Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя
(повне найменування вищого навчального закладу)
Комп'ютерно-інформаційних систем і програмної інженерії
(назва факультету)
Комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

магістр
(назва освітнього ступеня)

на тему: **Системний підхід до виявлення комп'ютерних атак засобами електронних приманок**

Виконав(ла): студент(ка) 6 курсу, групи САм-61
спеціальності 124 «Системний аналіз»
(шифр і назва спеціальності)

(підпис) Павлов І.Р. (прізвище та ініціали)

Керівник (підпис) Матійчук Л.П. (прізвище та ініціали)

Нормоконтроль (підпис) Мацюк О. В. (прізвище та ініціали)

Завідувач кафедри (підпис) Боднарчук І. О. (прізвище та ініціали)

Рецензент (підпис) Жаровський Р.О. (прізвище та ініціали)

Тернопіль
2021

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет Комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра Комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
« » 20___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня магістр
(назва освітнього ступеня)

за спеціальністю 124 «Системний аналіз»
(шифр і назва спеціальності)

студенту Павлову Іллі Руслановичу
(прізвище, ім'я, по батькові)

1. Тема роботи Системний підхід до виявлення комп'ютерних атак засобами електронних приманок

Керівник роботи Матійчук Любомир Павлович, к.е.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «*28*» *жовтня* 2021 року № *4/7-910*.
2. Термін подання студентом завершеної роботи
3. Вихідні дані до роботи Наукові публікації, електронні ресурси, підручники , посібники з тематики дослідження

4. Зміст роботи (перелік питань, які потрібно розробити) Вступ. 1. Аналіз предметної області постановка задачі дослідження. 2. Розробка модуля ідентифікації атак. 3. Розроблення програмного модуля виявлення атак. 4. Охорона праці та безпека в надзвичайних ситуаціях. Висновки..

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1. Актуальність. 2. Кількість атак в мережі Інтернет.3. Життєвий цикл типової атаки. 4. Характеристики атак. 5. Схема мережі Honeynet. 6. Теоретичне обґрунтування методу виявлення атак. 7.Архітектура багаторівневого персептрону. 7.Цикл роботи MD5. 8. Алгоритм навчання нейронної мережі. 9.Модифікований алгоритм MD5 та алгоритм класифікції нейронної мережі. 10. Узагальнена структура системи ідентифікації атаки. 11. Процес навчання нейронної мережі. 12. Тестування системи. 13. Висновки. 14. Завершальний слайд.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	Приймак М.В., проф.		
Безпека в надзвичайних ситуаціях	Клепчик В.М., ст.викл.		

7. Дата видачі завдання 27 вересня 2021 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	27.09.2020-29.09.2020	*Виконано*
2.	Підбір наукових джерел щодо виявлення	30.09.2020-03.10.2020	*Виконано*
	комп'ютерних атак засобами електронних приманок		
3.	Переклад та опрацювання наукових джерел щодо	04.10.2020-10.10.2020	*Виконано*
	виявлення комп'ютерних атак засобами електронних		
	приманок		
4.	Виконання дослідження щодо виявлення	11.10.2020-17.10.2021	*Виконано*
	комп'ютерних атак засобами електронних приманок		
5.	Оформлення розділу «Аналіз предметної області	18.10.2021-24.10.2021	*Виконано*
	постановка задачі дослідження»		
6.	Оформлення розділу «Розробка модуля ідентифікації	25.10.2021-31.10.2021	*Виконано*
	атак»		
7.	Оформлення розділу «Розроблення програмного	01.11.2021-07.11.2021	*Виконано*
	модуля виявлення атак»		
8.	Виконання завдання до підрозділу «Охорона праці»	08.11.2021-11.11.2021	*Виконано*
9.	Виконання завдання до підрозділу «Безпека в	12.11.201-14.11.2021	*Виконано*
	надзвичайних ситуаціях»		
10.	Оформлення кваліфікаційної роботи	15.11.201-24.11.2021	*Виконано*
11.	Нормоконтроль	25.11.2021-28.11.2021	*Виконано*
12.	Перевірка на плагіат	01.12.2021	*Виконано*
13.	Попередній захист кваліфікаційної роботи	07.12.2021	*Виконано*
14.	Захист кваліфікаційної роботи	20.12.2021	

Студент _____ Павлов І.Р.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Матійчук Л.П.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Системний підхід до виявлення комп'ютерних атак засобами електронних приманок // Кваліфікаційна робота освітнього рівня «Магістр» // Павлов Ілля Русланович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра комп'ютерних наук, група САм-61 // Тернопіль, 2021 // с. 71, рис. – 26, табл. – 1,бібліогр. – 59, додат. – 6.

Ключові слова: комп'ютерні атаки, структура Honeynet, ідентифікації атак.

У кваліфікаційній роботі запропоновано системний підхід до виявлення комп'ютерних атак засобами електронних приманок. Розроблено метод ідентифікації комп'ютерних атак. Запропонований алгоритм ідентифікації комп'ютерних атак.

На основі системного підходу та дослідження було розроблено програмний модуль для виявлення комп'ютерних атак.

ANNOTATION

System approach to cyber-attacks detection using electronic lures // Qualification thesis Master Degree // Pavlov Ilya Ruslanovich // Ternopil Ivan Puluj National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Computer Sciences, group SAM-61 // Ternopil, 2021 // p. 71, Fig. - 26, table. - 1, bibliogr. - 59, add. - 6.

Keywords: computer attacks, Honeynet structure, attack identifications.

The qualification work proposes a systematic approach to the detection of computer attacks by means of electronic lures. A method of identifying computer attacks has been developed. An algorithm for identifying computer attacks has been proposed.

Based on a systems approach and research, a software module for detecting computer attacks has been developed.

ЗМІСТ

ВСТУП

Актуальність теми. Створення і надзвичайно інтенсивний ріст глобальної мережі Інтернет спричинив появу глобального інформаційного суспільства. Розвиток бізнесу на основі розподілених інформаційних ресурсів може ефективно функціонувати на міжнародному рівні, при цьому інтенсивно обмінюючись інформацією між різними підрозділами, учасниками, постачальниками та покупцями. Уряди різних країн використовують глобальну мережу для представлення інформації своїм громадянам та світовій спільноті в цілому, також вони все більше використовують Інтернет для заміни ручних методів збору інформації та забезпечення урядових потреб.

Такий бурхливий розвиток інформаційних технологій і глобальної мережі зокрема, має і деякі негативні сторони. Людина з необхідними знаннями та здібностями, в наявності якої є комп'ютер, підключений до Інтернет має можливість завдати значної шкоди. Комп'ютерний вірус "I love you", наприклад, завдав людству збитків на приблизно 6,7 мільярдів доларів США [1, 54]. Такого роду дані наводяться досить обережно, так як методика їх обчислення не завжди правильна. Однак, це може бути невеликим аргументом про величину збитків.

Усі переваги мережі Інтернет і глобального інформаційного суспільства, включаючи підтримку базових і дуже важливих служб, від яких залежить держава, є мішенню для атак, які використовують глобальну комп'ютерну мережу. На сьогоднішній день зловмисники використовують швидкість і глобальний зв'язок мережі Інтернет для здійснення атак. Добре підготовлені атаки практично неможливо відслідковувати, використовуючи наявні на сьогоднішній день відповідні інструменти. Комп'ютерні зловмисники сьогодення дуже суворо дотримуються принципу анонімності під час здійснення неправомірних дій в області несанкціонованого доступу до мережевих ресурсів.

Із самого початку мережа створювалася як незахищена відкрита система призначена для інформаційного спілкування все зростаючого числа користувачів, а Інтернет як незахищена система, не призначена для зберігання і обробки конфіденційної інформації. Більш того, захищений Інтернет не зміг би стати тією системою, якою він зараз є і не перетворився б на інформаційний образ світової культури її минулого і сьогодення.

Розвиток засобів безпеки Інтернет на сьогодні є досить складною задачею. Більш правомірна постановка питання про створення спеціалізованої безпечної світової інфраструктури, призначеної для управління світовим виробництвом транспорту, геополітикою. У недалекому майбутньому, прогрес приведе до необхідності створення єдиної системи, де середовище спілкування володітиме архітектурою безпеки і гарантувати цілісність і конфіденційність інформації. Така система повинна буде забезпечити дотримання політичних і економічних інтересів світових суб'єктів.

Мета і задачі дослідження. Метою роботи є класифікувати та описати існуючі методи виявлення комп’ютерних атак, методи та засоби ідентифікації неправомірного доступу до інформації, а також розробити власний підхід до часткового або цілковитого вирішення даної проблеми.

Для вирішення поставленої мети вирішуються наступні завдання:

1) проаналізувати недоліки існуючих підходів до виявлення комп’ютерних атак;

2) розробити і реалізувати методи та засоби ідентифікації неправомірного доступу до інформації;

3) здійснити програмну реалізацію запропонованих математичних методів;

4) провести тестування та апробацію розроблених методів та програмних засобів.

Об’єкт дослідження – процеси виявлення комп’ютерних атак.

Предмет дослідження – методи та програмні засоби виявлення комп’ютерних атак.

Методи дослідження. Дослідження проводилися на базі комплексного системного аналізу, теорії комп'ютерних мереж.

Наукова новизна одержаних результатів. Запропоновано та удосконалено метод ідентифікації атак на основі сигнатур.

Апробація результатів кваліфікаційної роботи. Основні положення та результати проведених досліджень доповідались та обговорювались на IX науково-технічній конференції «Інформаційні моделі, системи та технології» Тернопільського національного технічного університету імені Івана Пулюя (м. Тернопіль, 2021 р.). Публікації. Основні результати кваліфікаційної роботи опубліковані у двох працях науково-технічної конференції (Див. додаток А).

Структура й обсяг кваліфікаційної роботи. Кваліфікаційна робота складається зі вступу, чотирьох розділів, висновків, списку літератури з 59 найменувань та 6 додатків.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

1.1. Методи і засоби здійснення комп'ютерних атак

Атака – це будь-яка дія порушника, яка призводить до реалізації загрози шляхом використання вразливостей ОС [8].

В загальному випадку життєвий цикл будь-якої атаки може бути розділений на чотири стадії (рис. 1.1).

Рекогносцировка – порушник старається отримати якомога більше інформації про об'єкт атаки, щоб на її етапі спланувати подальші етапи вторгнення.

Рисунок 1.1 — Життєвий цикл типової атаки

Об'єктами атаки можуть бути різноманітні сервіси та додатки, які мають певні недоліки у функціонуванні на різних рівнях OSI (рис. 1.2, 1.3, 1.4).

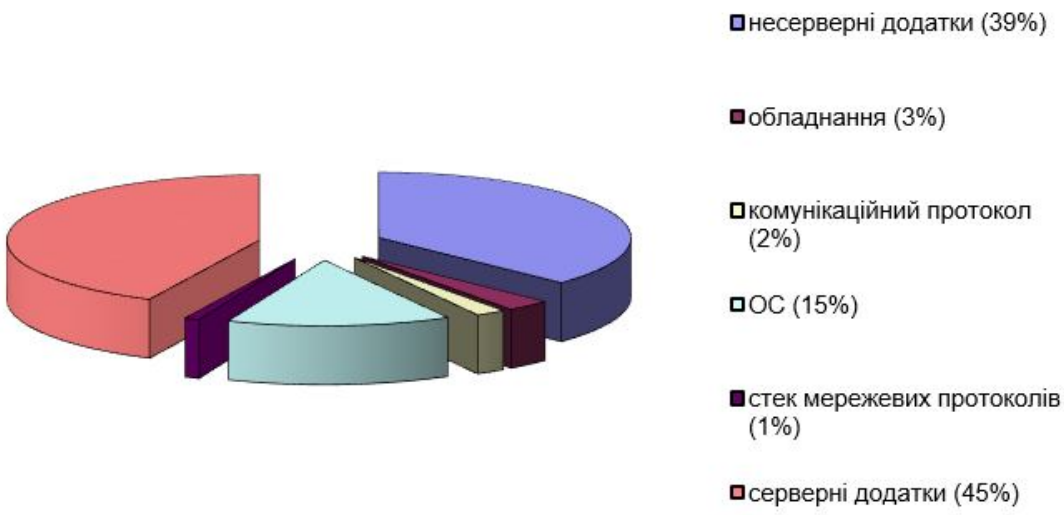


Рисунок 1.2 — Об’єкти атак

Вторгнення - на цьому етапі порушник отримує несанкціонований доступ до ресурсів тих хостів, які є об’єктами атаки [8].

Рисунок 1.3 — Доля атак за їх типами

Атакуючі дії – це набір заходів, внаслідок яких реалізуються ті цілі, заради яких здійснювалася атака. Наприклад, порушення працездатності ІС, крадіжка конфіденційної інформації, що зберігається в системі, знищення чи модифікація даних і т.д. Зловмисник старається розширити об’єкти атаки, щоб продовжити несанкціоновані дії на інших частинах ІС.

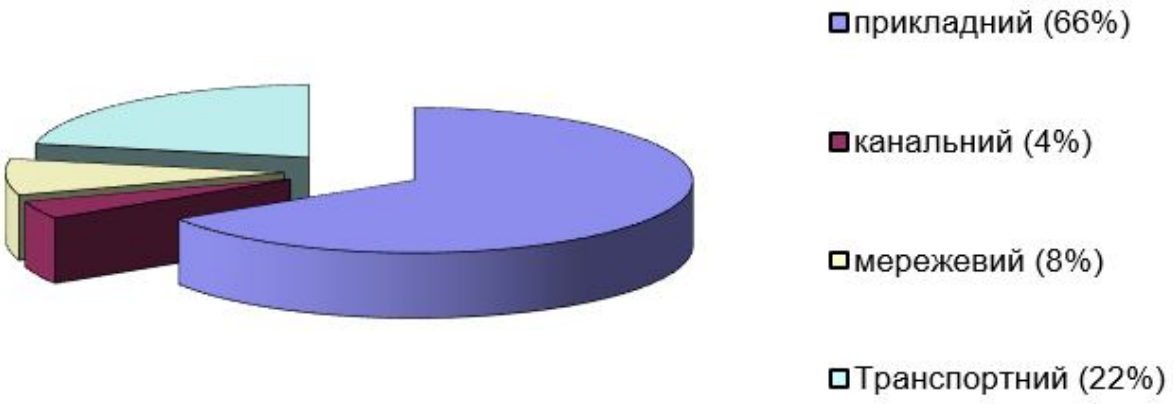


Рисунок 1.4. — Доля реалізованих атак за рівнями OSI

Атаки класифікуються на п’ять груп за наступними ознаками:

1) По характеру дії:
   - пасивні
   - активні

Пасивні дії на розподілену ОС – дії, які не здійснюють безпосереднього впливу на роботу системи, але можуть порушувати її політику безпеки. Пасивні віддалені дії практично неможливо виявити [9]. Активні дії на розподілену ОС – дії, які здійснюють безпосередній вплив на роботу системи (зміни конфігурації, порушення працездатності т.д.) і порушують прийняту в ній політику безпеки. Практично всі типи віддалених атак являються активними діями. Особливістю активної дії в порівнянні з пасивною є принципова можливість її виявлення, так як в результаті її здійснення в системі відбуваються певні зміни. На відміну від активної дії, при пасивному впливі не залишається ніяких слідів [10].

2) В залежності від цілевого призначення дії:
   - порушення конфіденційності інформації
   - порушення цілісності інформації
   - порушення працездатності (доступності) системи

При перехопленні інформації порушується її конфіденційність. При зміні інформації порушується її цілісність. При порушенні працездатності не відбувається несанкціонованого доступу, тобто зберігається цілісність і

конфіденційність інформації, однак доступ до неї легальних користувачів також неможливий [11].

3) За умовою початку здійснення дій:

- Атака за запитом від об’єкту, що здійснює атаку
- Атака за настанням очікуваної події на об’єкті атаки
- Безумовна атака

У випадку запиту, зловмисник очікує передачі від потенційної цілі атаки запиту визначеного типу, який і буде умовою початку здійснення певних дій. Ініціатором здійснення початку атаки є атакуючий об’єкт. Прикладом можуть бути DNS і ARP запити в стеці TCP/IP. У випадку настання події, атакуючий здійснює постійний нагляд за станом операційної системи віддаленої цілі атаки і при виникненні певної події в цій системі починає діяти. Ініціатором здійснення початку атаки являється об’єкт атаки. Прикладом може бути переривання сеансу роботи користувача з сервером в мережених ОС без виконання команди LOGOUT. У випадку безумовної атаки, початок її здійснення безумовний по відношенню до цілі атаки, тобто атака здійснюється негайно і незалежно до стану системи і атакуючого об’єкта. Відповідно, в цьому випадку атакуючий і є ініціатором початку здійснення атаки [11].

4) По кількості атакуючих:

- Розподілена
- Нерозподілена

Розподілена атака – атака, в якій приймають участь два або більше атакуючих об’єктів на одну і ту ж ОС, об’єднані одним задумом, з однаковою метою. Нерозподілена атака здійснюється одним атакуючим.

5) За джерелом здійснення:

- Зовнішня
- Внутрішня

Внутрішні атаки здійснюються з локальної мережі, в більшості випадків її виконавцями є самі співробітники певної фірми чи організації. Зовнішня

атака здійснюється з глобальної мережі Інтернет. Проте найбільш поширеними є зовнішні атаки (рис. 1.5).

Методи виявлення атак на мережеві ресурси є дуже важливою ланкою в побудові системи комп'ютерної безпеки. Виявлення атак – це процес ідентифікації і реагування на підозрілу діяльність, направлену на обчислювальні або мережеві ресурси [8]. Всі механізми виявлення атак базуються на декількох загальних методах. Необхідно зауважити, що всі нижчеописані методи не являються взаємовиключними. В багатьох системах використовується комбінація декількох методів.

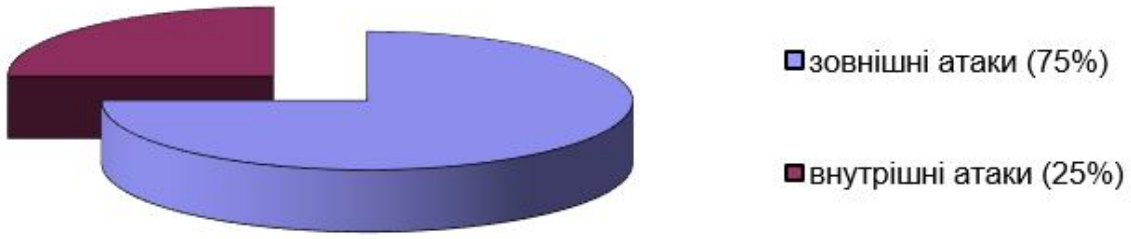


Рисунок 1.5 — Доля атаки за джерелом їх здійснення

Аналіз журналів реєстрації (ЖР) – це один із найперших реалізованих методів виявлення атак. Він заключається в аналізі ЖР (log, audit, trail), які створюються операційною системою, прикладним програмним забезпеченням, маршрутизаторами і т.д. Записи ЖР аналізуються і інтерпретуються системою виявлення атак [12].

До переваг цього методу відноситься простота його реалізації. Однак за цією простотою скривається немало недоліків. По-перше, для достовірного виявлення цієї чи іншої підозрілої діяльності необхідна реєстрація в журналах великого об'єму даних, що негативно відбивається на швидкості роботи системи. По-друге, при аналізі ЖР дуже важко обійтися без допомоги спеціалістів, що суттєво знижує область поширення цього методу. По-третє, до даного моменту немає уніфікованого формату зберігання журналів. І, на

кінець, аналіз вже записаних в ЖР записів говорить про те, що аналіз здійснюється не в реальному режимі часу [13].

Як правило аналіз ЖР являється доповненням до інших методів виявлення атак, а саме, до виявлення атак в реальному масштабі часу. Використання цього методу дозволяє проводити подальші дії вже після того, коли була зафіксована атака, для того щоб виробити ефективні міри запобігання аналогічних атак в майбутньому. Аналіз в реальному режимі часу – це метод, який заключається в моніторингу мереженого трафіку в реальному або близькому до реального часу і використовуванні відповідних алгоритмів виявлення. Дуже часто використовується механізм пошуку в трафіку визначених стрічок, які можуть характеризувати несанкціоновану діяльність. До таких стрічок можна віднести '\\WINNT\SYSTEM32\CONFIG' (дана стрічка описує шлях до файла Sam, Security і т.д.) або '/etc/shadow' (дана стрічка описує шлях до списку паролів ОС Unix) [14].

Використання методу виявлення атак в мереженому трафіку дає дві основні переваги. По-перше, один агент системи виявлення атак може продивлятися цілий сегмент мережі з багатьма хостами, в той час як, для попереднього методу необхідно на кожен вузол, що аналізується встановлювати свій агент. Цей метод дозволяє виявляти атаки проти всіх елементів підприємства, починаючи з атак на маршрутизатори і закінчуючи атаками на прикладні програми. По-друге, системи, побудовані з врахуванням цього методу, можуть визначати атаки в реальному масштабі часу і зупиняти атаки до досягнення ними цілі.

Використання профілів „нормальної” поведінки також досить часто використовується для виявлення комп’ютерних атак. Профілі нормальної поведінки використовуються для спостерігання за користувачами, системною діяльністю або МТ. Дані спостережень порівнюються з очікуваними значеннями профілю нормальної поведінки, який будується в період навчання системи виявлення атак [15].

Цей метод рідко використовується в сучасних системах захисту інформації (хоча такі спроби і робляться). Використання профілів знайшло

своє практичне застосування в системах виявлення шахрайства (fraud detection systems), що використовуються в фінансових структурах або операторами зв'язку.

Використання сигнатур атак – метод, який дуже часто співпадає з аналізом в реальному режимі часу. Метод заключається в описі атаки у вигляді сигнатури (signature) і пошуку даної сигнатури в підконтрольному просторі (МТ, ЖР і т.д.). В якості сигнатури атаки може виступати шаблон дій або стрічка символів, що характеризують аномальну діяльність. Ці сигнатури зберігаються в базі даних, яка аналогічна до тієї, яка використовується в антивірусних системах. Власне кажучи, антивірусні резидентні монітори являються окремим випадком системи виявлення атак, але так як ці напрямки початково розвивались паралельно, то прийнято розділяти їх. Перевагою сигнатурного методу (СМ) є його висока точність роботи, а очевидним недоліком - неможливість виявлення тих атак, сигнатури яких не визначені за допомогою методів [16].

Ефективне виявлення атак на етапах атакуючих дій і розвитку атаки можливе тільки за допомогою поведінкових методів, оскільки дії порушників залежать від цілі атаки і фіксованої множини сигнатур атак однозначно не визначаються. Враховуючи той факт, на двох останніх стадіях життєвого циклу інформаційної атаки, найхарактернішими об'єктами являються хости, в цьому випадку найбільш доцільне застосування хостових датчиків (таблиця 1.1).

Таблиця 1.1.

Застосування поведінкового і СМ для визначення різних стадій атак

Стадія атаки	Метод визначення	
	Сигнатурний	Поведінковий
Рекогносцировка	+СХ	-
Вторгнення в ІС	+СХ	+СХ
Атакуючі дії	-	+Х
Розвиток	-	+Х

Примітка: + – допустимо; - – недопустимо; СХ – використовуються мережеві і хостові датчики; Х – використовуються хостові датчики.

Незважаючи на ефективність і простоту реалізації цих методів, в системах, які їх реалізують, проблеми також існують. Перша проблема заключається в створенні механізму опису сигнатур, тобто мови опису атак. А друга проблема, плавно випливає з першої, – як записати атаку, щоб зафіксувати всі її можливі модифікації. Необхідно відзначити, що перша проблема вже частково вирішена в деяких продуктах. Наприклад, система опису мережевих атак Advanced Packets Exchange, реалізована компанією Internet Security Systems, Inc. і пропонується спільно з розробленою системою аналізу захищеності Internet Scanner.

При застосуванні статистичного методу у системі, що аналізується, спочатку визначаються профілі для всіх її суб'єктів. Будь-яке відхилення (дисперсія) профілю від еталонного вважається несанкціонованою діяльністю. Основні переваги статистичного підходу – це адаптація до поведінки суб'єкта і використання вже розробленого і зарекомендованого себе апарату математичної статистики. Однак при використанні цих методик виникає і декілька проблем.

По-перше, „статистичні" системи можуть бути з часом „навчені" порушниками так, щоб атакуючі дії розглядалися як нормальні.

По-друге, „статистичні" системи не чутливі до порядку слідування подій. А в деяких випадках одні і ті ж події в залежності від порядку їх слідування можуть характеризувати аномальну або нормальну діяльність. І, на кінець, дуже важко задати граничні (порогові) значення характеристик, які відслідковуються системою виявлення атак, щоб адекватно ідентифікувати аномальну діяльність.

Використання прогнозованих шаблонів – спосіб, який дозволяє „передбачати" майбутні події на основі вже минулих. Проблема в цьому випадку полягає в тому, що якщо в базі знань не описані деякі сценарії виявлення атак, то такі дії не будуть визначені як атакуючі [10]. Даний метод має декілька переваг. По-перше, правила, які базуються на послідовності шаблонів, можуть визначати аномальну активність, яку важко ідентифікувати традиційними методами. По-друге, системи, побудовані на цій моделі, дуже

добре пристосовані до змін. Це зв’язано з тим, що шаблони, які використовуються рідко, безперервно знищуються, залишаючи найбільш часто використовувані шаблони. По-третє, простіше виявити користувачів, які „навчають” систему виявлення аномалій протягом вивчення прикладної системи. По-четверте, аномальні дії можуть бути виявлені протягом декількох секунд після генерації події [17].

У випадку аналізу переходів із стану в стан, будь-яка дія в системі, в тому числі атака, записуються як послідовність переходу із одного стану в інший. Система виявлення атак, яка працює по цьому принципу, контролює поточний стан системи, що аналізується шукаючи події, які змінюють даний стан (тобто шукають умову переходу). Цей процес триває до тих пір, поки не буде досягнутий кінцевий стан, тим самим характеризуючи виявлення атаки [18].

1.2. Структура Honeynet та оцінка існуючих систем виявлення атак

Honeynet Project - це наукова організація, що займається дослідженнями в області систем безпеки і спеціалізується на вивченні тактики і мотивів, що використовується зловмисниками. До складу організації входять фахівці з питань безпеки з різних країн, які на добровільній основі надають свої ресурси для розгортання і вивчення мереж-приманок, основне призначення яких стати об'єктом атаки хакерів [19].

Проект був ініційований в 1999 році невеликою групою фахівців як неформальний список розсилки. Проте незабаром стало зрозуміло, що жоден фахівець сам з себе не володіє всім досвідом, необхідним для аналізу зібраних відомостей про атаки. Ряд учасників проекту поступово розширювався, і в червні 2000 року він отримав офіційну назву Honeynet Project. Проект розділяє свою діяльність на чотири етапи [20]:

- Перший етап почався в 1999 році і продовжувався два роки. Його метою було підтвердження основоположної ідеї: створення, розгортання і тестування технології мереж-приманок і їх можливості збирати інформацію

про діяльність хакерів. За першими мережами-приманками закріпилася назва GENI. Їх структура була вельми далека від досконалості, вони спиралися лише на базові механізми і не мали методів для збору відомостей про зашифровані дії зловмисників. Проте вони ефективно виявляли більшість автоматизованих атак.

- Другий етап почався в 2002 році і тривав приблизно два роки. Його основна мета - удосконалити можливості мереж-приманок і спростити роботу з ними. Передбачалося розвернути мережі GENII, які відрізнятимуться досконалішими методами моніторингу і контролю дій хакерів. У 2002 році у Вашингтоні була розгорнена перша бездротова мережа-приманка. Вона значно покращувала можливості збору інформації, що особливо стосується зашифрованих передач.
- Третій етап почався в 2003 році і розрахований був приблизно на рік. За цей період планувалося створити засоби, що дозволяють розміщувати технології розгортання мереж-приманок GENII на завантажуваному компакт-диску. Передбачається, що організації просто завантажуватимуть диск, що діє як шлюз мережі-приманки, розгортаючи тим самим все, що необхідне для роботи мереж-приманок, зокрема засобу реєстрації в глобальній базі даних повної інформації про діяльність хакерів.
- Четвертий етап почався в 2004 році. Його мета - розробити централізовану систему збору даних, яка погоджує відомості, що отримуються з безлічі розподілених мереж-приманок, і надає інтерфейси для їх аналізу. Робота в рамках даної концепції, пов'язана із створенням двох призначених для користувача інтерфейсів.

Honeynet - це засіб навчання, мережа машин з системами, що використовуєься в повсякденній діяльності і призначена для компрометації. В процесі компрометації інформація фіксується, а потім аналізується, з метою вивчення поведінки чорних капелюхів [21]. Ідея така, що і у honeypots, але є декілька відмінностей. Honeypot по суті звичайна приманка, тобто система спеціально побудована щоб бути атакованою. Взагалі кажучи, терміном "honeypot" прийнято називати системи, що моделюють відомі уразливості,

емулюючи інші системи або робочі системи, модифіковані так, щоб на них створилося замкнуте середовище. Приклади таких систем-пасток - The Deception Toolkit, CyberCop Sting і Mantrap.

Honeynet має свої особливості. Два її найголовніших конструктивних відмінності від класичного honeypot полягають в наступному [19]:

- Це не одинична система, а мережа, що складається з багатьох комп'ютерів. Ця мережа розташовується за міжмережевим екраном (firewall), на якому всі вхідні і витікаючі дані фіксуються і контролюються. Отримана інформація надалі аналізується з метою вивчення засобів, тактики і мотивів співтовариства чорних капелюхів. Honeynet може використовувати безліч систем одночасно: Solaris, Linux, Windows NT, маршрутизатор Cisco, комутатор Alteon, Web-сервери. Це створює мережеве середовище, яке реалістичніше моделює реальну мережу. Також, маючи різні системи з різними сервісами, наприклад, Linux як сервер DNS, Windows NT як веб-сервер-сервер і Solaris як сервер FTP, ми можемо дізнатися про різні засоби і тактиків. Можливо, деякі чорні капелюхи шукають тільки певні системи, додатку або уразливості. Маючи набір різних операційних систем і додатків, ми зможемо точніше профілювати специфічні дії чорних капелюхів і їх характеристики.
- Всі системи поміщені в межах Honeynet, стандартні системи, що використовуються в звичайних мережах. У цих системах нічого не емулюється, нічого також і не зроблено для штучного ослаблення захисту. Ризик і вразливості, що виявляються за допомогою Honeynet - ті ж самі, які існують сьогодні в багатьох організаціях.

Як правило захист інформації у системі носить оборонний характер. Міжмережеві екрани (firewalls), системи виявлення вторгнень, що мають назву Intrusion Detection Systems (IDS), криптографічні методи - всі ці механізми використовуються оборонно, щоб захистити власні ресурси [22, 38]. Така стратегія дозволяє захистити власність організацій і полягає в тому, щоб виявляти будь-які недоліки в системі захисту і тут же їх усувати. Проблема цього підходу в тому, що він повністю оборонний, тоді як супротивник активно атакує. Honeynet покликана змінити цю ситуацію, вона дозволяє

перейти від глухої оборони до активних дій, узяти ініціативу на себе. Первинна мета Honeynet полягає в тому, щоб зібрати розвідувальні дані про супротивника, вивчити його засоби, тактику і мотиви. Збираючи таку інформацію можна краще зрозуміти, що нам загрожує і як краще захиститися від цих погроз. Захист інформації часто порівнюють з військовими діями, такими як захист замку або партизанська війна. Незалежно від вибраної нами аналогії, ми завжди можемо взяти ініціативу в свої руки і вивчити супротивника перш, ніж він завдасть удару.

Одне з основних джерел інформації доступних Honeynet - спілкування чорних капелюхів, наприклад на Internet Relay Chat (IRC) [23]. Чорні капелюхи вільно спілкуються в своєму середовищі, розповідаючи про свої мотиви, цілі і "подвиги". За допомогою Honeynet можна зафіксувати ці переговори буквально у вигляді дослівних діалогів, а також тримати відеозображення чорних капелюхів, що атакують нашу систему в режимі реального часу. Це дає нам уявлення, яким чином чорні капелюхи намічають жертву і здійснюють атаку.

У побудові Honeynet є два основних елементи: управління даними і їх накопичення(збір) [20]. Управління даними означає, що ми можете контролювати пересилку пакетів, тобто куди і які саме пакети прямують. Основне призначення полягає в тому, щоб бути упевненим, що при компрометації honeypot (у складі Honeynet) мережа не буде використана для здійснення атак на інші системи. Другий елемент - збір даних, що має на увазі фіксацію всіх дій чорного капелюха, починаючи від натиснень клавіш і закінчуючи пакетами, що пересилаються. Після накопичення ці дані використовуються для аналізу з метою з'ясування які засоби були використані чорними капелюхами.

Складність полягає в тому, щоб зловмисники не відмітили нічого підозрілого. Після компрометації системи чорні капелюхи зазвичай розраховують на нормальне з'єднання з Інтернет для завантаження інструментарію, встановлення IRC-з'єднань. Родзинка honeynet якраз і полягає в тому, щоб дати зловмиснику деяку свободу дій, щоб дозволити їй

використовувати скомпрометовану мережу для шкідливих дій по відношенню до інших систем, таких як атаки на відмову в обслуговуванні, сканування і використання експлоїтов [24].

Honeynet спроектована так, щоб контролювати всі вхідні і витікаючі з'єднання. Це зроблено шляхом розміщення перед Honeynet міжмережевого екрану (МЕ), через який проходить весь трафік. МЕ відстежує скільки з'єднань ініційовано з Honeynet в Інтернет. Після досягнення деякого порогового значення екран блокує подальші спроби. Це дає чорному капелюху певну свободу дій, в теж час, автоматично утримуючи ситуацію під контролем. Емпірично встановлено, що дозвіл 5-10 витікаючих з'єднань не порушить його щастя, оберігаючи в той же час інші системи від атак. Це захищає Honeynet від використання як плацдарм для сканування, дослідження або атак на більшість інших систем. Якщо дозволити собі, щоб хтось здійснював цілодобовий моніторинг Honeynet, то можна не обмежувати число витікаючих з'єднань. Додатково, між міжмережевим екраном і Honeynet розміщений маршрутизатор це зроблено із двох причин [25].

Перша - маршрутизатор приховує МЕ. Після компрометації honeypot зловмисник виявляє звичайний маршрутизатор між Honeynet і зовнішніми мережами. Це створює реалістичніше середовище і оберігає МЕ від виявлення. Друга причина полягає в тому, що маршрутизатор діє як другий ешелон управління доступом. Він може доповнювати МЕ, не дозволяючи використовувати скомпрометований honeypot для нападу на системи за межами Honeynet.

На рисунку 1.6 представлена схема мережі Honeynet, де можна бачити МЕ, що розділяє поле дій на три частини, а саме: Honeynet, Інтернет і адміністративна мережа. Всі вхідні і витікаючі пакети проходять крізь МЕ і маршрутизатор. МЕ - перший ешелон, що здійснює контроль з'єднань. Маршрутизатор використовується як додатковий засіб фільтрації. МЕ настроєний так, щоб дозволяти будь-які вхідні і витікаючі з'єднання. Проте, він обмежує будь-який honeypot, що входить до з складу Honeynet можливістю

відкривати не більше 5 з'єднань з Інтернет. Всі подальші спроби встановлення з'єднань блокуються.

Маршрутизатор діє як другий ешелон управління доступом. Перш за все це використовується, щоб захиститися від спуфінга і атак на базі протоколу ICMP [26]. Маршрутизатор випускає назовні лише пакети з IP-адресою джерела, що належить діапазону Honeynet.

У системі блокується витікаючий ICMP-трафік, це зроблено із-за обмежених здібностей зробити повний автоматизований аналіз ICMP.

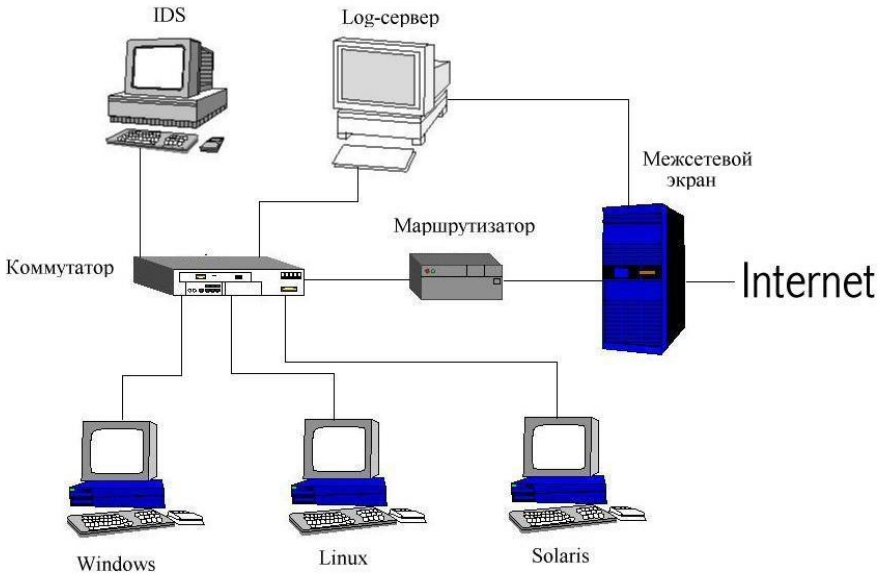


Рисунок 1.6 — Схема мережі Honeynet

Обмеження ICMP захищає проти SMURF-атаки, також можна побачити, що велику кількість пакетів ICMP маршрутизатор скидає [19, 57]:

router#show ip access-list 100

Extended IP access list 100

deny icmp any any (5446314 matches)

permit ip 172.16.1.0 0.0.0.255 any (66372 matches)

deny ip any any log (59990 matches)

Збір даних - фіксація дій усіх правопорушників. Особливість цього процесу в тому, щоб зафіксувати якомога більше даних, не даючи при цьому зловмиснику приводу запідозрити, що вона працює під контролем. Це

досягається внесенням як можна меншого числа модифікацій в системи, з яких складається Honeynet. Також, отримані дані не можуть зберігатися безпосередньо на honeypot, оскілики можуть бути виявлені чорним капелюхом, який відразу ж зрозуміє, що це – пастка.

МЕ являє собою перший рівень реєстрації будь-яких дій. Вище обговорювалось використання міжмережевого екрану для контролю за потоками даних, який можна використовувати і для реєстрації. МЕ записуватиме в журнали інформацію про всі вхідні і витікаючі з'єднання. Ця інформація носить критичний характер, оскільки будь-які з'єднання між Honeynet і Інтернет підозрілі. Як правило МЕ сконфігурований так, щоб він не тільки фіксував дані з'єднань, але і оповіщав нас про всі факти їх встановлення. Наприклад, якщо хтось спробує з'єднатися по протоколу telnet з будь-якою з систем в Honeynet, МЕ зафіксує ці дані і пошле нам відповідне попередження. Це надзвичайно корисно для відстежування спроби сканування. Ще одне його призначення - виявлення залишених чорних входів. Більшість експлоїтов залишають після себе можливість входу для запуску оболонки, при цьому використовується один з незайнятих (приватних) портів з верхньої частини діапазону. Наявність чорного входу виявляється по спробі встановити з'єднання з одним з таких портів [13]. МЕ також оповістить про спробу встановлення витікаючого з'єднання при цьому екран не тільки зафіксує дані з'єднання, але і пошле попередження. Воно може бути вислане як по електронній пошті, так і на пейджер.

Другим рівнем джерелом даних є система виявлення атак (IDS), що має два призначення. Перше і основне це фіксація всієї мережевої активності. IDS перехоплює і записує всі пакети, що проходять в мережевому кабелі. Якщо подивитися на рисунок 1.6 то можна побачити, що IDS підключений до комутатора, використовуваного всіма системами Honeynet. IDS підключений до порту, що знаходиться в режимі моніторингу, таким чином їй доступний для запису МТ. Ці записи в подальшому використовуватимуться для аналізу діяльності чорних капелюхів. Друга функція IDS - оповіщати нас про підозрілу активність. Більшість IDS мають власні бази сигнатур. Якщо перехоплений

пакет співпадає з сигнатурою, генерується попередження. Ця функція не така важлива для Honeynet, оскільки будь-яка активність в такій мережі підозріла за визначенням.

В даний час проект Honeynet використовує snort - систему виявлення атак (IDS), поширювану з відкритими початковими текстами [14, 19]. Snort настроєний на запис всього МТ в бінарному вигляді. Ці бінарні записи мають дуже велике значення, оскільки містять в собі всі вхідні і витікаючі пакети. Додатково проводиться фіксація в ASCII-вигляді даних сесій в окремі файли. Бінарні і ASCII файли записуються в каталоги з розбиттям по днях [20]. Для виконання цього завдання створений сценарій, який щодня запускається планувальником, перезапускаючий snort і що проводить розміщення реєстраційних файлів по каталогах. І, нарешті, всі попередження snort прямують на спеціальний реєстраційний сервер (syslog server). На цьому сервері отримані повідомлення поміщаються у файл /var/log/messages, який постійно сканується за допомогою Swatch [27]. Swatch аналізує вказаний файл в реальному часі і виконує два завдання. По-перше, він ідентифікує повідомлення від snort і направляє відповідне повідомлення системному адміністраторові. По-друге, він архівує всі повідомлення від snort в текстовому вигляді для подальшого аналізу.

Третій рівень - безпосередньо самі системи, що входять в Honeynet, які настроєні так, щоб всі системи вели ЖР не тільки локально, але і відправляти інформацію на спеціальний реєстраційний сервер. Для систем на базі UNIX і більшості мережевих пристроїв це досягається додаванням у файл конфігурації відповідного рядка. Для систем на базі Windows існують рішення третіх фірм, що дозволяють направляти інформацію на видалений syslog-сервер. Також, системні журнали можуть записуватися в загальний ресурс на syslog-сервері по протоколу NFS або SMB. Windows NT часто не володіє можливістю відправляти системну інформацію по протоколу syslog, але може робити це через NFS. Таким чином, критична системна інформація, така як активність процесів, з'єднання і спроби виконання експлоїтов безпечно зберігається у видаленій системі. Якщо правопорушник виявить це, він відключить syslog

(що є стандартним ходом для більшості з них). Фактично, з цієї миті ми втрачаємо це джерело даних, проте принаймні маємо інформацію звідки і яким чином він дістав доступ до системи.

Більш сучасні хакери можуть спробувати скомпрометувати syslog-сервер, щоб приховати сліди свого перебування. Даний сервер зазвичай захищеніша система, це означає, що для її злому чорні капелюхи повинні використовувати досконаліші методи, які можна буде зафіксувати. Якщо syslog-сервер буде скомпрометований, ми нічого не втратимо. Так, взломщик отримає контроль над системою і очистить реєстраційні журнали. Але не потрібно забувайти, що наш IDS продовжує пасивно накопичувати всі пакети, що проходять по мережі. Фактично, IDS є ще одним видаленим реєстраційним сервером, що фіксує інформацію (правда, від цього мало користі, якщо зловмисник скористається протоколом ssh).

Другий метод отримання інформації полягає в модифікації системи так, щоб вона фіксувала натиснення клавіш. Проект Honeynet в даний час готує декілька продуктів, що забезпечуватимуть дану функцію. Одним із них є модифікація версії оболонки протоколу bash, він використовуватиметься для заміни стандартного /bin/bash [19, 20]. Другий засіб - модифікована версія TTY Watcher, яка перенаправлятиме натиснення клавіш користувачем і знімки екранів на видалений сервер через нестандартне TCP-з'єднання [27, 57].

Honeynet як правило не вирішує повністю проблеми безпеки. Для цього потрібно використовувати методи, що зарекомендували себе, такі як строга аутентифікація, використання протоколів з криптографічним захистом, регулярний перегляд системних журналів і використання захищених рішень. Пріоритет слід віддати чіткій регламентації і опису процедур, це допоможе знизити ризик. Honeynet може допомогти лише коли перераховані вище заходи захисту вже прийняті, неухильно виконуються і супроводжуються.

1.3. Оцінка систем виявлення атак

На сьогоднішній день дуже інтенсивно розвиваються технології захисту корпоративних мереж, які включають в себе:

- МЕ (Firewall) – це програма або спеціалізована апаратна реалізація, що, ґрунтуючись на деяких правилах, дозволяє або забороняє передачу інформації, що проходить через неї, з метою обмеження деякої підмережі від зовнішнього доступу чи навпаки, для заборони виходу назовні. Міжмережеві екрани реалізують механізми контролю доступу із зовнішньої мережі до внутрішньої шляхом фільтрації всього вхідного і вихідного трафіку, пропускаючи тільки авторизовані дані. Всі міжмережеві екрани функціонують на основі інформації, яка отримується з різних рівнів еталонної моделі ISO/OSI [28], і чим вищий рівень OSI, на основі якого побудований МЕ, тим вищий рівень захисту, що ним забезпечується. Існують три основних типи міжмережевих екранів – пакетний фільтр (packet filtering), шлюз на сеансовому рівні (circuit-level gateway) і шлюз на прикладному рівні (application-level gateway). Існує дуже мало міжмережевих екранів, які можуть бути одночасно віднесені до одного з названих типів. Як правило, Firewall суміщає в собі функції двох або трьох типів.

Найбільш очевидний недолік МЕ – неможливість захисту від користувачів, які знають ідентифікатор і пароль для доступу в сегмент корпоративної мережі, який захищається. МЕ може обмежити доступ до ресурсів, але він не може заборонити авторизованому користувачу скопіювати цінну інформацію або змінити які-небудь параметри. А по статистиці не менше ніж 70% всіх загроз безпеці надходить зі сторони співробітників організації (див. рисунок 1.5).

- Віртуальна приватна мережа (VPN – Virtual Private Network). Технологія VPN призначена для побудови єдиного прозорого користувацького

середовища поверх будь-якої транспортної мережі [29]. Таке рішення дозволяє організувати: безпечний віддалений доступ персоналу до мережі підприємства чи організації з будь-якого робочого місця, підключеного до мережі Інтернет; достовірний підрахунок вживаних абонентом ресурсів в широкомовних транспортних мережах (наприклад, в мережах Ethernet); безпечну передачу конфіденційної інформації по мережі Інтернет без побудови додаткових фізичних каналів зв'язку. При побудові таких мереж можливо використовувати як комутовані канали зв'язку (dial-up), так і некомутовані (виділені лінії). При цьому для забезпечення конфіденційності інформації, що передається, не потребується організація додаткової виділеної лінії, а можливе використання вже існуючої, що значно знижує вартість побудови мереж VPN. Безпека інформації, що передається по мережі забезпечується шляхом шифрування з використанням будь-якого з наявних криптоалгоритмів.

- Сканер безпеки. Класичним сканером, який поставляється з усіма *nix подібними операційними системами є nmap [30]. Програма призначена для сканування мереж з будь-якою кількістю об'єктів, визначаючи стан об'єктів мережі, що сканується а також портів і відповідних служб. Для цього nmap використовує багато різних методів сканування таких як UDP, TCP connect(), TCP SYN (напіввідкрите), FTP proxy (прорив через ftp), Reverse-ident, ICMP (ping), FIN, ACK, Xmas tree, SYN і NULL сканування.
- Одним з найновіших сканерів безпеки є Nessus. Nessus являє собою безплатний сучасний сканер безпеки локальних і віддалених систем [31, 52]. Початок Nessus Project було покладено в 1998 році, перший реліз вийшов в квітні. На той період найпоширенішим сканером безпеки був SATAN. Задачею Nessus являється визначення запущених служб і вразливостей, включаючи найпопулярніші повідомленні про „дірки" wu-ftpd, наявність демонів DDOS, проблеми ipfw FreeBS і ін. Основний принцип полягає в тому, що вся інформації потребує перевірки, тобто інформація багерів основних служб не вважається основоположною.
- Систему виявлення вторгнень (IDS). Для запобігання комп'ютерним атакам, необхідно розробляти та налаштовувати системи

захисту інформації та системи виявлення атак (додаток А) [13]. Системи виявлення комп'ютерних атак – це один із найважливіших елементів систем інформаційної безпеки мереж. Враховуючи зростання в останні роки число проблем зв'язаних з комп'ютерною безпекою постійно зростає, як і пов'язаних з ними число хакерських атак (рис. 1.7) [12]. Системи виявлення вторгнень включають в себе: виявлення спроб несанкціонованого доступу та захист від атак типу „відмова в обслуговуванні" (DOS-атак).

Виявлення атак потребує виконання однієї із двох умов: розуміння очікуваної поведінки підконтрольного об'єкта системи або знання всіх можливих атак і їх модифікацій.

При створенні систем виявлення атак використовуються два основні підходи [6]:

– виявлення аномальної поведінки, використовуючи апарат математичної статистики, який досить добре себе зарекомендував. Даний підхід використовується , як правило, при виявленні DoS-атак, які використовують посилку великої кількості трафіку за короткий інтервал часу [25];

– виявлення зловживань, використовуючи сигнатури, що описують послідовність байт і дій, які характеризують несанкціоновану діяльність. Цей підхід знайомий по антивірусних системах, які побудовані саме за цим принципом.

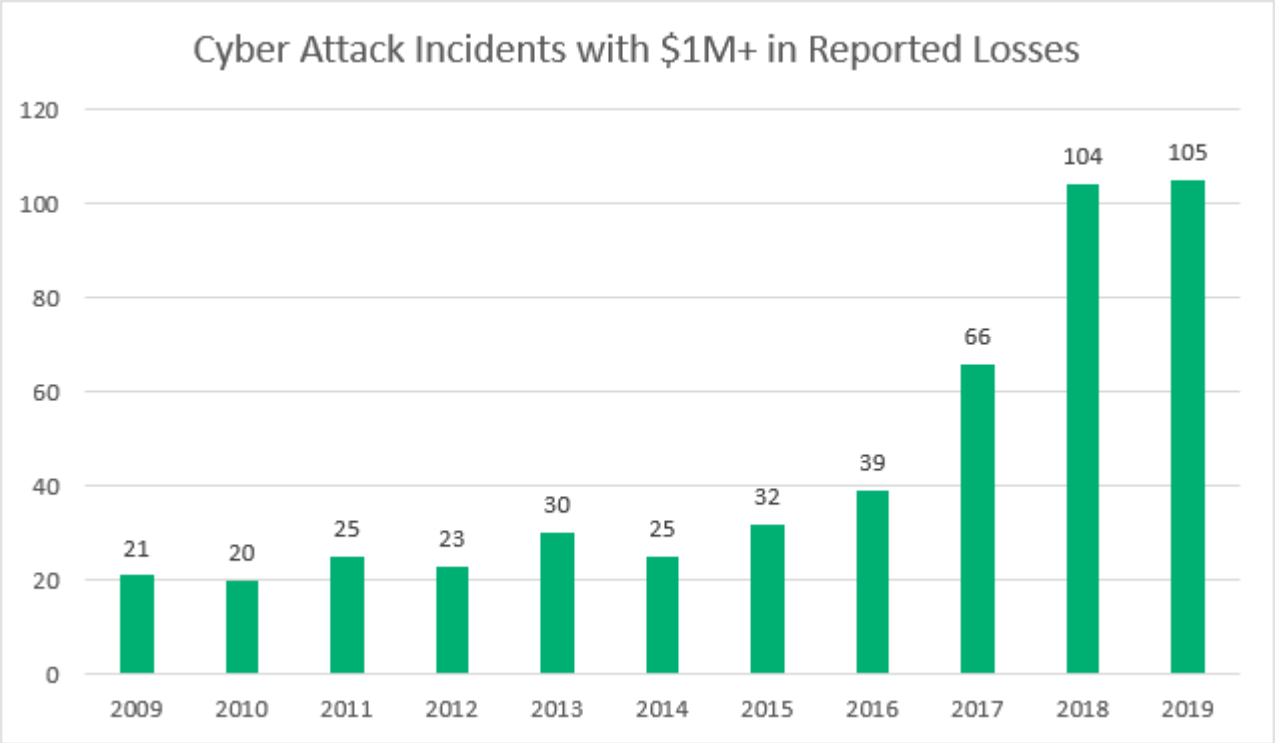


Рисунок 1.7 — Кількість атак в мережі Інтернет

1.4. Постановка задачі дослідження

Вирішенням задачі ідентифікації комп'ютерних атак у мережі на сьогодні займається чимало різних структур, організацій та просто адміністраторів комп'ютерних мереж. Всі вони мають свої напрацювання, нововведення, рекомендації але практично у всіх мало що зроблено для комплексного захисту так би мовити „на всі випадки". Практично всі відомі організації що займаються розробкою програмних продуктів надають гарантії знешкодження вірусу чи блокування атаки на протязі 24год, що в деяких мірах можуть стати смертельно небезпечними для функціонування цілої системи.

Тому оптимальним вирішенням поставленої задачі є використання сучасних систем-приманок [24]. Однією із організацій по створеню мережі приманки є відома наукова організація Honeynet Project, що займається дослідженнями в області систем безпеки і спеціалізується на вивченні тактики і мотивів ідентифікації атак, які використовуються зловмисниками [20]. До складу організації входять фахівці з питань безпеки з різних країн, які на

добровільній основі надають свої ресурси для розгортання і вивчення мереж-приманок, основне призначення яких стати об'єктом атаки хакерів.

Сучасні системи виявлення атак однією з яких є IDS основані на пошуку відповідних ознак (сигнатур) атак або на комбінації цих ознак (шаблонів). Якщо пошук таких сигнатур і шаблонів виконується на мереженому рівні, то IDS працює на мережквому рівні, а якщо пошук ведеться у системному журналі то говорять про системний рівень. Однією із перспективних технологій IDS являється нейромережева технологія виявлення атак на комп'ютерну мережу з використанням нейронних мереж. Тому б доцільно було поєднати ситему виявленя атак Snort і нейронну мережу (НМ) реалізовану в системі Matlab [32]. Система Snort, яка входить до складу honeynet веде список як своїх реєстраційних журналів так і журналів, які вона отримує від honeypot.

Як обгрунтовано вище, задачу ідентифікації комп'ютерної атаки доцільно вирішити виходячи із наступних міркувань:

– вдосконалити СМ виявлення атаки із додаванням до структури Honeynet засоби НМ. Це дозволить ідентифікувати ще невідомі атаки як вторгнення в систему;

– для попередньої обробки даних та їх представлення у НМ застосувати алгоритм шифрування MD5 [33, 42];

– забезпечити можливість виявлення атаки на web-сервер. Що дозволить знешкоджувати найпоширеніші атаки, такі як cgi-атаки;

– розробити стійкий алгоритм для ідентифікації зловмисника, та надати йому риси переносимості у разі зміни апаратного забезпечення;

– забезпечити можливість функціонування даних із реєстраційного журналу до НМ, що унеможливить зникнення їх у разі виявлення порушником.

Висновки до першого розділу

В даному розділі було охарактеризовано і представлено аналіз захищеності ІС. Проведено аналіз існуючих засобів, методів виявленя та здійснення атак на комп’ютерні системи та мережі, наведено їх переваги та недоліки. Представлено структуру мережі-приманки honeynet, зроблено її порівняння із системою honeypot. Також охарактеризовано рішення, які були прийняті для вирішення проблеми ідентифікації атак. На основі опрацьованого матеріалу було поставлено задачу по виявленю зловмисників у комп’ютерній мережі.

2. РОЗРОБКА МЕТОДУ ІДЕНТИФІКАЦІЇ АТАК

2.1. Теоретичне обґрунтування методу виявлення атак

Мета виявлення вторгнень на перший погляд дуже проста: виявити проникнення в ІС. Проте це вельми складне завдання. Насправді, системи виявлення вторгнень ніяких вторгнень взагалі не виявляють - вони тільки виявляють ознаки вторгнень під час таких атак. Якщо ніяких проявів немає або якщо інформація є, але не вселяє довіри система не в змозі виявити вторгнення [2].

Системи виявлення атак призначені для виявлення і протидії мережевим атакам зловмисників. Вони є спеціалізованим програмно-апаратним забезпеченням з типовою архітектурою, що включає наступні компоненти (рис. 2.1)[14, 40]:

- модулі-датчики для збору необхідної інформації про МТ в ІС;
- модуль виявлення атак, що виконує обробку даних, зібраних датчиками, з метою виявлення інформаційних атак;
- модуль реагування на виявлені атаки;
- модуль зберігання конфігураційної інформації, а також інформації про виявлені атаки. Таким модулем, як правило, виступає стандартна СУБД, наприклад MS SQL Server;
- модуль управління компонентами системи виявлення атак.

Для точного виявлення вторгнень необхідні надійні і вичерпні дані про те, що відбувається в системі, яка захищається. Взлом системи можливий як із сторини комп’ютера, що знаходиться в локальній мережі так і через глобальну мережу Інтернет. Проте сучасні атаки (DDOS-атаки - distributed denial-of-service) для здійснення взлому системи можуть використовувати і проміжні комп’ютери, які прийнято називати зомбі (рис. 2.2).

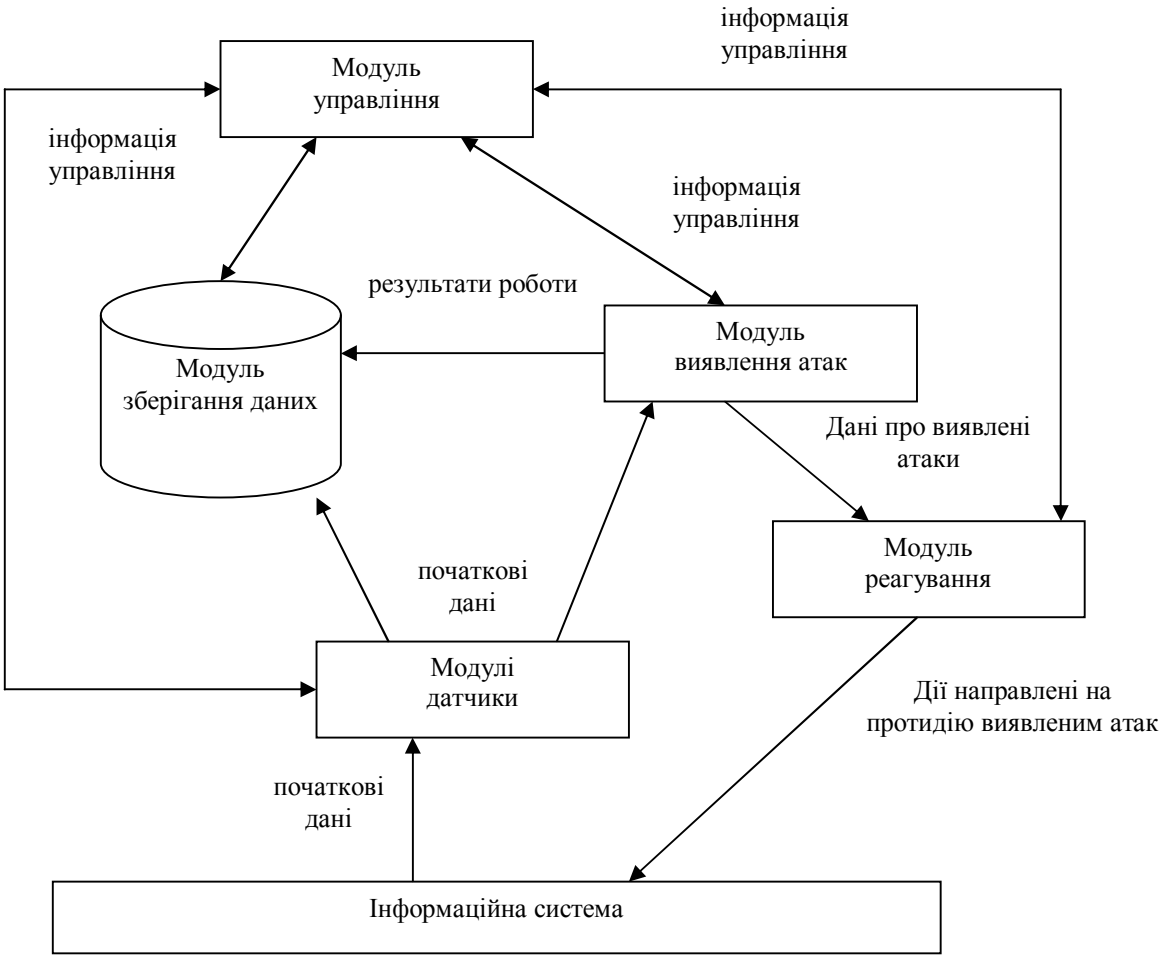


Рисунок 2.1 — Типова архітектура виявлення атак

Такі системи у мережі Інтернет є незахищені або мало захищені. Зловмисник взломавши їх, бере під свій контроль і при цьому інсталює відповідне програмне забезпечення на кожному з них [6, 50]. Такі компютери після того стають підвладні йому.

Виходячи із відомих методів виявлення атак розглянутих у попередньому розділі, найкращим методом для вирішення задачі ідентифікації атак є застосування СМ на базі нейронних мереж [34, 35]. Вони описують кожну атаку у вигляді спеціальної моделі або сигнатури. Як сигнатура атаки можуть виступати: рядок символів, семантичний вираз на спеціальній мові, формальна математична модель. Алгоритм роботи СМ полягає в пошуку сигнатури атак в початкових даних, зібраних мережевими і хостовими

датчиками системи. У разі виявлення шуканої сигнатури, система фіксує факт інформаційної атаки, яка відповідає знайденій сигнатурі.

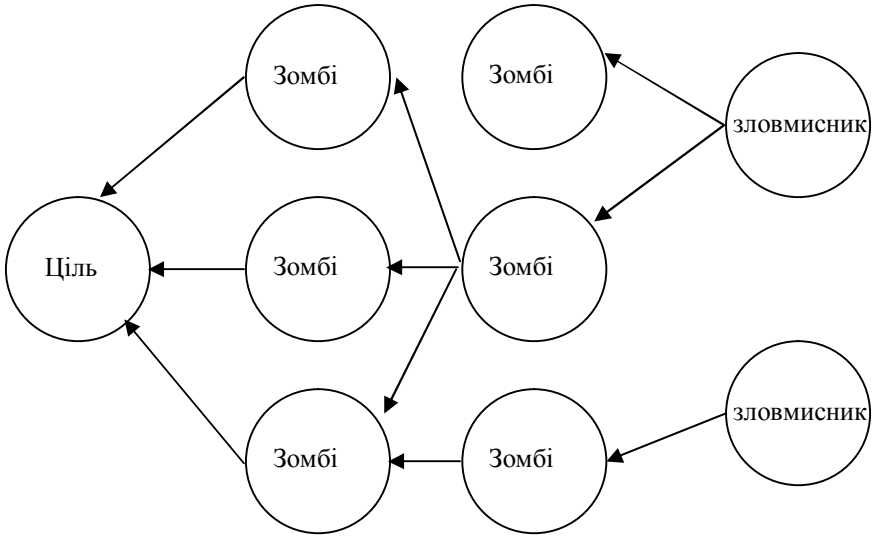


Рисунок 2.2 — Здійснення DDOS-атаки

Задача ідентифікації атак на комп'ютерну систему буде розглядатись як задача класифікації, реалізація якої представлена з допомогою однорівневого або багаторівневого персептрону (БП) [35]. Нехай М – множина вхідних даних системи виявлення атак, а А і В її підмножини. До А віднесемо дані, які вважаються атакою, а до В – дані, які є безпечними. Тоді справедлива рівність [4]:

$$A \subset M, B \subset M, A \cap B \neq 0, A \cup B = M\ . \qquad (2.1)$$

Як відомо – БП є ідеальним класифікатором. Він представляє собою структуру, яка складається із персептронних нейронів. Передавальна функція у такій структурі, як правило є порогова.

Серед великої кількості архітектур оптимальною для сторення мережі виявилась архітектура БП. Оскільки БП може класифікувати сигнатуру записану у лог-файл як підозрілу або непідозрілу. Вибір імено такої архітектури НМ обумовлений наступними міркуваннями. Рекурсивні та само-організовані мережі не підходять, так як вони навчаються задачі класифікації

образів виходячи із своїх внутрішніх цілей (наприклад по принципу знаходження мінімальної відстані). Використання однорівневого персептрону неприйнятно оскільки недоказана лінійна розподіленість образів [36]. Архітектура БП зображена на рис. 2.3, де у кожному нейроні використовується сигмоїдальна функція активації

$$F(n) = (1 - \exp(-n)) - 1. \qquad (2.2)$$

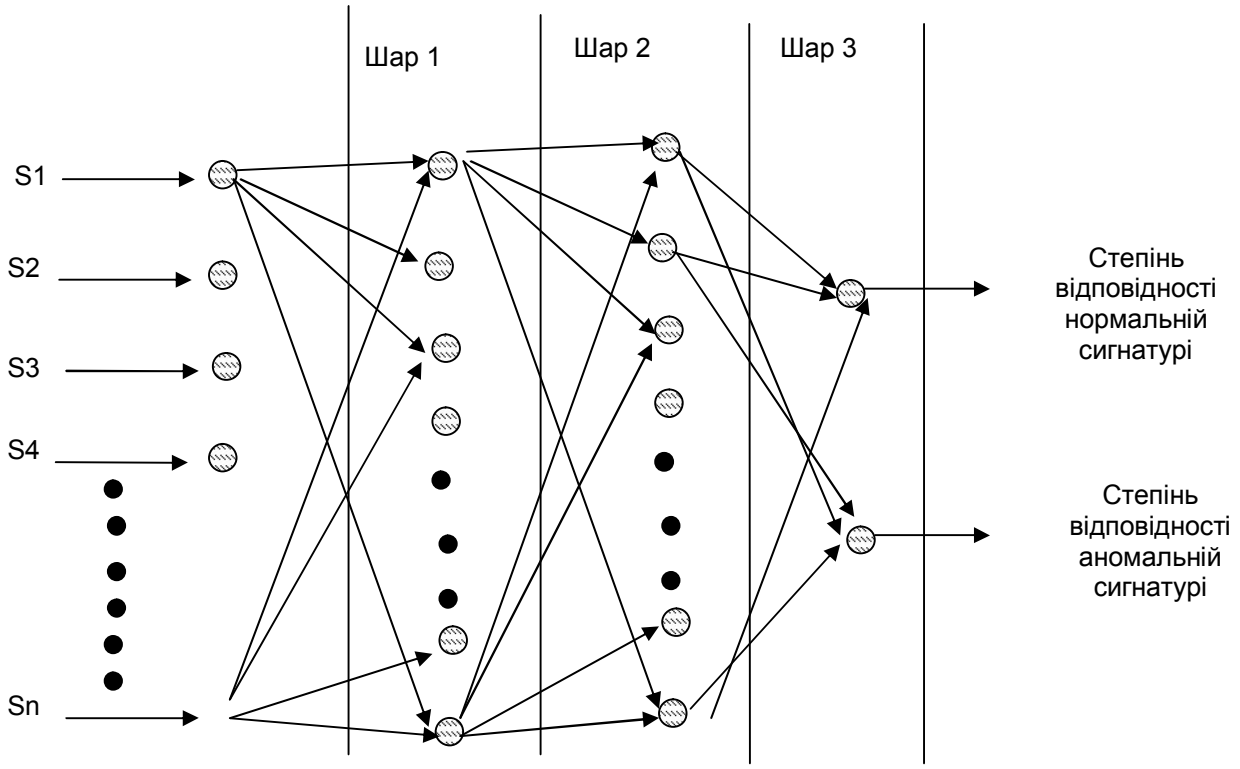


Рисунок 2.3 — Архітектура БП

Для реалізації алгоритму спочатку потрібно математично описати роботу НМ, яка полягає у наступному [37]. Нехай j- типовий елемент вихідного шару, а елемент і - це елемент шару, який передує вихідному. Активність елементу вихідного шару визначається двокроковою процедурою. Спочатку обчислюється сумарний зважений вхід.

$$X_j = S_i (Y_i * W_{ij}), \qquad (2.3)$$

де Yi - рівень активності і-го елементу в попередньому шарі і Wij - вага зв'язку між і-м і j-м елементами.

Далі елемент обчислює активність Yj за допомогою деякої функції від сумарного зваженого входу. Зазвичай застосовується сигма-функція.

$$Y_j = \frac{1}{(1+e^{(-X_j)})}. \tag{2.4}$$

Після того, як активності всіх вихідних елементів визначені, мережа обчислює помилку, яка визначається виразом.

$$E = \frac{1}{2} * S(Y_j - D_j)^2 \tag{2.5}$$

де *Yj* - рівень активності j-го елементу у верхньому шарі, а *Dj* -бажаний вихід j-го елементу.

Щоб навчити НМ рішенню якої-небудь задачі, потрібно підправляти ваги кожного елементу так, щоб зменшувалася помилка розбіжності між дійсним і бажаним виходом. Іншими словами, вона повинна обчислювати, як змінюється середньоквадратична помилка при невеликому збільшенні або зменшенні кожної ваги. Найчастіше для цього обчислення застосовується алгоритм зворотнього розповсюдження помилки (ВРА). Цей метод був розроблений як загальний метод Відроу-Хоффа для багатошарових мереж з нелінійними диференціальними функціями активації.

Тому для задачі класифікацій доцільно використати ВРА [39], який складається з чотирьох наступних кроків:

1) Визначають наскільки швидко міняється помилка при зміні вихідного елементу. Ця похідна середньоквадратичної помилки є різницею між дійсною і очікуваною активністю.

$$EA_j = \frac{dE}{dY_j} = Y_j - D_j . \quad (2.6)$$

2) Обчислюють наскільки швидко змінюється помилка у міру зміни сумарного входу, що отримується вихідним елементом. Ця величина є результат кроку 1, помножений на швидкість зміни вихідного елементу із зміною його сумарного входу.

$$EI_j = \frac{dE}{dX_j} = \frac{dE}{dY_j} * \frac{dY_j}{dX_j} = EI_j Y_j (1 - Y_j) . \quad (2.7)$$

3) Обчислюється, як швидко змінюється помилка у міру зміни ваги на вхідному зв'язку вихідного елементу. Ця величина є результатом кроку 2, помноженого на рівень активності елементу, з якого виходить зв'язок.

$$EW_{ij} = \frac{dE}{dW_{ij}} = \frac{E}{dX_j} * \frac{dX_j}{dX_{ij}} = EI_j Y_i . \quad (2.8)$$

4) Обчислюється наскільки швидко змінюється помилка із зміною активності елементу з попереднього шару. Цей ключовий крок дозволяє застосовувати ВРА до багатошарових мереж. Якщо активність елементу з попереднього шару змінюється, це впливає на активності всіх вихідних елементів, з якими він пов'язаний. Тому, щоб підрахувати сумарну дію на помилку, складаємо всі ці дії на вихідні елементи. Цей результат кроку 2, помноженого на вагу зв'язку до відповідного вихідного елементу.

$$EA_i = \frac{DE}{dY_{ij}} = S(\frac{dE}{dX_j} * \frac{dX_j}{dY_{ij}}) = S(EI_j W_{ij}) . \quad (2.9)$$

Є різна кількість модифікацій ВРА, з яких виділяють наступні п’ять груп [38, 40]:

- Алгоритмічні – алгоритм Quickprop, метод вторинного порядку, RPROP;
- Еврістичні – еврістичний метод ініціалізації вагів, оцінка оптимальної скорості навчання;
- Регуліровки - падіння вагів;
- Модифікації мережі – нестандартні нейрони та функції активації, рекурентні вузли та мережі прямого розповсюдження;
- Мережі з динамічною структурою – каскадна кореляція.

Для того, щоб НМ змогла розпізнати сигнатуру предсталену у вигляді стрічки доцільно застосувати алгоритм MD5. Він полягає у наступному - після деякої первинної обробки, обробляється вхідний текст 512-бітовими блоками, розбитими на 16 32-бітових підблоки. Виходом алгоритму є набір з чотирьох 32-бітових блоків, які об'єднуються в єдине 128-бітове хэш-значення.

Повідомлення, котре подається на вхід доповнюється так, щоб його довжина була на 64 бита коротшою за число кратному 512. Цим доповненням є 1, за якою аж до кінця повідомлення слідує стільки нулів скільки потрібно. Потім, до результату додається 64-бітове представлення довжини повідомлення. Ці дві дії служать для того, щоб довжина повідомлення була кратна 512 бітам (що потрібно для частини алгоритму, який залишився), щоб гарантувати, що різні повідомлення не виглядатимуть однакові після доповнення. Спочатку проходить ініціалізація чотироьх змінних, які копіюються в інші змінні: A в a, B в b, C в c і D в d, вони називаються змінними зчеплення.

Головний цикл складається з чотирьох етапів, що зображено на рисунку 2.4. На кожному етапі 16 разів використовуються різні операції [40]. Кожна операція є нелінійною функцією над трьома змінними із a, b, c і d. Цикл роботи однієї операції зображений на рисунку 2.5. Потім вона додає цей результат до четвертої змінної, підблоку тексту і константи. Далі результат циклічно зміщується вправо на змінне число бітів і додає результат до однієї з перемінних a, b, c і d.

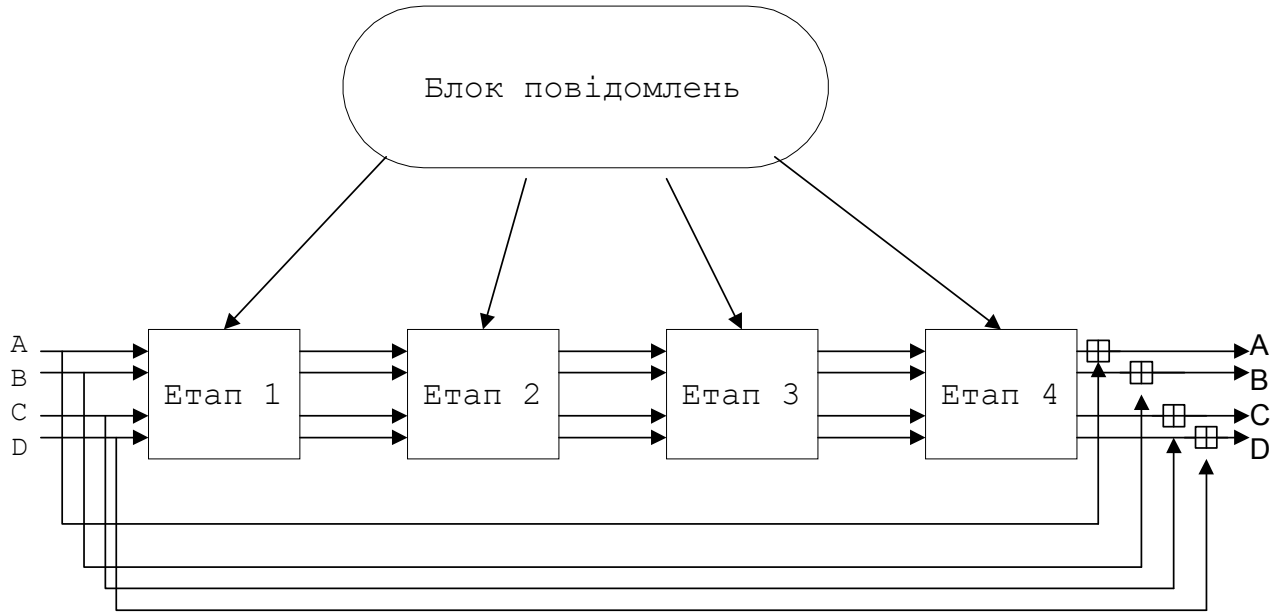


Рисунок 2.4 — Цикл роботи MD5

У кінці результат замінює одну із змінних a, b, c і d. Існують чотири нелінійні функції, використовувані по одній в кожній операції.

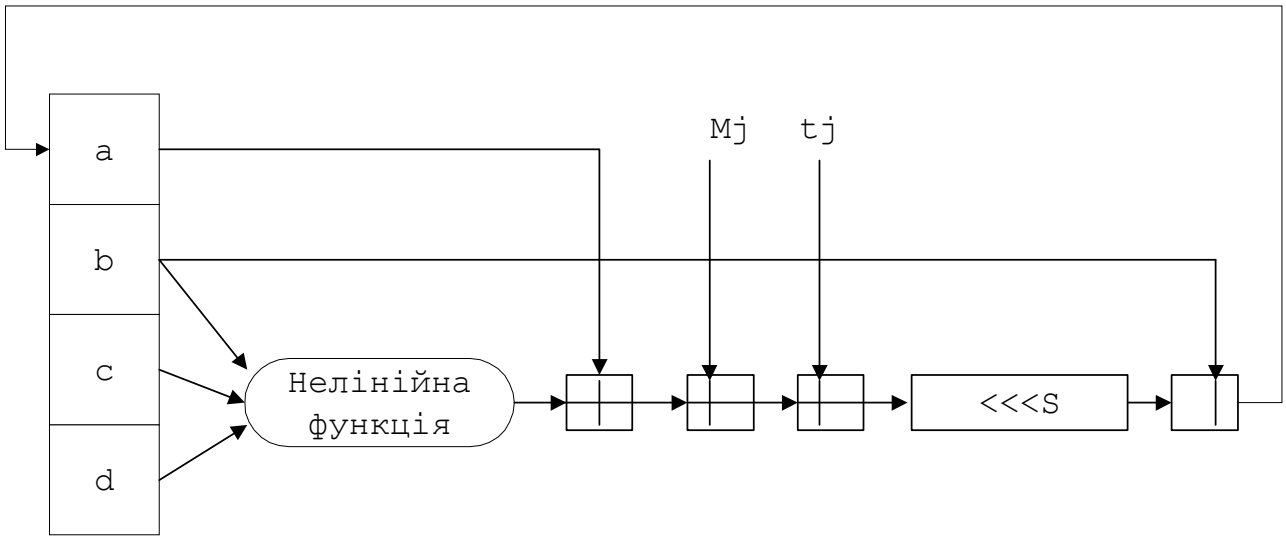


Рисунок 2.5 — Цикл виконання одної операції

Для кожного етапу використовується інша функція. Ці функції спроектовані так, якщо відповідні біти X, Y і Z незалежні і незміщені, то кожен біт результату також буде незалежним і незміщеним. Функція F - це побітова умова: якщо X то Y, інакше Z. Функція H - побітова операція парності.

Якщо Mj позначає j-ий підблок повідомлення (від 0 до 15), а <<<s позначає циклічне зрушення вліво на s бітів, то використовуються наступні чотири операції [33, 41]:

FF(a,b,c,d,Mj,s,ti) означає a = b + ((a + F(b,c,d)+ Mj + ti) <<<s)

GG(a,b,c,d,Mj,s,ti) означає a = b + ((a + G(b,c,d)+ Mj + ti) <<<s)

HH(a,b,c,d,Mj,s,ti) означає a = b + ((a + H(b,c,d)+ Mj + ti) <<<s)

II(a,b,c,d,Mj,s,ti) означає a = b + ((a + I(b,c,d)+ Mj + ti) <<<s)

Слід розрізняти – ще 64 дії, які реалізовані у програмному коді та представленні в додатку Б. Причому константи ti вибираються таким чином, що на i-му етапі ti є цілою частиною:

$$2^{32} * abs(\sin(i)), \qquad (2.10)$$

де i вимірюється в радіанах.

Після цього a, b, c і d додаються до A, B, C і D. Відповідно алгоритм продовжується для наступного блоку даних. Кінцевим результатом служить об'єднання A, B, C і D.

Алгоритм MD5 має наступні переваги [42, 35]:

1) У порівнянні з іншими тут додається четвертий етап.

2) У кожній дії використовується унікальна константа, що додається.

3) Функція G на етапі 2 з $((X \cap Y) \cup (X \cap Z) \cup (Y \cap Z))$ була змінена на $(X \cap Z) \cup (Y \cap (\neg Z))$, щоб зробити G менш симетричною.

4) Кожна дія додається до результату попереднього етапу. що забезпечує більш скоріший лавинний ефект.

5) Змінився порядок, в якому використовувалися підблоки повідомлення на 2 і 3 етапах, щоб зробити шаблони менш схожими.

2.2. Розроблення алгоритму ідентифікації атаки

Система honeynet є недосконалою для ідентифікації атак. Оскільки система виявлення атак Snort може знаходити тільки відомі на сьогоднішній час атаки здійснювані хакерами. Тому було запропоновано алгоритм класифікації атак з допомогою НМ. Для цього застосовується сигнатурний

метод виявлення хакерських атак описаний у попередньому розділі. Він дозволяє виявляти невідомі атаки з певною імовірністю. Розроблений алгоритм ідентифікації комп'ютерних атак поєднує у собі алгоритм навчання НМ та алгоритм класифікації НМ [35].

Алгоритм навчання НМ зображений на рисунку 2.6. Робота алгоритму починається із збору даних і віддаленого їх зберігання у системі honeynet, при цьому використовуэться віддаленоий сервер syslog у внутрішній мережі honeynet [21,47]. Завдання syslog полягає в зборі всіх системних журналів honeynet. Системні журнали - це відмінне джерело інформації, оскільки вони зазвичай реєструють те, як хакер зламав систему і дістав до неї доступ. Проте після атаки взломщики часто змінюють або стирають саме системні журнали, які знаходяться на web-серверах. З цієї причини потрібно зберігати інформацію віддалено на захищеному сервері. Сервер syslog має ще одне завданя, він є складною системою honeypot, а тому має найбільш захищену систему в honeynet.

На прикладі цієї honeypot можна вивчити найбільш витончені інструменти і тактику співдружності blackhat [25,48]. Коли вони зламують одну з менш захищених систем honeynet, то можуть відзначити, що system logs переправляються на видалений сервер. Багато з тих, які атакують спробують взломати віддалений сервер, щоб приховати свої сліди і знищити записи.

Проте віддалений реєстраційний сервер - набагато захищеніша система, для злому якої потрібні інструменти і складна тактика. Таким чином, можна дізнатися набагато більше, якщо взломщик націлюватиметься на регістраційний сервер.

Недоліком використання сервера syslog є те, що він може бути взломаний і всі записи будуть стерті, проте нічого не загубиться. Оскільки сервер IDS котрий записує всі пакети, також фіксує всі реєстраційні файли, які посилаються на віддалений сервер syslog, оскільки ця інформація пересилається в межах мережі. IDS виступає як вторинний, але пасивний сервер syslog.

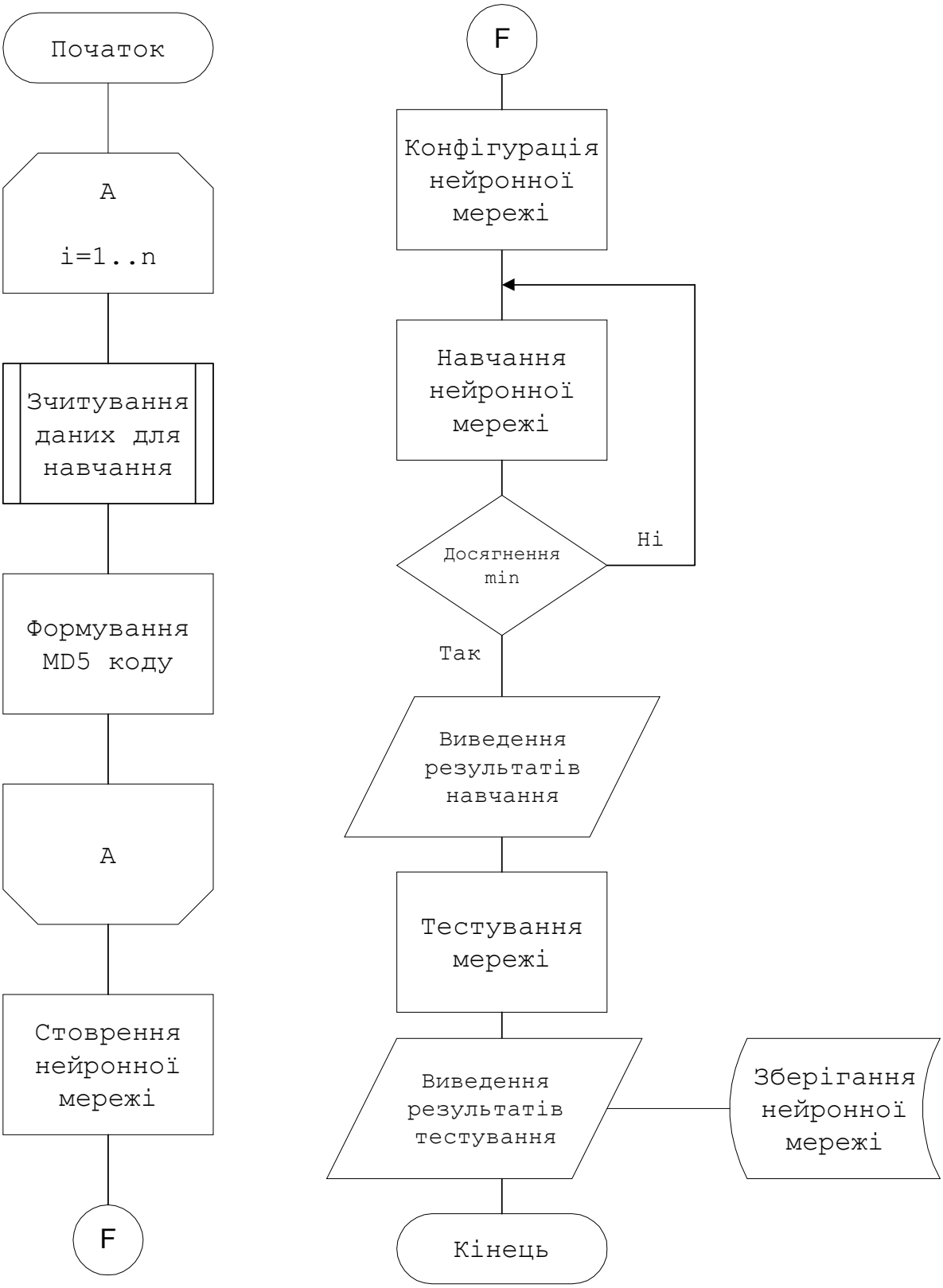


Рисунок 2.6 — Алгоритм навчання НМ

Таким чином, не тільки реєстраційні файли віддалено реєструються на сервері syslog, але і всі system logs пасивно записуються в IDS. Дуже важливо пам'ятати, що багаторівневий запис даних має величезне значення.

Наступним кроком після збору даних є їх представлення у шістнадцятковому форматі.

Для цього використовується модифікований алгоритм MD5 (рис. 2.7).

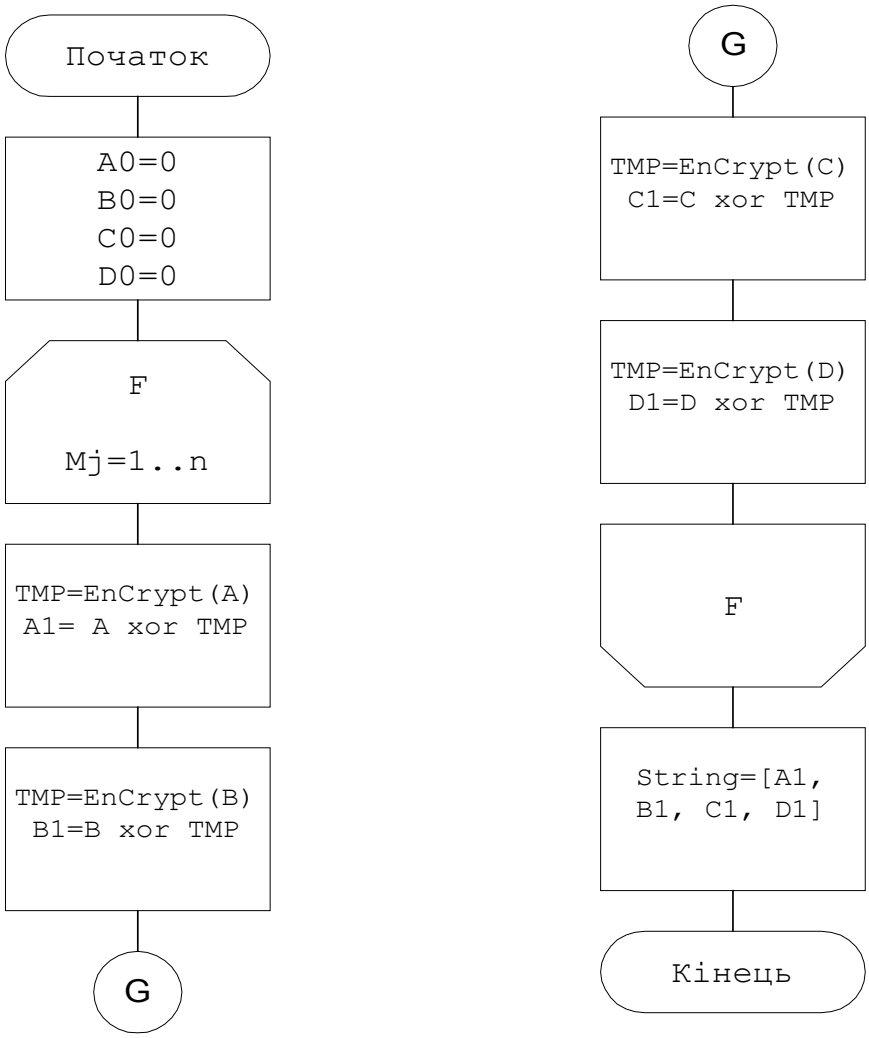


Рисунок 2.7 — Модифікований алгоритм MD5

Він представляє стрічку як 32 байтове число. Після його запуску із командної стрічки, він автоматично завантажує дані із реєстраційного журналу, перекодовує їх та зберігає у txt форматі.

Після того як дані перетворені у цифровий формат, наступним кроком алгоритму є їх завантаження у НМ, а у кінцевому етапі їх зберігання. Коли НМ уже навчена і ми отримали дані, логічним етапом є її перевірка на те як вона навчена. Для цього застосовується алгоритм класифікації (рис. 2.8).

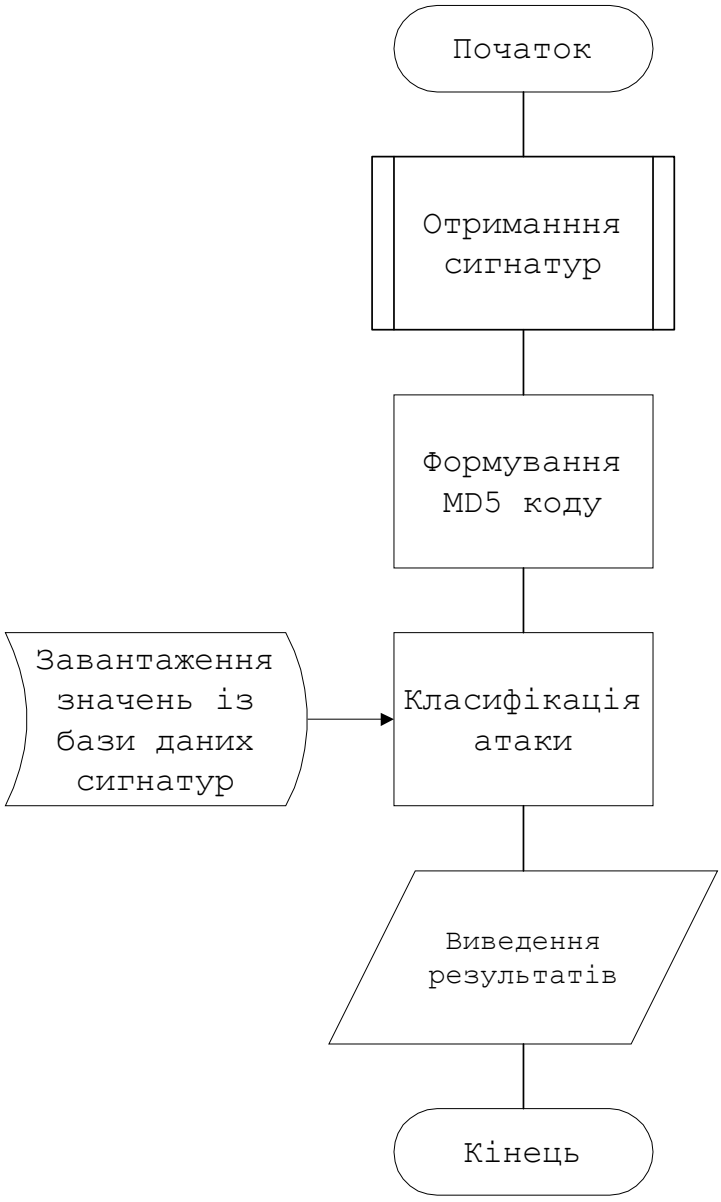


Рисунок 2.8 — Алгоритм класифікції НМ

Алгоритм класифікації є схожим до алгоритму навчання, проте він має свої відмінності. Першим етапом є створення вибірки даних із регістраційного журналу. Після цього дані поступають в алгоритм MD5, де відбувається їх коректне представлення для НМ. Даний алгоритм проводить навчання мережі беручи значення, які мають в собі стрічку проведеної атаки так і не атаки.

В алгоритмі використовуються значення із бази даних сигнатур на основі, яких училась НМ і нові дані раніше не знанні для неї. Навчання НМ доцільно проводити із застосуванням ВРА.

При цьому функція навчання представлена собою, як функція trainbr, яка реалізована по методу Байсена [39-40, 45].

2.3. Структура розробленої системи

На основі поставленої задачі в підрозділі 1.3 і розроблених та описаних в підрозділах 2.1, 2.2 методу і алгоритму виявлення атак, можна синтезувати узагальнену структуру системи ідентифікації комп'ютерних атаки (рис. 2.9).

Виходячи з основних функцій, в якості структури доцільно обрати багаторівневу систему, яка складається з певної впорядкованої сукупності програмних та апаратних блоків, які називаються рівнями.

При цьому виконуються такі вимоги:

- на кожному рівні нічого не відомо про властивості (і навіть існування) наступних (більш високих) рівнів;
- кожний рівень може взаємодіяти по управлінню (звертатися до компонентів) з попереднім (більш низьким рівнем) через наперед визначений інтерфейс, при цьому не знаючи нічого про внутрішню будову всіх попередніх рівнів;
- кожний рівень володіє певними ресурсами, які він або приховує від інших рівнів, або надає їх безпосередньо наступному рівню (через вказаний інтерфейс).

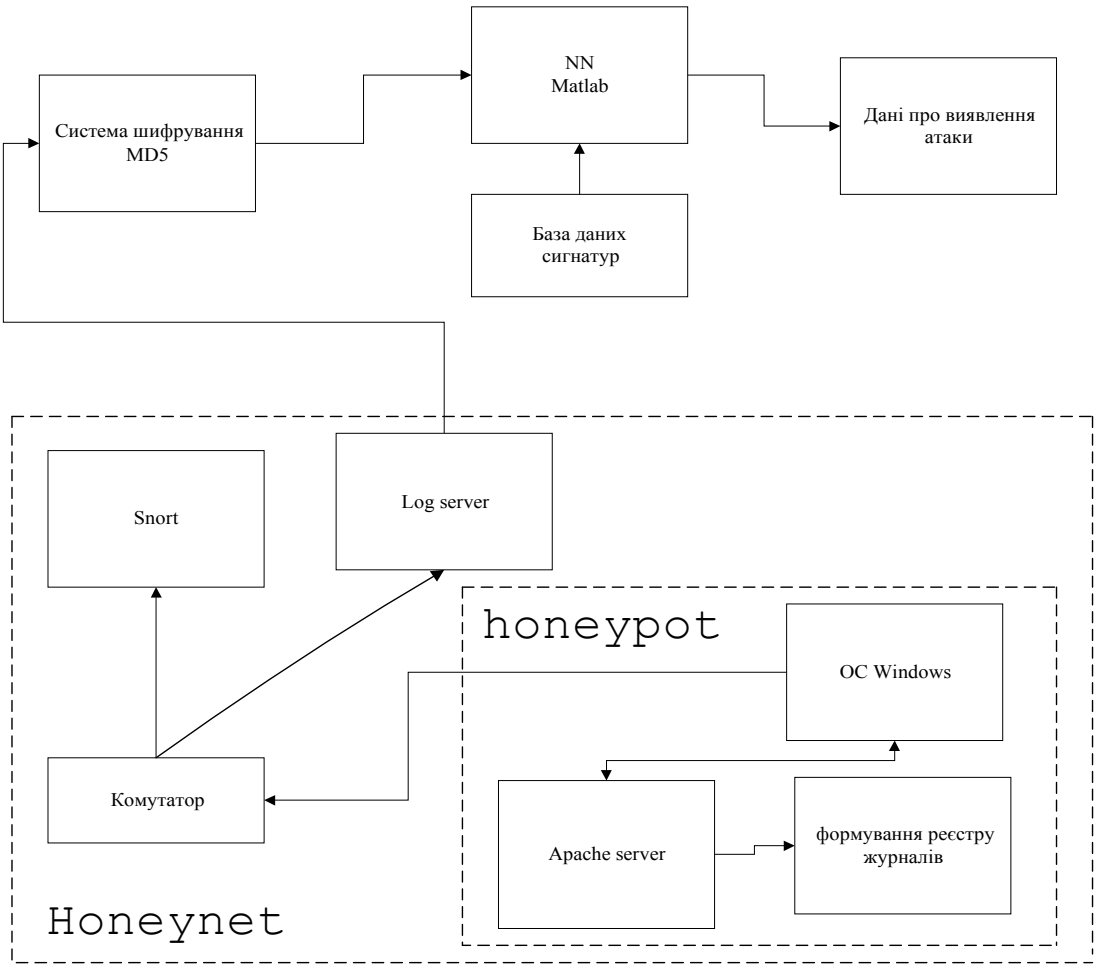


Рисунок 2.9 — Узагальнена структура системи ідентифікації атаки

Виходячи з представленої структури (див. рис. 2.9), доцільно виділити кілька блоків (рис. 2.10), які виконуватимуть лише певні функції, передаючи значення між собою у текстовому форматі:

- Блок формування даних (система honeynet);
- Блок шифрування даних (система шифрування MD5);
- Блок виявлення атаки (нейронна мережа).

З врахуванням запропонованої структури (див. рис. 2.10) інформація про виявлення атаки буде формуватись на основі послідовності наступних процедур [19]:

1) Дані про дії правопорушника, який взломав систему зберігатимуться у реєстраційних журналах. Система приманки з цими даними представляє собою web-server, котрий встановлений під операційну систему Windows;

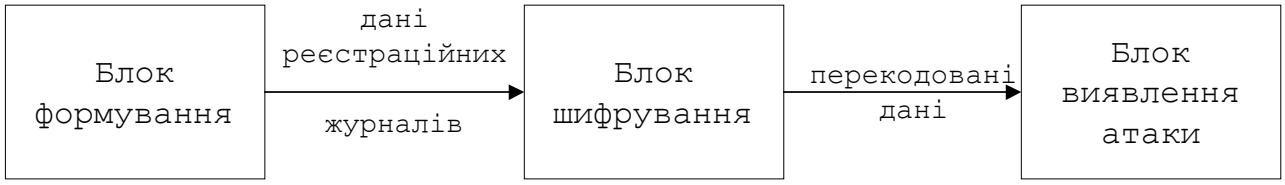


Рисунок 2.10 — Схема передачі інформації

2) Збережена на web-server інформація направляється на систему Snort та syslog сервер через комутатор, з метою недопущення того, що дана інформація щезає або стирається хакером з web-server;

3) Коли Snort просканував дані та сформував свій результат, дані із syslog серверу подаються на модифіковану систему шифрування MD5;

4) Система MD5 представляє ці дані у форматі, прийнятному для НМ;

На останьому етапі дані завантажуться у НМ для навчання, а після цього проводиться тестування уже на іншій відібраній парі даних.

Висновки до друго розділу

Таким чином, у другому розділі розроблено метод виявлення атак на основі якого побудовано алгоритм ідентифікації комп’ютерних атак. Розроблено алгоритм функціонування системи модуля виявлення атаки, з використанням нейронної мережі. Як наслідок створено підґрунтя для розробки набору програмних модулів з надійним виявленням комп’ютерних атак, що описаний у наступному розділі.

3. РОЗРОБЛЕННЯ ПРОГРАМНОГО МОДУЛЯ ДЛЯ ВИЯВЛЕННЯ АТАК

3.1. Структура системи програмних модулів і програмна реалізація модуля виявлення атак

Запропонований метод виявлення атак повинен бути досить швидким і в водночас простим, оскільки опрацьовується та аналізується значний об’єм інформації, який надходитиме із комп’ютера, підключеного до Інтернет. Враховуючи, що швидкість програмного модуля залежить від мови програмування, для реалізації НМ було обрано пакет програмних засобів Matlab [32, 47]. Matlab представляє собою, так би мовити надбудову над мовою програмування С з оптимізацією під математичні потреби. При цьому програмування в даному пакеті дуже схожі з програмуванням на звичайній мові С, з тією лише різницею, що завдяки широкому виборі математичних функцій та спрощенням синтаксису значно скорочується час отримання потрібного результату. Але для роботи нейронної мережі використовувались стандартні бібліотеки Matlab C Match [46, 49].

Для виявлення атаки із використанням нейронної мережі спочатку потрібно реалізувати атаку на сервер. У системі honeynet, як систему-приманку(honeypot) обрано web-сервер на якому налаштовано apache сервер версії 1.3.23. Атака на web-сервер формувалась використанням найпоширеніших на сьогоднішній час cgi-атак (рис. 3.1) [26], які найбільш часто зустрічаються у web-додатках, так і на web-серверах. CGI-атаки зв'язані з генерацією нестандартних рядків URL, деякі з них працюють з розрахунком на переповнювання параметрів буфера. Вони використовуються хакерами або черв'яками для зміни директорій на сервері, щоб дістати доступ до потрібних додатків.

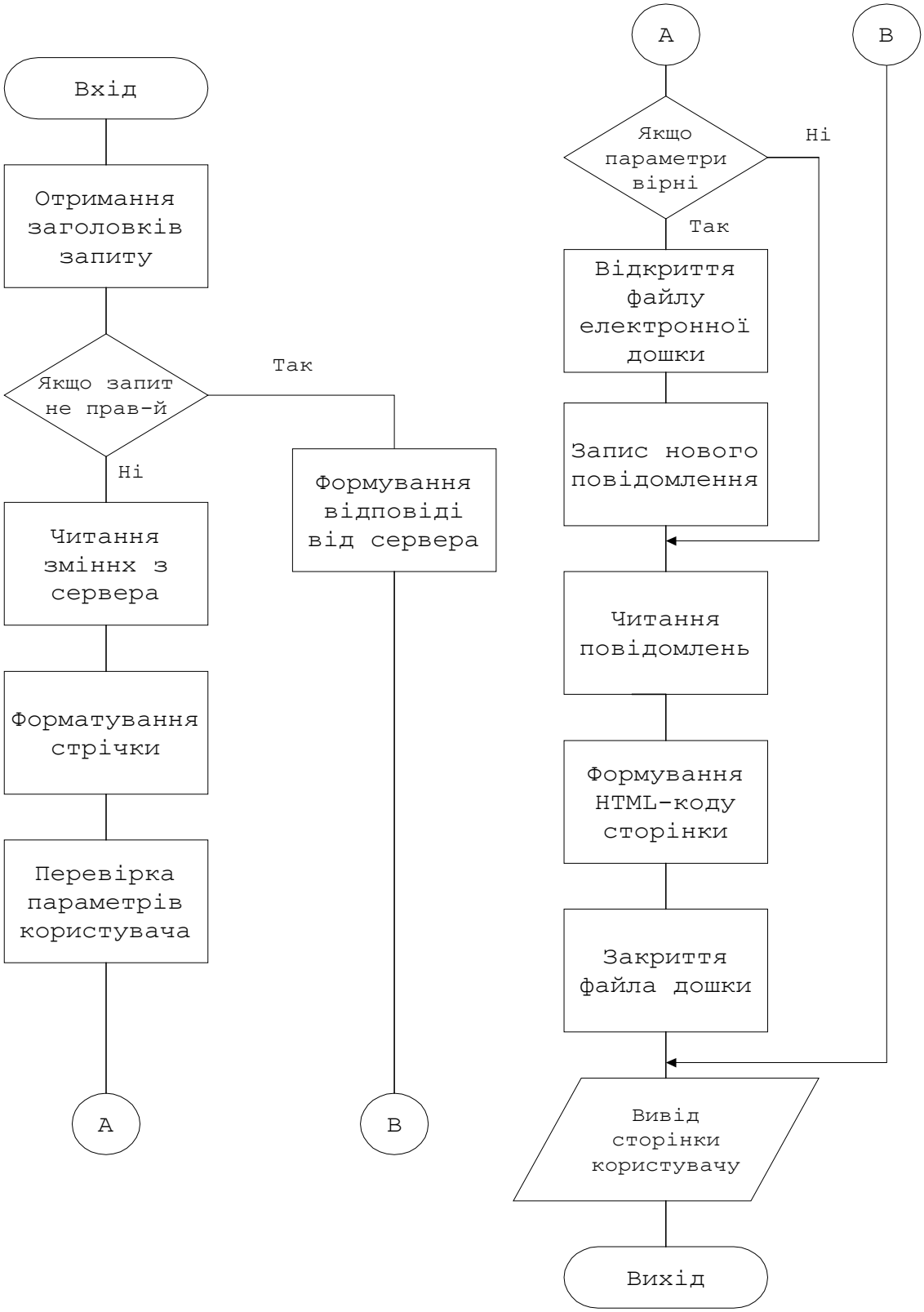


Рисунок 3.1 — Схема здійснення CGI-атаки

Програмний код здійснення cgi-атаки з використанням мови програмування Perl наведено у додатку В.

Perl – це є скриптова мова програмування в якій програмний код виконується не самою ОС, а передається на інтерпретатор. Після цього програма починає виконуватись забезпечуючи можливість перенесення програмного коду на різні платформи. Perl надає широкий спектр можливостей для створення ефективних програм. Нижче наведені деякі найбільш корисні переваги мови Perl [51]:

1) Асоціативні масиви ,які індексуються програмами з використанням нецілих ключів;

2) Функції вводу/виводу файлів;

3) Функції форматованого виводу (генерації звітів на основі шаблонів template [40]);

4) Повний набір операторів.

Для атаки на web - сервер використовуються запити для виконання програмного коду. Наприклад, на рисунку 3.2 зображено запит до сервера з застосуванням створеного cgi скрипту з наступним переліком команд [53]:

- Type – вивід на екран вміщуваних текстових файлів;
- Time – вивід системного часу;
- Ver – вивід відомостей про версію Windows;
- Xcopy – копіювання файлів та каталогів дерев;
- Rmdir – видалення папки;
- Rename – перейменування файлів папок;
- Print – вивід на друк місткості текстового файлу
- Mode – конфігурація системних пристроїв;
- Cmd – запуск командної стрічки;
- Cd – ввід або заміна діючої папки;
- Open – відкриття кореневого каталогу;
- Dir – перехід до потрібного каталогу;
- Echo – виведення повідомлень і переключення режиму відображення команд на екрані.

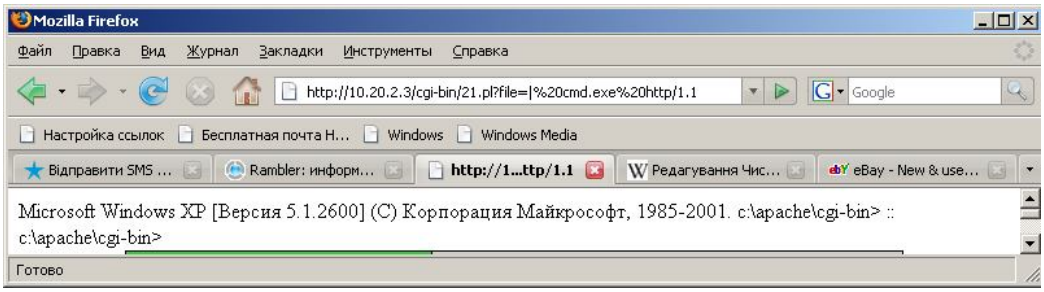


Рисунок 3.2 — Здійснення запиту до сервера

Створення такого cgi скрипта дає змогу одержати права адміністратора через запити до створеного скрипта у HTML сторінці (рис. 3.3).

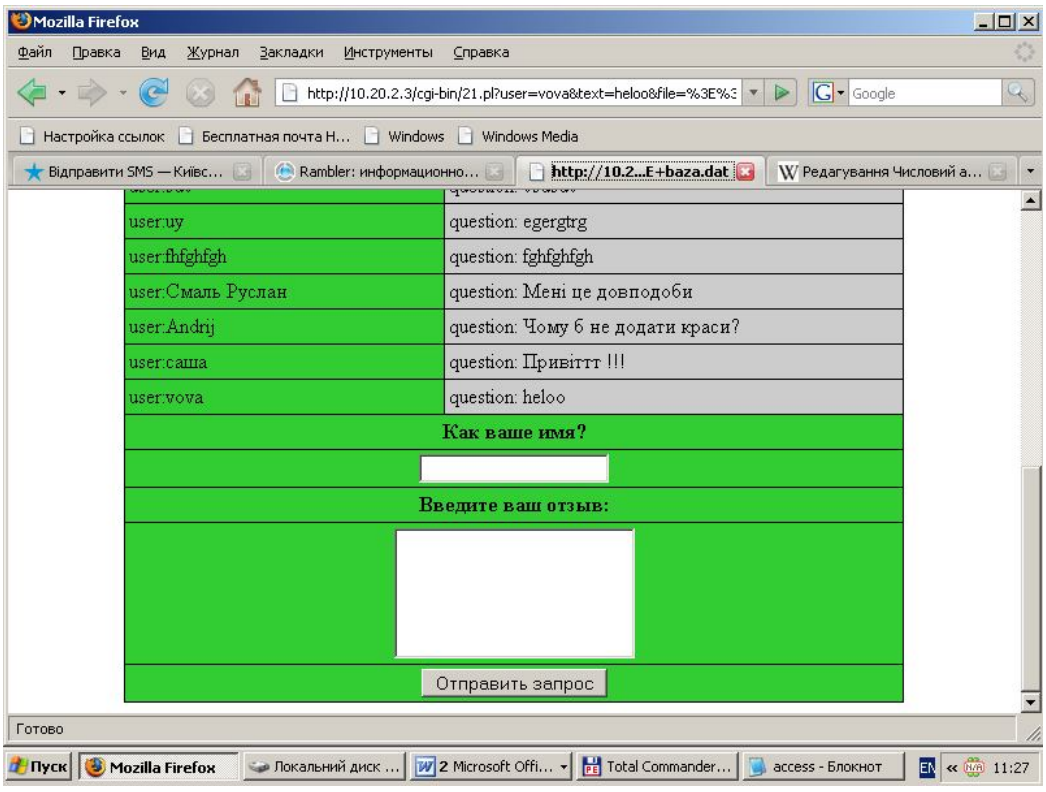


Рисунок 3.3 — Приклад HTML сторінки web – серверу на основі створеного скрипта

Ця сторінка представляє собою своєрідну дошку оголошень, при цьому користувач мережі (після завантаження відповідної сторінки web-сервера), має змогу розмістити потрібну для нього інформацію у вигляді оголошення. Кожен запит до сервера, який здійснюється через “дірявий” скрипт, виконує покладену на нього дію. Прикладом можуть послужити відповідні запити:

"GET /cgi-bin/21.pl?v1=mkdir%20c:\\proba HTTP/1.1" – стрічка дає змогу створити папку на диску C;

"GET /cgi-bin/hello.bat?%3E%3Ehello.txt HTTP/1.1" – цей запит дозволяє створити файл hello.txt та запустити відповідний bat файл;

"GET /cgi-bin/21.pl?v1=date HTTP/1.1" – виводить запит по часу;

"GET /cgi-bin/21.pl?v1=start%20regedit HTTP/1.1" – дозволяє запустити редактор реєстру у windows;

"GET /cgi-bin/21.pl?v1=tree%20c:\\ HTTP/1.1" – можливість перегляду файлів та папок на диску (рис 3.4).

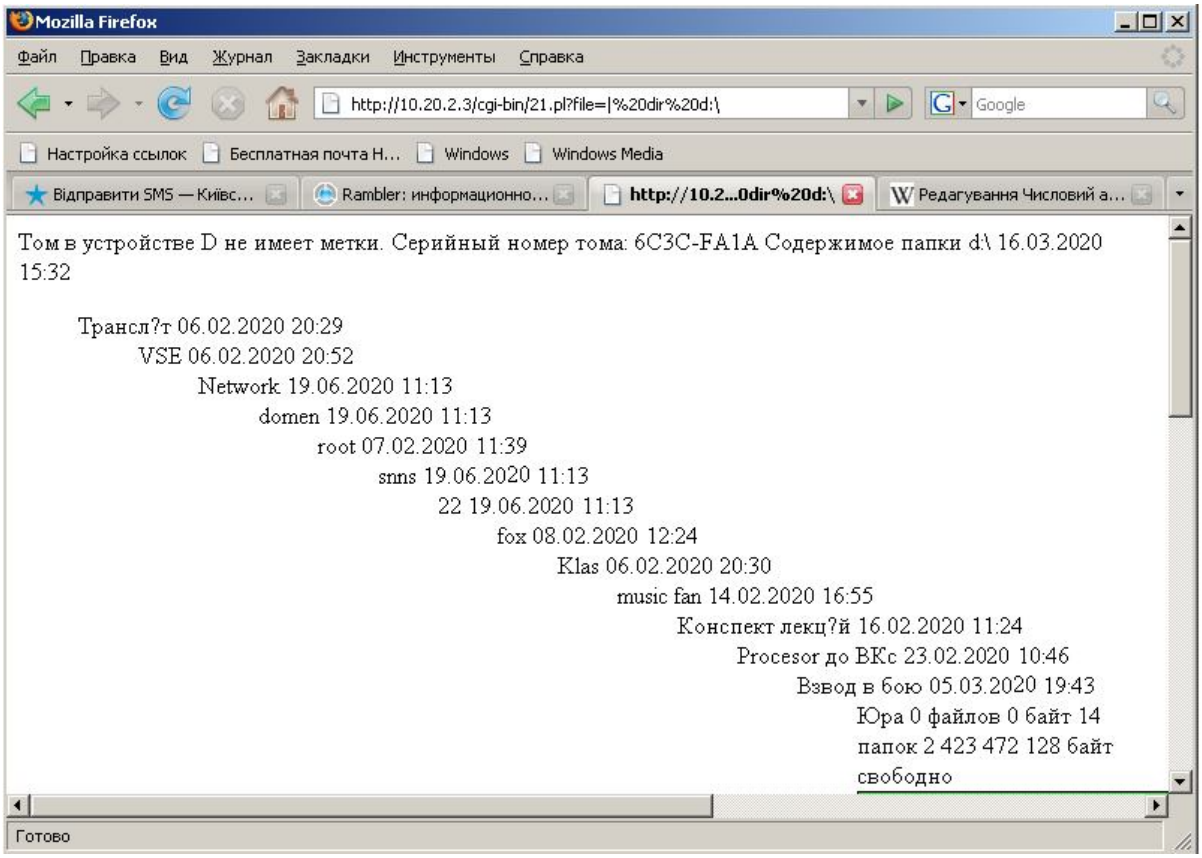


Рисунок 3.4 — Відображення структури папок на диску

Відповідно такі посилання на web-сервер, прописувались та зберігались у access та error файлах у txt форматі. У файлі access прописувались ті стрічки на котрі сервер дав згоду на виконання, а в error - відповідно стрічки, які не отримали згоди сервера.

Отримання access файлу дало змогу сформувати вибірку значень для навчання НМ. При цьому із відповідного файлу відбирались 10 допустимих

значень та інші 10 для формування атаки. Після цього із командної стрічки (рис. 3.5) запускалась програма MD5, яка конвертувала стрічку символів у шістнадцятковий формат, реалізація модифікованої програми MD5 наведена у додатку Б. Згідно алгоритму описаного у 2 розділі кожна стрічка перетворювалась наступним чином:

На вході маємо – “GET /cgi-bin/21.pl?v1=mkdir%20c:\winnt\boot.bat”

На виході отримуємо – “62D5C9297526F1A5B2E9FB616BF8F874”

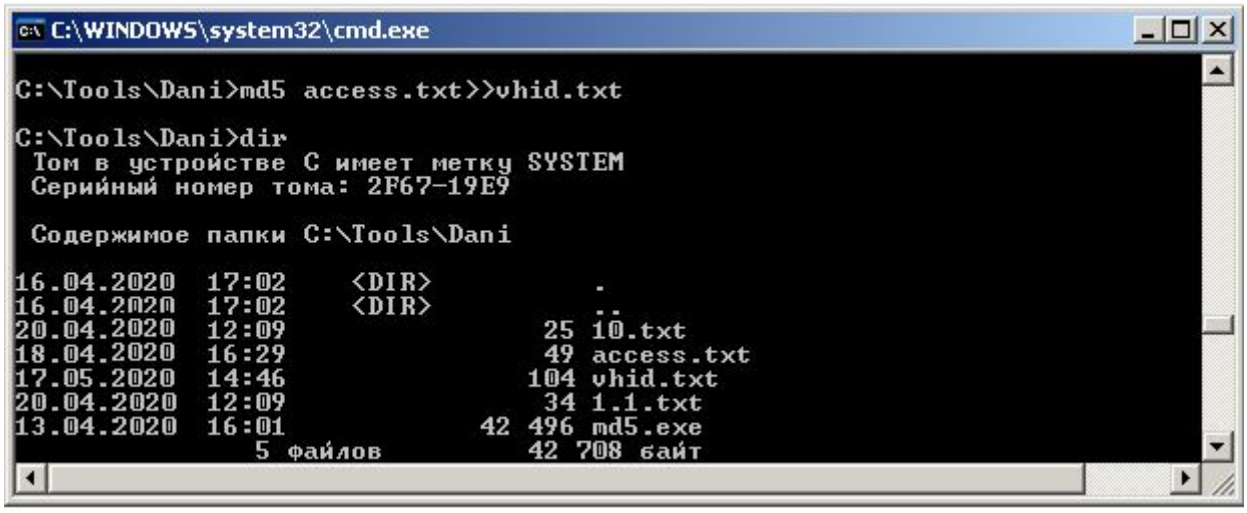


Рисунок 3.5 — Запуск програми MD5

Після перетворення у цифровий формат запускається Matlab, разом зі створеною НМ, при цьому дані автоматично заносяться у НМ у вигляді матриці розміром 20*32. Реалізація навчання мережі подана у додатку Г. На рисунку 3.6 показано процес тренування НМ із застосуванням алгоритму ВРА [35].

НМ створюється за допомогою функції *newff,* яка характеризується наступними параметрами. Масивом мінімальних та максимальних значень образів, кількістю нейронів у схованому та вихідному рівнях, функцією активації та алгоритм навчання:

```
net{nn} = newff(minmax(TrainSet)) [NumbHidNeurons
1], { ‘tansig’ ‘tansig’ }, TrainAlg);
```

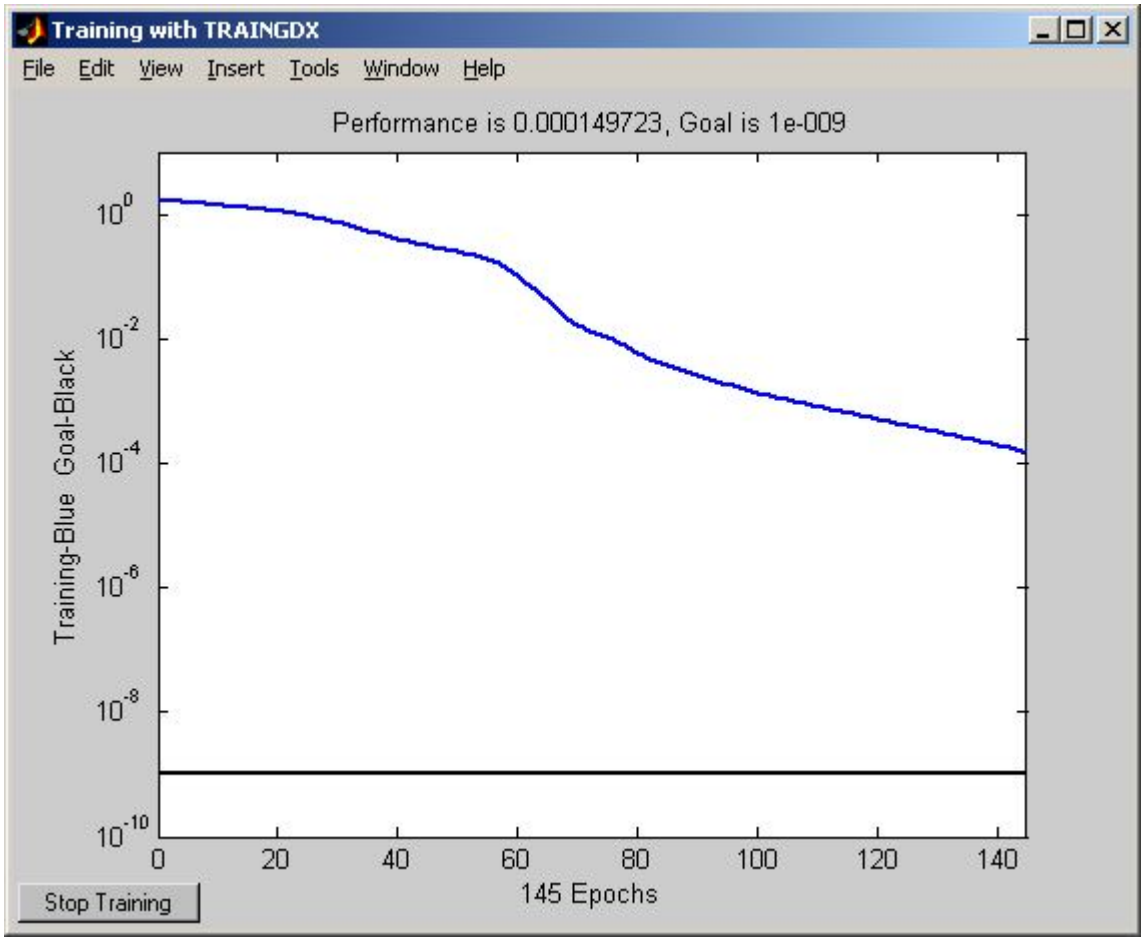


Рисунок 3.6 — Процес навчання нейронної мережі

Навчання НМ проведено з похибкою 10^{-8}, при цьому кількість епох була рівна 10^4, а число нейронів прихованого рівня 30. На рисунку 3.7 показано графік навченої мережі, час навчання становив 17.1740 секунд.

Із графіку навченої НМ (див. рис. 3.7) видно, що після того, як мережа навчилася на 100% вхідних значень, вона змогла чітко розрізнити стрічки, які представляли собою атаку в діапазоні від 1 до -1.

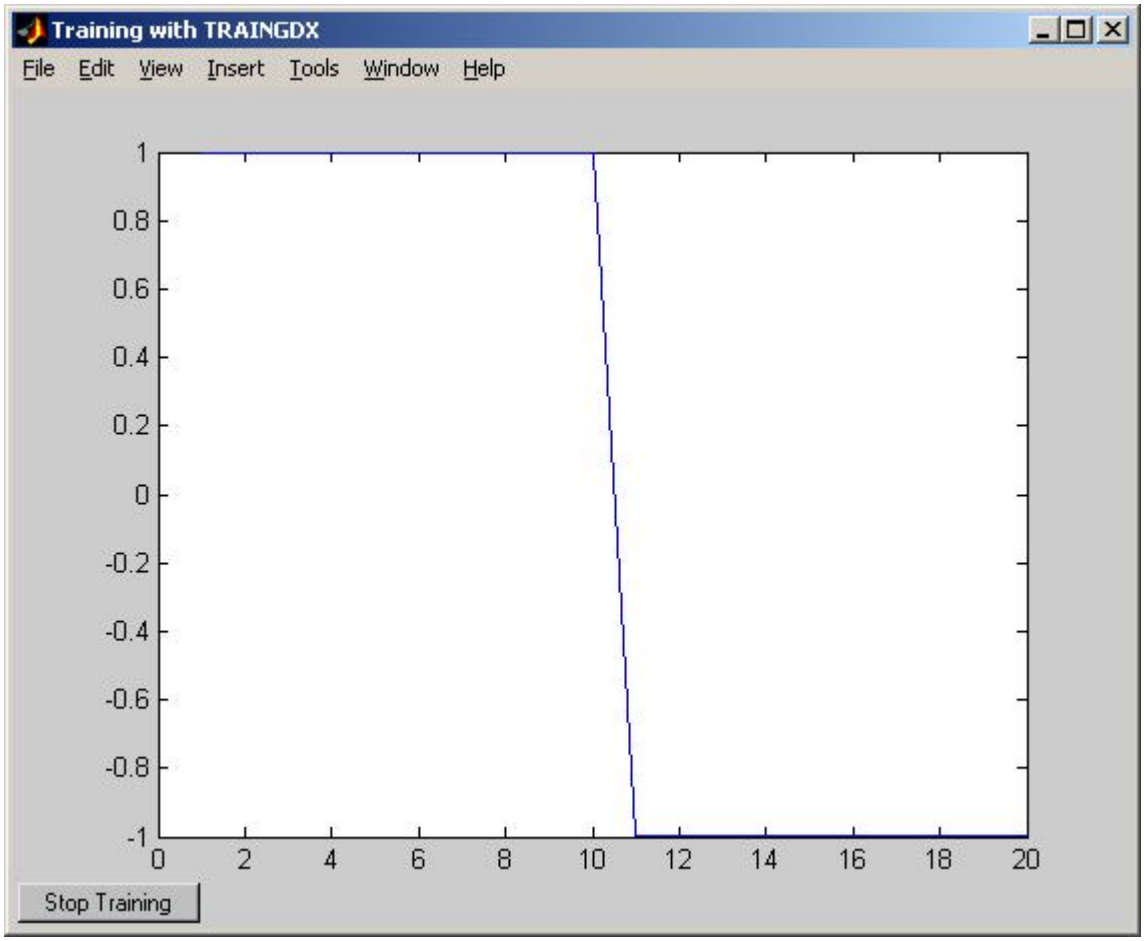


Рисунок 3.7 — Графік навченої НМ

3.2. Тестування системи

Тестування програмного модуля проводилось із використанням засобів Matlab. Для цього використовувалась навчена НМ, програмний код якої наведено у додатку Д, а в якості вхідних значень брали вибірку даних із 20 та 30 значень, які відрізнялись від тих, котрі були задіяні для навчання. На рисунку 3.8 зображено процес тестування операції над даними, які сформовані із access файлу. Значення про виявлення атаки лежать у діапазоні від 1 до -1.

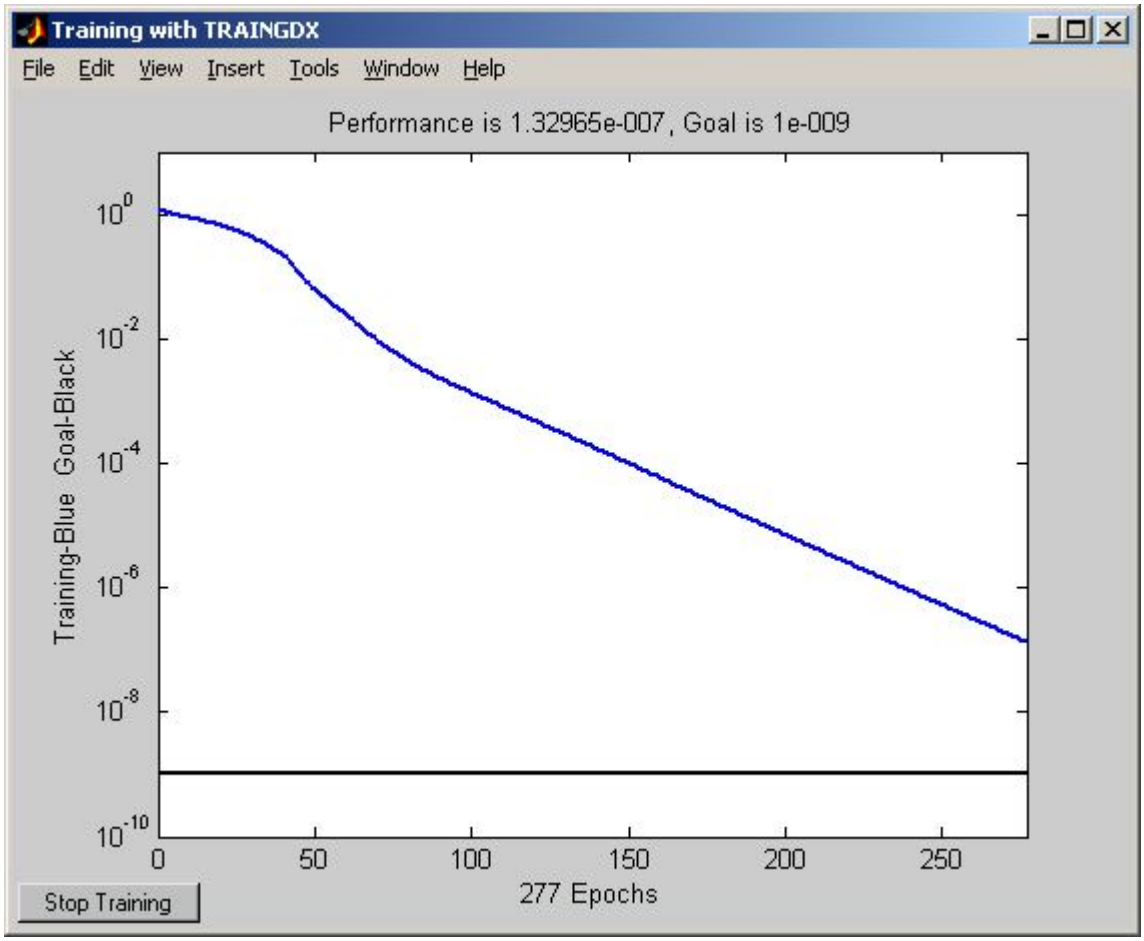


Рисунок 3.8 — Процес верифікації НМ

На рисунку 3.9 показано графік кінцевої верифікації. Експерементальні дослідження розробленої НМ дали змогу розпізнавати та класифікувати наявність атаки приблизно на 80%, тобто із 20 поданих значень мережа чітко розпізнала 7 стрічок як атаку та 2 стрічки як наявність нової атаки, ще не ідентифікованої. Решта образів - це значення, які НМ показала як позитивні, тобто атаки на web – сервер як такої здійснено не було (таблиця 3.1).

Для тренування НМ використовувалась функція train з синтаксисом функції - [net,tr]=train(net,P,T), де

net – об‘єкт, що містить відомості про мережу;

P – вхідні значення мережі;

T – цілі навчання мережі.

Таблиця 3.1

Результати виявлення атаки експериментальним способом

№ Образу	Вхідна кількість значень	Кількість значень про наявність вторгнень	Кількість значень нової атаки	Кількість значень не атаки	Час виконання розпізнавання вхідних образів
1	20	10	—	10	17,676
2	20	5	2	15	17,437
3	20	7	2	13	18,737
4	20	4	1	16	17,215

При цьому функція повертає такі значення:

net – нова мережа;

tr – записи отримані після навчання нейронної;

tr.perf – результат навчання(значення середньоквадратичної похибки);

tr.epoch – кількість епох навчання.

Під час верифікації програмного модуля застосовувалась функція симуляції з синтаксисом функції - A = sim(net,P), де

net – для симуляції використовується об‘єкт, що описує НМ;

P – вхідні дані для симуляції;

A – результат симулювання.

При тестуванні розробленого програмного забезпечення було виявлено наступні загальні переваги [37 55]:

a) Швидкість реакції на втручання та вірність прийнятого рішення;
b) Легкість до видозмінення, тобто можливість змінювати послідовність виконання та використання окремих модулів;
c) Зрозумілість, інтуїтивно зрозумілий інтерфейс усіх ланок системи;
d) Краща здатність виявляти та ідентифікувати зовнішні атаки.

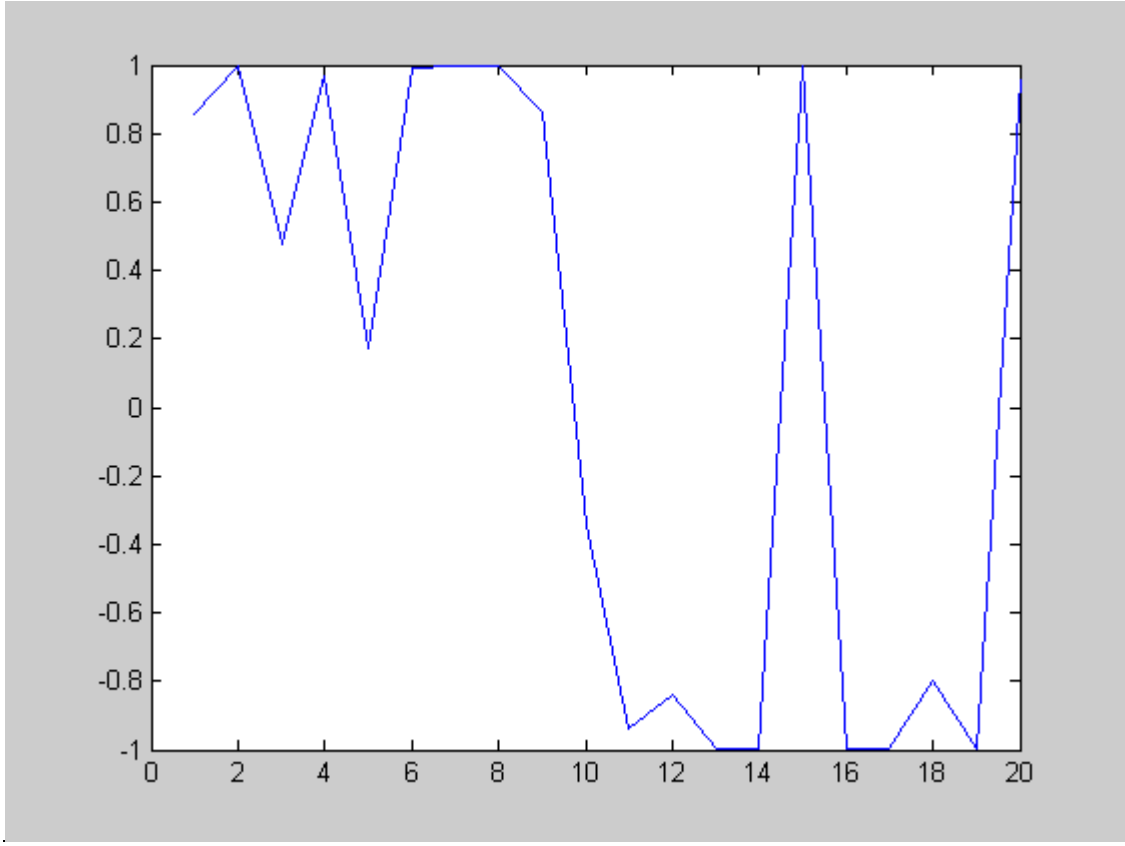


Рисунок 3.9 — Графік класифікації атак з використання НМ

3.3. Можливі сфери використання і впровадження розробленої системи

Оскільки розроблений програмний модуль входить до складу системи honeynet, він працює під операційними системами сімейства Windows. З невеликими змінами його можна використовувати і під операційними системами типу Linux, Solaris. Система дозволяє ідентифікувати противника та взнавати інформацію про нього із застосуванням нейронної мережі включеної до засобів honeynet. А це не так і мало, адже хто попереджений, той озброєний. Забезпечення безпеки інформації традиційно носить оборонний характер, використання honeynet дозволяє перейти від пасивного захисту до активних дій [56].

Honeynet можна використовувати як тест на захищеність, наприклад при розробці нової системи електронної комерції. Звичайно, при розробці спробували врахувати всі нюанси, що стосуються забезпечення безпеки інформації, провели практичне дослідження за допомогою сканера безпеки (RealSecure, Nessus). Проте їй можна влаштувати суворе стендове випробування в "бойових" умовах. Для цього досить просто помістити таку систему в honeynet і подивитися чи не знайдуться "добровільні тестери" яких-небудь вад її безпеки. Було б абсурдом стверджувати, що система "абсолютно безпечна", на підставі того лише факту, що вона простояла у складі honeynet декілька місяців і не була зламана, але такі випробування допоможуть виявити явні опущення в підсистемі інформаційної безпеки.

Створення програмного модуля дозволяє ідентифікувати уже зафіксовані атаки та виявляти нові не зареєстровані з певною імовірністю. Це стало можливим із використанням технологій штучних нейронних мереж. На сьогоднішній день моделювання таких задач здійснюється в середовищі Matlab, що забезпечує зручність моделювання та адекватність отримуваних результатів.

Розроблена система може бути застосована для розпізнавання образів і рішення задачі класифікації, оптимізації і прогнозування. Нижче наведено перелік можливих промислових застосувань, на базі яких вже створені комерційні продукти або реалізовані демонстраційні прототипи, зокрема:

Військова промисловість і аеронавтика:

- обробка звукових сигналів (розділення, ідентифікація, локалізація);
- обробка радарних сигналів (розпізнавання цілей, ідентифікація джерел);
- узагальнення інформації;
- автоматичне пілотування.

Служба безпеки - розпізнавання осіб, голосів, відбитків пальців.

Доцільно запримітити, що розроблений програмний продукт вирішує проблему захисту інформаційних ресурсів комп’ютерних мереж від атак, здійснюваних як правило хакерами, зокрема у сфері виявлення cgi-атак і є складовою частиною структури системи honeynet.

Розроблений в результаті роботи алгоритм доцільно впровадити в системах з цінною інформацією, які мають постійний доступ до Інтернет. Наприклад сервери, які розміщують в аеропортах, комерційних структурах, науково – дослідних та державних установах, тощо.

Результати магістерської роботи частково використані в рамках співпраці між Науково-дослідним інститутом інтелектуальних комп’ютерних систем (НДІ ІКС) та французькими науковцями наукової організації - Honeynet Project [56, 57]. Співпраця передбачає дослідження систем захисту комп'ютерних мереж від мережевих вторгнень в реальному часі. Зокрема планується створити сегмент мережі з двох Honeypot, з'єднаних із зовнішньою мережею через маршрутизатор. Таким чином - буде побудована повноцінна Honeynet, де можна буде реалізовувати всі наявні технології наприклад: Sebek, Honeyd та ін. В НДІ ІКС було обладнано комп'ютер (Honeypot) під'єднаний в комутатор мережі ТНЕУ, який використовувався як мережевий сенсор (на базі технології Honeynet). Під час проходження практики в НДІ ІКС автором встановлено програмне забезпечення на комп'ютер Honeypot НДІ ІКС і на протязі часу квітень – травень 2007р були проведені дослідження комп’ютерних атак з метою створення навчальної вибірки (додаток Е). Розроблені програмні засоби впровадженні та пройшли тестування в локальній мережі НДІ ІКС. Автором на основі використаного нейромережевого підходу до виявлення атак, досліджено існуючі види атак та отримано відповідні результати (див. підрозділ 3.2).

Висновки до третього розділу

У третьому розділі наведено опис розробленого програмного комплексу, його особливостей побудови та структуру написання програмного коду. Опис програми проводиться з використанням екранних форм. Проведено верифікацію модуля та окреслено можливі сфери використання. В результаті отримано програмний додаток, що дасть змогу виявляти порушників мережі Інтернет у поєднанні із такими системами як IDS та Firewall.

РОЗДІЛ 4. ОХОРОНА ПРАЦІ ТА БЕЗПЕКА У НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1 Методи підвищення мотивації безпеки праці

Потрібно знайти такі способи впливу на людей, щоб вони усвідомили необхідність працювати безпечно, створити такі «правила гри», в межах яких людині було б вигідно дотримуватися встановлених норм.

Безпечна поведінка на виробництві залежить не тільки від професійних знань, навичок і здібностей, а й значною мірою від мотивів поведінки працівника. Відповідно управляти діями людини можна тільки за допомогою управління її мотивами.

В обмін за свій труд працівники очікують не тільки високої оплати, а й створення умов для особистісного росту, отримання задоволення від власної роботи, інших компенсацій, які адекватні професійному рівню та відповідають особистим інтересам. Ефективна праця допомагає швидшому розвиванню підприємства.

Для заохочення працівників потрібно підбадьорення та підтримка з сторони начальства. Стимулювати ефективну роботу можуть матеріальні методи, наприклад премії, винагороди, безкоштовне харчування, додатковий дохід та інше. Мотиваційний комплекс взагалі й безпечної поведінки людини зокрема носить полімотивований характер, містить у собі широкий спектр мотиваційних регуляторів як матеріального, так і нематеріального характеру та має певну ієрархічність. На особистісному рівні працювати продуктивно та безпечно вигідно самій людині; від цього залежить успіх роботи її підрозділу; і нарешті - це потрібно підприємству (компанії). Тобто для вирішення охоронних проблем у праці потрібно зацікавити працівників трудитися безпечно не тільки для себе, а й для оточуючих.

Практично будь-якого працівника можна зацікавити будь-яким мотивом, оскільки абсолютно ні на що не мотивованих людей немає! Очевидно, тільки

закликами, зверненнями, деклараціями, пропагандою ці проблеми навряд чи вдасться вирішити.

Потрібно знайти такі способи впливу на людей, щоб вони усвідомили необхідність працювати безпечно, створити такі «правила гри», в межах яких людині було б вигідно дотримуватися встановлених регламентів. І цей вплив вона повинна відчувати безпосередньо в процесі всієї трудової діяльності.

Проаналізувавши загальні методи мотивації для підвищення роботи працівників можемо охарактеризувати методи які потрібні для підвищення мотивації безпеки праці осіб на різних підприємствах. Найголовнішим чинником для будь-якого підприємства має бути на першому місці створення безпечних умов праці та дотримання всіх необхідних безпечних заходів для своїх працівників. Важливим показником охорони праці на підприємстві є внутрішнє стимулювання для безпечного ведення робіт. При можливості на об'єкті можуть працівники долучитись до охорони праці та запропонувати свої варіанти, підвівши підсумки можуть скласти договір. У цій угоді, яку склали колективно можуть вказати свої матеріальні та нематеріальні вимоги. При цьому не може бути системи стимулювання, яка мотивує всіх співробітників однаково.

Система стимулів має бути персоніфікованою, ретельно дозованою та розроблятися для кожної людини або певної групи людей з подібними домінуючими потребами, або загальна система має індивідуалізуватися. Тому моніторинг домінуючих потреб персоналу - необхідна умова функціонування мотиваційного механізму.

Виходячи з цього, можна визначити види стимулюючих винагород. Вони можуть бути матеріальними, моральними, соціально значимими, морально – психологічними.

Якщо на підприємстві працюють бригади, цехи то корисно буде відзначити їх та видати премію за дотримання усіх вимог щодо безпеки на робочих місцях, без травм чи інших пошкоджень.

Якщо на підприємстві працівник виконує роботу у небезпечних для його здоров'я ділянках то йому необхідно надавати надбавку до заробітної плати,

адже він ризикує своїм здоров’ям та ставиться до роботи з високою обережністю.

Крім матеріального дуже високою цінністю буде моральна підтримка та похвала з сторони керівника, організація відпочинку, екскурсії, влаштування пікніку для робітників які сумлінно дотримувались правил з охорони праці.

Таким методом не тільки користуються у нашій країні, але і використовують закордонні фірми.

До методів для заохочення можемо виділити матеріальні як уже писалось вище, також можуть відноситись моральні тобто подяка у усній чи письмовій формі, відзначення перемоги та інше. Крім методів для заохочення також можуть бути методи покарання за недотримання правил та вимог щодо безпеки охорони праці.

До методів покарання можемо віднести матеріальні покарання у вигляді штрафів, менша вартість виплат, а до моральних ми можемо віднести критику з сторони начальства та з сторони співробітників, також можуть бути проведені окремі бесіди з працівником та обговорення в колективі.

В загальному створення на підприємстві безпечних умов та мотивації робітників принесе тільки позитивний результат праці а також зменшить випадки аварій на підприємствах та травм для здоров’я працівників. Для того, щоб підвищити та утримувати мотивацію працівників на необхідному рівні, забезпечити результативність і безпеку роботи, потрібно сформувати цілісну систему стимулів.

Ця система не повинна зводитися лише до росту зарплати. Вона може включати просування по службі, планування професійної кар'єри, можливість підвищити рівень знань тощо.

Тобто слід використовувати повний спектр матеріальних і нематеріальних важелів стимулювання. У зв'язку з цим найважливішим механізмом реалізації мотиваційних принципів і управління персоналом можуть бути мотиваційні програми, які розробляються в деяких західних компаніях. Прикладом може бути спеціальна система «Pay for Performance» («Плата за виконання»), що дозволяє визначити відповідність ефективності

діяльності конкретного працівника з розміром винагороди, яку він отримає. Чітка система цілей і критеріїв оцінки, реалізація їх кожним працівником - одна з умов розроблення програм.

При розробленні мотиваційної програми слід враховувати, що для будь-якої людини природнім є задоволення, насамперед, особистих потреб - підвищення рівня добробуту, самореалізація та самовираження, підвищення соціального статусу, віра в можливість досягнення бажаного, а крім того - характер роботи (сам процес).

Ефективне розроблення програм мотивації персоналу в першу чергу спрямоване на спонукання працівників до безпечної діяльності шляхом формування внутрішніх мотивів поведінки.

Причому розрив між особистою метою кожного працівника та загальною метою діяльності підприємства має бути мінімальним. Виходячи з цього, можна використовувати такі методи впливу на мотиви, які стимулюють безпечну поведінку працівників: установити працівникам чітку мету щодо дотримання правил безпеки; створити умови для можливості досягнення цієї мети; визначити винагороду, яку хотіли б отримати працівники; домогтися, щоб вони розуміли залежність між дотриманням правил безпеки та отриманням винагороди.

4.2 Забезпечення захисту працівників суб'єкта господарювання від іонізуючих випромінювань

Іонізуюче випромінювання або радіоактивність є небезпечним явищем для людського організму. При взаємодії впливу іонізаційних випромінювань у навколишнє середовище можуть відбутись різні утворення зарядів . Існують два різновиди випромінювання – «альфа» та «бета».

В залежності від носія та енергії, вони мають різну проникаючу здатність. Альфа це випромінювання яке проявляється важкими частинами складеними з протонів і нейтронів.

В свою чергу бета випромінювання являє собою ланцюг електронів та позитронів які є більшу здатність проникати у середовище. Працюючи на таких територіях, де існує радіаційна атмосфера можуть виникнути різні випадки.

На підприємстві можуть виникнути інциденти при користуванні ядерними матеріалами , зберіганні радіоактивних відходів в наслідок чого працівники можуть отримати травму у вигляді дози опромінення, використання іонізуючих джерел випромінювання.

Також у випадку такої радіаційної аварії забруднюється навколишнє середовище, люди можуть отримати травму у вигляді потужної дози опромінення. Призвести аварію на підприємстві може також якщо активна реакційна речовина знаходиться у роботі та це відбувається незаконно.

Це може привезти до опромінення жителів та перевищити межу дози опромінення. Частинки з цього випромінювання можуть залишати сліди на дихальній системі на травній системі людського організму. Також ці елементи можуть бути у водних каналах, які постачають питну воду людям.

На підприємстві де проводяться роботи з радіаційними речовинами обов’язково мають вживатись заходи проти радіації. Протирадіаційні захисти це така система правових, організаційних норм та санітарної гігієни.

До переліку таких захистів можна включити медичні заходи для забезпечення радіаційної безпеки персоналу та проектно-конструкторські. Для організації заходів проти іонізації опромінювання підприємство має ввести обов’язкові методи щоб подбати про безпеку працюючого персоналу. До таких методів можуть належати заходи які обмежують допуск працівників до джерел які випромінюють радіацію.

До таких працівників можемо віднести таких, які не підходять за віком, за статтю та працівники які вже отримали дозу випромінення. Підприємство мусить створити сприятливі умови що дотримуються встановлених норм та вимог для працівників та застосовувати індивідуальні засоби для захисту працівника цього підприємства.

Організація повинна контролювати рівні опромінювання та вести інформаційну систему про стан радіації на підприємстві та призначених місць для праці.

На підприємстві повинні бути проведені заходи щодо організації безпеки для робіт які проводяться у радіаційних ділянках а саме:

-організація роботи нарядів та розпоряджень;

-організація та перевірка пропусків до робочих місць;

-оформлення контролю за процесом виконання роботи;

-введення примусового часу на перерву та вчасне закінчення робочого процесу.

До фізичних норм захисту проти радіації існують перешкоди поширення іонізації опромінень. Для поширення дози випромінювання може бути ряд перешкод, залежать вони від кількості годин, перешкоджати може дистанція , також перешкодою може бути чисельність.

Реалізувати заходи проти радіації за певний відрізок часу можливо, тим що працівники , які працюють з іонізованими випромінюваннями можуть виконувати вчасно свою роботу ,відповідно керівництво може за якісну роботу зменшити кількість робочих днів у тижні.

Цим самим вони застереженням вони зменшать знаходження працівників у зоні випромінювання та відповідно буде менше контактування з радіаційними приладами. Захистити працівників за допомогою відстані підприємство може шляхом доцільного розміщення приміщення, правильно розставити та розрахувати робочі місця для працівників а також забезпечити приладами, які зможуть контактувати, керувати робочим процесом з технікою яка має радіаційний вплив на відстані.

Слугувати захистом може покриття свинцем меблів які присутні у приміщенні (двері, вікна, робочі столи), створення перекриття між поверхами та перегородки. Працівникам обов’язково має бути виданий спеціальний одяг ,такі як фартухи, шапочки та рукавиці зшиті з просвинцевої тканини.

Розміщення робочих місць повинно мати правильний розрахунок на загальну кімнату, не робити перенабір та забезпечити відповідним та

необхідним обладнанням робочі кабінети. При користуванні відкритими приладами іонізованого опромінення провести герметизації цих систем, при можливості використовувати роботу техніки. Підприємство повинне вжити усіх санітарно-гігієнічних заходів та соціальних, а також важливо необхідний є медичний захист робочих на об’єкті.

4.3 Висновок до розділу 4

В даному розділі описано заходи та методи із забезпечення радіаційних впливів та іонізації опромінювання на підприємствах. Описані вимоги для керівництва та підлеглих працюючих на об’єктах щодо їхніх дій в разі виникнення радіації .

Також описані вимоги для мотивації робітників щодо дотримання правил охорони праці на підприємстві. Мотивація - одна з центральних функцій управління як персоналом, так і охороною праці. Вона може відігравати важливу роль як фактор спонукання персоналу діяти адекватним способом у власних і корпоративних інтересах.

Для цього потрібно, щоб мета підприємства збігалася з метою працівників. Однак мотивація одночасно є не тільки рушійним механізмом, а й фактором залучення, наприклад, до охорони праці, високопрофесійних спеціалістів. Це механізм, що спонукає вдосконалювати систему управління. Крім того, рівень мотивації працівників відіграє важливу роль у загальному успіху підприємства (компанії).

Розглянуто методи для заохочення працівників дотримуватись правил охорони праці, поділено та описано матеріальні та нематеріальні заходи. Розглянуто у розділі заходи для безпеки від іонізуючого опромінення та як потрібно реалізовувати ці заходи на відповідних підприємствах.

ВИСНОВКИ

У даній роботі для виявлення атак описано існуючі методи, найбільш використовувані на сьогодні є сигнатурний метод, метод аномалій, комбінований та статистичний. Приведено структуру системи honeynet, етапи її становлення та зроблено оцінку сучасних систем виявлення атак. В основі запропонованого методу лежить сигнатурний метод. Запропонований метод дає переваги по швидкості виконання у порівнянні із статистичними методами. Таким, чином він є зручнішим для опису і аналізу засобів виявлення атак, як тих котрі раніше зустрічались, так і нових не зареєстрованих.

Запропоновано та удосконалено метод ідентифікації атак на основі сигнатур. Розроблено метод виявлення атак на основі якого побудовано алгоритм ідентифікації комп'ютерних атак. Алгоритм ідентифікації комп'ютерних атак організований і представлений як алгоритм навчання та класифікації нейронної мережі. Наведено його переваги та недоліки. Представлено розроблену структуру модуля виявлення атаки, для реалізіції цього використанно нейронну мережу.

Проведено опис розробленого програмного комплексу, його особливостей побудови та структуру написання програмного коду. Опис програми проводиться з використанням екранних форм. Проведено верифікацію модуля та окреслено можливі сфери використання. В результаті отримано програмний додаток, що дає змогу виявляти порушників мережі Інтернет у поєднанні із такими системами як IDS та Firewall.

Програмний модуль є складовою частиною структури honeynet, а тому дозволяє виявляти правопорушників, прослідковувати та аналізувати їх тактику та мотиви здійснення злому. У недалекому майбутньому використання нейронних мереж для ідентифікації атак у honeynet відкриває нову область для розвитку та досліджень у цьому напрямку. А саме впровадження нейронних мереж у цю структуру дозволить здійснювати спроби ідентифікації атак таких як DDos-атаки, атаки типу „розподілена відмова в обслуговуванні".

ПЕРЕЛІК ДЖЕРЕЛ

1. Маклаков С.В. BPwin и ERwin: CASE-средства для разработки информационных систем. – М.: Диалог-Мифи, 1999. - 295 с.

2. Федорова Д.Э., Семенов Ю.Д., Чижик К.Н. CASE-технологии. - М.: Горячая линия Телеком, Радио и связь, 2005. – 160 с.

3. Самойлов, В. Д. Модельное конструирование компьютерных приложений / В. Д. Самойлов. - К.: Наукова думка. - 2007. - 198 с.

4. Love Bug Damage costs Rise to 6.7 Billion available at http://www.businesseconomic.com/cei/press.index.html.

5. А.В. Лукацкий. Вопросы инфомационной безопасности.- http://www.infosec.ru/press/pub/t_v_1.zip.

6. А.В. Лукацкий. Адаптивное управление защитой. – Сети, №10, 1999р.

7. Э.С. Абрамов. Разработка комбинированной архитектуры системы обнаружения и выявления сетевых атак // Материалы 3-ей международной научно-практической конференции «Информационная безопасность», Таганрог 2001р.

8. Вильям Столлинге. Криптография и защита сетей. Принципы и практика, 2е издание. - М.: “Вильямс”, 2002.

9. М. Ранум. Обнаружения атак: реальность и мифы - http://security.tsu.ru/.

10. А.В. Лукацкий. Обнаружения атак. – 2-е изд., перераб. и доп. СПб.:БХВ – Петербург, 2003.

11. Edward Amoroso. Intrusion Detection. An Introduction to Internet Surveillance, Correlation, Trace Back, Traps, and Response. Intrusion.Net Books, 1999.

12. Б. Анин. Защита компьютерной информации - http://bugtraq.ru/library/books/attack1/chapter7/c72.html.

13. Richard Power, "2000 CSI/FBI Computer Crime and Security Survey", Computer and Security Issues and Trends, Vol. 6, No. 1, Spring 2000.

14. "UK e-business at risk from hackers, reveals report" available at http://www.ananova.com/news/story/internet_security_79176.html.

15. А.В. Лукацкий. Безопасность сети банка глазами специалистов - http://securitylab.ru/tools/22111.html.

16. В.В. Домареев. Защита информации и безопасность компьютерных систем - http://www.softline.ru/course.

17. А.В. Лукацкий. Системы обнаружения атак – СПб.: БХВ – Санкт-Петербург, 1999. – 58 С.

18. CERT Coordination Center. CERT Advisory CA-1998-01: Smurf IP Denial-of-Service Attacks. <http://www.cert.org/advisories /CA-1998-01.html> (January 5, 1998; last revised March 13, 2000).

19. Y.Ho, D. Frinke, D. Tobin. Planning, Petri-Nets and Intrusion Detection, Taylor & Francis, 1998 324 pages.

20. Justin J. Lister. Latest Developments and New Technologies for Detecting and Preventing Computer, Communication and Financial Fraud.

21. О.Ю. Гаценко. Защита информации. М.: Сентябрь, 2001. – 228с.

22. В.В. Маснянкин. Перевод на русский язик материалов honeynet project - http://cybervlad.net/lspitz/enemy/index.html.

23. The Honeynet Project. Know Your Enemy: Honeynets. Whitepaper, 2003 — http://project.honeynet.org/papers/honeynet/index.html.

24. The Honeynet Project. Know Your Enemy: The motives and psychology of the blackhat community. Whitepaper, 2000 - http://project.honeynet.org/papers/motives/index.html.

25. В.И. Завгородний. Комплексная защита информации в компьютерных системах. - М.: Логос. 2001. – 264 С.

26. The Honeynet Project. Know Your Enemy: The Tools and Methodologies of the Script Kiddie. Whitepaper, 2000 — http://project.honeynet.org/papers/enemy/index.html.

27. Л. Спитцнер. Honeynet Project: ловушка для хакеров// Открытые системы. - № 7-8, 2003. — http://www.osp.ru/os/2003/07-08/061.htm.

28. Spitzner L. Honeypots: Tracking Hackers. — Addison-Wesley

Professional, 2002. - 480 p.

29. Postel, J. (ed.), "Internet Protocol - DARPA Internet Program Protocol Specification," RFC 791, USC/Information Sciences Institute, September 2002 - http://www.citforum.ru/internet/tifamily/icmpspec.shtml.

30. Spitzner L. Honeypots: Tracking Hackers. — Addison-Wesley Professional, 2002. — 480 p.

31. Ю.В. Романец, П.А. Тимофеев, В.Ф. Шаньгин. Защита информации в компьютерных системах и сетях. М.: Радио и связь. 2001. – 376 С.

32. Е.А. Степанов, И.К. Корнеев. Информационная безопасность и защита информации. Учебное пособие. М.: Инфа-М, 2001. – 304 С.

33. А.В. Галицкий, С.Д. Рябко, В.Ф. Шаньгин. Защита информации в сети. Анализ технологий и синтез решений. М.: ДМК Пресс, 2004. – 616 С.

34. А.В. Соколов, В.Ф. Шаньгин. Защита информации в распределенных корпоративных сетях и системах. М.: ДМК Пресс, 2002. – 656 С.

35. H. Demuth., M. Beale. Neural Network Toolbox/ForUse with MATLAB - http://www.mathworks.com, ftp.mathworks.com. – 1992. – 1997 by The MathWorks, Inc.

36. R. Malayter. By MD5 and SHA. - www.secure-hash-algorithm-md5-sha-1.co.uk.

37. M.T. Hagan., H. Demuth. Neural Network Design, Boston: PWS Publishing Company, 1996.

38. А.О. Шеременда. Використання нейронних мереж(НМ) для виявлення комп’ютерних атак // Матеріали 10-ї наукової конференції Тернопільського державног технічного університету ім. Івана Пулюя.- Тернопіль:ТДТУ, 2007- С. 146.

39. С. Осовский. Нейронные сети для обработки информации. – М.: Финансы и статистика, 2002р.

40. H. Debar., M. Becke., & D. Siboni. (1992). A Neural Network Component for an Intrusion Detection System. In Proceedings of the IEEE Computer Society Symposium on Research in Security and Privacy.

41. L. Kevin., Henning, R. Rhonda., and H. Jonathan. (1990). A Neural Network Approach Towards Intrusion Detection. In Proceedings of the 13th National Computer Security Conference.

42. О.А. Савенкова. Исследование алгоритмов обучения модели нейронной сети при распознавании речевых сигналов - http://www.dgma.donetsk.ua/%7Ekiber/index.htm.

43. Houle, Kevin J.; Weaver, George M.; Long, Neil; & Thomas, Rob. Trends In Denial of Service Attack Technology, CERT Coordination Center, October 2001 - http://www.cert.org/archive/pdf/DoS_trends.pdf.

44. F.H. Vonwangelin. Context menu MD5 - http://www.vonwangelin.com/md5/index.html.

45. Robert Rothenburg. MD5 implementation for Turbo Pascal - http://www.squid/spylog.com.

46. В.М. Варакін. Структура системы и ее компоненты. - http://www.unicyb.kiev.ua/~kga/pi/lec7.doc.

47. И. Трифаленков, В. Макоев. Критерии выбора средств защиты информации. pdf CIO N5/2002. - http://www.jetinfosoft.ru/download/publication.html.

48. Маняшин С.М., Жуков И.Ю., Свирин И.С., Юраков Ю.Д. Защита информации в автоматизированных системах военного назначения в условиях современных угроз // Безопасность сетей и средств связи. – 2006. – №1. – С. 137 – 146.

49. Свирин И.С. Методология построения и оценки адекватности модели штатного поведения системы // Методы и технические средства обеспечения безопасности информации: СПб., 2006. – С. 104.

50. Фролов А.В., Фролов Г.В. Глобальные сети компьютеров. Практическое введение в Internet, E-mail, FTP, WWW, и HTML, программирование для Windiws Sockets. - Диалог - МИФИ, 1996. - 283 с.

51. Левин В.К. Защита информации в информационно-вычислительных системах и сетях // Программирование. - 1994. - N5. - С. 5-16.

52. В.Г. Олифер, Н.А. Олифер Компьютерные сети. Принципы, технологии, протоколы. — СПб.:Питер, 2001 – 672 С.

53. Максим Кульгин. Компьютерные сети. Практика построения. П.: Питер, 2003. – 464 С.

54. www.lhg.ru – сайт програмування на мові PERL.

55. Джеймс Ф., Куроуз В., Кит В. Компьютерные сети. Многоуровневая архитектура Интернета. П.: Питер, 2004. – 765 С.

56. Марк Спортаж, Френк Паппас. Компьютерные сети и сетевые технологии. М.: ТИД «ДС», 2002. – 736 С.

57. Р.М. Романяк. Методи і засоби відслідковування та локалізації віддаленого комп’ютера // Магістерська робота. – Тернопіль: ТАНГ, 2004. – 87 С.

58. Carole A. Lane. Naked in cyberspace, Wilton, CT, 1997. – 513 pages.

59. Proceedings of the NATO Advanced Research Workshop on Cyberspace Security and Defense: Research Issues. Gdansk, Poland, 6-9 September 2004.

ДОДАТКИ

Додаток А

М А Т Е Р І А Л И

ІХ НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ, СИСТЕМИ ТА ТЕХНОЛОГІЇ»

8–9 грудня 2021 року

ТЕРНОПІЛЬ
2021

УДК 004.031.6
Л. Матійчук, кан. екон. наук, доцент, В. Сташук, І. Павлов
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

ОЦІНКА ІСНУЮЧИХ СИСТЕМ ВИЯВЛЕННЯ АТАК

UDC 004.031.6
L. Matiychuk, PhD, Assoc. Prof., I. Pavlov, V. Stashuk

EVALUATION OF EXISTING ATTACK DETECTION SYSTEMS

На сьогоднішній день дуже інтенсивно розвиваються технології захисту корпоративних мереж, які включають в себе:

- ME (Firewall) – це програма або спеціалізована апаратна реалізація, що, грунтуючись на деяких правилах, дозволяє або забороняє передачу інформації, що проходить через неї, з метою обмеження деякої підмережі від зовнішнього доступу чи навпаки, для заборони виходу назовні. Міжмережеві екрани реалізують механізми контролю доступу із зовнішньої мережі до внутрішньої шляхом фільтрації всього вхідного і вихідного трафіку, пропускаючи тільки авторизовані дані. Всі міжмережеві екрани функціонують на основі інформації, яка отримується з різних рівнів еталонної моделі ISO/OSI, і чим вищий рівень OSI, на основі якого побудований ME, тим вищий рівень захисту, що ним забезпечується. Існують три основних типи міжмережевих екранів – пакетний фільтр (packet filtering), шлюз на сеансовому рівні (circuit-level gateway) і шлюз на прикладному рівні (application-level gateway). Існує дуже мало міжмережевих екранів, які можуть бути одночасно віднесені до одного з названих типів. Як правило, Firewall суміщає в собі функції двох або трьох типів.
- Найбільш очевидний недолік ME – неможливість захисту від користувачів, які знають ідентифікатор і пароль для доступу в сегмент корпоративної мережі, який захищається. ME може обмежити доступ до ресурсів, але він не може заборонити авторизованому користувачу скопіювати цінну інформацію або змінити які-небудь параметри. А по статистиці не менше ніж 70% всіх загроз безпеці надходить зі сторони співробітників організації.
- Віртуальна приватна мережа (VPN – Virtual Private Network). Технологія VPN призначена для побудови єдиного прозорого користувацького середовища поверх будь-якої транспортної мережі. Таке рішення дозволяє організувати: безпечний віддалений доступ персоналу до мережі підприємства чи організації з будь-якого робочого місця, підключеного до мережі Інтернет; достовірний підрахунок вживаних абонентом ресурсів в широкомовних транспортних мережах (наприклад, в мережах Ethernet); безпечну передачу конфіденційної інформації по мережі Інтернет без побудови додаткових фізичних каналів зв'язку. При побудові таких мереж можливо використовувати як комутовані канали зв'язку (dial-up), так і некомутовані (виділені лінії). При цьому для забезпечення конфіденційності інформації, що передається, не потребується організація додаткової виділеної лінії, а можливе використання вже існуючої, що значно знижує вартість побудови мереж VPN. Безпека інформації, що передається по мережі забезпечується шляхом шифрування з використанням будь-якого з наявних криптоалгоритмів.
- Сканер безпеки. Класичним сканером, який поставляється з усіма *nix подібними операційними системами є nmap. Програма призначена для сканування мереж з будь-якою кількістю об'єктів, визначаючи стан об'єктів мережі, що сканується а також портів і відповідних служб. Для цього nmap використовує багато різних методів сканування таких як UDP, TCP connect(), TCP SYN (напіввідкрите), FTP proxy (прорив через ftp), Reverse-ident, ICMP (ping), FIN, ACK, Xmas tree, SYN і NULL сканування.
- Одним з найновіших сканерів безпеки є Nessus. Nessus являє собою безплатний сучасний сканер безпеки локальних і віддалених систем. Початок Nessus Project було покладено в 1998 році, перший реліз вийшов в квітні. На той період найпоширенішим сканером безпеки був SATAN. Задачею Nessus являється визначення запущених служб і вразливостей, включаючи

найпопулярніші повідомленні про „дірки" wu-ftpd, наявність демонів DDOS, проблеми ipfw FreeBS і ін. Основний принцип полягає в тому, що вся інформації потребує перевірки, тобто інформація багерів основних служб не вважається основоположною.

- Систему виявлення вторгнень (IDS). Для запобігання комп'ютерним атакам, необхідно розробляти та налаштовувати системи захисту інформації та системи виявлення атак. Системи виявлення комп'ютерних атак – це один із найважливіших елементів систем інформаційної безпеки мереж. Враховуючи зростання в останні роки число проблем зв'язаних з комп'ютерною безпекою постійно зростає, як і пов'язаних з ними число хакерських атак (рис. 1). Системи виявлення вторгнень включають в себе: виявлення спроб несанкціонованого доступу та захист від атак типу „відмова в обслуговуванні" (DOS-атак).

Виявлення атак потребує виконання однієї із двох умов: розуміння очікуваної поведінки підконтрольного об'єкта системи або знання всіх можливих атак і їх модифікацій.

При створенні систем виявлення атак використовуються два основні підходи:

– виявлення аномальної поведінки, використовуючи апарат математичної статистики, який досить добре себе зарекомендував. Даний підхід використовується , як правило, при виявленні DoS-атак, які використовують посилку великої кількості трафіку за короткий інтервал часу [25];

– виявлення зловживань, використовуючи сигнатури, що описують послідовність байт і дій, які характеризують несанкціоновану діяльність. Цей підхід знайомий по антивірусних системах, які побудовані саме за цим принципом.

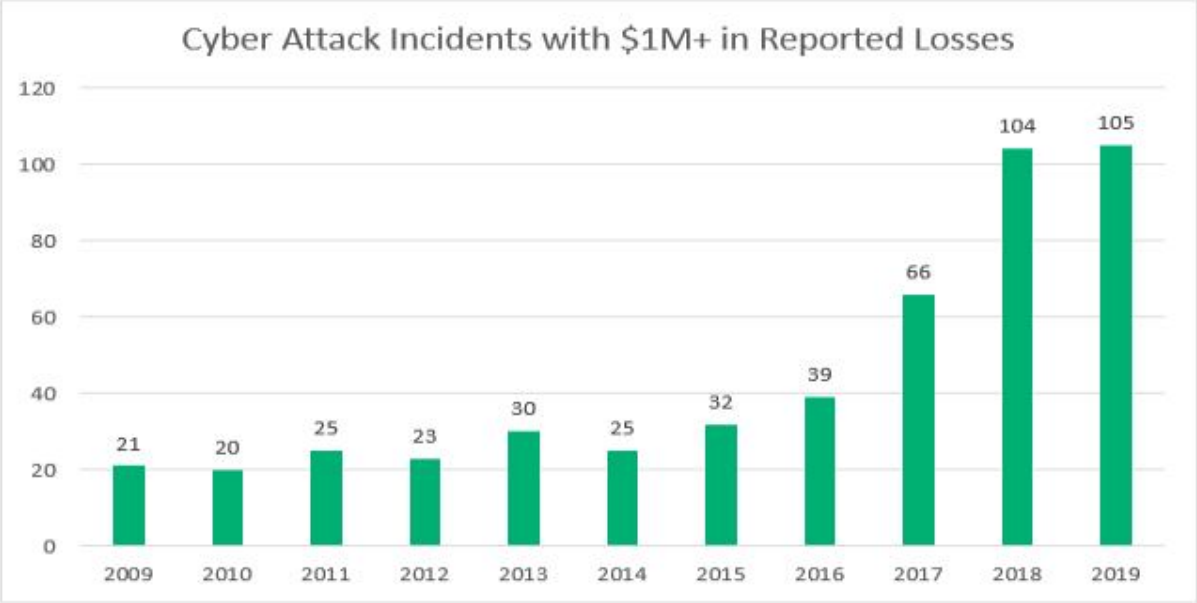


Рисунок 1. Кількість атак в мережі Інтернет

УДК 004.031.6

Л. Матійчук, кан. екон. наук, доцент, І. Павлов, В. Сташук
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

ТЕОРЕТИЧНЕ ОБҐРУНТУВАННЯ МЕТОДУ ВИЯВЛЕННЯ КОМП'ЮТЕРНИХ АТАК

UDC 004.031.6

L. Matiychuk, PhD, Assoc. Prof., I. Pavlov, V. Stashuk

THEORETICAL JUSTIFICATION OF THE METHOD OF DETECTION OF COMPUTER ATTACKS

Мета виявлення вторгнень на перший погляд дуже проста: виявити проникнення в ІС. Проте це вельми складне завдання. Насправді, системи виявлення вторгнень ніяких вторгнень взагалі не виявляють вони тільки виявляють ознаки вторгнень під час таких атак.
Системи виявлення атак призначені для виявлення і протидії мережевим атакам зловмисників. Вони є спеціалізованим програмно-апаратним забезпеченням з типовою архітектурою, що включає наступні компоненти (рис.1): модулі-датчики для збору необхідної інформації про МТ в ІС; модуль виявлення атак, що виконує обробку даних, зібраних датчиками, з метою виявлення інформаційних атак; модуль реагування на виявлені атаки; модуль зберігання конфігураційної інформації, а також інформації про виявлені атаки. Таким модулем, як правило, виступає стандартна СУБД, наприклад MS SQL Server; модуль управління компонентами системи виявлення атак.

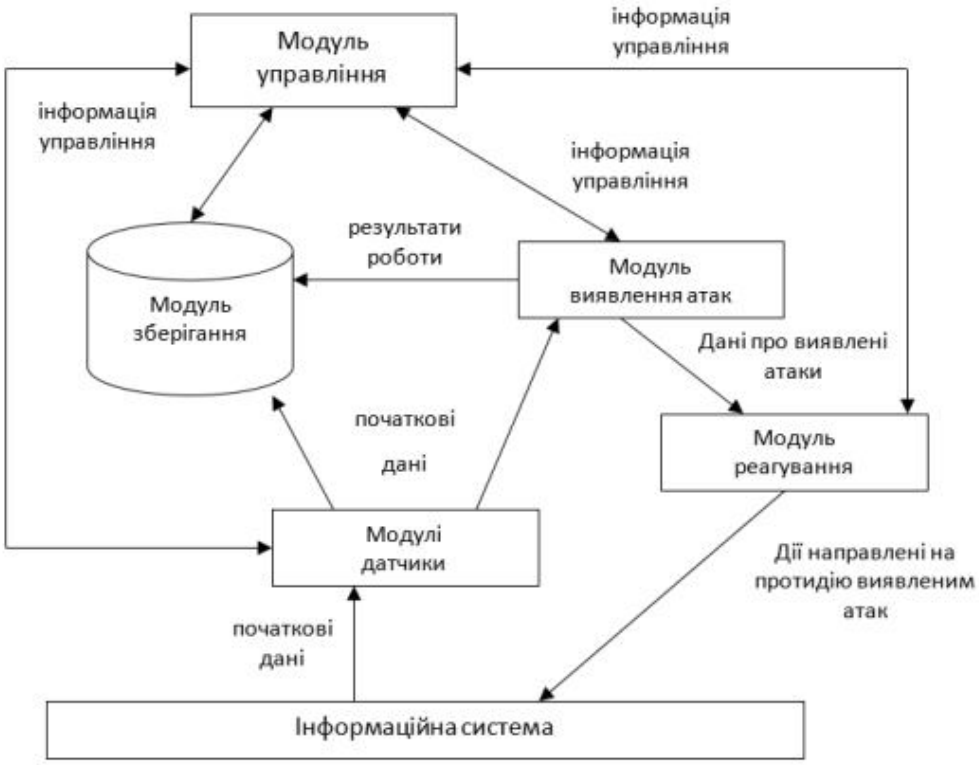


Рисунок 1. Типова архітектура виявлення атак

Для точного виявлення вторгнень необхідні надійні і вичерпні дані про те, що відбувається в системі, яка захищається. Взлом системи можливий як із сторини комп'ютера, що знаходиться в локальній мережі так і через глобальну мережу Інтернет. Проте сучасні атаки (DDOS-атаки –

distributed denial-of-service) для здійснення взлому системи можуть використовувати і проміжні комп'ютери, які прийнято називати зомбі (рис.2).

Такі системи у мережі Інтернет є незахищені або мало захищені. Зловмисник взломавши їх, бере під свій контроль і при цьому інсталює відповідне програмне забезпечення на кожному з них. Такі компютери після того стають підвладні йому.

Виходячи із відомих методів виявлення атак розглянутих у попередньому розділі, найкращим методом для вирішення задачі ідентифікації атак є застосування СМ на базі нейронних мереж. Вони описують кожну атаку у вигляді спеціальної моделі або сигнатури. Як сигнатура атаки можуть виступати: рядок символів, семантичний вираз на спеціальній мові, формальна математична модель. Алгоритм роботи СМ полягає в пошуку сигнатури атак в початкових даних, зібраних мережевими і хостовими датчиками системи. У разі виявлення шуканої сигнатури, система фіксує факт інформаційної атаки, яка відповідає знайденій сигнатурі.

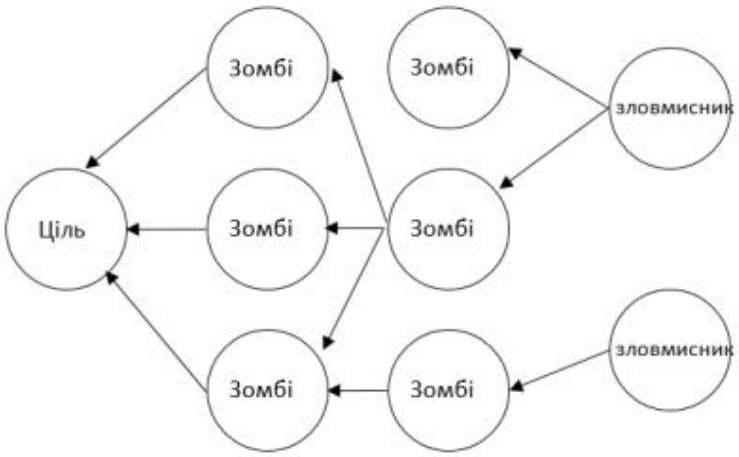
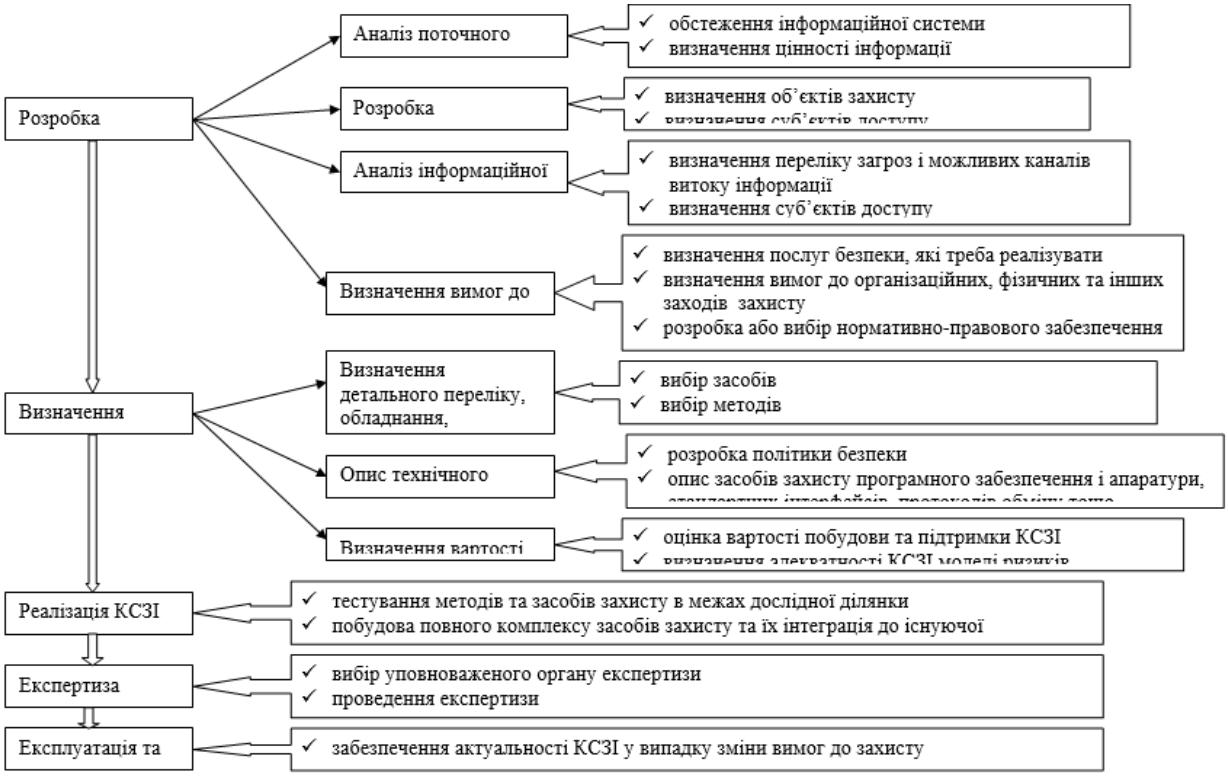


Рисунок 2. Здійснення DDOS-атаки

Додаток Б

Алгоритм побудови Комп'ютерних Систем Захисту Інформації (КСЗІ)



Додаток В

Програмний модуль MD5

```
#include "global.h"
#include "md5.h"
/* Constants for MD5Transform. */
#define S11 7
#define S12 12
#define S13 17
#define S14 22
#define S21 5
#define S22 9
#define S23 14
#define S24 20
#define S31 4
#define S32 11
#define S33 16
#define S34 23
#define S41 6
#define S42 10
#define S43 15
#define S44 21
static void MD5Transform (UINT4 state[4], unsigned char block[64]);
static void MD5_memcpy (POINTER output, POINTER input,unsigned int len);
static void Encode (unsigned char *output, UINT4 *input,unsigned int len);
static void Decode (UINT4 *output, unsigned char *input, unsigned int len);
static void MD5_memset (POINTER output,int value,unsigned int len);
static unsigned char PADDING[64] = {
  0x80, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
};
/* F, G, H and I are basic MD5 functions. */
#define F(x, y, z) (((x) & (y)) | ((~x) & (z)))
#define G(x, y, z) (((x) & (z)) | ((y) & (~z)))
#define H(x, y, z) ((x) ^ (y) ^ (z))
#define I(x, y, z) ((y) ^ ((x) | (~z)))
/* ROTATE_LEFT rotates x left n bits. */
#define ROTATE_LEFT(x, n) (((x) << (n)) | ((x) >> (32-(n))))
/* FF, GG, HH, and II transformations for rounds 1, 2, 3, and 4.
Rotation is separate from addition to prevent recomputation. */
#define FF(a, b, c, d, x, s, ac) { \
 (a) += F ((b), (c), (d)) + (x) + (UINT4)(ac); \
 (a) = ROTATE_LEFT ((a), (s)); \
 (a) += (b); \
  }
#define GG(a, b, c, d, x, s, ac) { \
 (a) += G ((b), (c), (d)) + (x) + (UINT4)(ac); \
 (a) = ROTATE_LEFT ((a), (s)); \
 (a) += (b); \
  }
#define HH(a, b, c, d, x, s, ac) { \
 (a) += H ((b), (c), (d)) + (x) + (UINT4)(ac); \
 (a) = ROTATE_LEFT ((a), (s)); \
 (a) += (b); \
  }
#define II(a, b, c, d, x, s, ac) { \
 (a) += I ((b), (c), (d)) + (x) + (UINT4)(ac); \
 (a) = ROTATE_LEFT ((a), (s)); \
 (a) += (b); \
  }
/* MD5 initialization. Begins an MD5 operation, writing a new context. */
extern "C" void __stdcall MD5Init (MD5_CTX *context)
```

```
                {
                        context->count[0] = context->count[1] = 0;
                        /* Load magic initialization constants.*/
                        context->state[0] = 0x67452301;
                        context->state[1] = 0xefcdab89;
                        context->state[2] = 0x98badcfe;
                        context->state[3] = 0x10325476;
                }
/* MD5 block update operation. Continues an MD5 message-digest
  operation, processing another message block, and updating the
  context. */
extern "C" void __stdcall MD5Update (MD5_CTX *context,unsigned char *input,unsigned int inputLen)
{
              unsigned int i, index, partLen;
             /* Compute number of bytes mod 64 */
             index = (unsigned int)((context->count[0] >> 3) & 0x3F);
             /* Update number of bits */
             if ((context->count[0] += ((UINT4)inputLen << 3))
              < ((UINT4)inputLen << 3))
            context->count[1]++;
             context->count[1] += ((UINT4)inputLen >> 29);
             partLen = 64 - index;
             /* Transform as many times as possible.*/
             if (inputLen >= partLen) {
            MD5_memcpy
              ((POINTER)&context->buffer[index], (POINTER)input, partLen);
            MD5Transform (context->state, context->buffer);
            for (i = partLen; i + 63 < inputLen; i += 64)
              MD5Transform (context->state, &input[i]);
           index = 0;
  }
              else
            i = 0;
             /* Buffer remaining input */
             MD5_memcpy
            ((POINTER)&context->buffer[index], (POINTER)&input[i],
             inputLen-i);
}
/* MD5 finalization. Ends an MD5 message-digest operation, writing the
  the message digest and zeroizing the context. */
extern "C" void __stdcall MD5Final (unsigned char digest[16],MD5_CTX *context)
{
  unsigned char bits[8];
  unsigned int index, padLen;
  /* Save number of bits */
  Encode (bits, context->count, 8);
  /* Pad out to 56 mod 64.*/
  index = (unsigned int)((context->count[0] >> 3) & 0x3f);
  padLen = (index < 56) ? (56 - index) : (120 - index);
  MD5Update (context, PADDING, padLen);
  /* Append length (before padding) */
  MD5Update (context, bits, 8);
  /* Store state in digest */
  Encode (digest, context->state, 16);
  /* Zeroize sensitive information.*/
  MD5_memset ((POINTER)context, 0, sizeof (*context));
}
/* MD5 basic transformation. Transforms state based on block. */
static void MD5Transform (UINT4 state[4], unsigned char block[64])
{
  UINT4 a = state[0], b = state[1], c = state[2], d = state[3], x[16];
 Decode (x, block, 64);
  /* Round 1 */
  FF (a, b, c, d, x[ 0], S11, 0xd76aa478); /* 1 */
```

```
 FF (d, a, b, c, x[ 1], S12, 0xe8c7b756); /* 2 */
 FF (c, d, a, b, x[ 2], S13, 0x242070db); /* 3 */
 FF (b, c, d, a, x[ 3], S14, 0xc1bdceee); /* 4 */
 FF (a, b, c, d, x[ 4], S11, 0xf57c0faf); /* 5 */
 FF (d, a, b, c, x[ 5], S12, 0x4787c62a); /* 6 */
 FF (c, d, a, b, x[ 6], S13, 0xa8304613); /* 7 */
 FF (b, c, d, a, x[ 7], S14, 0xfd469501); /* 8 */
 FF (a, b, c, d, x[ 8], S11, 0x698098d8); /* 9 */
 FF (d, a, b, c, x[ 9], S12, 0x8b44f7af); /* 10 */
 FF (c, d, a, b, x[10], S13, 0xffff5bb1); /* 11 */
 FF (b, c, d, a, x[11], S14, 0x895cd7be); /* 12 */
 FF (a, b, c, d, x[12], S11, 0x6b901122); /* 13 */
 FF (d, a, b, c, x[13], S12, 0xfd987193); /* 14 */
 FF (c, d, a, b, x[14], S13, 0xa679438e); /* 15 */
 FF (b, c, d, a, x[15], S14, 0x49b40821); /* 16 */
/* Round 2 */
 GG (a, b, c, d, x[ 1], S21, 0xf61e2562); /* 17 */
 GG (d, a, b, c, x[ 6], S22, 0xc040b340); /* 18 */
 GG (c, d, a, b, x[11], S23, 0x265e5a51); /* 19 */
 GG (b, c, d, a, x[ 0], S24, 0xe9b6c7aa); /* 20 */
 GG (a, b, c, d, x[ 5], S21, 0xd62f105d); /* 21 */
 GG (d, a, b, c, x[10], S22,  0x2441453); /* 22 */
 GG (c, d, a, b, x[15], S23, 0xd8a1e681); /* 23 */
 GG (b, c, d, a, x[ 4], S24, 0xe7d3fbc8); /* 24 */
 GG (a, b, c, d, x[ 9], S21, 0x21e1cde6); /* 25 */
 GG (d, a, b, c, x[14], S22, 0xc33707d6); /* 26 */
 GG (c, d, a, b, x[ 3], S23, 0xf4d50d87); /* 27 */
 GG (b, c, d, a, x[ 8], S24, 0x455a14ed); /* 28 */
 GG (a, b, c, d, x[13], S21, 0xa9e3e905); /* 29 */
 GG (d, a, b, c, x[ 2], S22, 0xfcefa3f8); /* 30 */
 GG (c, d, a, b, x[ 7], S23, 0x676f02d9); /* 31 */
 GG (b, c, d, a, x[12], S24, 0x8d2a4c8a); /* 32 */
 /* Round 3 */
 HH (a, b, c, d, x[ 5], S31, 0xfffa3942); /* 33 */
 HH (d, a, b, c, x[ 8], S32, 0x8771f681); /* 34 */
 HH (c, d, a, b, x[11], S33, 0x6d9d6122); /* 35 */
 HH (b, c, d, a, x[14], S34, 0xfde5380c); /* 36 */
 HH (a, b, c, d, x[ 1], S31, 0xa4beea44); /* 37 */
 HH (d, a, b, c, x[ 4], S32, 0x4bdecfa9); /* 38 */
 HH (c, d, a, b, x[ 7], S33, 0xf6bb4b60); /* 39 */
 HH (b, c, d, a, x[10], S34, 0xbebfbc70); /* 40 */
 HH (a, b, c, d, x[13], S31, 0x289b7ec6); /* 41 */
 HH (d, a, b, c, x[ 0], S32, 0xeaa127fa); /* 42 */
 HH (c, d, a, b, x[ 3], S33, 0xd4ef3085); /* 43 */
 HH (b, c, d, a, x[ 6], S34,  0x4881d05); /* 44 */
 HH (a, b, c, d, x[ 9], S31, 0xd9d4d039); /* 45 */
 HH (d, a, b, c, x[12], S32, 0xe6db99e5); /* 46 */
 HH (c, d, a, b, x[15], S33, 0x1fa27cf8); /* 47 */
 HH (b, c, d, a, x[ 2], S34, 0xc4ac5665); /* 48 */
 /* Round 4 */
 II (a, b, c, d, x[ 0], S41, 0xf4292244); /* 49 */
 II (d, a, b, c, x[ 7], S42, 0x432aff97); /* 50 */
 II (c, d, a, b, x[14], S43, 0xab9423a7); /* 51 */
 II (b, c, d, a, x[ 5], S44, 0xfc93a039); /* 52 */
 II (a, b, c, d, x[12], S41, 0x655b59c3); /* 53 */
 II (d, a, b, c, x[ 3], S42, 0x8f0ccc92); /* 54 */
 II (c, d, a, b, x[10], S43, 0xffeff47d); /* 55 */
 II (b, c, d, a, x[ 1], S44, 0x85845dd1); /* 56 */
 II (a, b, c, d, x[ 8], S41, 0x6fa87e4f); /* 57 */
 II (d, a, b, c, x[15], S42, 0xfe2ce6e0); /* 58 */
 II (c, d, a, b, x[ 6], S43, 0xa3014314); /* 59 */
 II (b, c, d, a, x[13], S44, 0x4e0811a1); /* 60 */
 II (a, b, c, d, x[ 4], S41, 0xf7537e82); /* 61 */
 II (d, a, b, c, x[11], S42, 0xbd3af235); /* 62 */
```

```
  II (c, d, a, b, x[ 2], S43, 0x2ad7d2bb); /* 63 */
  II (b, c, d, a, x[ 9], S44, 0xeb86d391); /* 64 */
  state[0] += a;
  state[1] += b;
  state[2] += c;
  state[3] += d;
  /* Zeroize sensitive information.*/
  MD5_memset ((POINTER)x, 0, sizeof (x));
}
/* Encodes input (UINT4) into output (unsigned char). Assumes len is
  a multiple of 4. */
static void Encode (unsigned char *output, UINT4 *input,unsigned int len)
{
              unsigned int i, j;
              for (i = 0, j = 0; j < len; i++, j += 4) {
             output[j] = (unsigned char)(input[i] & 0xff);
             output[j+1] = (unsigned char)((input[i] >> 8) & 0xff);
             output[j+2] = (unsigned char)((input[i] >> 16) & 0xff);
             output[j+3] = (unsigned char)((input[i] >> 24) & 0xff);
  }
}
/* Decodes input (unsigned char) into output (UINT4). Assumes len is
  a multiple of 4. */
static void Decode (UINT4 *output, unsigned char *input, unsigned int len)
{
              unsigned int i, j;
              for (i = 0, j = 0; j < len; i++, j += 4)
             output[i] = ((UINT4)input[j]) | (((UINT4)input[j+1]) << 8) |
               (((UINT4)input[j+2]) << 16) | (((UINT4)input[j+3]) << 24);
}
/* Note: Replace "for loop" with standard memcpy if possible. */
static void MD5_memcpy (POINTER output, POINTER input,unsigned int len)
{
              unsigned int i;
              for (i = 0; i < len; i++)
             output[i] = input[i];
}
/* Note: Replace "for loop" with standard memset if possible. */
static void MD5_memset (POINTER output,int value,unsigned int len)
{
              unsigned int i;
              for (i = 0; i < len; i++)
             ((char *)output)[i] = (char)value;
}
```

Додаток Г

Програмний код здійснення cgi-атаки

```
#!c:/Perl/bin/perl
print "Content-type: text/html\n\n";
use CGI qw(param);
$user=param("user");
$text=param("text");
$file=param("file");
if (($user ne " ")||($text ne " ")){
open (Config, "$file");
print Config "$user :: $text \n";
close Config;
}
print "<html>";
print "<head>";
print "<body>";

#########################################################################
print '<table BORDER=0 CELLPADDING=3 CELLSPACING=1 WIDTH=600 BGCOLOR="#000000"
ALIGN="CENTER">';
open(TEMP, "baza.dat");
while (<TEMP>)
{




chop;
($user, $text) = split(/::/);

     if (($user ne " ")&&($text ne " "))  {push (@point, "<tr VALIGN=baseline BGCOLOR=#CCCCCC><td
BGCOLOR=#32CD32>user:$user</td><td>question:$text</td></tr>\n")};

}
print @point;
#########################################################################
print "<form action=/cgi-bin/21.pl>";
print "<tr BGCOLOR=#32CD32 align=center><td colspan=2><b>Как ваше имя?</b></td></tr>";
print "<tr BGCOLOR=#32CD32 align=center><td colspan=2><input type=text name=user size=20></td></tr>";
print "<tr BGCOLOR=#32CD32 align=center><td colspan=2><b>Введите ваш отзыв:</b></td></tr>";
print "<tr BGCOLOR=#32CD32 align=center><td colspan=2><textarea name=text rows=5
cols=20></textarea></td></tr>";
print "<input type=hidden name=file ",'value=">> baza.dat" >';
```

```
print "<tr BGCOLOR=#32CD32 align=center><td colspan=2><input type=submit></tr>";
print "</form>";
print "</table>";

print "</body>";
print "</html>";
```

Додаток Д

Програмний модуль створення та навчання НМ

```
function status = createFFNN ()

%CREATEFFNN create a feed-forward neural network
%   STATUS = CREATEFFNN()
%
%    Structure:
%     - 20 neurons in input layer
%     - user selected neurons in hidden layer (10 by default)
%     - 2 neuron in output layer
%
%   The network input is a 20x32 window of the picture in vector form.
%
%   The output parameter STATUS is a logical 1 if the
%   network was created succesfully, and 0 if an error occurs.
%
%   ------------------------------
%   Module identification of computer attacks

status = false;

if exist('net.mat','file')
    button = questdlg(...
        'The network is already initialized. Would you like to create new
one?',...
        '','Yes','No','Yes');
    if strcmp(button,'No')
        status = true;
        return;
    end
end

fprintf ('Creating a feed-forward backpropagation network ...\n');

while 1
    answer = inputdlg({'Training algorithm',...
                       'Sum-squared error goal',...
                       'Max. number of epoches to train Neural Network',...
                       'Number of neural networks',...
                       'Number of hidden neurons'},...
                      'Configure Feed-Forward Neural Network Parameters',...
                      1,...
                      {'traingdx',...
                       '1e-5',...
                       '10000',...
                       '1',...
                       '25'});
    if isempty(answer)
        button = questdlg('Do you realy want to
exit?','Exit','Yes','No','Yes');
        if strcmp(button,'Yes')
            return;
        end
    else
        TrainAlg = answer{1};
        TrainAccuracy = str2num(answer{2});
        TrainEpochs = str2num(answer{3});
        NumbNNets = str2num(answer{4});
        NumbHidNeurons = str2num(answer{5});
        break;
    end
end
```

```
if exist('trainSet.mat','file')
    load trainSet;
else
    if loadTrainSet
        load trainSet;
    else
        fprintf ('Error on loading train set of images\n');
        return;
    end
end

net = {NumbNNets};
for nn = 1:NumbNNets
    [TrainSet, mintrain, maxtrain] = premnmx(double(TrainSet));

    net{nn} = newff(minmax(TrainSet), [NumbHidNeurons 1], {'tansig' 'tansig'},
TrainAlg);

    net{nn}.performFcn = 'mse';
    net{nn}.trainParam.goal = TrainAccuracy;
    net{nn}.trainParam.show = 10;
    net{nn}.trainParam.epochs = TrainEpochs;
    net{nn}.adaptFcn = 'trainr';

end

fprintf ('Initializing complete!\n\n');

save net net;
status = true;
```

```
function status = trainNet ()

%TRAINNET trains a feed-forward neural network
%   STATUS = TRAINNET()
%
%   TRAIN trains a network NET according to NET.trainFcn and
%   NET.trainParam.Structure:
%
%   The output parameter STATUS is a logical 2 if the
%   network was trained succesfully, and 0 if an error occurs.
%
%   ------------------------------
%   Module identification of computer attacks

status = false;

% Load neural network, if exists. If not, create the new one
if exist('net.mat','file')
load net;
else
button = questdlg(...
'The network is not initialized yet. Would you like to initialize it now?',...
'','Yes','No','Yes');
if strcmp(button,'Yes')
if createFFNN
load net;
else
fprintf ('Error on initializing neural network\n');
```

```
return;
end
else
fprintf ('NOTE! The network was not trained.\nYou may train it later from
menu.\n');
return;
end
end
% ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

if exist('trainSet.mat','file')
load trainSet;
else
if loadTrainSet
load trainSet;
else
fprintf ('Error on loading train set of images\n');
return;
end
end

fprintf ('Training a feed-forward backpropagation network ...\n');

for nn = 1:size(net)
[TrainSet, mintrain, maxtrain] = premnmx(double(TrainSet));
net{nn} = train(net{nn}, TrainSet, PatternSet);

Result = sim(net{nn}, TrainSet);
figure;
plot([1:size(Result,2)],Result,...
[1:size(Result,2)],PatternSet,'ro');
end

fprintf ('Training complete!\n\n');

save net net;
                                status = true;
```

Додаток Е

Тестування програмного модуля

```
function status = testNet ()

%TESTNET tests a neural network on training images set
%   STATUS = TESTNET()
%
%   The output parameter STATUS is a logical 2 if the
%   network was tested succesfully, and 0 if an error occurs.
%
%   ------------------------------
%   Module identification of computer attacks

status = false;

% Load neural network, if exists. If not, create the new one
if exist('net.mat','file')
    load net;
else
    button = questdlg(...
        'The network is not initialized yet. Would you like to initialize it
now?',...
        '','Yes','No','Yes');
    if strcmp(button,'Yes')
        if createFFNN
            load net;
        else
            fprintf ('Error on initializing neural network\n');
            return;
        end
    else
        fprintf ('NOTE! The network was not trained.\nYou may train it later
from menu.\n');
        return;
    end
end
% ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

if exist('trainSet.mat','file')
    load trainSet;
else
    if loadTrainSet
        load trainSet;
    else
        fprintf ('Error on loading train set of images\n');
        return;
    end
end

fprintf ('Testing a feed-forward backpropagation network ...\n');

for nn = 1:size(net)
    [TrainSet, mintrain, maxtrain] = premnmx(double(TrainSet));
    Result = sim(net{nn}, TrainSet);
    figure;
    plot([1:size(Result,2)],Result,...
        [1:size(Result,2)],PatternSet,'ro');
end

fprintf ('Testing complete!\n\n');

status = true;
```