro необхідно реалізовувати нового клієнта вручну, тому що для неї немає готового сенсора) ставить її на ранг вище інших СВА. Саме в такій системі можно запропонувати використання нейромережних методів виявлення аномалій, що далі розглянуті в роботі.

3. АНАЛІЗ ДАНИХ ЗА ДОПОМОГОЮ НЕЙРОННИХ МЕРЕЖ ТА ОПИС ЗРОСТАЮЧОГО НЕЙРОННОГО ГАЗУ (GNG)

3.1 Нейромережеві методи

В основі нейромережевих методів аналізу даних лежить спроба комп'ютерного моделювання процесів навчання, що використовуються в живих організмах. Когнітивні здібності живих істот пов'язані з функціонуванням мереж пов'язаних між собою біологічних нейронів - клітин нервової системи. Для моделювання біологічних нейромереж використовуються мережі, вузлами яких є штучні нейрони (тобто математичні моделі нейронів), Можна виділити три типи штучних нейронів: нейрони-рецептори, внутрішні нейрони і реагуючі нейрони [7]. Кожен внутрішній або реагуючий нейрон має безліч вхідних зв'язків, до яких надходять сигнали від рецепторів або інших внутрішніх нейронів. Приклад моделі внутрішнього або реагуючого нейрона представлений на рисунку. 3.1.

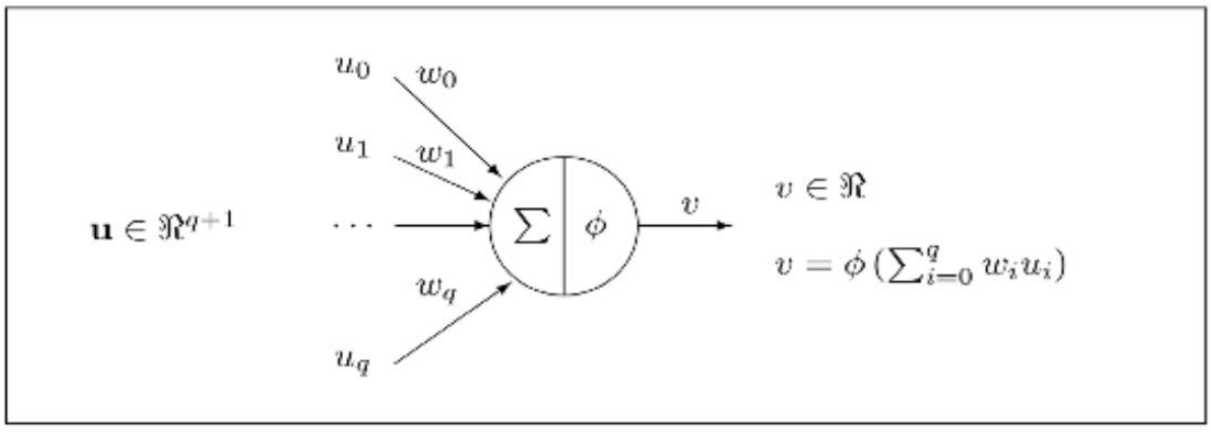

Рисунок. 3.1 - Моделі внутрішнього або реагуючого нейрона

Представлений на рисунку 3.1 нейрон має r зовнішніх зв'язків, за якими на нього надходять вхідні сигнали $u_1, \ldots, u_r$. Надіслані сигнали підсумуються з

вагами $w_1, \ldots, w_r$. На виході нейрона виробляється сигнал $\Phi(z)$, де $z = w_0 + \sum_{i=1}^{r} w_i u_i$, $w0$ – параметр зсуву. Може бути використана також форма запису $z = \sum_{i=0}^{r} w_i u_i$, де фіктивний «сигнал» u_0 тотожно дорівнює 1. Функцію $\Phi(z)$ зазвичай називають активаційною функцією. Можуть використовуватися різні види активаційних функцій. Методи, засновані на використанні штучних нейронів можуть бути використані для вирішення найрізноманітніших завдань розпізнавання. При цьому сигнали, що надходять на вхід персептрона, інтерпретуються як вхідні ознаки $X_1, \ldots, X_n$.

Першою нейромережною моделлю став персептрон Розенблатта, запропонований в 1957 році. У даній моделі використовується єдиний реагуючий нейрон. Модель, яка реалізує лінійну розподільчу функцію в просторі вхідних сигналів, може бути використана для вирішення завдань розпізнавання з двома класами, поміченими мітками 1 або -1. В якості активаційної функції використовується порогова функція: $\Phi(z) = 1$ при $z \geq 0, \Phi(z) = -1$ при $z < 0$. Особливістю моделі Розенблатта є дуже проста, але в той же час ефективна, процедура навчання, що обчислює значення вагових коефіцієнтів $(w_0, \ldots, w_n)$. Налаштування параметрів проводиться за навчальними вибірками, абсолютно аналогічних тим, які використовуються для навчання статистичних алгоритмів. На першому етапі проводиться перетворення векторів сигналів для об'єктів навчальної вибірки. У набір вихідних ознак додається тотожно рівна 1 нульова компонента. Потім вектори описів з класу K_2 помножуються на -1. Вектори описані із класу K_1 не змінюються.

Процедура навчання персептрона - нульове наближення вектору вагових коефіцієнтів $(w_0^{(0)}, \ldots, w_n^{(0)})$ вибирається випадково. Перетворені опису об'єктів навчальної вибірки $\widetilde{S_t}$ послідовно подаються на вхід перцептору. У разі якщо опис $x^{(k)}$, подано на k-му кроці класифікується неправильно, то відбувається корекція за правилом $w^{(k+1)} = w^{(k)} + x$. В випадку правильної класифікації $w^{(k+1)} = w^{(k)}$. Відзначимо, що правильної класифікації завжди

відповідає виконання рівності $\left(w^{(k)}, x^{(k)}\right) < 0$, а неправильної (w (k) , x (k)) < 0. Процедура повторюється до тих пір, поки не буде виконано одну з наступних умов:

- досягається повне розділення об'єктів з класів K_1 та K_2;

- повторення підряд заздалегідь заданого числа ітерацій не приводить до поліпшення поділу;

- виявляється вичерпаним заздалегідь заданий ліміт ітерацій.

Для описаної процедури навчання справедлива наступна теорема:

У разі, якщо опису об'єктів навчальної вибірки лінійно нероздільні в просторі приознакових описів, то процедура навчання персептрона побудує лінійну гіперплощину, що розділяє об'єкти двох класів за кінцеве число кроків.

Відсутність лінійної роздільності двох класів призводить до нескінченного зациклення процедури навчання персептрона. Істотно вищою апроксимуючої здатність мають нейромережеві методи розпізнавання, що задаються комбінаціями пов'язаних між собою нейронів. Таким методом є Багатошаровий персептрон. У методі Багатошаровий персептрон мережу формується з декількох шарів нейронів. У їх число входить шар вхідних рецепторів, які подають сигнали на нейрони з внутрішніх шарів. Шари внутрішніх нейронів здійснюють перетворення сигналів. Шар реагуючих нейронів виробляє остаточну класифікацію об'єктів на підставі сигналів, що надходять від нейронів, що належать внутрішнім верствам. Зазвичай дотримуються наступні правила формування структури мережі. Допускаються зв'язку між тільки між нейронами, що знаходяться в сусідніх шарах. Зв'язки між нейронами всередині одного шару відсутні. Активаційні функції для всіх внутрішніх нейронів ідентичні. Для вирішення завдань розпізнавання з L класами $K_1, \ldots, K_L$ використовується конфігурація з L реагують нейронами. Схема багатошарового персептрона з двома внутрішніми шарами представлена на рисунку 3.2.

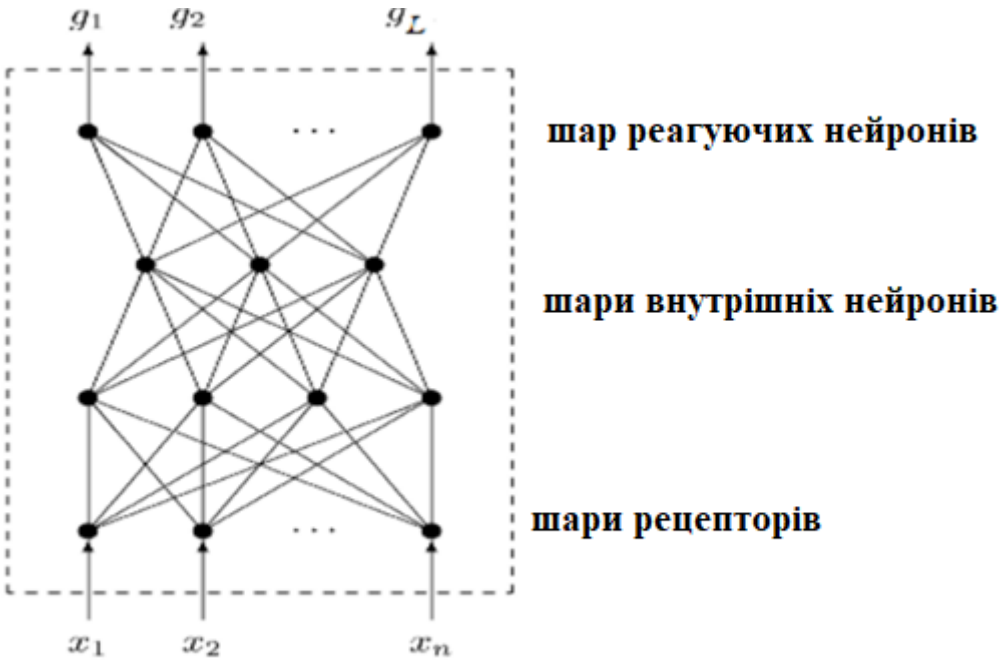

Рисунок. 3.2 - Схема багатошарового персептрона

Відзначимо, що сигнали $g_1,\ldots,g_L$, обчислювані на виході реагуючих нейронів, інтерпретуються як оцінки за класи $K_1,\ldots,K_L$. Вагові коефіцієнти w зіставлені кожного із зв'язків між нейронами з різних шарів. Розглянемо процедуру розпізнавання об'єктів з використанням багатошарового персептрона. Припустимо, що конфігурація нейронної мережі включає поряд із шаром рецепторів і шаром реагують нейронів також H внутрішніх шарів штучних нейронів. Задані також кількості нейронів в кожному шарі. Нехай n – число вхідних нейронів-рецепторів, $r(h)$ - число нейронів у міністерстві внутрішніх справ шарі h. На першому етапі вектор рецептори формують за інформацією, що надходить із зовнішнього середовища, вектор вхідних змінних (сигналів) $u_1^0,\ldots,u_n^0$. Відзначимо, що вхідні сигнали $u_1^1,\ldots,u_n^0$ можуть інтерпретуватися як ознаки $X_1,\ldots,X_n$ в загальній постановці завдання розпізнавання. Припустимо, що для і-го нейрона 1-го внутрішнього шару зв'язок з рецепторами здійснюється за допомогою вагових коефіцієнтів $w_1^{i0},\ldots,w_n^{i0}$. Суматор і-го нейрона першого внутрішнього шару обчислює зважену суму $\xi^{i0}=\sum_{t=0}^{n} w_t^{i0}\,u_t^0$. Сигнал на виході і-го нейрона першого

внутрішнього шару обчислюється за формулою $u_i^1 = \Phi(\xi^{i0})$. Аналогічним чином обчислюються сигнали на виході нейронів другого внутрішнього шару. Сигнали $g_1, \ldots, g_L$ розраховуються за допомогою тієї ж самої процедури, яка використовується при обчисленні сигналів на виході нейронів з внутрішніх шарів. Тобто при обчисленні g_i на першому кроці відповідний акумулятор обчислює зважену суму

$$\xi^{iH} = \sum_{i=0}^{n} w_t^{iH} u_t^H, \tag{3.1}$$

де $w_1^{iH}, \ldots, w_{r(H)}^{iH}$ - вагові коефіцієнти, що характеризують зв'язок і -го реагує нейрона з нейронами останнього внутрішнього шару H, $u_1^H, \ldots, u_{r(H)}^H$ - сигнали на виході внутрішнього шару H. Сигнал на виході і-го реагує нейрона обчислюється за формулою $g_i = \Phi(\xi^{iH})$. Очевидно, що вектор вихідних сигналів є функцією вектору вхідних сигналів u^0 (Вектору ознак x) і матриці вагових коефіцієнтів зв'язків між нейронами.

3.1.1 Метод зворотного розповсюдження помилки

Найбільш поширеним способом навчання нейромережевих алгоритмів є метод зворотного поширення помилки. позначимо через $\alpha_* = (\alpha_{*1}, \ldots, \alpha_{*L})$ вектор індикаторних функцій класів $K_1, \ldots, K_L$ на об'єкті s_* з описом x_*. Тобто $\alpha_{*i} = 1$, якщо $s_* \in K_i$ и $\alpha_{*i} = 0$ в іншому випадку. Нехай на виході і-го реагує нейрона обчислюється оцінка $g_i(x_*)$ за клас K_i, що належить відрізку [0, 1]. Відзначимо, що оцінка $g_i(x_*)$ обчислюється активаційною функцією реагує нейрона. Далі буде передбачатися, що дана активаційна функція є сігмоідной. Таке ж припущення робиться для активаційних функцій кожного з внутрішніх нейронів. Втрати, пов'язані з класифікацією об'єкта s∗ природно оцінювати за допомогою функціоналу

$$\sum_{i=1}^{L} [\, \alpha_{*i} - g_i(x_*)]^2. \tag{3.2}$$

Якість апроксимації на навчальній вибірці $\widetilde{S_t} = \{(\alpha_1, x_1), \ldots, (\alpha_m, x_m)\}$ оцінюється за допомогою функціоналу

$$E(\widetilde{S}_t\widetilde{w}) = \sum_{j=1}^{m}\sum_{i=1}^{L}[\alpha_{ji} - g_i(x_j)]^2. \quad (3.3)$$

Де $\widetilde{w} = \{w_t^{ih} \mid h = 0,\dots,H; t = 1,\dots,r(h); i = 1,\dots,(r_{h+1})\}$ – безліч вагових коефіцієнтів зв'язків меду нейронами. Навчання полягає в пошуку значень коефіцієнтів з $\widetilde{w}$, при яких досягає мінімуму функціонал $E(\widetilde{S}_t\widetilde{w})$. В основі навчання лежить метод градієнтного спуску. Метод градієнтного спуску є ітераційним методом оптимізації довільного функціоналу F , залежить від параметрів $(\theta_1,\dots,\theta_r)$ і дифференцируемого по кожному з параметрів в довільній точці $\mathbb{R}^n$. Нові значення вектору параметрів на k-ій ітерації $\theta^{(k)}$ обчислюється через вектор $\theta^{(k-1)}$, отриманий на попередній ітерації. При цьому використовується формула

$$\theta^{(k)} = \theta^{(k-1)} + \eta \times grad[F(\theta_1,\dots,\theta_r)], \quad (3.4)$$

де $\eta > 0$ - речовинний параметр, що задає розмір кожного кроку. На попередньому етапі навчання ваговим коефіцієнтам з w випадковим чином присвоюються початкові значення. На навчання подається деякий об'єкт навчальної вибірки $s_j = (\alpha_j, x_j)$, за описом якого обчислюються вхідні і вихідні сигнали внутрішніх нейронів мережі, а також вихідні сигнали реагують нейронів $g_1(x_j, \widetilde{w}),\dots, g_L(x_j, \widetilde{w})$. Проведемо корекцію вагових коефіцієнтів зв'язків і-го реагує нейрона з нейронами попереднього внутрішнього шару:

$$(w_0^{iH},\dots,w_{r(H)}^{iH}). \quad (3.5)$$

Для спрощення формул $g_i(x_j, \widetilde{w})$ будемо позначати $g_i(x_j)$ або просто g_i. Від вагових коефіцієнтів $(w_0^{iH},\dots,w_{r(H)}^{iH})$ залежить тільки компонента $[\alpha_{ji} - g_i(x_j)]^2$ помилки прогнозування для об'єкта s_j, дорівнює $E(s_j, \widetilde{w}) = \sum_{i=1}^{L}[\alpha_{ji} - g_i(x_j)]^2$.

Тому

$$\frac{\partial E(s_j,\ \widetilde{w})}{\partial w_t^{iH}} = \frac{\partial E(s_j,\widetilde{w})}{\partial g_i(x_j)}\frac{\partial g_i}{\partial w_t^{iH}} = -2[\alpha_{ji} - g_i(x_j)]\frac{\partial g_i(x_j)}{\partial w_t^{iH}} \quad (3.6)$$

Однак

$$\frac{\partial g_i(x_j)}{\partial w_t^{iH}} = \frac{\partial g_i(x_j)}{\partial \xi^{iH}}\frac{\partial \xi^{iH}}{\partial w_t^{iH}}, \quad (3.7)$$

Де $\xi^{iH} = \sum_{t=1}^{r(H)} w_t^{iH}\, u_t^H,\ u_t^H$ – сигнал на виході нейрона з номером t з шару H. Оскільки g_i є сігмоідной функцією від ξ^{iH}, то

$$\frac{\partial g_i}{\partial w_t^{iH}} = (1 - g_i) g_i u_t^H. \tag{3.8}$$

Таким чином

$$\frac{\partial E(s_j,\ \tilde{w})}{\partial w_t^{iH}} = \delta^{iH} u_t^H, \tag{3.9}$$

де

$$\delta^{iH} = \frac{\partial E(s_j, \tilde{w})}{\partial \xi^{iH}} = -2[\alpha_{ji} - g_i(x_j)][1 - g_i(x_j)] g_i(x_j). \tag{3.10}$$

Скориставшись методом градієнтного спуску, запишемо нові значення вагових коефіцієнтів $w_t^{iH}(k)$, обчислювані на k ітерації k у вигляді

$$w_t^{iH}(k) = w_t^{iH}(k - 1) + \eta \times \delta^{iH} u_t^H \tag{3.11}$$

Розглянемо тепер корекцію вагових коефіцієнтів $[w_0^{i(H-1)}, \ldots, w_{r(H-1)}^{i(H-1)}]$, відповідних зв'язків нейрона i з шару H с нейронами попереднього внутрішнього шару $(H - 1)$. Внесок цих коефіцієнтів в величину помилки здійснюється тільки через сигнал u_i^H на виході нейрона i з шару H.

Тому

$$\frac{\partial E(s_j,\ \tilde{w})}{\partial w_t^{i(H-1)}} = \frac{\partial E(s_j,\ \tilde{w})}{\partial u_t^H} \frac{\partial u_t^H}{\partial w_t^{i(H-1)}} \tag{3.12}$$

Однак

$$\frac{\partial E(s_j,\ \tilde{w})}{\partial u_t^H} = \sum_{l=1}^{L} \frac{\partial E(s_j, \tilde{w})}{\partial \xi^{lH}} \frac{\partial \xi^{lH}}{\partial u_t^H} \tag{3.13}$$

При цьому $\frac{\partial E(s_j, \tilde{w})}{\partial \xi^{lH}} = \delta^{lH}$, а $\frac{\partial \xi^{lH}}{\partial u_t^H} = w_t^{lH}$, отримаємо

$$\frac{\partial E(s_j,\ \tilde{w})}{\partial u_t^H} = \sum_{l=1}^{L} \delta^{lH} w_t^{lH} \tag{3.14}$$

Виходячи з припущення про те, що активаційна функція кожного з нейронів є сігмоідной, неважко показати також, що

$$\frac{\partial u_t^H}{\partial w_t^{i(H-1)}} = u_t^H(1 - u_t^H) u_t^{H-1} \tag{3.15}$$

В підсумку

$$\frac{\partial E(s_j,\ \widetilde{w})}{\partial w_t^{i(H-1)}} = \left(\sum_{l=1}^{L} \delta^{lH}\, w_t^{lH}\right) u_i^H \left(1 - u_i^H\right) u_i^{H-1} = \delta^{i(H-1)} u_i^{H-1}, \qquad (3.16)$$

де

$$\delta^{i(H-1)} = \frac{\partial E(s_j,\widetilde{w})}{\partial \xi^{i(H-1)}} = \left(\sum_{l=1}^{L} \delta^{lH}\, w_t^{lH}\right) u_i^H \left(1 - u_i^H\right) \qquad (3.17)$$

Скориставшись методом градієнтного спуску, запишемо нові значення вагових коефіцієнтів $w_t^{i(H-1)}(k)$, обчислювані на ітерації k в формі

$$\mathrm{w_t^{i(H-1)}(k)} = \mathrm{w_t^{i(H-1)}(k-1)} + \eta \times \delta^{\mathrm{i(H-1)}} \mathrm{u_t^{H-1}} \qquad (3.18)$$

Розглянемо тепер процедуру корекції вагових коефіцієнтів w для зв'язків між штучними нейронами з шару h с штучними нейронами з шару $h + 1$ при $h < H - 1$. Нехай

$$[w_0^{i(H-h)}, \ldots, w_{r(H-h)}^{i(H-h)}] \qquad (3.19)$$

- вагові коефіцієнти, що зв'язують нейрон з номером i з шару $H - h + 1$ с нейронами з шару $H - h$. Очевидно, що справедливо рівність:

$$\frac{\partial E(s_j,\ \widetilde{w})}{\partial w_t^{i(H-1)}} = \frac{\partial E(s_j,\ \widetilde{w})}{\partial u_i^{H-h+1}} \frac{\partial u_i^{H-h+1}}{\partial w_t^{i(H-h)}} \qquad (3.20)$$

Неважко показати, що

$$\frac{\partial E\left(s_j,\ \widetilde{w}\right)}{\partial u_i^{H-h+1}} = \sum_{l=1}^{r(H-h+1)} \frac{\partial E\left(s_j,\ \widetilde{w}\right)}{\partial \xi^{H-h+1}} \frac{\partial \xi^{H-h+1}}{\partial u_i^{H-h+1}} =$$

$$= \sum_{l=1}^{r(H-h+1)} \delta^{l(H-h+1)} w_i^{l(H-h+1)} \qquad (3.21)$$

З огляду на, що активаційна функція для кожного внутрішнього нейрона є сігмоідной, і беручи до уваги визначення $\xi^{i(H-h)}$, отримуємо

$$\frac{\partial u_i^{H-h+1}}{\partial w_t^{i(H-h)}} = \frac{\partial u_i^{H-h+1}}{\partial \xi^{(H-h)}} \frac{\partial \xi^{(H-h)}}{\partial w_t^{i(H-h)}} = u_i^{H-h+1}(1 - u_i^{H-h+1}) u_i^{H-h} \qquad (3.22)$$

В підсумку

$$\frac{\partial E(s_j,\ \widetilde{w})}{\partial w_t^{i(H-h)}} = \delta^{i(H-h)} u_i^{H-h}, \qquad (3.23)$$

де

$$\delta^{i(H-h)} = \left[\sum_{l=1}^{r(H-h+1)} \delta^{i(H-h+1)} w_i^{i(H-h+1)}\right] u_i^{i(H-h+1)} \left(1 - u_i^{i(H-h+1)}\right) \qquad (3.24)$$

Корекція згідно з процедурою градієнтного спуску, здійснюється за формулою:

$$w_t^{i(H-h)}(k) = w_t^{i(H-h)}(k-1) + \eta \times \delta^{i(H-h)} u_t^{H-h} \quad (3.25)$$

Таким чином може бути представлений загальна схема методу зворотного поширення помилки для багатошарового персептрона. На попередньому етапі вибирається архітектура мережі: задається число внутрішніх шарів і кількості нейронів в кожному шарі. Випадковим чином задаються вихідні вагові коефіцієнти. На вхід багатошарового персептрона черзі подаються векторні опису об'єктів навчальної вибірки. З використанням описаного вище способу проводиться корекція вагових коефіцієнтів. З використанням нових скоригованих вагових коефіцієнтів $\widetilde{w}$ обчислюється значення функціоналу $E(s_j, \widetilde{w})$.

Навчання закінчується при виконанні одного з заздалегідь заданих умов:

а) Величина функціоналу помилки виявляється менше обраного порогового значення: $E(s_j, \widetilde{w}) < \varepsilon$;

б) Зміни функціоналу помилки протягом декількох останніх ітерацій виявляється меншим деякого порогового значення.

в) загальний час навчання перевищує допустиму межу;

3.2 Нейронний газ

Нейронний газ – це нейронна мережа, що самоорганізується (реалізує навчання без вчителя), яка в міру надходження вхідних даних дозволяє виявити їх топологічну структуру. Принципова відмінність даного алгоритму від безлічі інших алгоритмів навчання без вчителя полягає в тому, що нейронний газ спочатку не вимагає ніяких даних про предметну область (наприклад, кількості кластерів). На вхід цього алгоритму подаються параметри, що впливають на спосіб побудови нейронної мережі, яка формується на підставі вхідних даних. Нейронна мережа, отримана в результаті роботи алгоритму, відображає

топологію вхідних даних і може використовуватися, наприклад, для їх класифікації або кластеризації

Алгоритм нейронного газу виглядає наступним чином:

1) Ініціалізувати 2 вузла нейронної мережі (із вихідних даних) і з'єднати їх зв'язком вік якого дорівнює 0. В якості вагових коефіцієнтів цих вузлів беруться координати відповідних точок.
2) Подати на вхід алгоритму довільний вектор $\bar{x}$.
3) Знайти 2 нейрона s і t, найближчі до $\bar{x}$. Зазвичай в якості міри відстані вибирається Евклидова метрика, однак, в якості запобіжного відстані може бути також застосована міра Жаккар, косінусний коефіцієнт і т. п. Подальший опис алгоритму буде приведено для Евклідовій міри.
4) Оновити помилку нейрона-переможця (нейрон, для якого відстань до вхідного вектора $\bar{x}$ мінімальна) :

$$E_s = E_s + \|\overline{w_s} - \bar{x}\|^2 \tag{3.26}$$

Де E_s – помилка нейрона s, $\overline{w_s}$ – вектор вагів нейрона s.

5) Змістити нейрон s і всіх його сусідів:

$$\overline{w_s} = \overline{w_s} + \varepsilon_w(\overline{w_s} - \bar{x})$$
$$\overline{w_n} = \overline{w_n} + \varepsilon_n(\overline{w_n} - \bar{x}) \tag{3.27}$$

Де $\varepsilon_w, \varepsilon_n$ – коефіцієнти навчання нейрона-переможця і його сусідів відповідно. У класичному алгоритмі нейронного газу вони залишаються незмінними протягом всього процесу навчання. Ці параметри є вхідними параметрами алгоритму, при цьому зазвичай вибирають $\varepsilon_w = 100 \ldots 1000\varepsilon_n$.

6) Збільшити на 1 вік дуг, що виходять від s.
7) Якщо нейрони s і t з'єднані, то обнулити вік їх зв'язку; якщо зв'язку немає, то створити нову.
8) Видалити всі дуги з віком, що перевищує age_{max} (параметр алгоритму).

9) Якщо номер ітерації кратний $\lambda_{\max}$, то здійснити вставку нового вузла:
   - знайти нейрон u з максимальною локальної помилкою;
   - серед сусідів u знайти нейрон v з максимальною помилкою;
   - створити вузол r з вектором ваг рівним $\overline{W_r} = \frac{\overline{w_u} + \overline{w_v}}{2}$
   - замінити зв'язок (u, v) на (u, r) та (v, r)
   - зменшити помилки нейронів u і v, встановити значення помилки нейрона r наступним чином:

$$E_u = E_u * \alpha$$
$$E_v = E_v * \alpha$$
$$E_r = E_u \quad (3.28)$$

10) Зменшити помилки всіх нейронів j на частку β: $E_j = E_j * \beta$

11) Якщо критерій зупинки не виконано, то перейти на крок 2.

Варто зауважити, що чіткого критерію зупину класичний алгоритм нейронного газу не дає, тому вибір критерію зупину залишається на розсуд дослідника. Завершити роботу алгоритму можна, коли закінчаться вхідні дані, коли мережа досягне заданого розміру або коли середня локальна помилка всіх нейронів почне на кожній ітерації змінюватися менше, ніж на задане число ε.

Переваги даного алгоритму полягають в тому, що він дозволяє визначити топологію вхідних даних без будь-якої апріорної інформації про ці дані. Крім того, задаючи обмеження на розмір будується нейронної мережі, можна ефективно управляти ресурсами, які споживає алгоритм, і часом його роботи. До недоліків даного алгоритму відноситься те, що на його налаштування впливають 6 параметрів ($\varepsilon_w, \varepsilon_n, \lambda, age_{\max}, \alpha, \beta$) підібрати які оптимальним чином для конкретного завдання може виявитися непросто.

4. РОЗРОБКА МОДЕЛІ НЕЙРОННОЇ МЕРЕЖІ ДЛЯ ВИЯВЛЕННЯ АНОМАЛІЙ ТРАФІКУ

4.1 Опис даних (dataset)

У цій роботі для реалізації GNG у сфері виявлення аномалій було використовано набір даних NSL-KDD. NSL-KDD – це набір даних, що був запропонований [3], який дозвоняє моделювати будь-яке вторгнення у реальному маштабі та часі. Набір даних складається із списку файлів, що описані в таблиці 4.1

Таблиця 4.1 – Набір даних NSL-KDD

№	Ім’я файлу	Опис
1	KDDTrain+.csv	Повний NSL-KDD набір даних для навчання у форматі csv
2	KDDTrain+.txt	Повний NSL-KDD набір даних для навчання у форматі txt
3	KDDTrain+_20Percent.csv	Вибірка 20% даних із KDDTrain+.csv
4	KDDTrain+_20Percent.TXT	Вибірка 20% даних із KDDTrain+.txt
5	KDDTest+.csv	Повний NSL-KDD набір даних для тестування у форматі csv

Продовження таблиці 4.1

6	KDDTest+.txt	Повний NSL-KDD набір даних для тестування у форматі txt
7	KDDTest-21.txt	Вибірка із KDDTest+.txt, яка містить невірно класифіковані під час дослідження дані
8	Small Training Set.csv	Малий набір даних NSL-KDD для навчання у форматі csv

Кожне з’єднання характеризується сорока двома параметрами, з яких 41 описані в таблиці 4.2

Таблиця 4.2 – Параметри NSL-KDD

№	Ім’я параметру	Опис	Приклад
Основні параметри кожного мережевого з'єднання			
1	duration	Час тривалості підключення	0
2	protocol_type	Протокол, який використовується при підключенні	TCP
3	service	Мережева служба, яка використовується підключенням	ftp_data

Продовження таблиці 4.2

4	src_bytes	Кількість відправлених байт за одне з'єднання	491
5	dst_bytes	Кількість прийнятих байт за одне з'єднання	0
6	flag	Статус з'єднання - нормальне або з помилкою	SF
7	land	Якщо ip-адреси хоста джерела і призначення рівні, і аналогічна ситуація з портами, то параметр приймає значення 1, інакше 0.	1
8	wrong_fragment	Загальна кількість невірних фрагментів за це підключення	0
9	urgent	Кількість urgent-пакетів в цьому підключенні	0
10	hot	Кількість hot-індикаторів, наприклад таких як: вхід в системні директорії, створення програм, виконання програм.	0
11	num_failed_logins	Кількість невдалих спроб входу	0
Параметри, пов'язані з контентом кожного мережевого з'єднання			
12	logged_in	Логін статус. 1 - якщо успішно увійшли в систему, інакше 0	1
13	num_compromised	Число скомпрометованих станів	0
14	root_shell	1, якщо root-права отримані, інакше 0	1

Продовження таблиці 4.2

15	su_attempted	1, якщо su root-права отримані, інакше 0	1
16	num_root	Число root-доступів	0
17	num_file_creations	Число операцій по створенню файлів під час з'єднання	0
18	num_shells	Число викликів shell-оболонки	0
19	num_access_files	Число операцій з отримання контролю доступу до файлів	0
20	num_outbound_cmds	Число вихідних команд в FTP-сесії	0
21	is_hot_login	1, якщо логін належить hot-листу тобто якщо є root або адміністратором, інакше 0	1
22	is_guest_login	1, если логин является гостевым, иначе 0	1
Параметри, пов'язані з тимчасовими характеристиками кожного мережевого з'єднання			
23	count	Кількість підключень до одного і того ж хосту призначення за останні дві секунди	2
24	serror_rate	Відсоток з'єднань з хостом з count з SYN-помилками	0
25	rerror_rate	Відсоток з'єднань з хостом з count з REJ-помилками	0

Продовження таблиці 4.2

26	same_srv_rate	Відсоток з'єднань з хостом з count використовують одні і ті ж служби	1
27	diff_srv_rate	Відсоток з'єднань з хостом використовуючи різні служби	0
28	srv_count	Число з'єднань з однієї і тієї ж службою за останні дві секунди.	2
29	srv_serror_rate	Відсоток з'єднань з SYN-помилками при з'єднанні по службі з srv_count	0
30	srv_rerror_rate	Відсоток з'єднань з REJ-помилками при з'єднанні по службі з srv_count	0
31	srv_diff_host_rate	Відсоток з'єднань з різними хостами при з'єднанні по службі з srv_count	0
32	dst_host_count	Число з'єднань з тим же самим ip-адресою хоста призначення	150
33	dst_host_srv_count	Число з'єднань з тим же самим номером порту	25
34	dst_host_same_srv_rate	Відсоток з'єднань по тій же самій службі під час з'єднання з ip з dst_host_count	0.17
35	dst_host_diff_srv_rate	Відсоток з'єднань по різних служб під час з'єднання з ip з dst_host_count	0.03

Продовження таблиці 4.2

36	dst_host_same_src_port_rate	Відсоток з'єднань до того ж самому хосту приймача під час зв'язок через порт з dst_host_srv_count	0.17
37	dst_host_srv_diff_host_rate	Відсоток з'єднань з різними хостами приймачами під час зв'язок через порт з dst_host_srv_count	0
38	dst_host_serror_rate	Відсоток з'єднань з хостом з dst_host_count з SYN-помилками	0
39	dst_host_srv_serror_rate	Відсоток з'єднань з SYN-помилками при з'єднанні по службі з dst_host_srv_count	0
40	dst_host_rerror_rate	Відсоток з'єднань з хостом з dst_host_count з REJ -помилки	0.05
41	dst_host_srv_rerror_rate	Відсоток з'єднань з REJ -помилки при з'єднанні по службі з dst_host_srv_count	0

Кожен з сорока одного параметрів належить до певного типу даних (Таблиця 4.3).

Таблиця 4.3 – Типи даних параметрів NSL-KDD

Тип даних	Атрибути
Nominal	protocol_type, service, flag
Binary	land, logged_in, root_shell, su_attempted, is_host_login, is_guest_login

Продовження таблиці 4.3

Numeric	duration, src_bytes, dst_bytes, wrong_fragment, urgent, hot, num_failed_logins, num_compromised, num_root, num_file_creations, num_shells, num_access_files, num_outbound_cmds, count, srv_count, serror_rate, srv_serror_rate, rerror_rate, srv_rerror_rate, same_srv_rate, diff_srv_rate, srv_diff_host_rate, dst_host_count, dst_host_srv_count, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_serror_rate, dst_host_srv_serror_rate, dst_host_rerror_rate, dst_host_srv_rerror_rate

Сорок другий параметр містить інформацію про один з п'яти можливих типів з'єднання: нормальне, DOS-атака, Probing-атака, U2R-атака, R2L-атака. Нижче представлені короткі описи атак:

1) DOS (Denial of service). Відмова в обслуговуванні є атакою, яка виснажує ресурси жертви, тим самим роблячи її нездатною обробляти нормальні запити.

2) Probing. Мета атаки полягає в віддаленому спостереженні і зборі інформації про жертву.

3) U2R (User to root). Даний тип атак, використовує доступ до існуючої обліковим записом користувача в комп'ютерній системі жертви, отриманий, наприклад, методами соціальної інженерії, щоб, використовуючи вразливості, отримати доступ до облікового запису суперкористувача.

4) R2L (Remote to local attack). Неавторизований доступ з віддаленого комп'ютера дозволяє зловмисникові увійти в віддалену машину і отримати локальний доступ до зараженого комп'ютера [4].

Кожна з представлених вище атак, по суті своїй, є цілим класом, кожен з яких включає в себе безліч конкретних прикладів. Приклади класів та типів атак приведені в таблиці 4.4.

Таблиця 4.4 – Класи та приклади типів атак в NSL-KDD

Клас атак	Тип атак
DoS	Back, Land, Neptune, Pod, Smurf, Teardrop, Apache2, Udpstorm, Processtable, Worm
Probe	Satan, Ipsweep, Nmap, Portsweep, Mscan, Saint
R2L	Guess_Password, Ftp_write, Imap, Phf, Multihop, Warezmaster, Warezclient, Spy, Xlock, Xsnoop, Snmpguess, Snmpgetattack, Httptunnel, Sendmail, Named
U2R	Buffer_overflow, Loadmodule, Rootkit, Perl, Sqlattack, Xterm, Ps

Всі сорок два параметри створюють рядок, приклад якого зображений на рисунку 4.1.

```
13,tcp,telnet,SF,118,2425,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,26,10,0.38,0.12,0.04,0.00,0.00,0.00,0.12,0.30,guess_passwd,2
0,udp,private,SF,44,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,3,0.00,0.00,0.00,0.00,0.75,0.50,0.00,255,254,1.00,0.01,0.01,0.00,0.00,0.00,0.00,0.00,snmpguess,12
0,tcp,telnet,S3,0,44,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1.00,1.00,0.00,0.00,1.00,0.00,0.00,255,79,0.31,0.61,0.00,0.00,0.21,0.68,0.60,0.00,processtable,18
0,udp,private,SF,53,55,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,511,511,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,0.87,0.00,0.00,0.00,0.00,0.00,normal,17
0,tcp,private,SH,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1.00,1.00,0.00,0.00,1.00,0.00,0.00,16,1,0.06,1.00,1.00,0.00,1.00,1.00,0.00,0.00,nmap,17
0,tcp,http,SF,54540,8314,0,0,0,2,0,1,1,0,0,0,0,0,0,0,0,0,2,9,0.00,0.00,0.50,0.11,1.00,0.00,0.22,255,229,0.90,0.01,0.00,0.00,0.00,0.00,0.00,0.01,0.00,back,10
0,tcp,imap4,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,228,9,0.00,0.00,1.00,1.00,0.04,0.06,0.00,255,9,0.04,0.07,0.00,0.00,0.00,0.00,1.00,1.00,neptune,19
7570,tcp,telnet,SF,0,44,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,165,0.65,0.33,0.00,0.00,0.49,0.76,0.33,0.01,processtable,18
0,udp,private,SF,56,52,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,294,294,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,0.93,0.00,0.00,0.00,0.00,0.00,normal,17
0,tcp,ftp_data,SF,192,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.00,0.00,93,51,0.09,0.04,0.09,0.04,0.00,0.00,0.01,0.00,normal,20
0,tcp,other,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,511,1,0.14,0.00,0.86,1.00,0.00,1.00,0.00,255,1,0.00,0.94,0.00,0.00,0.13,0.00,0.87,1.00,satan,20
0,tcp,other,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,490,1,0.07,0.00,0.93,1.00,0.00,1.00,0.00,255,1,0.00,1.00,0.00,0.00,0.07,0.00,0.93,1.00,saint,19
0,tcp,telnet,SF,21,97,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,14,0.00,0.21,0.75,0.57,0.50,0.75,0.93,255,40,0.16,0.02,0.00,0.00,0.00,0.00,0.24,0.50,mscan,11
0,udp,private,SF,45,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,4,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,254,1.00,0.01,0.00,0.00,0.00,0.00,0.00,0.00,snmpguess,16
0,tcp,telnet,S3,0,44,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1.00,1.00,0.00,0.00,1.00,0.00,0.00,255,115,0.45,0.02,0.00,0.00,0.53,0.36,0.01,0.01,processtable,18
```

Рисунок 4.1 – Приклад даних NSL-KDD

В таблиці 4.5 наведен перелік параметрів бази даних NSL-KDD, що використовуються детектором для вивлення аномалій.

Таблиця 4.5 – Параметри NSL-KDD, для нейронного деетектору

duration	Час тривалості підключення
protocol_type	Протокол, який використовується при підключенні
service	Мережева служба, яка використовується підключенням
src_bytes	Кількість відправлених байт за одне з'єднання
dst_bytes	Кількість прийнятих байт за одне з'єднання
flag	Статус з'єднання - нормальне або з помилкою
land	Якщо ip-адреси хоста джерела і призначення рівні, і аналогічна ситуація з портами, то параметр приймає значення 1, інакше 0.
wrong_fragment	Загальна кількість невірних фрагментів за це підключення
urgent	Кількість urgent-пакетів в цьому підключенні
count	Кількість підключень до одного і того ж хосту призначення за останні дві секунди
srv_count	Число з'єднань з однієї і тієї ж службою за останні дві секунди.
serror_rate	Відсоток з'єднань з хостом з count з SYN-помилками
diff_srv_rate	Відсоток з'єднань з хостом використовуючи різні служби
srv_diff_host_rate	Відсоток з'єднань з різними хостами при з'єднанні по службі з srv_count
dst_host_srv_count	Число з'єднань з тим же самим номером порту

4.2 Архітектура модулю

Дані з вузлів пересилаються агентами і для кожного параметра обчислюється середнє. Ці дані об'єднуються в один вектор з даними мережевої активності, одержуваними від IDS.

У роботі дані хостів генеруються функцією, яка збільшує споживання ресурсів в певному відсотку атак. Дані мережі надаються модулю IDS в стандартизованном вигляді. У прототипі це записи NSL-KDD. Реалізован алгоритм виявлення відхилень від середнього, що зображений на рисунку 4.2

З метою поліпшення якості виявлення, вважається не одне відхилення, а середне відхилення для нейронів, що найкраще підходять. Потім для кластера обчислюється корінь із середнього арифметичного всіх значень відхилень, що входять до нього нейронів.

Для поданого на перевірку семпла даних шукається найближчий нейрон, розраховується відстань між ними і порівнюється з порахуваним среднеквадратическим відхиленням для кластера, до якого входить нейрон. Якщо відстань між семплом і нейроном більше відхилення по кластеру, даний семпл вважається аномальним.

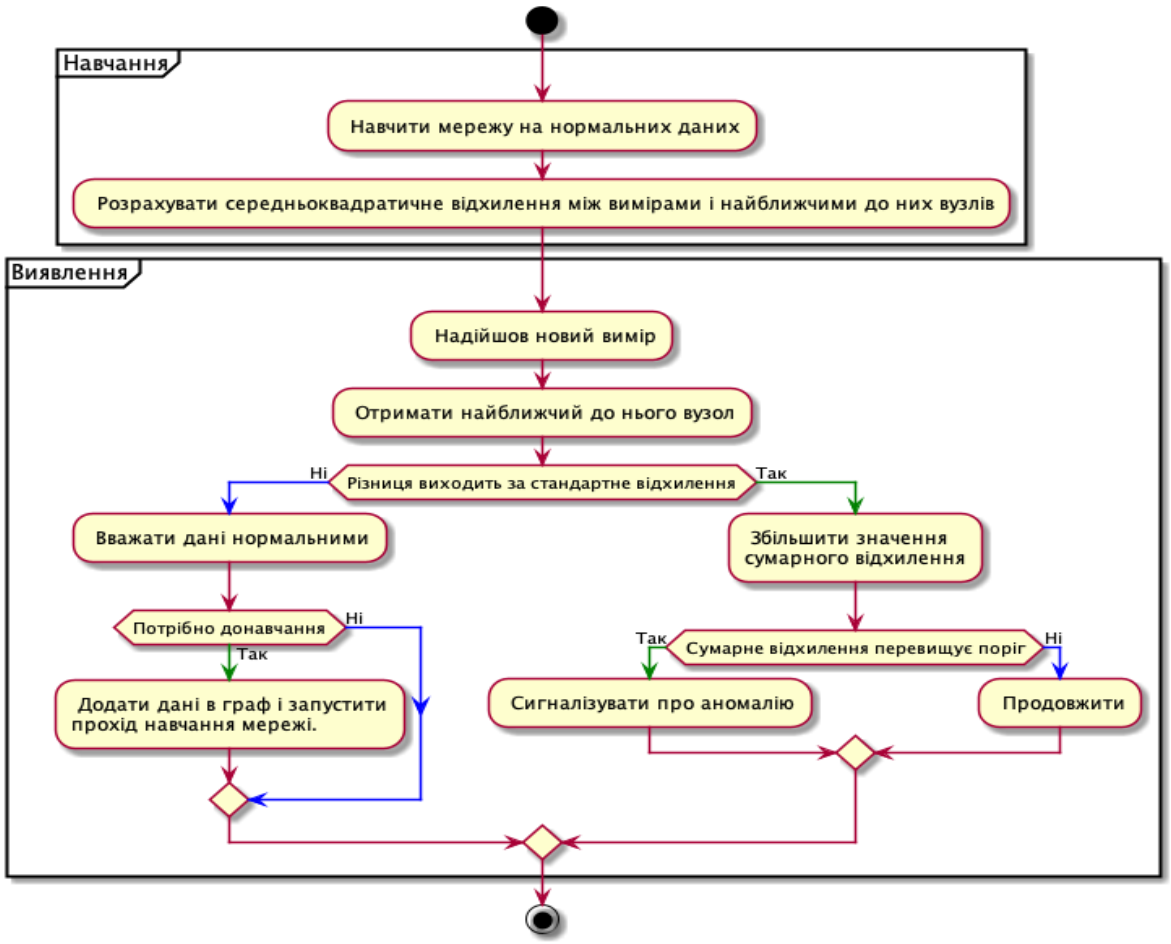

Рисунок 4.2 – Алгоритм виявлення відхилень

Ядро детектора є об'єктом, який реалізує один з алгоритмів "зростаючого нейроного газу" або GNG, який навчений на мережевій активності під час нормального функціонування системи передачі даних. GNG дозволяє кластеризувати дані, самостійно визначаючи необхідну кількість кластерів.

Модуль зберігає дані, на яких він навчався, у вигляді структури мережі і отримує нові дані під час роботи. На виході модуль повинен постійно виводити результат: є аномальна активність, або немає, або ймовірність наявності даної активності.

4.3 Побудова моделі GNG

Для моделювання роботи детектора на основі роботи алгоритму зростаючого нейроного газу було створено два набіра даних: навчалний та тестовий(Таблиця 4.6).

Таблиця 4.6 – Дані для моделювання праці детектора

	Повний набір даних	Кількість записів, які не містять аномалій	Кількість аномальних записів
Навчальна вибірка	1011	516	495
Тестова вибірка	11850	2152	9698

Спочатку детектор був навчений на навчальній вибірці. Кількість кроків навчання дорівнює 7000(рис. 4.3).

```
----------------------------------------------------------------------
GNG detector training...
----------------------------------------------------------------------
Reading normal activity from the file "NSL_KDD/Small Training Set.csv" [generated host data was not included]...
Records count: 516
Output images will be saved in: images
Training time = 0.0 s, Time per record = 6.006669628527737e-09 s Training step = 1 / 7000, Clusters count = 1, Neurons = 2
Training time = 0.05 s, Time per record = 9.2009703318278e-05 s Training step = 2 / 7000, Clusters count = 1, Neurons = 2
Training time = 0.09 s, Time per record = 0.00017719998840213746 s Training step = 3 / 7000, Clusters count = 1, Neurons = 2
Training time = 0.14 s, Time per record = 0.00026501934657725254 s Training step = 4 / 7000, Clusters count = 1, Neurons = 2
Training time = 0.18 s, Time per record = 0.0003568063410677651 s Training step = 5 / 7000, Clusters count = 1, Neurons = 2
Training time = 0.23 s, Time per record = 0.0004523197809855143 4 s Training step = 6 / 7000, Clusters count = 1, Neurons = 2
Training time = 3164.42 s, Time per record = 6.132590836794802 s Training step = 6995 / 7000, Clusters count = 78, Neurons = 351
Training time = 3165.15 s, Time per record = 6.134005873702293 s Training step = 6996 / 7000, Clusters count = 78, Neurons = 351
Training time = 3165.87 s, Time per record = 6.13540747784829 s Training step = 6997 / 7000, Clusters count = 78, Neurons = 351
Training time = 3166.6 s, Time per record = 6.136821319428525 s Training step = 6998 / 7000, Clusters count = 78, Neurons = 351
Training time = 3167.32 s, Time per record = 6.138216309307158 s Training step = 6999 / 7000, Clusters count = 78, Neurons = 351
Training complete, clusters count = 78, training time = 3168.05 s
Saving GIF file...
```

Рисунок 4.3 – Навчання детектору

Детектор пройшов навчання за 3169 секунд.

Після застосували детектор до нормальної активності навчальної вибірки, щоб отримати відсоток помилкових спрацьовувань.

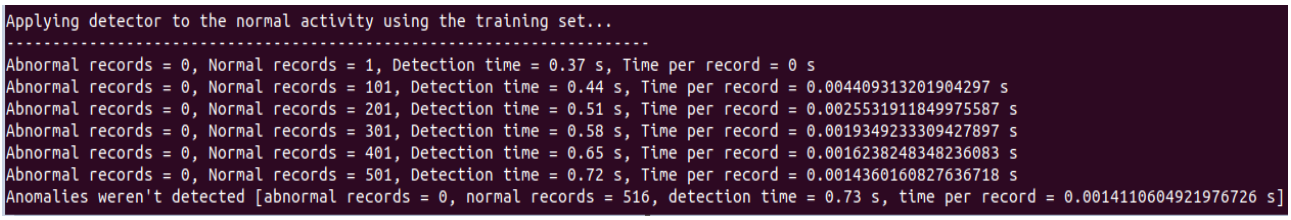

Рисунок 4.4 – Застосування детектору до нормальної активності навчального набору даних

Як видно із рисунку 4.4 відсоток помилкових повідомлень про аномалії на навчальному наборі даних дорівнює 0.

Наступним кроком було застосування детектора на аномальній активності навчальної вибірки для того щоб отримати відсоток знайдених аномалій для повного набору даних із якого була сформована навчальна вибірка.

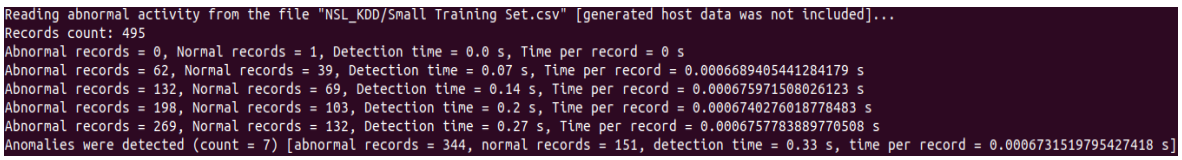

Рисунок 4.5 – Застосування детектору до аномальної активності навчального набору даних

З 495 аномальних записів детектор виявив 344(рис.4.5). Відсоток знайдених аномалій для повного набору даних із якого була сформована навчальна вибірка дорівнює 69.5.

Отже як було отримано результати на навчальному наборі даних переходимо до тестової вибірки.

Спочатку, як і для навчального набору, застосовуєм детектор до нормальної активності тестовою вибірки, яка містить 2152 записи, для отримання відсотку помилкових повідомлень про аномалію.

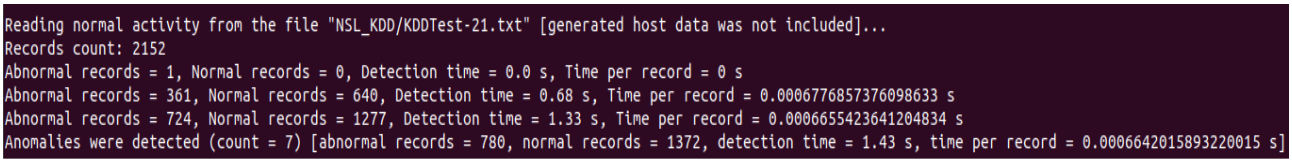

Рисунок 4.6 – Застосування детектору до нормальної активності тестового набору даних

Детектор знайшов 780 аномалій в 2152 записах, які не містять аномальної поведінки, шо зображено на рисунку 4.6. Отже відсоток помилкових спрацівать сладає 36.2.

Наступним кроком є застосування детектору на аномальній активності тестого набору даних.

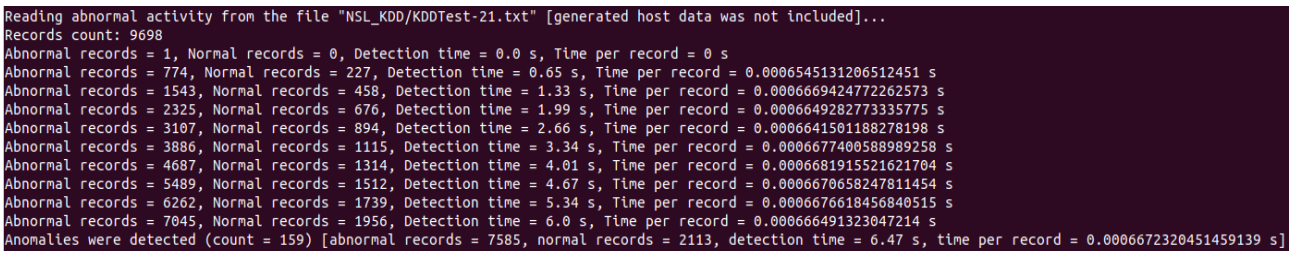

Рисунок 4.7 – Застосування детектору до аномальної активності тестового набору даних

Було виявленно 7585 аномальних записів із 9698. Відсоток знайдених аномалій для повного набору даних тестової вибірки.

Внесемо результати у таблицю 4.7

Таблиця 4.7 – Результати детектору

Кількість аномалій в навчальному наборі даних	495
Кількість знайдених аномалій в навчальному наборі даних	344
Відсоток знайдених аномалій в навчальному наборі даних	69.5
Кількість аномалій в тестовому наборі даних	9698
Кількість знайдених аномалій в тестовому наборі даних	7585
Відсоток знайдених аномалій в тестовому наборі даних	78.2
Відсоток помилкових повідомлень про аномалію на навчальному наборі даних	0
Відсоток помилкових повідомлень про аномалію на тестовому наборі даних	36.2

ВИСНОВКИ

На сьогоднішній день з метою виявлення мережевих атак досить широко використовуються системи виявлення вторгнень, робота яких базується на використані різних моделій аналізу мережевого трафіку. Проведені дослідження показали, що на сучасному етапі з метою блокування гібридних атак досить ефективно можуть використовуватись моделі побудовані на основі нейронних мереж. Подібні системи досить легко впроваджуються в існуючі інструментальні засоби та мають можливість до адаптованого навчання, а також здатні працювати з великими об’ємами даних, що є актуальним при опрацювані мережного трафіку.

У даній роботі було розглянуто типи аномалій мережевого трафіку та види систем виявлення вторгнень(СВВ), методи побудови систем виявлення вторгнень та проведений їх порівняльний аналіз.

Проаналізовані методи інтелектуального аналізу даних за допомогою нейроних мереж та запропонована модель нейронного детектору на основі зростаючого нейронного газу(GNG). Модель успішно пройшла навчання на вибірці. Під час тестування були виявлено, що детектор виявив 69,5 відсотків аномалій на навчальній вибірці з помилкою 0 відсотків, а на тестовій вибірці відсоток виявлених аномалій складає 78,2, а відсоток помилки 36,6.

Проведені дослідження показали, що створена система має право на існування та здатна виявляти левову частку аномалій, але використовувати її слід у комбінації з іншими подібними системами, оскільки даний алгоритм нейронної мережі має багато гіперпараметрів, які потребують оптимізації та корегування під конкретну задачу. Перевагами даної мережі є здатність до самоорганізації та самонавчання після першого-базового навчання.

ПЕРЕЛІК ПОСИЛАНЬ

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. О.В. Сєвєрінов, А.Г. Хрєнов. Системи обробки інформації, випуск 6 (122). 2013.

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

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

5. 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

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

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

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

12. 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.

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

14. Paxson V. Bro: A system for detecting network intuders in realime // Proceedings of the 7th USENIX Security Symposium, San Antonio, TX, USA, January 1988.

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

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

17. 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.

18. Yaroshkin F. SnortNet – A Ditributed Intrusion Detection System // IVT-1/95, Kyrgyz Russian Slavic University, Bishkek, Kyrgystan, June 2000.

19. Bernd Fritzke. A Growing Neural Gas Network Learns Topologies [Електроний ресурс]. – Режим доступу: https://papers.nips.cc/paper/893-a-growing-neural-gas-network-learns-topologies.pdf.