МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
КАФЕДРА КОМП'ЮТЕРИЗОВАНИХ СИСТЕМ ЗАХИСТУ ІНФОРМАЦІЇ

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
_____ С.В. Казмірчук
«_____»_____ 20__ р.

На правах рукопису
УДК 004.056.55(004.421.5)

ДИПЛОМНА РОБОТА

ЗДОБУВАЧА ВИЩОЇ ОСВІТИ
ОСВІТНЬОГО СТУПЕНЯ «БАКАЛАВР»

Тема: Програмний модуль виявлення атак на веб-додатки

Виконавець: Ю.В. Венедіктова
Керівник: к.т.н, доцент О.О. Висоцька
Нормоконтролер: к.т.н, доцент О.О. Висоцька

Київ 2021

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет: Кібербезпеки, комп'ютерної та програмної інженерії

Кафедра: Комп'ютеризованих систем захисту інформації

Освітній ступінь: Бакалавр

Спеціальність: 125 «Кібербезпека»

Освітньо-професійна програма: «Безпека інформаційних і комунікаційних систем»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ С.В. Казмірчук

«__» _____ 20__ р.

ЗАВДАННЯ

на виконання дипломної роботи

здобувача вищої освіти Венедіктової Юлії Віталіївни

1. Тема: *Програмний модуль виявлення атак на веб додатки*
затверджена наказом ректора від «26» квітня 2021 р. № 652/ст.

2. Термін виконання: з 10.05.2021 р. по 20.06.2021 р.

3. Вихідні дані: проаналізувати існуючі методи та методики аналізу і оцінки атак на веб-сторінки; на основі аналізу виділити вхідні і вихідні параметри, завдяки яким можливо здійснити виявлення атак на веб-додатки, виявлення їх переваг і недоліків; розробити програмний модуль виявлення атак на веб додатки.

4. Зміст пояснювальної записки: аналіз існуючих систем та методик аналізу і оцінки атак на веб-сторінки; розробка методу захисту Веб-сторінок; розробка програмного модуля, верифікація отриманих результатів.

КАЛЕНДАРНИЙ ПЛАН

виконання дипломної роботи

№ п/п	Етапи виконання дипломної роботи	Термін виконання етапів	Примітка
1.	Уточнення постановки задачі	19.04.2021	*Виконано*
2.	Аналіз літературних джерел	20.04-01.05.2021	*Виконано*
3.	Обґрунтування вибору рішення	02-03.05.2021	*Виконано*
4.	Збір інформації	03-08.05.2021	*Виконано*
5.	Дослідження предметної області	09-12.05.2021	*Виконано*
6.	Дослідження сучасних атак на веб-додатки, а також методів їх виявлення	13-20.05.2021	*Виконано*
7.	Розробка алгоритму та програмного модуля для виявлення атак на веб-додатки	21-30.05.2021	*Виконано*
8.	Оформлення і друк пояснювальної записки	30.05-03.06.2021	*Виконано*
9.	Перевірка на антиплагіат	04.05.2021	*Виконано*
10.	Оформлення презентації	05-07.06.2021	*Виконано*
11.	Отримання рецензій від рецензента	08-11.06.2021	*Виконано*

Здобувач вищої освіти Ю. Венедіктова

(підпис, дата)

Керівник дипломної роботи О. Висоцька

(підпис, дата)

РЕФЕРАТ

Дипломна робота складається зі вступу, трьох розділів, загальних висновків, списку використаних джерел, додатків і має 62 сторінки основного тексту, 34 рисунка. Список використаних джерел містить 22 найменування. Загальний обсяг роботи 83 сторінки.

Метою роботи є розробка програмного модуля виявлення атак на веб-додатки.

Для **досягнення поставленої мети** необхідно розв’язати наступні задачі: дослідити предметну область, що включає в себе аналіз видів атак на веб-додатки, дослідження проблем виявлення та захисту веб-додатків, розгляд стандартних засобів захисту веб-додатків, а також аналіз програмних засобів для виявлення атак на веб-додатки; дослідити існуючі методи захисту веб-додатків; розробити метод виявлення атак на веб-додатки, що є комбінацією двох вже відомих методів; розробити програмний модуль, який реалізує запропонований метод.

Об'єктом дослідження є процес виявлення атак на веб-додатки.

Предметом дослідження є методи та системи для виявлення атак на веб-додатки, сканери вразливостей веб-сайтів.

Методи дослідження – аналіз та синтез.

Практична цінність – програмний модуль, який дозволяє виявляти атаки на веб-додатки, який реалізує комбінований метод двох методів dropout та backpropagation, та може бути використаний для підвищення рівня захищеності веб-додатків.

Актуальність дослідження вразливостей та їх виявлення в веб-додатках в рамках безпеки інформації обумовлена тим, що веб-технології, з одного боку, активно використовуються при реалізації сучасних інформаційних систем, в тому числі критичних з точки зору інформаційної безпеки, а, з іншого боку, проведення базових атак на подібні інформаційні системи не вимагають від порушників високої технічної компетентності, оскільки дані про типові

вразливості і атаках, включаючи інструментальні засоби проведення атак, в великому обсязі представлені в загальнодоступних джерелах інформації, а самі інформаційні системи, як правило, доступні з мереж зв'язку загального користування.

ВЕБ-ДОДАТОК, АТАКА, НЕЙРОННА МЕРЕЖА, ІНФОРМАЦІЙНА БЕЗПЕКА, ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНА СИСТЕМА, ЗАГРОЗА, СТУПІНЬ РИЗИКУ.

ЗМІСТ

ПЕРЕЛІК ПРИЙНЯТИХ СКОРОЧЕНЬ

ІС - інформаційна система

ПЗ - програмне забезпечення

СВА - система виявлення атак

DoS - Denial of Service «відмова в обслуговуванні»

IDS - система виявлення вторгнень

ІБ - інформаційна безпека

IPS - система запобігання вторгненням

NIDS - Network Intrusrnn Detection Systems

GrIDS - Graph-Based Intrusion Detection System

OIDS - Operational Intrusion Detection Systems

HIDS - Host-based Intrusion Detection System

НСД - несанкціонований доступ

ОС - операційна система

ВСТУП

Стрімкий розвиток інформаційних систем (ІС) та технологій всебічно впливає на всі сфери діяльності суспільства. Значна кількість сучасних державних та приватних підприємств використовує ІС для управління виробничими процесами, підтримки прийняття рішень, пошуку необхідних даних тощо. Разом з цим збільшується кількість уразливостей та загроз ІС і тому для забезпечення їх нормального функціонування та попередження вторгнень необхідні спеціалізовані засоби безпеки. Слід зазначити, що одним із актуальних напрямів, який активно розвивається у сфері інформаційної безпеки є виявлення кібератак і запобігання вторгнень в ІС з боку неавторизованої сторони (НАС). Наприклад, низка нещодавно реалізованих кібератак, які завдали шкоди багатьом державним установам та приватним підприємствам і організаціям (Ощадбанк, Укргазбанк, Укрпошта, Укрзалізниця, Укренерго, ДТЕК, Київенерго, Київводоканал, Міжнародні аеропорти «Бориспіль» і «Київ», Rozetka, Київстар, Vodafone Україна, Lifecell, Київський метрополітен, телеканали СТБ і ICTV, Нова пошта, мережа магазинів «Епіцентр», автозаправки WOG і ТНК тощо) показали неготовність та недосконаліть їх власних систем безпеки до раніше невідомих вторгнень. Масові кібератаки ініціюють створення спеціальних технічних рішень, засобів та систем протидії. Для виявлення мережевих вторгнень використовуються сучасні методи, моделі, засоби, програмне забезпечення (ПЗ) і комплексні технічні рішення для систем виявлення та запобігання вторгнень, які можуть залишатись ефективними при появі нових або модифікованих видів кіберзагроз. Але на практиці при появі нових загроз та аномалій, породжених атакуючими діями з невстановленими або нечітко визначеними властивостями, зазначені засоби не завжди залишаються ефективними і вимагають тривалих часових ресурсів для їх відповідної адаптації. Тому системи виявлення вторгнень повинні постійно 10

досліджуватись і удосконалюватись для забезпечення неперервності в їх ефективному функціонуванні. Серед таких систем є спеціалізовані програмні засоби, які направлені на виявлення підозрілої активності або втручання в інформаційну систему і прийняття адекватних заходів щодо запобігання кібератакам. До таких засобів відноситься: міжмережеві екрани, антивірусні системи, системи виявлення та запобігання вторгнень.

Оскільки на сьогодні веб-додатки є найбільш використовуваними програмними засобами, які використовує людина, то виявлення атак саме на них є найбільш актуальною задачею. Саме тому для дипломної роботи і була обрана відповідна тематика.

РОЗДІЛ 1
АТАКИ НА ВЕБ-ДОДАТКИ

1.1. Аналіз предметної області та постановка задачі

Проблемам захисту веб-ресурсів присвячене широке коло досліджень. Наприклад, [1, 2], які повністю присвячені опису методів та інструментів атак і захисту від них. Видані з інтервалом майже в 10 років, вони наочно ілюструють зміни в підходах до захисту веб-ресурсів. Якщо в [1] стверджується про можливість забезпечення захисту від будь-яких атак на веб-ресурси, то в [2] розглянуто конкретні методи для захисту від атак. Така зміна орієнтування викладу є дуже симптоматичною та є наслідком тієї обставини, що методи та інструменти атак досить важко піддаються класифікації, а сама атака часто використовує технології маскування цих методів та інструментів.

У [3] описано метод ідентифікації атак типу «відмова в обслуговуванні», оснований на застосуванні багатошарового персептрону, що дозволило отримати необхідну множину показників.

Розв’язання задачі детектування DDoS-атак на основі розробки спеціальної метрики є предметом статті [4]. В роботі [5] проаналізовано існуючі методи захисту від DDoS-атак і запропоновано новий метод, який базується на статистичному аналізі вхідного трафіка на сервері та надійній системі перевірки гіпотез.

Розробляють також комбіновані методи захисту веб-ресурсів, основані на використанні евристичного підходу [6], в рамках якого виділяється аномальна поведінка споживача, що підвищує ймовірність захисту порівняно із сигнатурним аналізом.

Перспективним також є використання моделей агента загроз для захисту веб-ресурсів від атак [7], що дозволяє формалізувати пошук вразливостей в інформаційних системах на всіх етапах взаємодії агента загроз із веб-

ресурсом.

Розгляд Cross-site scripting атак на Android WebView здійснено в [8], де розроблено метод моніторингу доступу для браузера з метою протидії.

Проблеми витоку інформації проаналізовано в [9, 10], де розглянуто як типові сценарії, так і методи та способи захисту від них.

У [11] відмічено, що злам паролю залежить від наявних обчислювальних ресурсів, часу, функції, що використовується для зберігання цього пароля, а також від багатьох інших характеристик. Запропоновано загальні рамки для оцінки складності пароля й оцінки його надійності.

Метою даної роботи є удосконалення захисту веб-ресурсів від атак, яке базується на використанні комбінованих методів захисту.

1.2. Аналіз актуальних атак на веб-ресурси

Більшість Web-ресурсів не відповідають сучасним вимогам безпеки. Вразливості Web-ресурсів можуть поставити під загрозу імідж, фінанси, активи, персональні дані та інші цінні ресурси організацій, а, отже, як підсумок можливе банкрутство або повна ліквідація компанії.

У цьому пункті наведені найпопулярніші атаки 2019 року [1]. Серед найпоширеніших атак на веб-додатки можна виділити наступні: «Міжсайтове виконання сценаріїв» (39.1% використання), «Впровадження операторів SQL» (24.9% використання), «Вихід за межі призначеної директорії» (6.6% використання відповідно) та ін.

1) Впровадження операторів SQL

SQL-ін'єкція [2, 3] більше не є новою концепцією, але все ж є одним з найбільш поширених типів мережевих атак. SQL-ін'єкція – це метод, який використовує уразливість в запитах для отримання даних небезпечних веб-сайтів в Інтернеті, що є дуже популярним методом атаки. Його успіх також відносно високий. SQL-ін'єкція (для стислості називається SQLi)

організована шляхом відправки шкідливих команд SQL на сервери БД за допомогою запитів, дозволених сайтом, таких, як команди входу в систему.

Будь-які вхідні дані сайту організацій (теги введення, рядки запиту, файли cookie тощо). Можуть бути використані для відправки шкідливого коду.

Існують поширені типи помилок SQL-ін'єкцій:

• Якщо неправильно поводитися: Помилки впровадження SQL такого типу зазвичай виникають через те, що програміст або користувач неясно визначає введення даних або не виконує етап перевірки і фільтрації типу вхідних даних. Це може статися, коли числове поле використовується в запиті SQL, але у програміста відсутні помилки під час введення для перевірки типів даних, які користувач вводить як число.

• Помилка конфігурації СУБД на сервері: іноді уразливості можуть існувати в ПЗ БД сервера, як у випадку з функцією mysql_real_escape_string () серверів MySQL. Це дозволить зловмисникові виконати успішну атаку SQL-ін'єкцією на основі незвичайних символів Юнікоду, навіть коли введення завершується.

• Зміна значення умови запиту: Цей тип помилки дозволяє зловмисникові змінити значення умови в запиті, що спотворює відображення додатки, що містить цю помилку.

• Час запізнювання: Цей тип помилки впровадження SQL існує, коли час обробки одного або декількох запитів SQL залежить від введених логічних даних або процес обробки запитів механізму SQL займає багато часу. Зловмисники можуть використовувати цей тип помилки SQL-ін'єкції, щоб визначити точний час завантаження сторінки, коли введене значення вірне. Помилки SQL-ін'єкцій відбуваються через небезпечного програмування, тому кращим рішенням є те, що програмістам потрібно бути обережними при розробці веб-додатків. Для помилок впровадження SQL фахівець може використовувати метод, застосовуючи Prepare Statement для

виправлення, тоді вхідні дані від користувача не будуть виконуватися в запиті. Для кожного конкретного мови та бази даних будуть різні способи застосування. Що стосується атаки SQL-ін'єкцією в реченні ORDER BY, оскільки місце розташування цієї пропозиції не може використовувати оператор Prepare, то для виправлення атаки необхідно використовувати дійсний метод білого списку.

2) Виконання команд ОС

Впровадження команд ОС (також зване впровадженням оболонки) - це вразливість в веб-сайтах, яка дозволяє зловмиснику виконувати довільні команди операційної системи (ОС) на сервері, на якому виконуються певні служби. Зловмисник може використовувати цю вразливість для використання, вилучення інформації, передачі атак на інші системи всередині організації.

Багато випадків впровадження команд ОС є сліпими уразливими. Це означає, що вихідні дані не будуть повернуті у відповіді HTTP. Тому результат не буде відображатися на екрані.

Затримки можуть бути використані для виявлення сліпих вразливостей. Це викличе затримку, що дозволяє адміністратору підтвердити, чи була команда виконана або немає, грунтуючись на часі, яке потрібно додатком для відповіді. Команда ping - ефективна команда для цього, так як вона дозволяє адміністратору системи вказати пакет ICMP для відправки і час, необхідний для виконання команди.

Уразливість цього типу з'являється в плагіні WordPress DZS-VideoGallery (CVE діє до: 2014 -9094), Gemitel 3.50 - Віддалене включення файлів / впровадження команд (CVE: 2004 - 1934) і т. Д.

Найефективніший спосіб запобігання небезпечних команд - це припинити використання команд. Тобто ніколи не викликати команди ОС на рівні додатків.

У деяких випадках існують різні способи виконання необхідних функцій з використанням API на більш безпечної платформі. Якщо не можна уникнути використання команд ОС, необхідно виконати сувору перевірку справжності введення:

• перевірка вхідних значень;

• необхідність прийому тільки даних у вигляді чисел;

• використання введення тільки даних з букв і цифр, без спеціальних символів, пробілів ...

в. Вихід за межі призначеної директорії

Метою атаки типу Path Traversal [4] є отримання доступу до файлів і каталогів, які розташовані поза межами, визначеними конфігурацією (web root folder).

Зловмисник часто використовує в своїх запитах послідовності типу «... /», щоб потрапити в кореневій каталог. Для всіх органів належний контроль доступу до контенту сайту є ключовим фактором в роботі захищеного сервера. Дорога назад в каталогах - це експлойт HTTP, який дозволяє зловмисникам отримувати доступ до обмежених каталогам, виконуючи команди поза кореневого каталогу веб-сервера.

Веб-сервери забезпечують два основних рівня механізму безпеки:

• список контролю доступу (ACL);

• коренева директорія.

Список контролю доступу використовується під час аутентифікації. Це список, який адміністратор сервера використовує для визначення того, які користувачі або групи користувачів можуть отримувати за добу, такі, як зміна або виконання певних файлів на сервері, а також інші дозволи.

Коренева директорія – це спеціальна директорія в файлової системі сервера, в якій доступ користувачів обмежений.

Його користувачі не можуть отримати доступ до будь-чого в цій директорії. Наприклад, коренева директорія IIS за замовчуванням в системі

Windows - C: \ Inetpub \ wwwroot, і з вбудованими правилами користувачі не можуть отримати доступ до C: \ Windows, але можуть отримати доступ до C: \ Inetpub \ wwwroot \ news і будь-якого файлу в цієї директорії (за умови, що користувач дійсний ACL).

По-перше, адміністратор системи встановлює веб-сервер з останньою версією програмного забезпечення. По-друге, адміністраторові треба встановити фільтр для будь-якого користувача введення.

3) Міжсайтове виконання сценаріїв

Міжсайтове виконання сценаріїв (Cross-site Scripting або XSS) [5] є поширеною вразливістю в веб-додатках. Щоб скористатися вразливістю XSS, зловмисник вводить шкідливий код через скрипти, щоб виконати їх на комп'ютерах користувачів. Як правило, атаки XSS використовуються для обходу контролю доступу та уособлення користувачів. Сам програмний код зазвичай пишеться на HTML / JavaScript, але може бути також перенесений в VBScript, ActiveX, Java, Flash, або на будь-яку іншу підтримувану браузерами технологію.

XSS-уразливість дозволяє впровадити в згенерований, а потім передаваючу користувачу сторінку формату HTML якийсь довільний код, інколи дуже шкідливий. Коли зловмисник домагається, щоб браузер користувача виконав його програмний код, цей код буде запущений в безпечному середовищі сервера веб-сайту. З цим рівнем привілеїв програмний код цілком здатний прочитати, змінити або передати важливі дані, доступні браузеру. Складність цієї атаки полягає в тому, що алгоритм фільтрації вхідних даних не повинен створювати необгрунтованих обмежень легальним користувачам, але в той же час повинен робити неможливою XSS атаку з боку зловмисника.

4) Відмова в обслуговуванні

«Відмова в обслуговуванні» (DDoS) [6] – один з популярних типів атак на веб-додатки. Кількість атак даного типу в третьому кварталі 2017 р

збільшилася на 8% в порівнянні з другим кварталом 2017 р DDoS-атака - це тип атаки, при якому зловмисник робить систему непридатною для використання або істотно сповільнює роботу системи для звичайних користувачів, перевантажуючи ресурси системи . Хоча атака DDoS не може отримати доступ до фактичних даних системи, але вона може порушити роботу служб, що надаються системою. При атаці на систему будуть використовуватися найслабші уразливості системи. DDoS-атаки можуть використовуватися для приховування інших мережевих атак. Коли веб-сайт органу перебуває під атакою, внутрішня і зовнішня група фахівців з інформаційної безпеки зазвичай фокусується на закритті портів цих сайтів, очищення трафіку і відновлення його роботи. Ці великі зусилля надають величезну можливість для інших небезпечних атак таких як SQL-ін'єкція.

5) Підключення локальних файлів

У мові програмуванні PHP має команди include, require, include_once, require_one, які дозволяють поточного файлу викликати інший файл, які є середовищем народження LFI уразливості. LFI-уразливості (Local File Inclusion) дозволяють зловмисникам підключати внутрішні файли на сервері, наприклад, файли: passwd, php.ini, access_log, ... (знати конфіденційну інформацію) в залежності від рівня безпеки сервера. Причиною цієї помилки є те, що при використанні вищевказаних команд програміст знову викликає файл для відкриття через змінну. Ці змінні або ще не ініціалізовані, або визначаються користувачем. Помилки LFI часто пов'язані з помилками завантаження. Зловмисник завантажує файл, який містить код php на сервері, не обов'язково тип файлу .php. Потім зловмисник використовує LFI-уразливість, щоб прочитати вміст завантаженого файлу. Коли сервер читає ці файли, при виявленні php-коду ці коди виконуються і, таким чином, реалізуються наміри зловмисника. Для захисту веб-додатків, написаних на мові програмування PHP, програмісту треба обмежити використання змінних

у функціях (include і require), якщо вони використовуються, то ці змінні повністю і правильно оголошені.

6) Впровадження зовнішніх сутностей XML

Як відомо, PHP є поширеним інтерпретується мовою загального призначення з відкритим вихідним кодом. Уразливість була виявлена у вбудованих класах PHP SoapClient і SoapServer. У PHP дозволені зовнішні сутності при обробці SOAP wsdl-документів, що дозволяє атакуючому читати довільні файли.

7) Завантаження довільних файлів

Рівень небезпеки цієї уразливості дуже високий. В системі "eXtreme File Hosting" вразливість дозволяє віддаленому користувачу виконати довільний PHP сценарій на цільовій системі. Уразливість існує изза помилки при обробці завантаження, що містять кілька розширень. Дана уразливість може привести до виконання довільного коду і / або відмови в обслуговуванні.

8) Підробка міжсайтових запитів

CSRF (Cross-Site Request Forgery) - це атака з використанням аутентіфіцированний облікових даних користувача на інший сайт. Веб-додатки працюють, отримуючи від користувача команди HTTP, а потім виконують їх. Зловмисники використовують метод CSRF, щоб обдурити браузер користувача для відправки команд http веб-додатків.

Якщо сеанс користувача не закінчився, команди зловмисника будуть виконуватися з правами аутентифікації користувача.

CSRF дуже рідко з'являється серед CVE (поширених вразливостей і небезпек) - менше 0.1% в 2008 році, але насправді це «сплячий гігант». CSRF залишається важливим питанням безпеки. Хоча існує величезна кількість атак на веб-додатки, відомо безліч способів виявлення атак для захисту Веб-додатки.

Дослідження атак на веб-додатки за 2017 і 2018 року показало, що зловмисники активно атакують веб-додатки, переслідуючи при цьому різні цілі: пряму крадіжку грошових коштів, отримання фінансової вигоди шляхом вимагання, проникнення у внутрішню інфраструктуру, політичні цілі, шпигунство і т. д. Будь-який веб-додаток, навіть який не є безпосередньою метою кіберзлочинців, може піддатися атаці. Тому проблема захисту веб-додатків організації від атак стане ще більш актуальною.

Висновки по розділу

В даному розділі описані основні поняття предметної області, а також наведені актуальні атаки на веб-ресурси, які використовуються сьогодні.

РОЗДІЛ 2
СУЧАСНІ МЕТОДИ ВИЯВЛЕННЯ АТАК НА ВЕБ-РЕСУРСИ

У багатьох системах виявлення атак (СВА) і міжмережевих екранах для веб-додатки частіше використовуються наступні методи: сигнатурні методи [7], методи виявлення аномалій [8], методи з використанням машинного навчання [9, 10].

2.1. Сигнатурні методи

Як і інші програми сканування вірусів на основі сигнатур, більшість СВА намагаються виявити атаки на бази даних за ознаками атаки. Коли зловмисник намагається використовувати відому уразливість, СВА намагається помістити її в свою базу даних. Наприклад, Snort, безкоштовний продукт на основі сигнатур, розроблений як для Unix, так і для Windows.

Оскільки це програмне забезпечення з відкритим вихідним кодом, Snort здатний розробляти базу даних сигнатур швидше, ніж будь-який інший механізм бази даних. Підпис Snort використовується у всьому продуктах інформаційної безпеки, від комерційного брандмауера до проміжного програмного забезпечення, такого як Hogwash.

Сигнатурні методи засновані на спеціальних структурах з тих, хто вивчає атак. Системний адміністратор визначить поля в атаці і на підставі цього напише правила примусового застосування в системі для реагування на подібну атаку.

2.2. Методи виявлення аномалій

Метод виявлення аномалій – виявлення аномалій, пов'язаних з встановленням базової основи нормальної роботи системи або поведінки в системі, а потім оповіщення адміністратору про виникнення відхилень. Трафік в мережі змінюється незначно в нормальному стані роботи системи. Однак деякі мережі мають незвичайні структури, особливо військові або

розвідувальні мережі, і, з іншого боку, дії, які відбуваються на сервері, можуть бути некерованими. Слід зазначити, що системний адміністратор хоче розділити події СВА на основі ненормальних подій (на відміну від відомого опису руху) і ненормальних протоколів подій (відхилятися від стандартів мережевого протоколу).

Деякі ефективні моделі виявлення поведінкового активності включають в себе:

• статистичну модель;

• модель, засновану на теорії інформації;

• модель кластера;

• модель класифікації.

2.3. Підходи з використанням методів машинного навчання

Система виявлення атак виявляє несанкціонований доступ до комп'ютерних систем і їх експлуатацію, відстежуючи ненормальну активність користувача, грунтуючись на встановленні правил або використанні команди онлайн-прогнозування. Однак ці заходи виявилися неефективними, дорогими, ненадійними і нездатними оновити себе, щоб виявити нові атаки. Іншим підходом, який долає вищевказані обмеження та все більше демонструє перевагу, є застосування методів машинного навчання з використанням безлічі різних методів. У цьому пункті будуть розглянуті деякі основні способи машинного навчання для порівняння роботи цих методів в завданні класифікації.

Байєсова мережа

Одним з найбільш часто використовуваних методів машинного навчання для виявлення вторгнень є Байєсова мережу.

Байєсова мережа [11, 12] – це модель, яка кодує імовірнісні відносини між розглянутими подіями (перемінними) і надає певний механізм для обчислення умовних ймовірностей їх настання.

Окремий випадок цієї моделі – наївний байесовский класифікатор (байєсівської метод) зі строгими припущеннями про незалежність вхідних змінних. Алгоритм наївною Байєсова класифікації – це група простих класифікацій ймовірностей, заснованих на теоремі Байеса, яка передбачає незалежність між атрибутами.

Навіть якщо ці атрибути пов'язані один з одним, то цей метод вважає, що атрибути не залежать одне від одного.

Досліджуючи наївний Байєсівський класифікатор можна відзначити наступні:

• наївні байєсовські класифікатори часто використовуються в задачах класифікації тексту;

• алгоритм має швидкий час навчання і тестування. Це пов'язано з припущенням незалежності між атрибутами, якщо клас відомий;

• наївні байєсовські класифікатори дають кращі результати, ніж логістичну регресію при меншій кількості навчальних даних, якщо припущення про незалежність виконано (Ця інформація базується на характері даних);

• алгоритм може працювати з векторами ознак, які є безперервними частинами (з використанням гауссовского наївного байєсівського алгоритму), а інші в дискретної формі (з використанням многочлена або Бернуллі);

• при використанні поліноміальний наївного байєсівського згладжування часто використовується згладжування Лапласа, щоб уникнути того факту, що компонент в даних

тесту не з'являється в систему адаптації.

2.4. Нейронна мережа

Штучна нейронна мережа [13, 14] – це модель обробки інформації, яка змодельована на поведінці нервової системи організму, включаючи велику

кількість нейронів, встановлених для обробки інформації. Штучна нейронна мережа подібна людському мозку, навченому на досвіді (за допомогою навчання), здатному зберігати досвід знань (знання) і використовувати ці знання при прогнозуванні невідомих даних (невидимі дані).

В роботі [15] проводиться порівняльний аналіз можливостей штучної нейронної мережі і методу дерева рішень для вирішення завдань виявлення комп'ютерних атак.

Дослідники дійшли висновку, що штучна нейронна мережа ефективна для узагальнення і малопридатна для виявлення нових атак, в той час як дерева рішень ефективні для вирішення обох завдань.

У задачах класифікації нейронні мережі завжди дають хороші результати, коли кількість вхідних параметрів обмежена в порівнянні з іншими методами машинного навчання. У конкретному завданні класифікації атак з двома різними класами (атак і без атак) метод опорних векторів виявився домінуючим.

Метод k-найближчих сусідів

Методом k-найближчих сусідів (k-nearest neighbor, k-NN) є один з найпростіших алгоритмів машинного навчання. При навчанні цей алгоритм нічого не вивчає з навчальних даних, кожен розрахунок виконується, коли йому необхідно передбачити результат нових даних.

Для k-NN в завданні класифікації мітка нової точки даних виводиться безпосередньо з найближчих k точок даних в навчальному наборі. Мітка тестових даних може бути визначена шляхом порівняння між найближчими точками (найближчими сусідами), або вона може бути виведена за допомогою різної ваги для кожної точки з безлічі найближчих точок.

Переваги методу полягають в наступному:

• простота прогнозу результату нових наборів даних;

• відсутність необхідності припускати будь-який розподіл класів.

Недоліки методу k-NN:

• гостра чутливість k-NN до шуму, коли k мало;

• обчислення відстані до кожної точки даних в навчальному наборі займе багато часу, особливо для баз даних з великими вимірами і багатьма точками даних.

Зі збільшенням k складність також зросте;

• вплив зберігання всіх даних в пам'яті на продуктивність стану.

Su Ming-Yang в своєму дослідженні [16] використовував змішаний підхід: об'єднання генетичного алгоритму і класифікатор k-найближчих сусідів для виявлення атак типу «відмова в обслуговуванні». Цей підхід дає досить високу точність виявлення атак: 96.75%.

2.5. Дерево прийняття рішень

Дерево прийняття рішення (ДПР) [17] є одним з найпопулярніших алгоритмів машинного навчання, доступних сьогодні. Він використовується в задачах класифікації та регресії.

Дерево прийняття рішення – це дерево, в якому кожен вузол являє характеристику (властивість), кожна гілка представляє правило, а кожен лист представляє результат (конкретне значення або безперервну гілка). У порівнянні з іншими методами аналізу даних дерево рішень має кілька переваг:

• простота пояснення роботи методу дерева прийняття рішення;

• метод може обробляти як числові дані, так і дані імена категорій;

• дерево прийняття рішення є моделлю «білого ящика». Якщо в моделі можна спостерігати цю ситуацію, то це можна пояснити за допомогою булевої логіки. Нейронні мережі є прикладом моделі «чорного ящика», тому що пояснення результатів занадто складно для розуміння;

• дерево прийняття рішення може швидко обробляти великі обсяги даних. Персональні комп'ютери можуть використовуватися для аналізу

великих обсягів даних за досить короткий час, щоб дозволити стратегам приймати рішення на основі аналізу дерева рішень;

Недоліки цього методу полягає в тому, що:

• цільові атрибути методу брали тільки дискретні значення;

• результат класифікації даних залежить від якості навчальної вибірки;

• при вирішенні задач класифікації з великим числом класів цей метод не ефективний.

2.6. Метод опорних векторів

Метод опорних векторів є відомим методом навчання з учителем, що використовується для задач класифікації та регресійного аналізу.

Метод опорних векторів [18-20] відноситься до методів лінійної класифікації. До завдань класифікації з двома класами входить два безлічі точок, що належать до двох різних класів, що розділяються гиперплоскостью в цьому просторі. При цьому гіперплоскость будується так, щоб відстані від неї до найближчих кордонів обох класів (опорних точок) були максимальні, що забезпечує найбільшу точність класифікації.

Після перевірки деяких наборів даних із застосуванням методу опорних векторів слід відзначені його переваги:

• метод опорних векторів застосуємо для класифікації тексту, де розміри можуть бути надзвичайно великими;

• метод зводиться до вирішення задачі квадратичного програмування в опуклою області, яка завжди має єдине рішення;

• можливість застосовувати нове ядро, яке забезпечує гнучкість між лінійними і нелінійними алгоритмами, що підвищує продуктивність класифікації.

З недоліків можна відзначити наступні:

• метод ефективний тільки для вирішення завдань з двома класами;

• чутливість до шумів і стандартизації даних;

• відсутність автоматичного способу вибору функції ядра в разі лінійної нероздільності класів.

2.7. Аналіз сканерів вразливостей веб-сайтів

В даному розділі проаналізуємо інші сканери вразливостей веб-сайтів.

Сканер Shadow Security виявляє вразливі місця в системі безпеки серверів як на платформі * Unix, так і на платформі Windows. Shadow Security Scanner має унікальну технологію виявлення можливих дірок в системі. На даний момент база складає майже 10000 аудитів. В неї входить аудит таких модулів як TCP / IP, UDP, FTP, DNS, SMTP, POP3, HTTP, CGI, NetBIOS, Registry, Users accounts, Password checks, Services, LDAP, DoS атаки і багато іншого. Також є генератор звітів. Програма підтримується російською мовою. Інтерфейси сканера представлено на рис. 2.1 та 2.2.

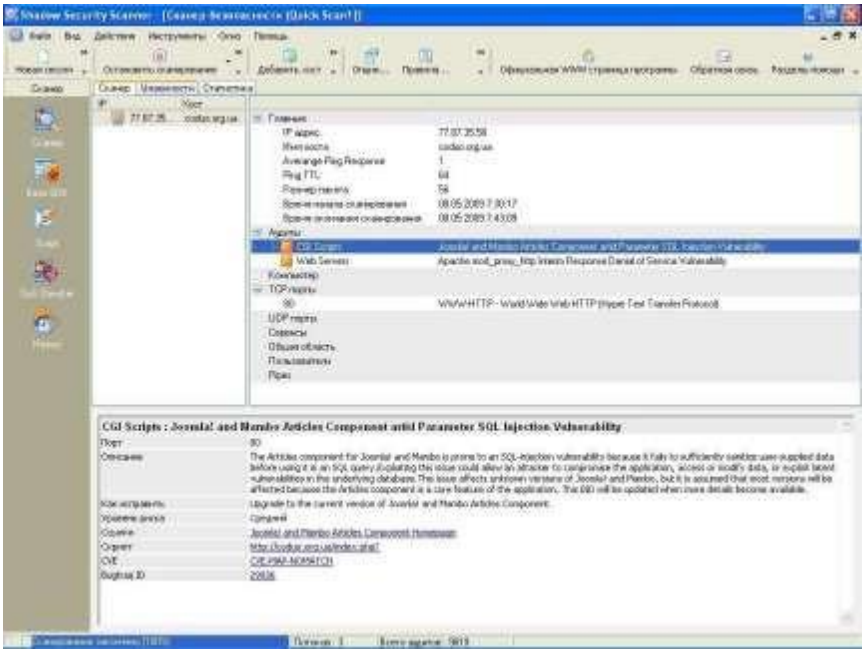


Рис. 3.1. Приклад використання Shadow Security Scanner

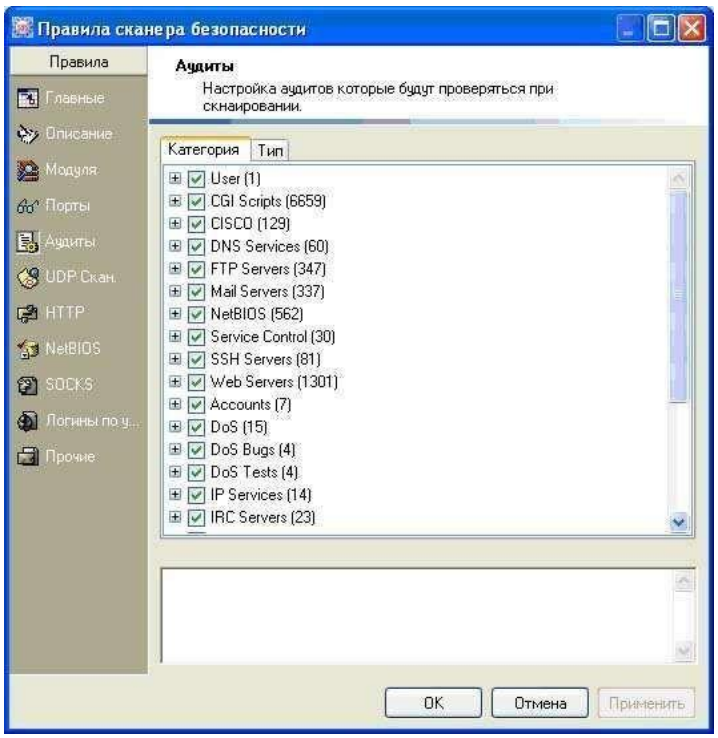


Рис. 2.2. Налаштування правил сканування

Acunetix Web Vulnerability Scanner

Компанія Acunetix – всесвітній лідер захисту веб-додатків. Компанія стала першовідкривачем у сфері дослідження технологій захисту веб-додатків. Ще з 1997 року компанія сконцентрувала свою діяльність на захисті веб-сайтів і поступово вдосконалила технічні переваги у тестуванні веб-сайтів та виявленні вразливих місць у системі їх захисту. Acunetix Web VulnerabilityScanner має такі інноваційні характеристики:

– автоматичний аналізатор JavaScript, що робить безпечне тестування додатків Ajax і Web 2.0;

– найбільш удосконалене тестування SQL ін'єкцій та Cross site scripting;

– visual macro recorder, що дозволяє побачити, в яких саме місцях були знайдені помилки полегшує роботу над веб-бланками і сторінками, захищеними паролями;

– програми докладних звітів, включаючи звіти відповідають

стандартам;

– VISA PCI;

– багатопотоковий сканер, здатний з блискавичною швидкістю ретельно перевірити сотні тисяч сторінок;

– пошуковий агент, який визначає тип веб-сервера і мовупрограми;

– дослідження та аналіз вмісту веб-сайтів, включаючи flash content протокол SOAP і AJAX.

Acunetix робить перевірку всіх вразливих місць веб-сайту, включаючи перевірку SQL ін'єкцій, Cross site scripting та інших вразливих місць у системі захисту веб-сайтів.

Для виявлення даних вразливостей, необхідний комплексний двигун. Основним у процесі перевірки сайта є не тільки кількість вразливостей, які сканер здатний виявити, а також комплексність і ретельність запуску різноманітних SQL ін'єкцій, Cross site scripting та інших хакерських атак. Acunetix має вдосконалений програмний код, що дозволяє з високою швидкістю і з найменшою вірогідністю помилкового результату знайти вразливі місця в системі захисту веб-сайту. Також він дозволяє виявити CRLF ін'єкції, Code execution, Directory Traversal, File inclusion та вразливості при аутентифікації.

Перевірка AJAX та технологій Web 2.0 на уразливості. Сучасний аналізатор JavaScript дозволяє всебічно перевірити новітні та найбільш складні AJAX та Web 2.0 Веб-додатки та знайти вразливі місця.

Детальні звіти, що дозволяють знайти відповідності прийнятим стандартам. Acunetix Web Vulnerability Scanner має схему докладних звітів, які крім усього також показують, чи відповідають Ваші Веб-додатки новим вимогам і стандартам VISA PCI.

Перевірка Вашого веб-сайту на захищеність від The Google Hacking Database. The Google Hacking Database (GHDB) - база запитів, якими

користуються хакери для виявлення незахищених даних на Вашому веб-сайті. До них відносяться сторінки реєстрацій на порталах, файли подій і так далі. Acunetix запускає GHDB запити на Ваш веб-сайт і розпізнає незахищені дані або дані, схильні до використання до того, як це зробить пошукова програма хакера.

Можливість налаштування програми тестування. Окрім програми автоматичного сканування, Acunetix також має вдосконалені сервісні модулі, що дозволяють тестувальником з точністю налаштувати перевірку безпеки Веб-додатків:

– HTTP Editor – за допомогою цієї сервісної програми Ви зможете без особливих зусиль зробити HTTP/HTTPS запити і проаналізувати реакцію Інтернет-сервера;

– HTTP Sniffer – зупиняти, розпізнавати і модифікувати весь мережевий трафік HTTP/HTTPS і виявляти всю інформацію, надіслану Вебдодатком

– HTTP Fuzzer – виконує всебічну перевірку переповнення буфера і правильності введення даних. Легко налаштовуються правила http Fuzzer дозволяють тестувати тисячі вхідних змінних. Тестування, які зазвичай займають цілі дні, зараз можуть бути виконані протягом декількох хвилин.

– також можна створити схему індивідуальної атаки або ж модифіковані вже існуючу за допомогою Web Vulnerability Editor.

Тестування сторінок, захищених паролями, а також веб-форм за допомогою програми автоматичного заповнення веб-форм HTML. Acunetix Web Vulnerability Scanner автоматично заповнює веб-форми і поля вебреєстрації. Більшість інших сканерів нездатні цього зробити або вимагають написання складних програм для тестування таких сторінок. Але з Acunetix все набагато простіше: за допомогою макро записуючого пристрою Ви зможете записати реєстрацію або процес заповнення форм і зберегти послідовність. Під час процесу тестування сканер знову відтворить

цю послідовність, а також автоматично заповнить веб-форму або увійде на сторінку, захищену паролем.

Вдосконалені характеристики:

– тестуючі профілі без особливих зусиль перевіряють веб-сайти з різними скануючими опціями і особливостями;

– розвинена система звітності;

– порівняння і пошук відмінностей від результатів сканування попередніми версіями;

– програма без особливих зусиль здійснює додаткові перевірки змін вебсайту;

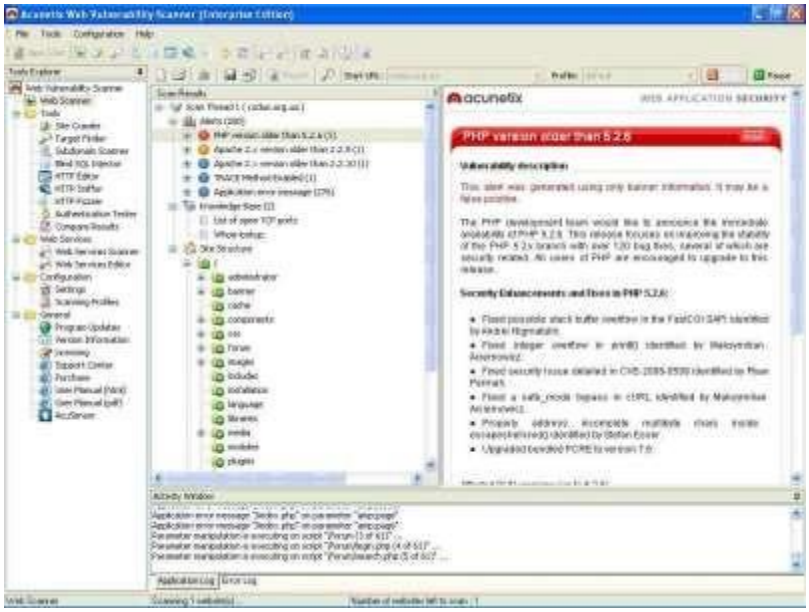


Рис. 2.3. Інтерфейс Acunetix Web Vulnerability Scanner

– виявлення каталогів (директорій), доступ до яких є небажаним;

– програма знаходить популярні Веб-додатки (наприклад, форуми, кошики для віртуальних покупок) і виявляє уразливі версії;

– визначення, небезпечних HTTP методів, активовані на веб-сервері.

Сканер Nessus має клієнт-серверну архітектуру. Відповідно до проведеного порталом securitylab.ru опитуванням, nessus використовують

17% респондентів.

Рис. 2.4. Інтерфейс сканера Nessus

Перш за все використовується для сканування портів і визначає сервіси, що використовують їх. Так само проводиться перевірка сервісів по базі вразливостей. Для тестування вразливостей використовуються спеціальні плагини, написані на мові NASL (Nessus Attack Scripting Language).

База вразливостей оновлюється щотижня, проте для комерційних абонентів є можливість завантажувати нові плагіни без семиденної затримки.

При відключеній опції "safe checks" деякі тести на уразливості, що використовуються nessus можуть привести до порушень в роботі сканованих систем. Ранні версії поширюються за ліцензією GPL. Нові версії є комерційним продуктом.

Сканер вразливостей XSpider 7 може в повністю автоматичному режимі перевіряти комп'ютери і сервіси в мережі на предмет виявлення вразливостей. База вразливостей постійно поповнюється фахівцями Positive Technologies, що в сумі з автоматичним оновленням баз і модулів програми постійно підтримує актуальність версії XSpider.

XSpider може виконувати перевірки за розкладом. Таким чином,

налаштувавши планувальник XSpider, автоматичне оновлення і надсилати звіти про результати перевірки поштою або їхнє збереження на мережному диску, можна значно полегшити процес виявлення вразливостей. Це дозволить сконцентрувати свою увагу на боротьбі з уже виявленими уразливими місцями та оновленні програмного забезпечення. У цьому XSpider надає так само неоціненну допомогу, виводячи до звіту про результати перевірки не тільки інформацію про знайдену уразливості, але і посилання, наприклад, на статті на сайті Microsoft, які описують виявлену XSpider вразливість і дають рекомендації по її усуненню.

Рис. 2.5. Повідомлення про DoS–атаку

Кожне налаштоване завдання зберігається у файлі. Якщо запланований запуск завдання з Планувальника, то результат її виконання буде збережений у файлі .tsk, який може бути відкритий у будь-який час за допомогою XSpider. У файлі зберігається результат не тільки останньої перевірки хоста або хостів, а вся історія перевірок. Таким чином, можна контролювати зміни рівня безпеки після ліквідації виявлених раніше вразливостей і поновлення операційних систем і сервісів. Приклад завантаженого завдання показаний на рисункунижче.

Рис. 2.6. Приклад завантаженого завдання

У даному прикладі на Windows XP SP3 з увімкненим файрволом був встановлений Apache, PHP, mySQL, безкоштовний скрипт для підписки на новини, демоверсія поштового сервера MERAK Mail Server (SMTP/ESMTP/POP3/IMAP4) з модулем віддаленого управління і налаштуваннями за замовчуванням, Remote Administrator 2.2 від компанії FamaTech. Для цієї перевірки був створений профіль, в якому було включено сканування всіх портів хоста і пошук всіх можливих вразливостей. В результаті аудиту було виявлено багато нарікань до скрипту підписки на новини і був виявлений паблікрелей на поштовому сервері (так як настроювання сервера після його встановлення не виконувалася). Всі інші сервіси, які працюють на тестовому комп'ютері, були виявлені і безпомилково ідентифіковані.

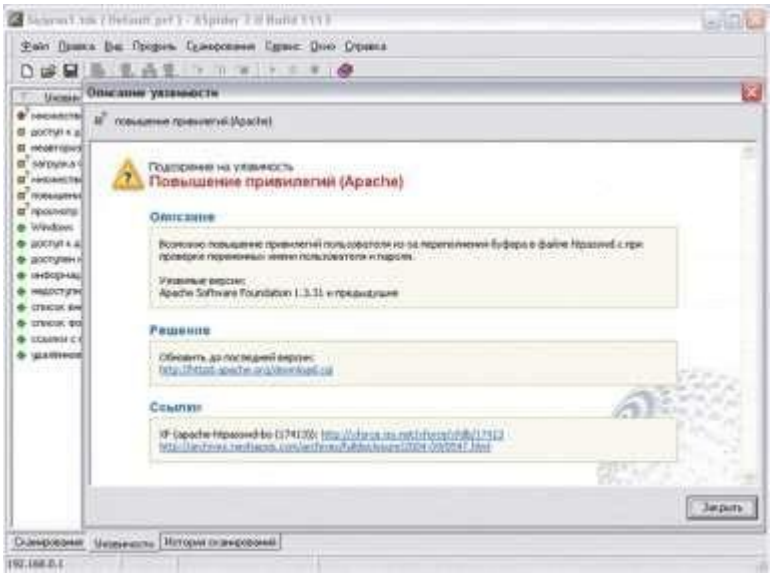


Рис. 2.7. Результати сканування

У результатах перевірки, крім інформації про виявлені вразливості, були наведені посилання на опис уразливості на сайтах, що спеціалізуються на безпеці і дані посилання на завантаження оновлених версій програмного забезпечення.

У другому тесті аудиту був підданий хост з ОС Windows XP без сервіс-пака з відключеним брандмауером. Вкладка Вразливості головного вікна XSpider показана на рисунку нижче.

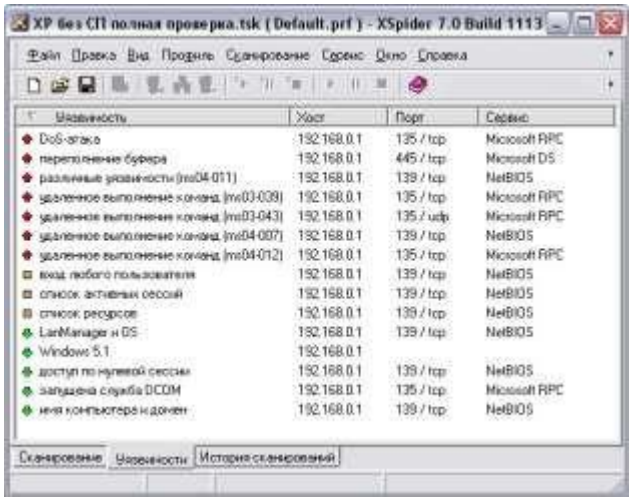


Рис. 2.8 Вкладка «Вразливості»

У ході сканування було виявлено кілька критичних вразливостей. У результатах роботи були дані посилання статті в Базі знань Microsoft, які

описують виявлену уразливість і посилання для завантаження виправлень, що усувають ці уразливості.

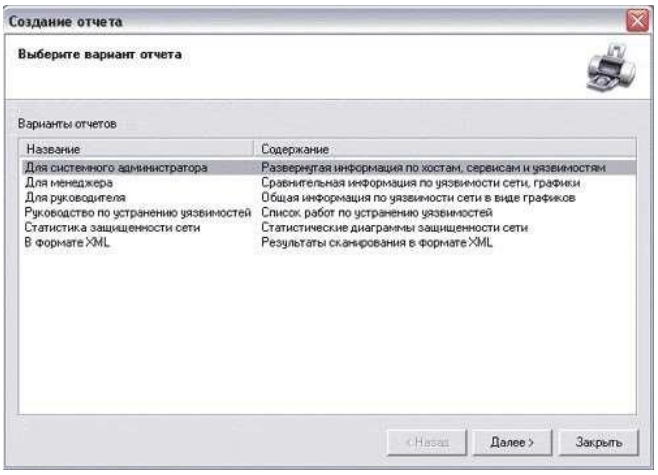


Рис. 2.9. Вибір варіантів представлення звіту

XSpider пропонує на вибір декілька стандартних типів звітів про результати перевірки.

Висновки по розділу

В даному розділі наведені найбільш широко використовувані методи для виявлення атак на веб-додатки, наведені їх особливості, переваги та недоліки.

РОЗДІЛ 3
РОЗРОБКА ПРОГРАМНОГО МОДУЛЯ ДЛЯ ВИЯВЛЕННЯ АТАК НА ВЕБ-ДОДАТКИ

3.1. Структура веб-додатку

Веб-застосунок (часом називають **веб-додаток)** – розподілений застосунок, в якому клієнтом виступає браузер, а сервером – вебсервер. Браузер може бути реалізацією так званих тонких клієнтів – логіка застосунку зосереджується на сервері, а функція браузера полягає переважно у зображенні інформації, завантаженої мережею з сервера, і передачі назад даних користувача. Однією з переваг такого підходу є той факт, що клієнти не залежать від конкретної операційної системи користувача, тому вебзастосунки є міжплатформовими сервісами. Унаслідок цієї універсальності й відносної простоти розробки вебзастосунки стали широко популярними в кінці 1990-х – початку 2000-х років.

Істотною перевагою побудови вебзастосунків для підтримки стандартних функцій браузера є те, що функції повинні виконуватися незалежно від операційної системи клієнта. Замість того, щоб писати різні версії для Microsoft Windows, Mac OS X, GNU/Linux й інших операційних систем, застосунок створюється один раз для довільно обраної платформи та на ній розгортається. Проте різна реалізація HTML, CSS, DOM й інших специфікацій в браузерах може викликати проблеми при розробці вебзастосунків і подальшої підтримки. Крім того, можливість користувача налаштовувати багато параметрів браузера (наприклад, розмір шрифту, кольори, відключення підтримки сценаріїв) може перешкоджати коректній роботі застосунку.

На початку 2000-х років був популярним інший (менш універсальний) підхід з використанням Adobe Flash або Java-аплетів для повної або часткової реалізації призначеного для користувача інтерфейсу. Ці технології надавали

програмістові більший контроль над інтерфейсом, і були здатні обходити багато несумісностей у конфігураціях браузерів (хоча несумісність між Java або Flash реалізаціями клієнта спричиняла різні ускладнення). Станом на 2020 рік Java-аплети та Flash-технологія практично вийшли з ужитку.

Через архітектурну схожість з традиційними клієнт-серверними застосунками, певним чином «товстими» клієнтами, існують суперечки щодо коректності зарахування подібних систем до вебзастосунків; альтернативний термін «Насичений інтернет-застосунок» (англ. Rich Internet Application).

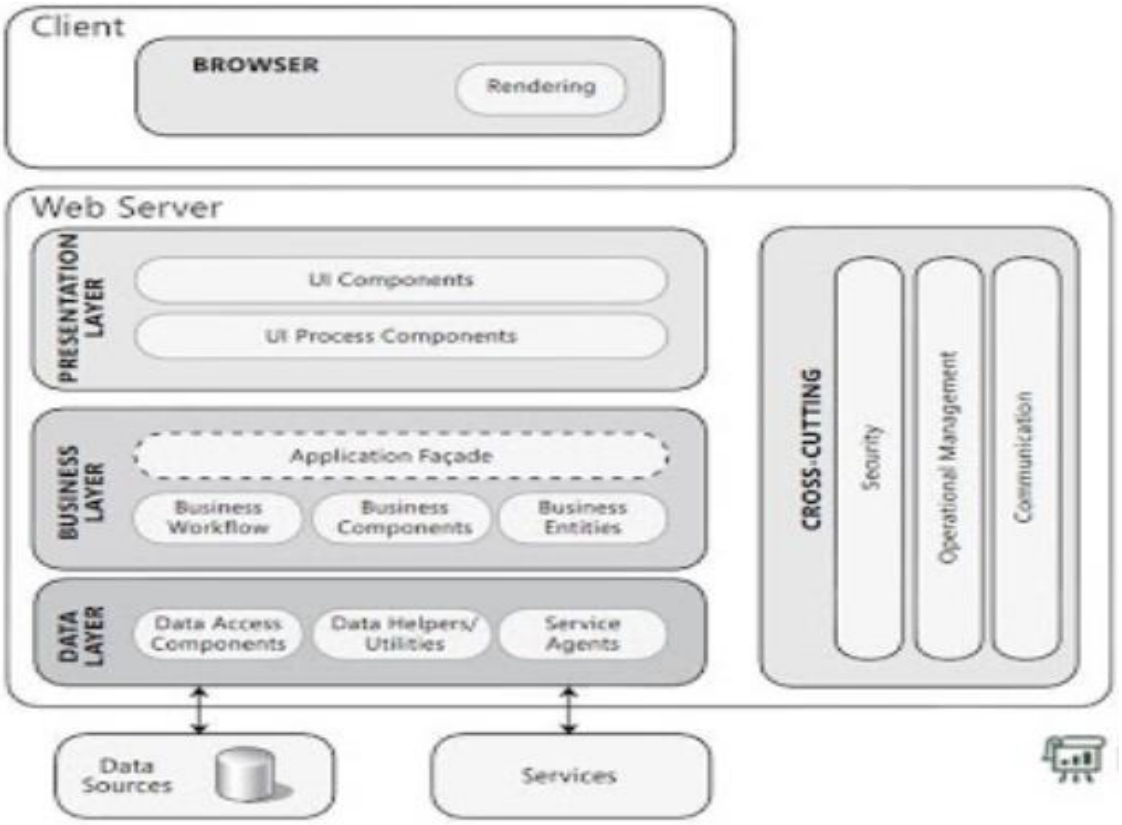


Рис. 3.1. Структура веб-застосунку

Вебзастосунок отримує запит від клієнта і виконує обчислення, після цього формує вебсторінку і відправляє її клієнтові мережею з використанням протоколу HTTP. Саме вебзастосунок може бути клієнтом інших служб, наприклад, бази даних або стороннього вебзастосунку, розташованого на іншому сервері. Яскравим прикладом вебзастосунку є система управління

вмістом статей Вікіпедії: безліч її учасників можуть брати участь у створенні мережевої енциклопедії, використовуючи для цього браузери своїх операційних систем (Microsoft Windows, GNU/Linux або будь-якої іншої операційної системи) без завантаження додаткових виконуваних модулів для роботи з базою даних статей.

Для більшої інтерактивності й продуктивності розроблений новий підхід до розробки вебзастосунків, названий AJAX, і який нині є стандартним де-факто. При використанні Ajax сторінки вебзастосунку здатні відправляти вебзапити до сервера у фоновому режимі, і не перезавантажуються цілком, а лише довантажують необхідні дані з сервера, що значно пришвидшує роботу і робить її зручнішою.

Для створення вебзастосунків використовуються різноманітні серверні технології та мови програмування.

Назва	**Ліцензія**	**Вебсервер**
ASP	власна	спеціалізований
ASP.NET	власниа	спеціалізований
Java	вільна	безліч, зокрема вільних
Groovy	вільна	практично будь-який
Perl	вільна	практично будь-який
PHP	вільна	практично будь-який
Python	вільна	практично будь-який
Ruby	вільна	практично будь-який

Клієнтська частина може використовувати:

- JavaScript
- Flash
- Java / JavaFX
- ActiveX
- Silverlight

3.2. Розробка структури та інтерфейсу модуля

При побудові системи виявлення атак зазвичай виходять з методу, за допомогою якого проводиться виявлення потенційно зловмисної активності. Розглянемо методи визначення атак, які використовуються в СВА веб-додатків.

У загальному вигляді виділяють два методу визначення атак:

- статичний;

- динамічний.

Статичний метод визначення атак з'явився першим і в цілому полягав в перегляді та аналізі системних журналів подій програмного забезпечення і операційної системи.

Системи виявлення атак класифікуються за способами визначення атак, однією з головних характеристик визначають ефективність СВА. Перевагою даного методу є простота реалізації, що не вимагає додаткових коштів і ресурсів. Однак даний метод містить безліч недоліків:

- можливість аналізувати події, що відбулися тільки після їх здійснення;

- аналіз журналів може бути дуже трудомістким через складність інформації, що міститься в журналах або розміру журналу;

- якщо зловмисник отримує доступ до системи, він може видалити всі записи в журналах.

Якщо розглядати даний метод з точки зору застосування його для захисту веб-додатків, то також можна виділити суттєві недоліки:

- неможливість аналізувати дані, що передаються методом POST;

- HTTP заголовки можуть аналізуватися лише частково.

Найбільш ефективним є динамічний метод, головною перевагою якого є аналіз і контроль цих процесів в режимі реального часу.

В даний час статичний метод застосовується як додатковий, а системи виявлення атак використовують виключно динамічний метод, при цьому вони класифікуються за способом реалізації даного методу.

Розрізняють такі способи визначення атак:

- сигнатурний спосіб;

- спосіб, заснований на виявленні аномалій в системі.

Системи виявлення атак, засновані на сигнатурному способі виявлення, включають в себе базу сигнатур відомих атак і працюють так само, як антивірусне програмне забезпечення, «піднімаючи тривогу», коли виникає збігу будь-якої сигнатури з вхідними даними. Сигнатури (також відомі як правила) зазвичай містять дані про відомих і широко використовуваних системах і додатках, для яких існують уразливості і інформація про них доступна публічно. Проте, як і антивірусне програмне забезпечення, яке не в змозі ідентифікувати невідомі віруси, коли в базі немає даних про їх сигнатури або вірусна база застаріла, сигнатурні СВА також не в змозі виявити невідомі атаки. Так як сигнатури спеціально розробляються, щоб відповідати відомим атакам, цей тип СВА зазвичай має дуже низький рівень помилкових тривог (помилкових спрацьовувань). Однак, через те, що сигнатури містять дані про конкретних атаках і носять статичний характер, системи виявлення атак, засновані на цьому методі, можуть не виявити навіть невеликі модифікації атаки. Даний факт є серйозним, так як атаки «нульового дня» і поліморфні атаки (в яких зміна корисного навантаження не впливає на ефективність) залишаться непоміченими системою виявлення атак, поки в її базах не з'являться відповідні сигнатури.

З огляду на, що нові уразливості з'являються часто, а точніше щодня, і що зазвичай існує проміжок часу, коли вразливість виявлена і коли для неї створять сигнатуру і вона буде додана в базу сигнатур, у зловмисника в цей час є потенційна перевага і він може ним скористатися, щоб отримати доступ до уразливої системи.

Проте, тема вразливостей «нульового дня» є складною і окремою темою, яка в даний час є безумовно важливою і над якою проводяться роботи по дослідженню даної проблеми і знаходженню ефективних методів і способів пошуку та усунення несправностей.

Щоб подолати обмеження, властиві сигнатурним СВА, які не в змозі виявити раніше невідомі атаки, дослідники шукали інші способи виявлення атак і розробили інший метод, заснований на визначенні аномалій.

Спосіб, заснований на виявленні аномалій в системі, працює шляхом створення статистичної моделі, яка використовує деякі шаблони, що описують нормальну поведінку контрольованих ресурсів (це не більше ніж набір характеристик, який визначається при нормальній роботі системи) і даний процес, як правило, називається етапом навчання . Після того, як статистична модель нормального поведінки системи створена, система виявлення атак починає порівнювати кожен наступний запит з моделлю і якщо визначаються відмінності, то СВА генерує попередження, так як вважає запит як аномальне подія. В основному атаки виявляються, тому що вони проводяться статистично-різному, тобто аномально, від поведінки, яке створювалося при складанні моделі.

Виявлення аномалій передбачає, що проведення атаки тісно пов'язане з неправильною поведінкою, на відміну від певного користувачем або системою виявлення атак. Основна ідея полягає в завданні базового нормального поведінки контрольованого об'єкта і виявленої активності, яка істотно відрізняється, є аномальною і яка може бути потенційною спробою проведення атаки на захищає систему.

Головна перевага СВА, заснованих на даному способі виявлення атак, - це здатність визначити раніше невідомі атаки (Або модифікації відомих). Оскільки системи, засновані на виявленні аномалій, в змозі визначити нові атаки, які обходять сигнатурні системи, то отримана інформація про

виявлення атаки згодом може бути використана для розробки нових правил для сигнатурних систем.

Істотним недоліком, який є у систем виявлення атак, заснованих на даному способі, є велика кількість помилкових спрацьовувань, а саме для випадків, коли моніторинг ресурсів не є стаціонарним і часто змінюється, наприклад, в разі веб-додатків. Якщо статистична моделі є занадто суворою і обмеженою, і система не може узагальнювати і враховувати зміни, то цілком ймовірно, що така система буде генерувати безліч помилкових спрацьовувань. Тим не менше, кілька продуктів, які використовують механізм виявлення аномалій, вже існують [5].

Ефективність розроблюваного проекту (системи) багато в чому залежить від організації процесу по його створенню. Перш ніж приступити до безпосередньої розробки системи необхідно пройти необхідні етапи проектування, які дозволяють суворо організувати життєвий цикл системи, виключити певні помилки і гнучко управляти всіма процесами.

Першим етапом проектування є визначення вимог до розроблюваної системі.

Вимоги до програмного забезпечення - сукупність тверджень щодо атрибутів, властивостей або якостей програмної системи, яка підлягає реалізації. Створюються в процесі розробки вимог до програмного забезпечення, в результаті аналізу [7].

Під розробкою вимог до ПЗ розуміється процес виявлення, формулювання, аналізу, документування та верифікації вимог, що підлягають виконанню під час розробки.

Як відомо аналізом вимог є процес збору вимог до програмного забезпечення, їх систематизації, документування, аналізу, виявлення суперечностей, неповноти, вирішення конфліктів в процесі розробки програмного забезпечення.

Модель вимог, що описується в UML, формують два основних типи вимог:

- функціональні вимоги – функції, які система повинна робити;

- нефункціональні вимоги - обмеження, що накладаються на систему.

Для розроблюваної системи виявлення атак визначаються наступні функціональні вимоги:

- СВА повинна бути кросплатформною;

- СВА повинна підтримувати найбільше число існуючих веб-серверів;

- СВА повинна бути простою в реалізації і використанні;

- СВА повинна забезпечувати достатній рівень захисту;

- СВА повинна забезпечити низький рівень помилкових спрацьовувань;

- СВА повинна надати досить просту інтеграцію з існуючими системами (додатками);

- СВА повинна забезпечити повноцінне робочий стан відразу після установки;

- СВА повинна бути незалежною від використання додаткових послуг (наприклад, послуг хостинг-компаній).

Нефункціональними вимогами є:

- СВА повинна бути розроблена на мові PHP;

- СВА повинна забезпечувати актуальність баз сигнатур.

Крім вироблення вимог при складанні моделі вимог існує інша форма і спосіб визначення і документування вимог до програмного забезпечення - це моделювання прецедентів. Результатом даної діяльності є модель прецедентів, яка складаються з чотирьох основних елементів:

- межа системи - це контекст системи, який відокремлює системи від решти світу. Позначається прямокутником, який окреслює прецедентів і позначає межі модельованої системи;

- актори - ролі, що виконуються людьми або сутностями, що використовують систему;

- прецеденти - то, що актори можуть робити з системою;

- відносини - значущі відносини між акторами і прецедентами.

Модель прецедентів є основним джерелом об'єктів і класів. Це основні вихідні дані для моделювання класів. Проведемо аналіз системи, що розробляється, з точки зору моделювання прецедентів. Як було зазначено вище, актори - це ролі, виконувані сутностями, безпосередньо взаємодіють з системою. У системі виявлення атак можна виділити два актора:

- адміністратор системи;

- користувач.

Актор адміністратор системи - це особа, яка в більшості випадків є власником певного веб-ресурсу, який використовує розроблювану СВА. Можна виділити основні прецеденти, які відповідають даним акторові:

- настройка параметрів СВА;

- контроль стану системи (моніторинг реєстрованих подій);

- забезпечення актуальності бази сигнатур шляхом її періодичного оновлення.

Актор користувач є описом деякої загальної особи, яке є користувачем веб-ресурсу, а значить і розроблюваної системи, так як система є інтегрованою в веб-додаток. При цьому користувач може мати у своєму розпорядженні різними намірами. Користувач, який має злі наміри, є зловмисником. В даному випадку, розглядаючи користувача в якості зловмисника, передбачається, що він в змозі визначити наявність системи виявлення атак і його дії будуть спрямовані потенційно на СВА, іншими словами зловмисник буде катувати обійти засоби захисту і здійснити атаку [8].

Користувач, який не має злого умислу, є звичайним користувачем веб-ресурсу, проте, існує ймовірність, що він може випадково ввести некоректні дані, які можуть містити небезпечні символи і логіка веб-додатки може бути порушена. Так як розробляється СВА здійснює моніторинг на однаковому

рівні і не залежить від намірів користувача, передбачається виробити узагальнення акторів в актора-батька, який буде позначати деякий особа, яка використовує веб-ресурс - користувач.

Виділяються наступні прецеденти, які виконує СВА при взаємодії з користувачем:

- при введенні даних або параметрів відбувається формування масиву вхідних даних;

- фільтрація вхідних даних, використовуючи пошук по сигнатурам;

- виявлення збігу з сигнатурою;

- перетворення даних в безпечний вид;

- реєстрація події в СВА;

- відображення зареєстрованих подій в консолі управління;

- передача управління веб-додатком.

Варто зазначити, що забезпечення безпеки, що здійснюється розробляється системою виявлення атак, направлено тільки безпосередньо на веб-додаток в яке інтегрована СВА. Тобто фільтрація і контроль вхідних даних, а також реєстрація що виникли подій здійснюється тільки на рівні веб-додатки. Моніторинг подій, що виникають безпосередньо в веб-сервері, інтерпретатор PHP, операційній системі сервера, СВА не здійснює.

Після завершення етапу визначення вимог до розроблюваної системі слід етап проектування архітектури системи.

Архітектурою програмного забезпечення є структура програми або обчислювальної системи, яка включає програмні компоненти, видимі зовні властивості цих компонентів, а також відносини між ними.

Розробляється система виявлення атак є динамічною системою, що виробляє обробку вхідного масиву даних. Для опису архітектури системи і її функціональності використовуються діаграми діяльності.

Діаграма діяльності - об'єктно-орієнтована діаграма, на якій показано розкладання деякої діяльності на її складові частини.

Під діяльністю розуміється специфікація виконуваного поведінки у вигляді координованого послідовного і паралельного виконання підлеглих елементів - вкладених видів діяльності і окремих дій, з'єднаних між собою потоками, які йдуть від виходів одного вузла до входів іншого [8].

Узагальнена діаграма діяльності системи виявлення атак представлена на рисунку 3.2.

Розробку СВА передбачається здійснити на мові PHP, який є вільним мовою для веб-розробки і надає Кросплатформені можливості. Його підтримка здійснюється максимально можливою кількістю веб-серверів і компаніями надають хостинг-послуги. Розробляється система є вбудованої системою, яка працює паралельно з веб-додатком і постійно здійснює моніторинг вхідних даних. Використання для реалізації цієї мети PHP дозволить без зусиль інтегрувати систему з веб-додатками, не витрачаючи додаткових ресурсів.

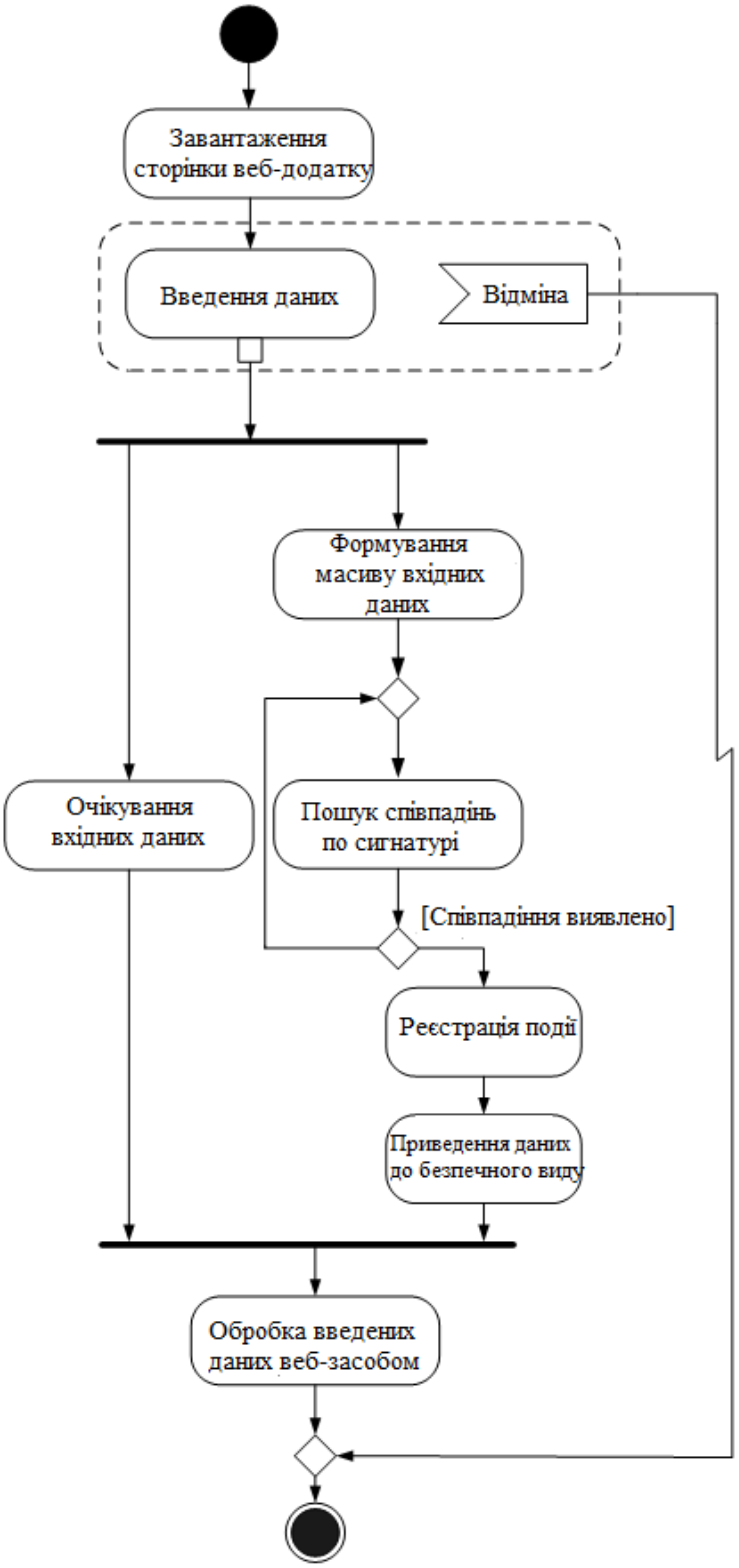


Рис. 3.2. Узагальнена діаграма діяльності СВА

Загальний алгоритм роботи СВА на веб-додатки, представлений на рисунках 3.3-3.4, складається з наступних етапів:

Етап 1. Введення даних (передача параметрів). Даний етап є початковим і має на увазі передачу параметрів, яка реалізується логікою веб-додатки, або введення даних користувачем для обробки веб-додатком. Слід зазначити, що даний етап може бути перерваний, в разі якщо користувач ввів дані, послав сигнал на їх відправку (зазвичай натисканням кнопки на веб-формі) і потім негайно скасував дію. В цьому випадку обробка вхідних даних не здійснюється як системою, так і веб-додатком.

Етап 2. Формування масиву вхідних даних. Передача вхідних даних веб-додатку може здійснюватися кількома способами. Як було зазначено раніше, мережеве взаємодія здійснюється за допомогою протоколу HTTP, який має методи POST і GET. Їм відповідають глобальні масиви $ _POST і $ _GET мови PHP. Крім того, для здійснення механізму Cookie існує масив $ _COOKIE. На даному етапі відбувається збір входять даних з усіх глобальних масивів і формування загального масиву даних для зручності здійснення процесу обробки.

Етап 3. Пошук збігів з сигнатурою. На даному етапі відбувається перевірка кожного параметра з масиву даних з сигнатурами на встановлення відповідності. При виявленні збігу відбувається реєстрація виник події, якщо подія не встановлено проводиться аналіз наступного параметра.

Етап 4. Виявлено збіг з сигнатурою. При виявленні збігу з сигнатурою відбувається приведення значення співпала параметра, шляхом фільтрації, до безпечного виду. Це здійснюється з використання спеціальних PHP функцій (strip_tags (string $ str), mysql_real_escape_string (string $ unescaped_string) і ін.). Крім цього інформація про цю подію реєструється в СВА для подальшого аналізу адміністратором системи.

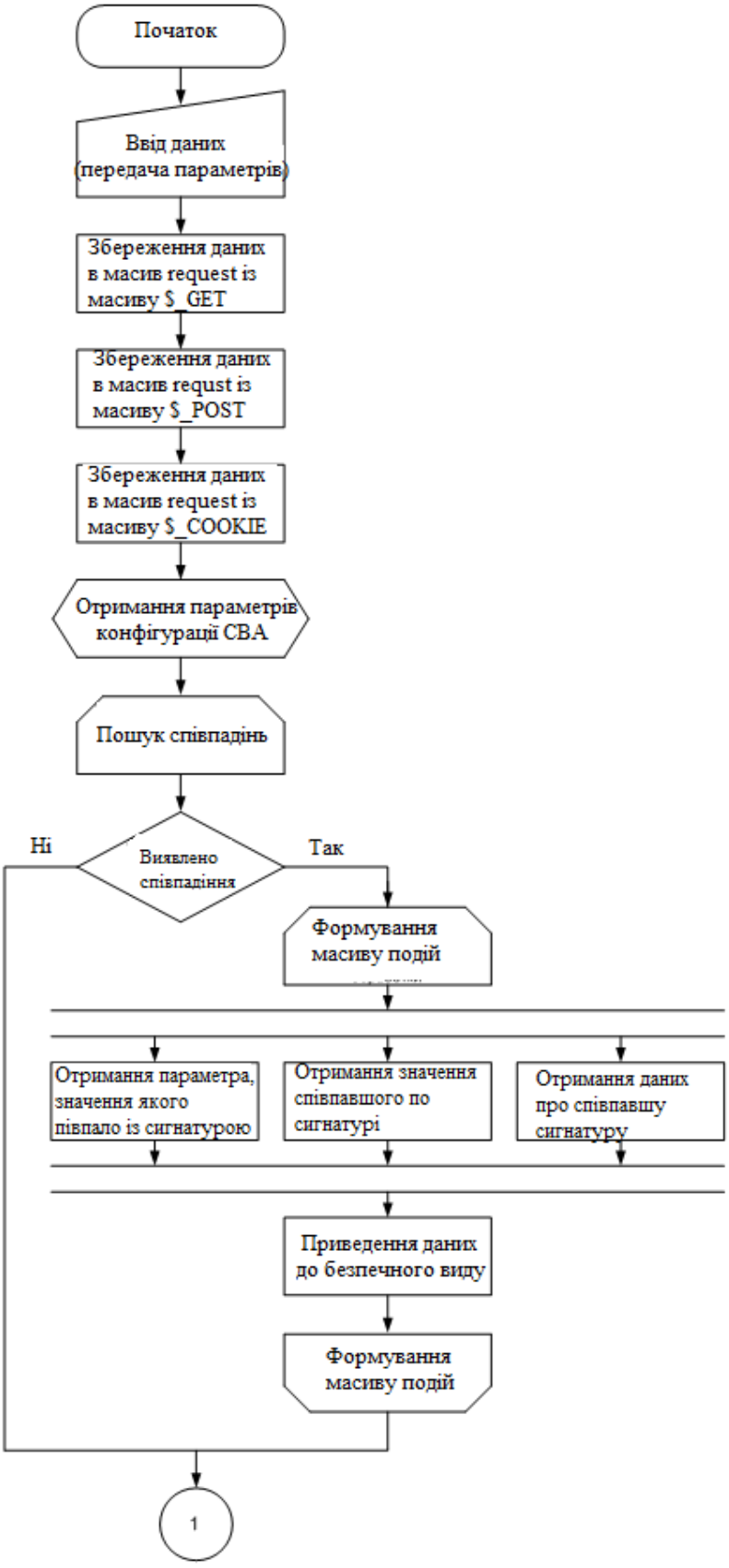


Рис. 3.3. Загальний алгоритм роботи СВА

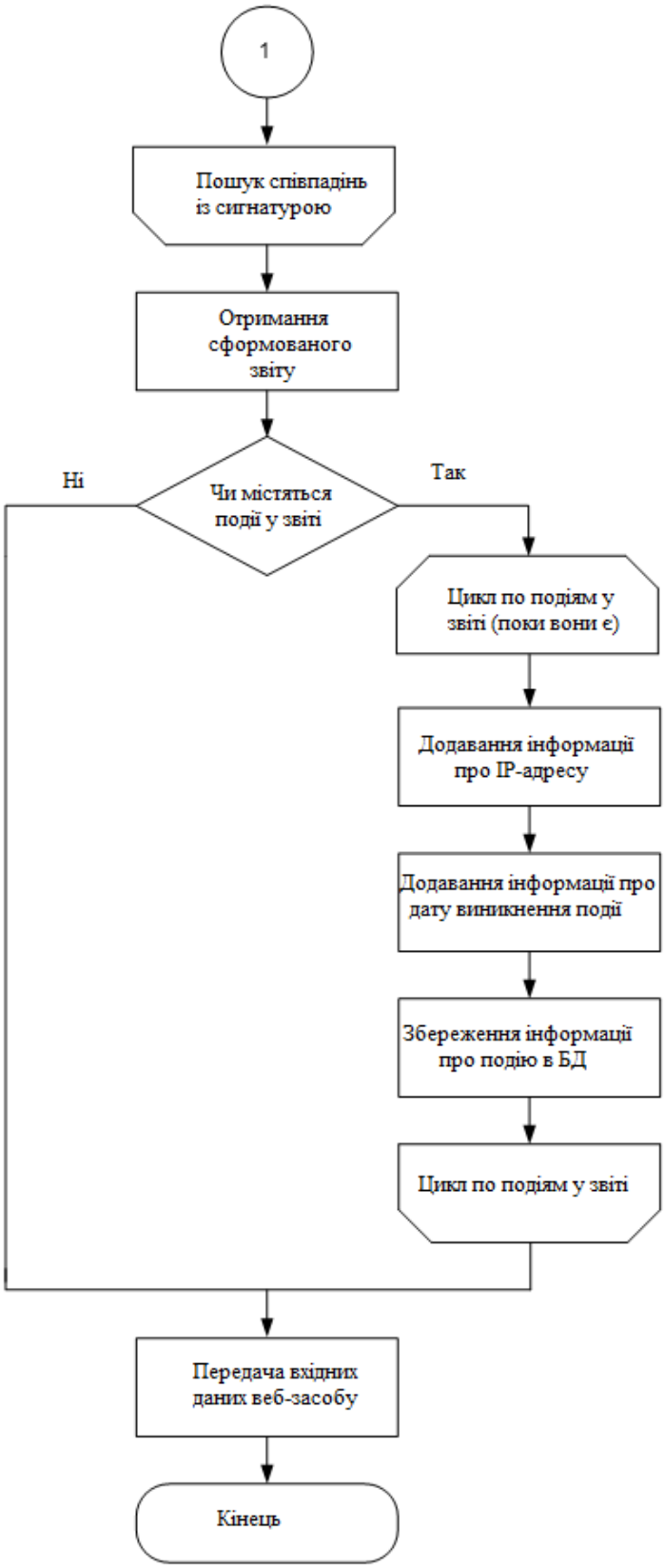


Рис. 3.4. Загальний алгоритм роботи СВА (продовження)

Етап 5. Передача управління веб-додатком. Після аналізу, всі вхідні параметри передаються для обробки логікою веб-додатки. До даного етапу

веб-додаток перебуває ніби в стані «очікування», але так як робота СВА відбувається з високою швидкістю, то в залежності від кількості параметрів воно не перевищує одиниць секунд.

Для детального опису архітектури системи необхідно провести її декомпозицію, виділивши окремо складові модулі (підсистеми).

Система виявлення атак веб-додатків складається з наступних модулів:

1. Модуль логіки СВА.

2. Модуль формування звіту.

3. Модуль реєстрації подій.

4. Модуль управління СВА.

Структурна схема взаємодій модулів представлена на рис. 3.5.

Модуль логіки системи виявлення атак є логічним ядром, яке здійснює аналіз і фільтрацію вхідних даних. Основний механізм пошуку збігу сигнатур побудований на використанні регулярних виразів.

Регулярні вирази - це формальна мова пошуку і здійснення маніпуляцій з підрядками в тексті, заснований на використанні метасимволов. Іншими словами це рядок-зразок (Шаблон), що складається з символів і метасимволов і задає правило пошуку [9].

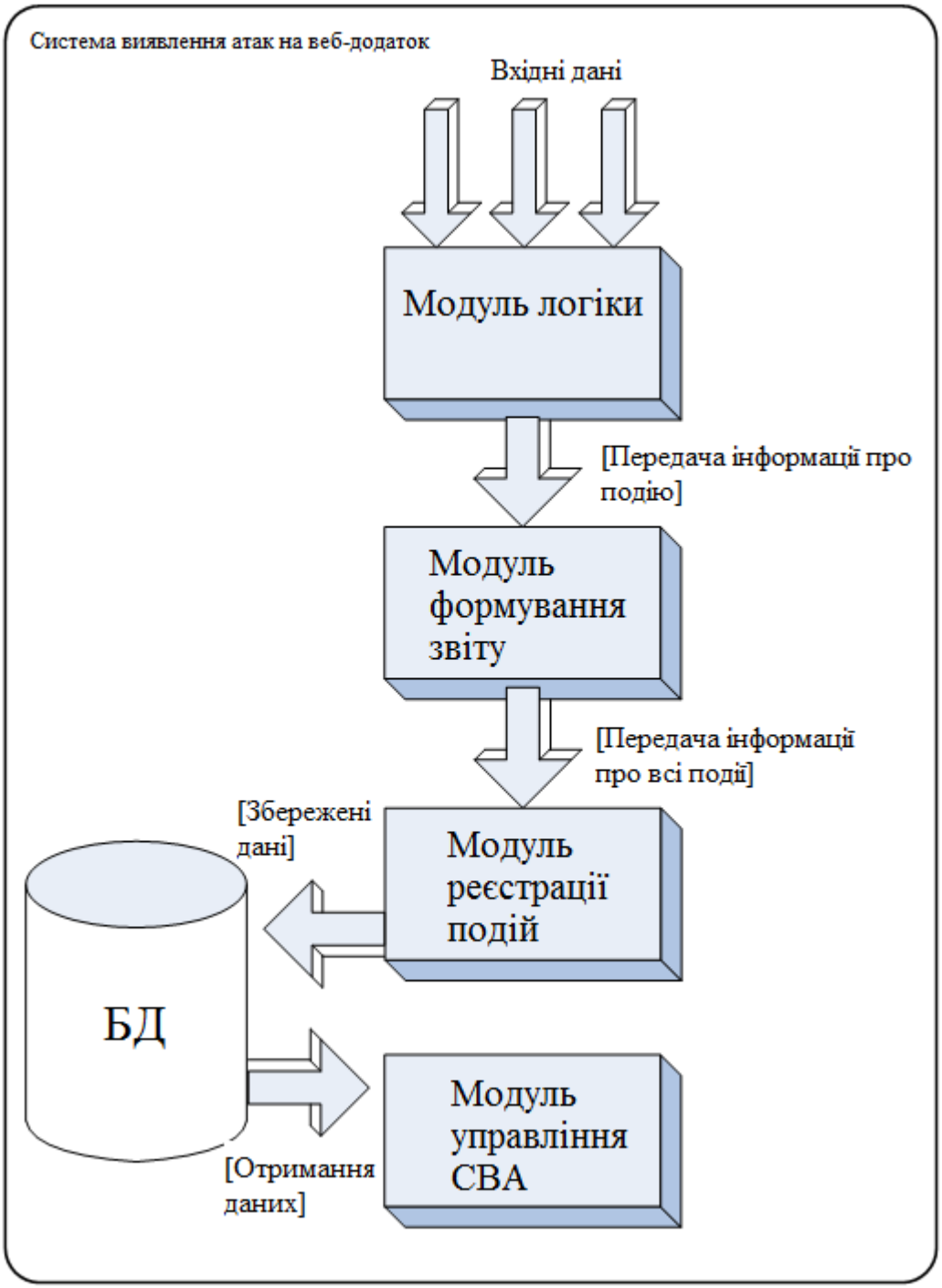


Рис. 3.5. Структурна схема СВА на веб-додатки

Потенційно небезпечні конструкції, описувані за допомогою мови регулярних виразів, є сигнатурами. Механізм пошуку збігу заснований на використанні спеціальної функції PHP - preg_match (string $ pattern, string $

subject, array $ matches). Ця функція приймає три параметри, регулярний вираз (параметр $ pattern), рядок для пошуку (параметр $ subject) і масив, що містить збіги (параметр $ matches).

Діаграма діяльності модуля логіки системи виявлення атак зображена на рисунку 3.6.

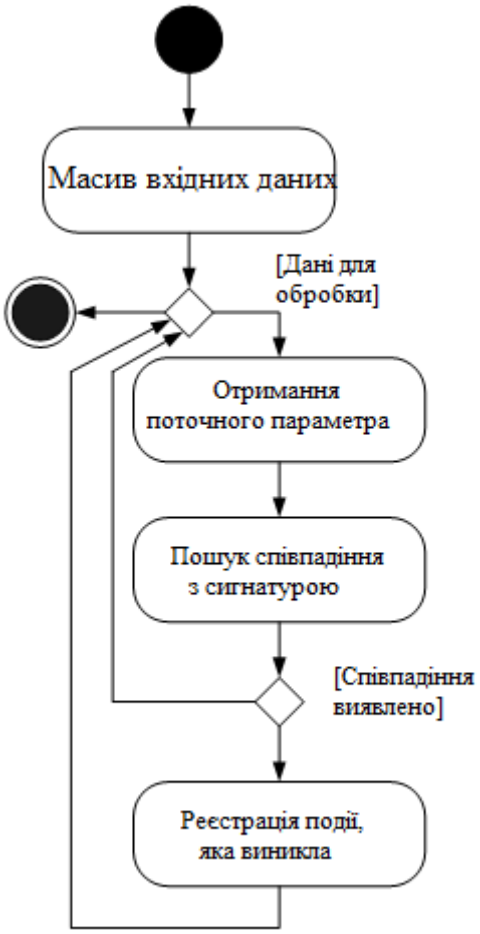


Рис. 3.6. Діаграма діяльності модуля логіки СВА

Модуль формування звіту отримує управління при виникненні збігу сигнатури з вхідними даними. При появі збігу створюється спеціальний об'єкт Event, який описує виникло подія. При створенні об'єкта події в нього передається інформація про параметр, значення даного параметра і сигнатуре, з якої збіглося значення параметра. Крім того, в даному модулі існує деяке сховище (масив), в яке поміщаються всі події, що виникають при обробці модулем логіки СВА вхідних даних. Таким чином, даний масив

містить звіт по всіх виниклих подій (збігів). Діаграма діяльності даного модуля представлена на рис. 3.7.

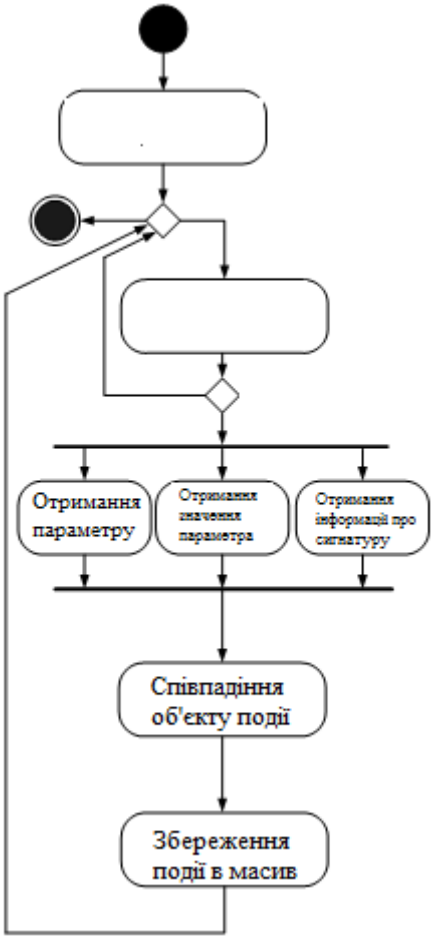


Рис. 3.7. Діаграма діяльності модуля формування звіту СВА

Модуль формування звіту передає управління в модуль реєстрації подій, який витягує зі звіту всі зібрані події та реєструє їх в системі. Реєстрація подій передбачає використання бази даних для хранений інформації про всі події.

Всі сучасні веб-додатки використовують бази даних для зберігання інформації. Для незалежності від обраної бази даних для веб-додатки робота модуля реєстрації подій заснована на використанні спеціального розширення - PDO.

PHP Data Objects (PDO) - розширення для PHP, що надає розробнику простий і універсальний інтерфейс для доступу до різних баз даних. PDO є деяким абстрактним шаром, який дозволяє уникнути залежності системи від

конкретної бази даних. З версії PHP 5.1 PDO входить в ядро PHP, тому встановлювати ніякі додаткові пакети не потрібно.

Діаграма діяльності даного модуля представлена на рис. 3.8.

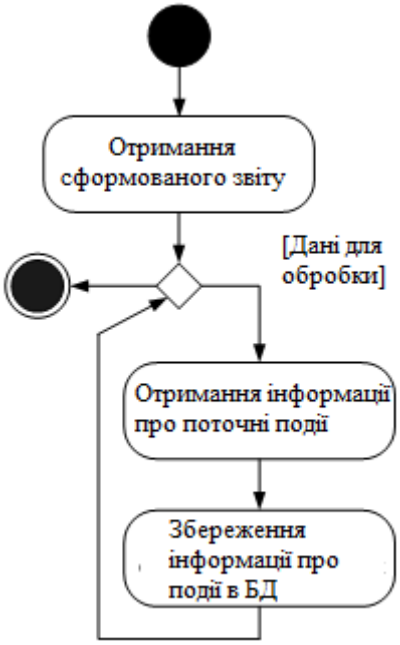


Рис. 3.8. Діаграма діяльності модуля реєстрації подій

В даному модулі відбувається запис інформації про кожну подію в таблицю з подіями. Цю таблицю використовує модуль управління, для надання адміністратору системи інформації про стан СВА.

Модуль управління системою виявлення атак є консоль управління адміністратора, виконану у вигляді веб-панелі.

Для отримання доступу до даного модулю необхідно пройти процедуру аутентифікації, після чого адміністратор отримує можливість управляти СВА. Дані про всі зареєстровані події Модулем реєстрації подій зберігаються в таблиці бази даних, з якою всю інформацію завантажує модуль управління для відображення в консолі.

Даний модуль надає адміністратору актуальну інформацію про всі зареєстровані події, які відображаються у вигляді таблиці з можливістю сортування і пошуком. Крім того, в даному модулі адміністратор здійснює

додавання нових сигнатур для підтримки актуального стану системи виявлення атак.

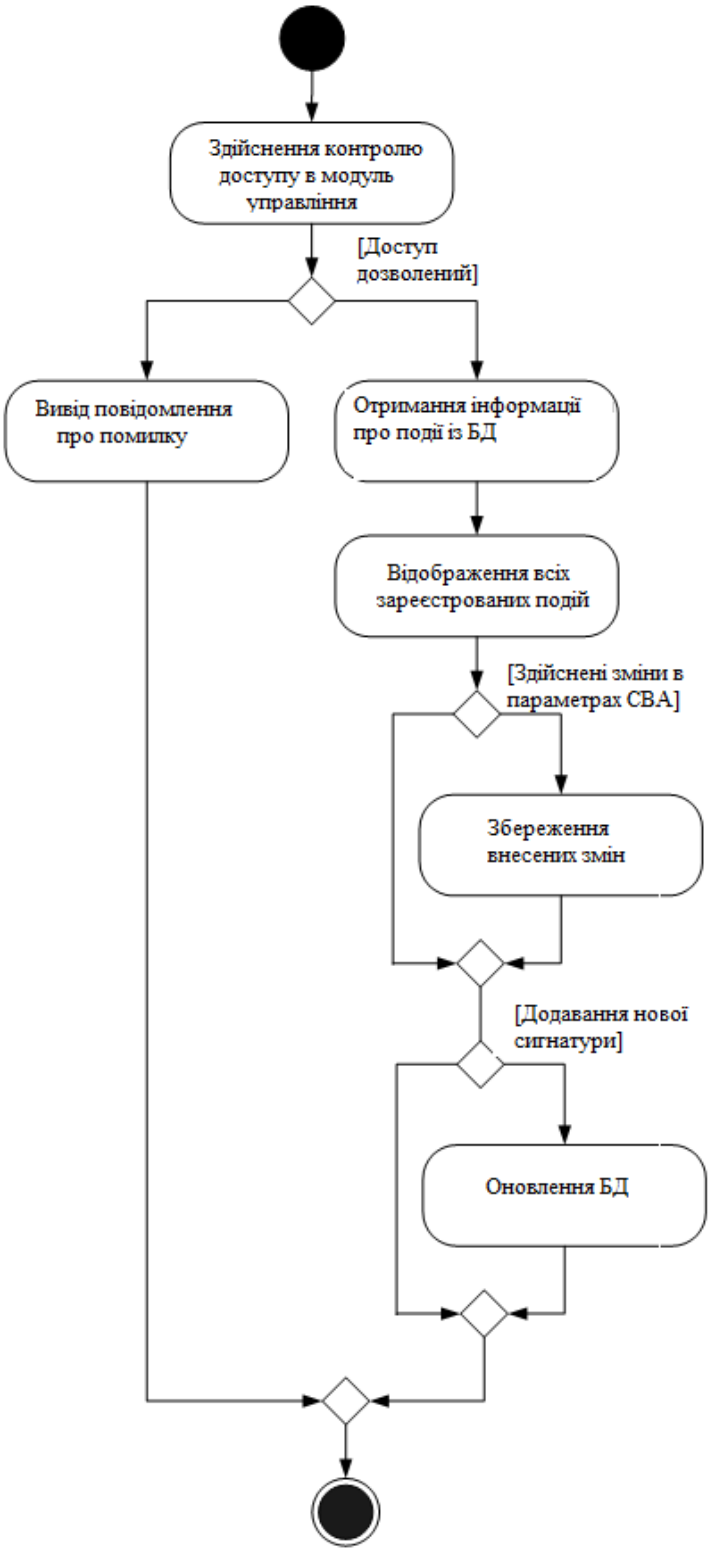


Рис. 3.9. Діаграма діяльності модуля управління СВА

Діаграма діяльності модуля управління системою виявлення атак зображена на рис. 3.10.

При розгляді діаграм діяльності було відзначено, що хоча ці діаграми і використовуються для специфікації динаміки поведінки системи, час в явному вигляді в них не присутня. Проте, тимчасової аспект поведінки може мати суттєве значення при моделюванні синхронних процесів, що описують взаємодії об'єктів. Для моделювання взаємодії об'єктів в часі в мові UML використовуються діаграми послідовності.

Діаграма послідовності - діаграма, на якій показані взаємодії об'єктів, впорядковані за часом їх прояву.

На діаграмі послідовності зображаються тільки ті об'єкти, які безпосередньо беруть участь у взаємодії. Ключовим моментом для діаграм послідовності є динаміка взаємодії об'єктів в часі.

Діаграма послідовності розробляється виявлення атак зображена рисунку 3.19.

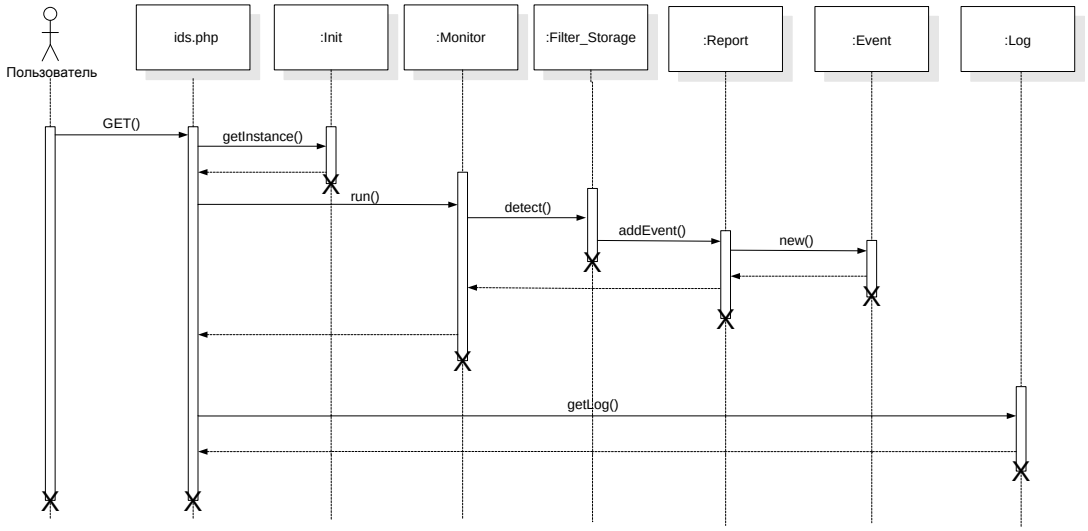


Рис. 3.10. Діаграма послідовності об'єктів СВА

В UML діаграма послідовності має два виміри. Перше зліва направо у вигляді вертикальних ліній, кожна з яких зображує лінію життя окремого об'єкта, який бере участь у взаємодії. Крайнім зліва на діаграмі зображений

об'єкт, який є ініціатором взаємодії. Праворуч розташовується інший об'єкт, який безпосередньо взаємодіє з першим. Таким чином, всі об'єкти на діаграмі послідовності утворюють деякий порядок, який визначається черговістю або ступенем активності об'єктів при взаємодії один з одним [30].

Графічно кожен об'єкт зображений прямокутником і розташовується у верхній частині своєї лінії життя. Усередині прямокутника записано ім'я об'єкта і ім'я класу, розділені двокрапкою. При цьому об'єкт може бути анонімним, в цьому випадку перед його ім'ям ставиться двокрапка без вказівки імені класу.

Другим виміром діаграми послідовності є вертикальна тимчасова вісь, спрямована зверху вниз. Початкового моменту часу відповідає сама верхня частина діаграми. Взаємодії об'єктів реалізуються за допомогою повідомлень, які надсилаються одними об'єктами іншим. Повідомлення зображені у вигляді горизонтальних стрілок з ім'ям повідомлення, а їх порядок визначається часом виникнення. Тобто, повідомлення, розташовані на діаграмі послідовності вище, ініціюються раніше тих, які розташовані нижче. Масштаб на осі часу не вказується, оскільки діаграма послідовності моделює лише тимчасову упорядкованість взаємодій.

Лінія життя об'єкту зображена пунктирною вертикальною лінією, асоційованою з єдиним об'єктом на діаграмі послідовності. Лінія життя служить для позначення періоду часу, протягом якого об'єкт існує в системі і, отже, може потенційно брати участь у всіх її взаємодіях.

Представлена діаграма послідовності дозволяє проаналізувати взаємодію об'єктів, які розташовуються у відповідних модулях СВА і які беруть участь в процесі роботи системи виявлення атак.

У висновку проектування можна зробити висновок, що створена архітектура системи, що розробляється виявлення атак на веб-додатки повністю задовольняє поставленим вимогам, тому наступним етапом є програмна реалізація даної системи.

Діаграма класів розробленої системи виявлення атак представлена на рис. 3.11.

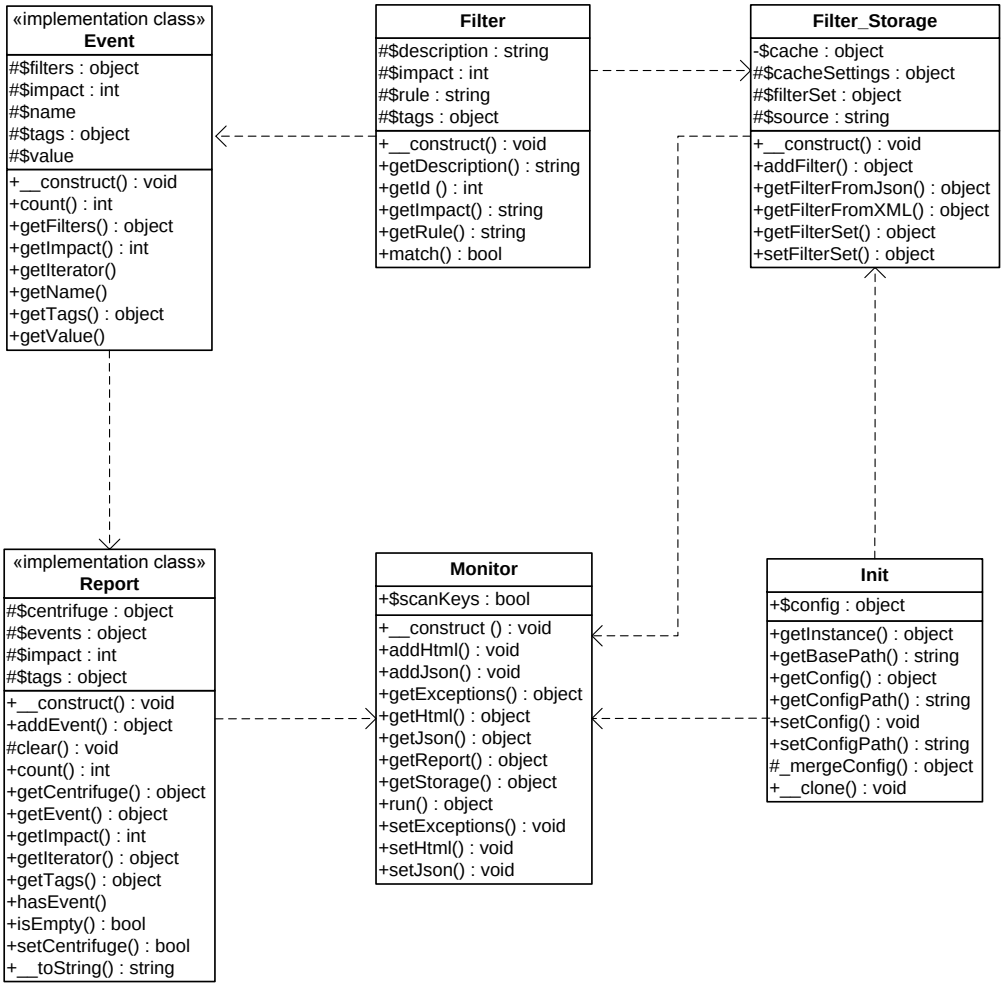


Рис. 3.11. Діаграма класів СВА

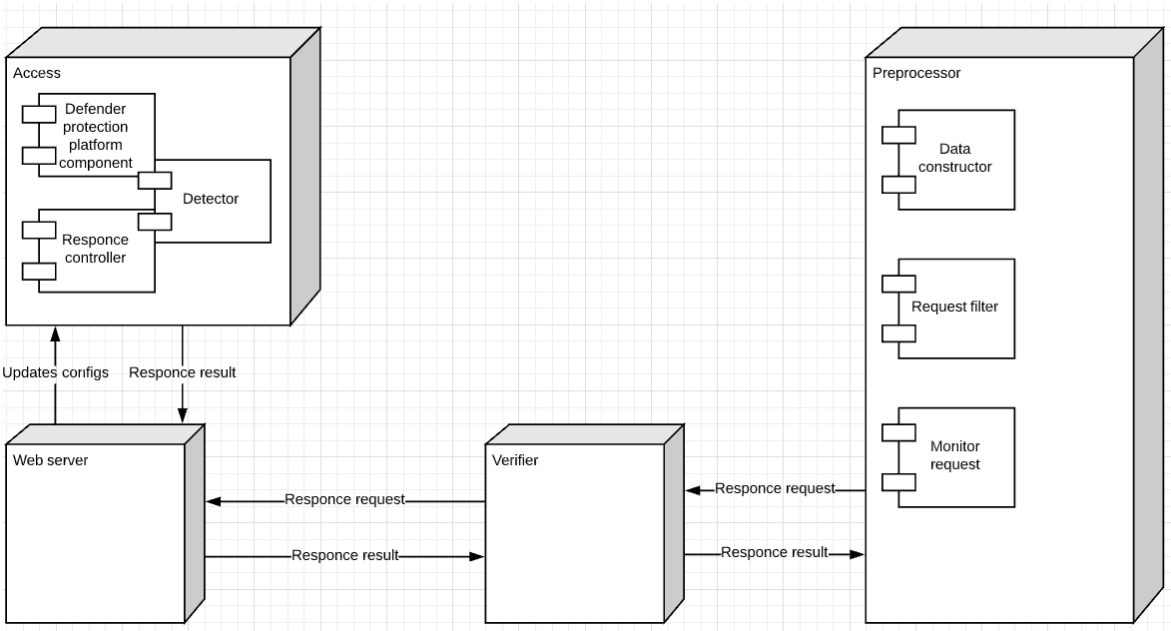


Рис. 3.12. Діаграма розгортання

При розробленні модуля спочатку був зроблений інтерфейс з інтуїтивно-зрозумілою формою для клієнтів, які будуть користуватися даним модулем (рис. 3.13). В результаті обрали оптимальні технології, які використовувалися при створенні веб-додатка.

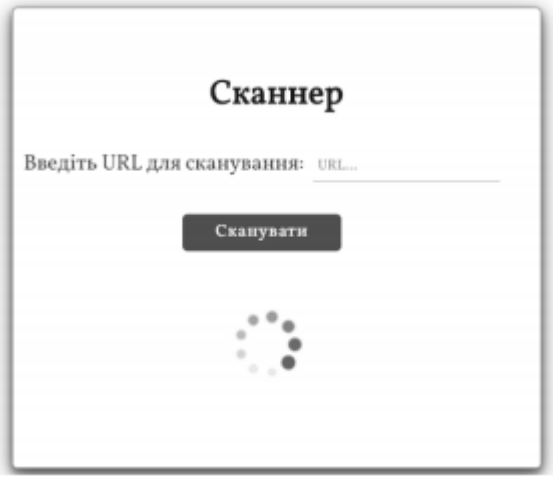


Рис. 3.13. Інтерфейс

Для створення інтерфейсу використовувались такі технології:

- Html
- Css
- VueJS
- Node.js

• Express

• PHP.

Відкривши додаток, користувач повинен знати, що робити без икористання інструкцій і довідок. Навігацію, розмір і стиль іконок потрібно проектувати так, щоб користувач не відчував незручність. Задоволення, яке отримує користувач при використанні програми - це важливий фактор. Якщо додаток інтерактивний і корисний, то до нього будуть повертатися знову. Оптимальність інтерфейсу і зручність навігації являється дуже важливою темою. Модуль має бути зручний для розуміння користувачам там має містити якомога зручніший інтерфейс. У веб-додатку використовувалась блокова верстка. Цей спосіб є найбільш вдалим для створення структури веб-додатка, яка здійснюється за допомогою контейнера (рис. 3.14).

```
<div class="container">
    <div class="scanner">
        <h1>Сканнер</h1>
        <label>Введіть URL для сканування:</label>
        <input type="text"  id="scanner-input" placeholder="URL..." required>
        <br>
        <button id="btn">Сканувати</button>

        <div id="result"></div>
    </div>
</div>
<script src="index.js"></script>
```

Рис 3.14. Використання блокової верстки

3.3. Синтезований алгоритм виявлення атак

Набір даних включає в себе такі атаки, як впровадження SQL, переповнення буфера, збір інформації, розкриття файлів, впровадження CRLF, міжсайтового виконання сценаріїв, підробку параметрів.

Процес тестування складається з двох фаз:

– фази навчання

– фази виявлення атак.

Фаза навчання складається з трьох модулів: модуль навчання dropout, модуль навчання back-propagation та модуль навчання за допомогою синтезованого методу.

Алгоритм треннування «Dropout»

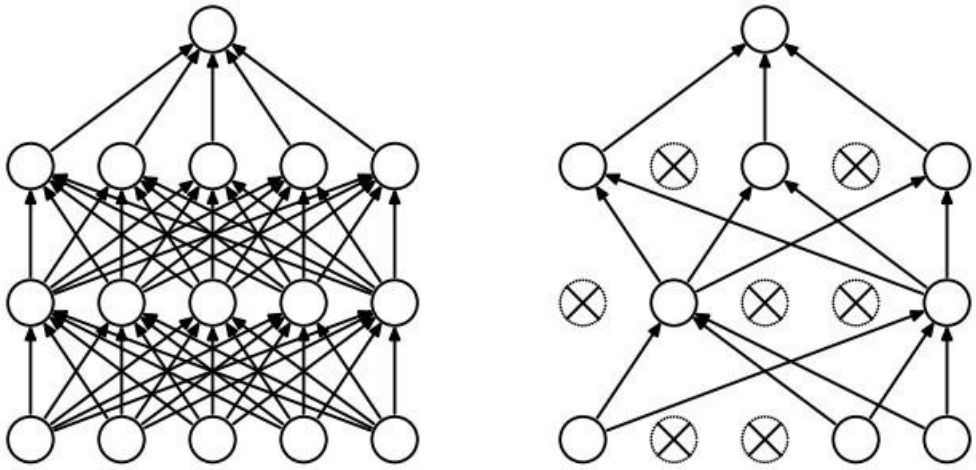


Рис. 3.15. Графічне представлення методу Dropout. Зліва - нейронна мережа до того, як до неї застосували Dropout, праворуч - та ж мережа після Dropout.

Глибокі нейронні мережі з великою кількістю параметрів - це дуже потужні системи машинного навчання. Однак перевиконання є серйозною проблемою в таких мережах. Великі мережі також є повільними у використанні, що ускладнює справу з надмірною обробкою, поєднуючи передбачення багатьох різних великих нейронних мереж у тестовий час. Dropout - це методика вирішення цієї проблеми.

Ключова ідея - випадкове скидання одиниць (разом із їх з'єднаннями) з нейронної мережі під час тренувань. Це не дозволяє занадто сильно адаптуватися. Під час тренінгу випадають зразки з експоненціальної кількості різних «стоншених» мереж. У тестовий час легко оцінити ефект усереднення прогнозів усіх цих стоншених мереж, просто використовуючи єдину неткану мережу, яка має меншу вагу. Це суттєво зменшує надмірний набір та дає значні покращення порівняно з іншими методами регуляризації. Показано, що випадання покращує продуктивність нейронних мереж у контрольованих навчальних завданнях із зору, розпізнавання мови,

класифікації документів та обчислювальної біології, одержуючи найсучасніші результати для багатьох наборів даних орієнтирів.

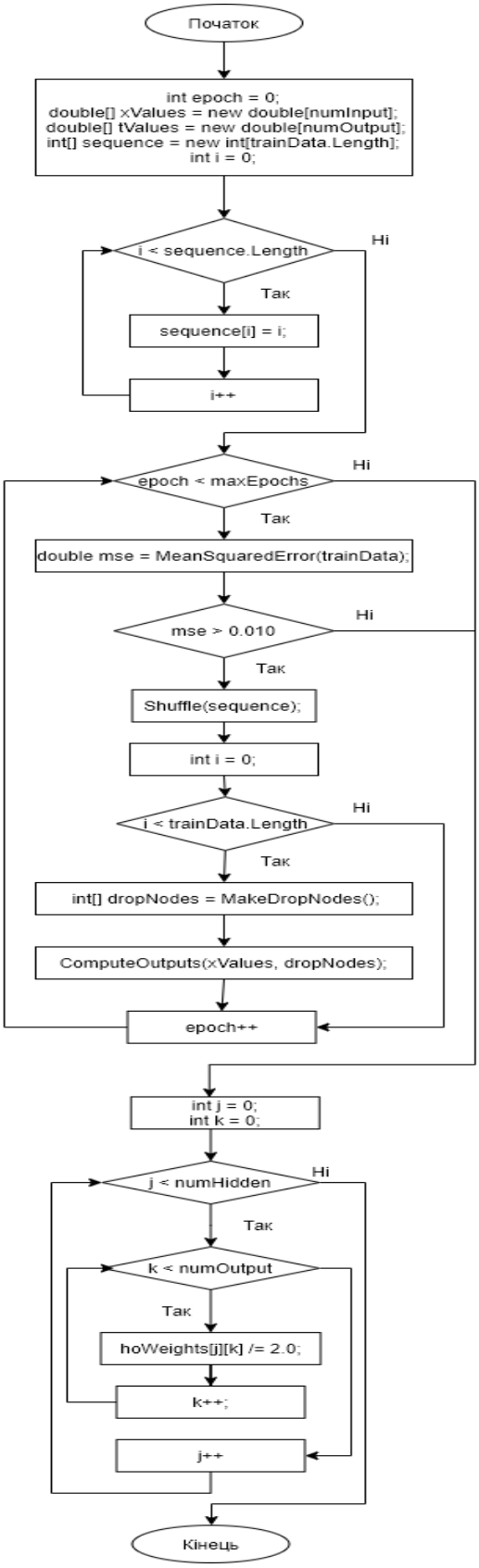
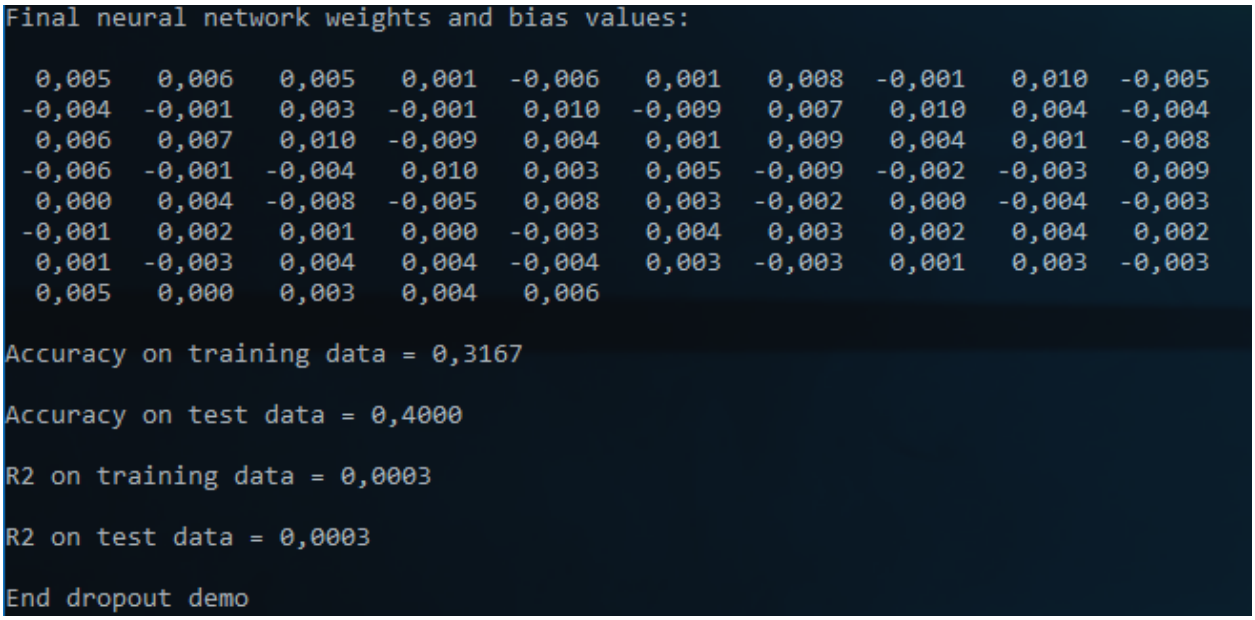


Рис. 3.15. Алгоритм тренування за допомогою dropout

Результат навчання мережі наведено нижче:



Алгоритм тренування «Back-Propagation»

Back-Propagation - це просто метод обчислення часткових похідних (або градієнта) функції, який має форму у вигляді функції функції (як у Нейронних мережах). Коли ви вирішуєте задачу оптимізації, використовуючи метод на основі градієнта (градієнтне спускання - лише один з них), ви хочете обчислити градієнт функції при кожній ітерації.

Для Нейронних мереж цільова функція має форму композиції. Як обчислити градієнт? Є два загальних способи це зробити: (i) аналітична диференціація. Ви знаєте форму функції. Ви просто обчислюєте похідні, використовуючи правило ланцюга (основне обчислення). (ii) Орієнтовна диференціація з використанням кінцевої різниці. Цей метод обчислювально дорогий, оскільки кількість оцінювання функції дорівнює O (N), де N - кількість параметрів. Це дорого, порівняно з аналітичною диференціацією. Кінцева різниця, однак, зазвичай використовується для перевірки реалізації резервної підтримки при налагодженні.

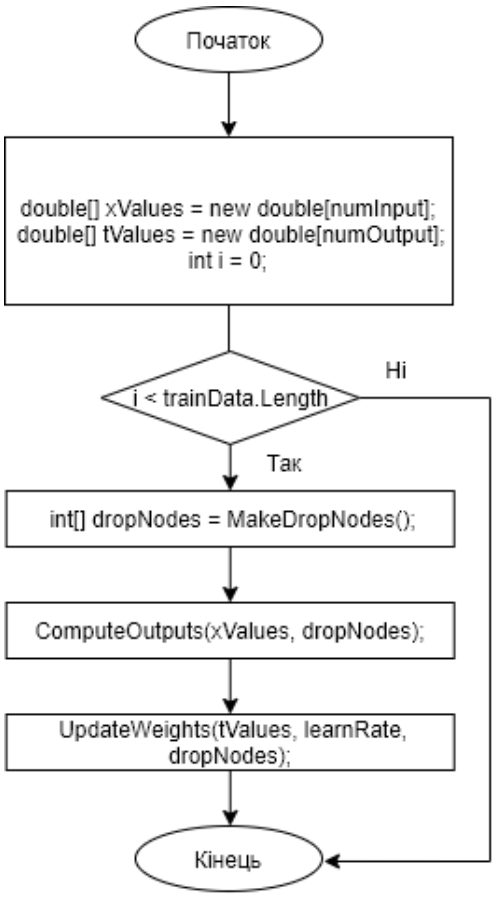


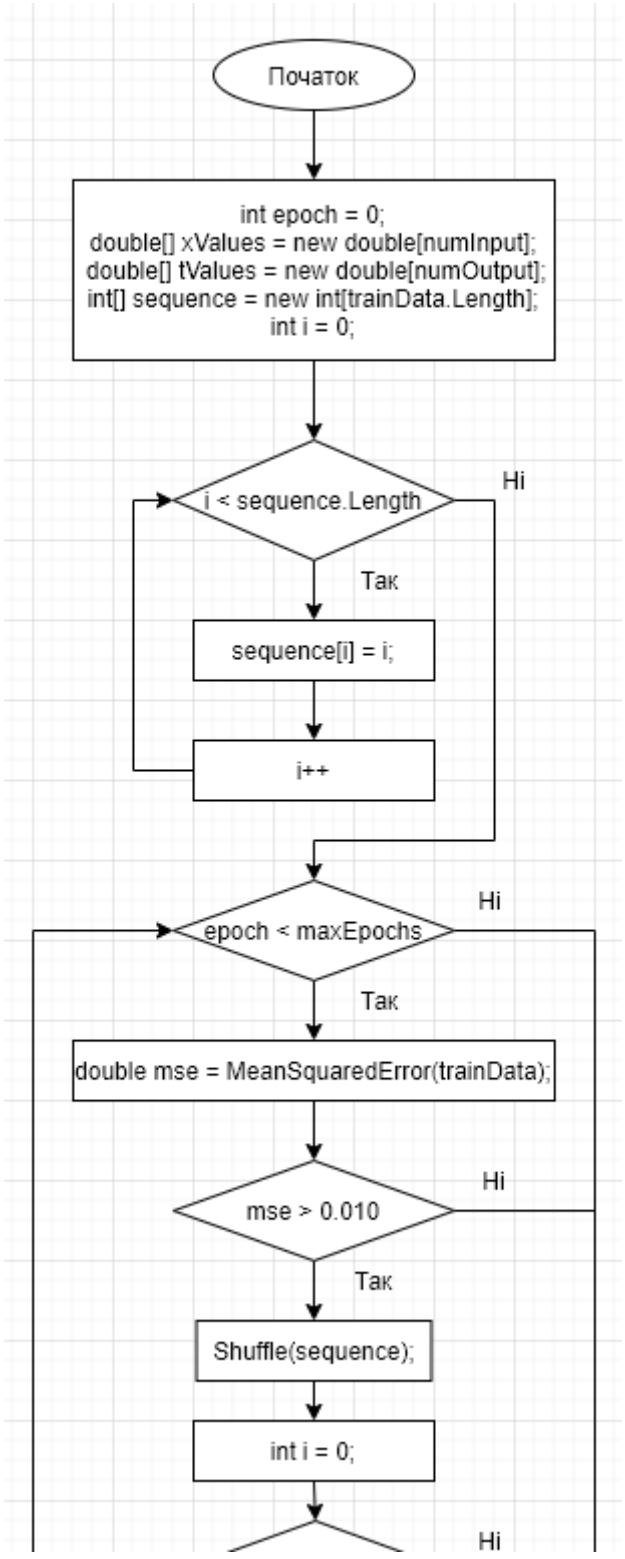
Рис. 3.26. Алгоритм тренування за допомогою dropout

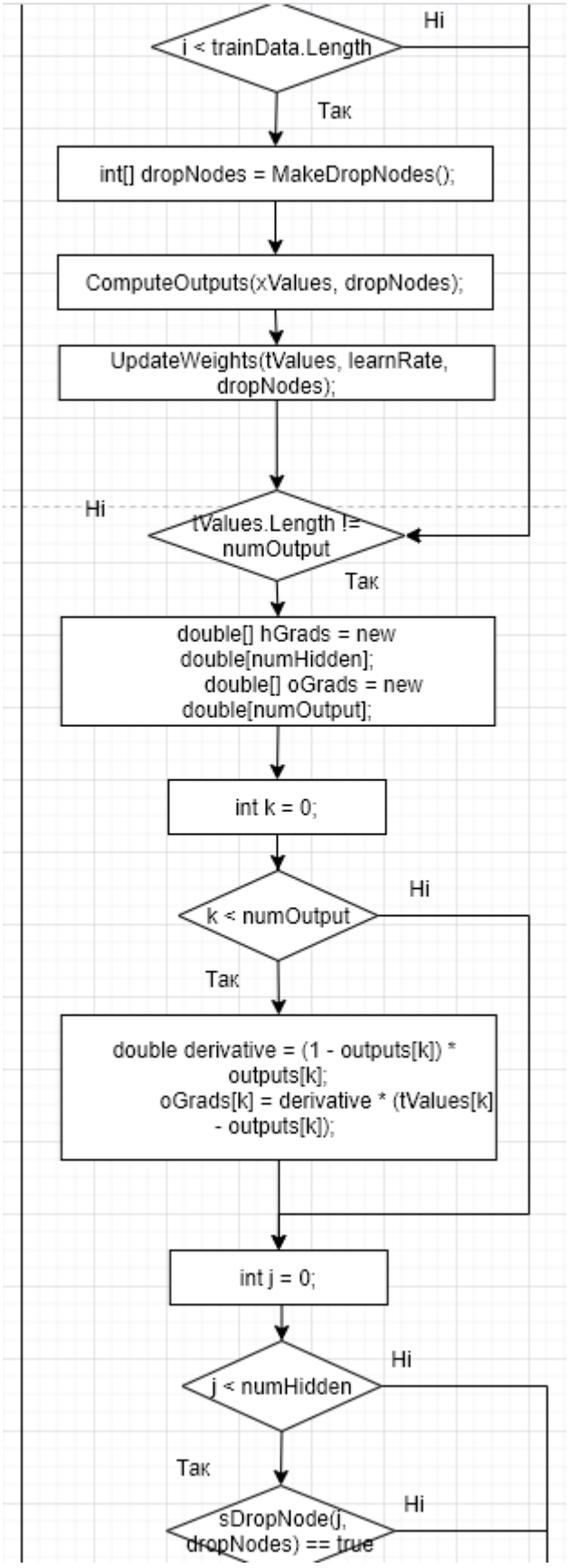
Результат

```
Final neural network weights and bias values:

  0,005   0,006   0,005   0,001  -0,006   0,001   0,008  -0,001   0,010  -0,005
 -0,004  -0,001   0,003  -0,001   0,010  -0,009   0,007   0,010   0,004  -0,004
  0,006   0,007   0,010  -0,009   0,004   0,001   0,009   0,004   0,001  -0,008
 -0,006  -0,001  -0,004   0,010   0,003   0,005  -0,011   0,001  -0,005   0,008
 -0,003   0,004  -0,010  -0,005   0,008   0,008  -0,001   0,001  -0,007  -0,005
 -0,002   0,005   0,003   0,000  -0,008   0,006   0,005   0,005   0,007   0,004
  0,002  -0,006   0,007   0,010  -0,006   0,009  -0,006   0,004   0,008  -0,007
  0,008  -0,002  -0,441  -0,440  -0,439

Accuracy on training data = 0,3417

Accuracy on test data = 0,4667

R2 on training data = 0,0009

R2 on test data = 0,0008
```

Синтез

Об’єднаємо обрані алгоритми.


Початок
int epoch = 0;
double[] xValues = new double[numInput];
double[] tValues = new double[numOutput];
int[] sequence = new int[trainData.Length];
int i = 0;
i < sequence.Length
Ні
Так
sequence[i] = i;
i++
epoch < maxEpochs
Ні
Так
double mse = MeanSquaredError(trainData);
mse > 0.010
Ні
Так
Shuffle(sequence);
int i = 0;
Ні


i < trainData.Length
Hi
Так
int[] dropNodes = MakeDropNodes();
ComputeOutputs(xValues, dropNodes);
UpdateWeights(tValues, learnRate, dropNodes);
Hi
tValues.Length != numOutput
Так
double[] hGrads = new double[numHidden];
double[] oGrads = new double[numOutput];
int k = 0;
k < numOutput
Hi
Так
double derivative = (1 - outputs[k]) * outputs[k];
oGrads[k] = derivative * (tValues[k] - outputs[k]);
int j = 0;
j < numHidden
Hi
Так
sDropNode(j, dropNodes) == true
Hi

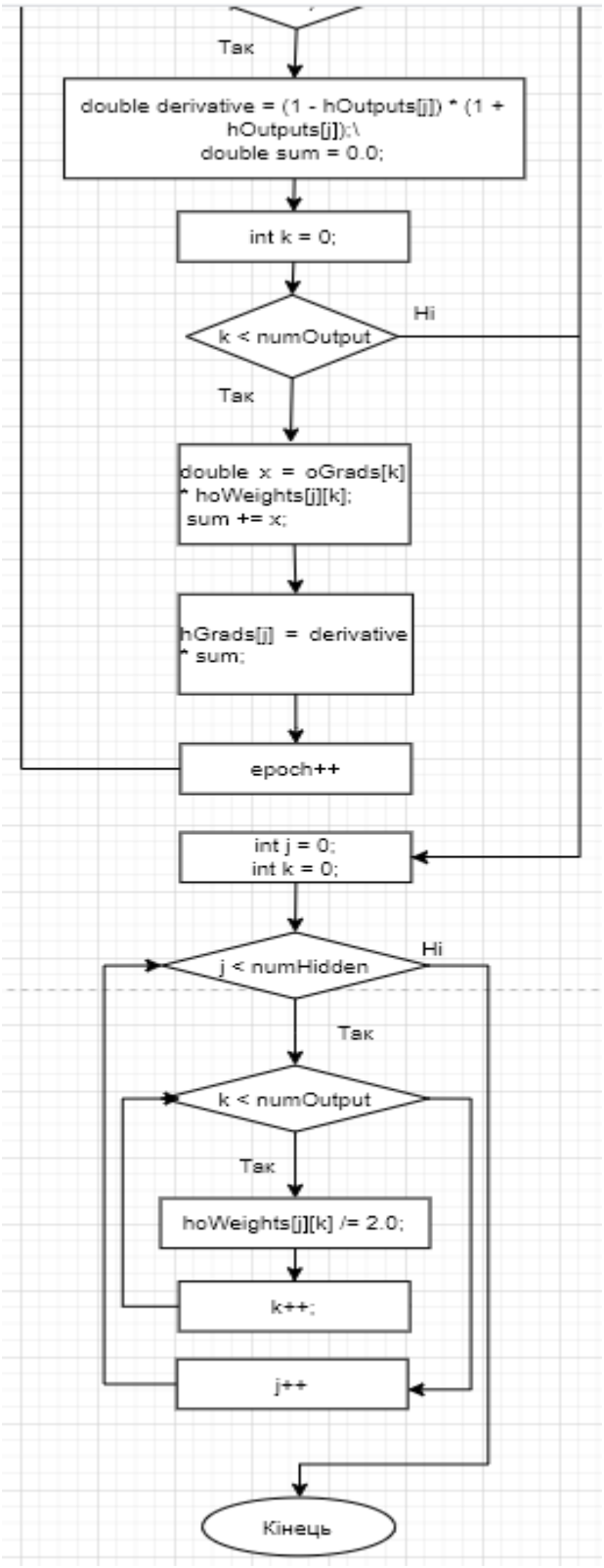


Рис. 3.27. Алгоритм тренування за допомогою синтезованого алгоритму

Результат:

```
Beginning training using back-propagation with dropout

Training complete

Final neural network weights and bias values:

 -1,938   1,761  -1,891   0,210  -0,249   1,907  -0,213   0,241  -1,884  -1,558
  1,563  -1,564   0,765  -0,781   1,692  -0,748   0,662  -1,633   2,890  -2,696
  2,782  -1,173   1,145  -2,967   1,155  -1,106   2,903   2,904  -2,668   2,947
 -0,588   0,548  -2,949   0,581  -0,690   2,910  -2,488   2,199  -2,403   0,163
 -0,150   2,456  -0,161   0,199  -2,417   0,872  -0,406  -0,550  -0,862   0,366
  0,530   0,875  -0,322  -0,597  -0,713  -0,682   1,035   0,743   0,672  -1,047
 -0,868   0,357   0,632   0,675   0,736  -1,053  -0,660  -0,684   1,084   0,877
 -0,376  -0,556   1,779   1,784   1,197

Accuracy on training data = 0,9833

Accuracy on test data = 0,9667

R2 on training data = 0,9997

R2 on test data = 1,0000
```

Метрики для порівняння

Точність (Accurancy) - це достовірне прогнозоване значення з підтримкою перевірених даних набору. Це число, що відповідає правильному і загальнодоступному числу вхідних даних. Ця метрика працює добре, якщо існує аналогічне число вибірок, приналежних до кожного класу. Чим ближче до 1,00, тим вище якість.

R-квадрат (R2) або коефіцієнт детермінації означає сукупну що прогнозує здатність моделі в діапазоні від -inf до 1,00. 1,00 означає, що є ідеальний збіг, але збіг може бути довільно поганим, тому оцінки можуть бути негативними. Оцінка 0,00 означає, що модель прогнозує очікуване значення для мітки. R2 вимірює, наскільки реальні значення даних близькі до прогнозованих. Чим ближче до 1,00, тим вище якість.

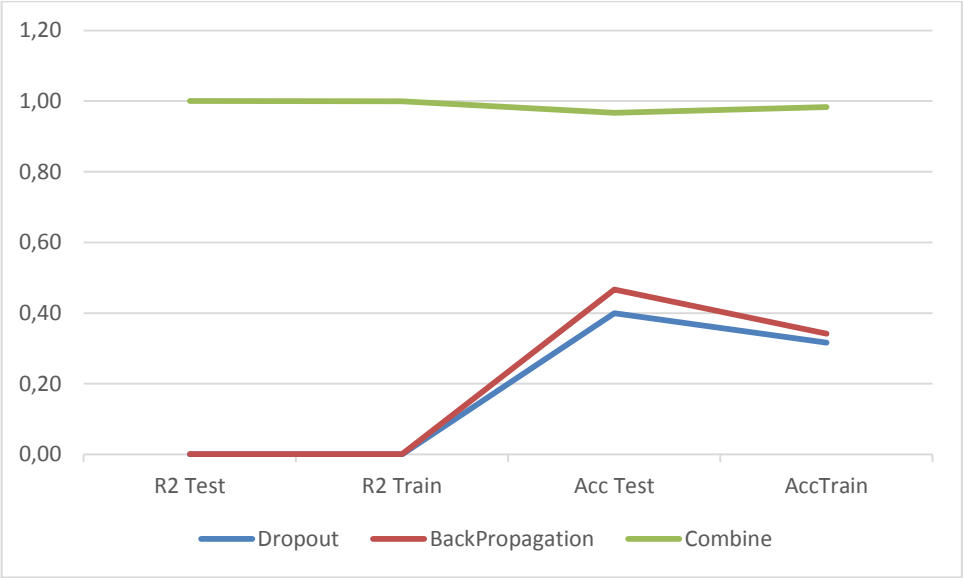


Рис. 3.19. Точність та коефіцієнт детермінації алгоритмів тренування

3.4. Рекомендації для користувача по використанню програмного засобу

Програмний засіб «Web Attack Detector» призначена для виявлення атак на веб-додатки, що дає можливість для автоматизації процесу виявлення та моніторингу веб-атак.

Запуск програми в ОС сімейства Windows здійснюється запуском виконуючого файлу WebAttackDetector.exe в робочій директорії програми.

Після запуску програми на екрані з’являється головне вікно програмного засобу (рис. 3.20).

При відкритті головного вікна програми з’являються 3 вкладення: «Program», «Settings», «About».

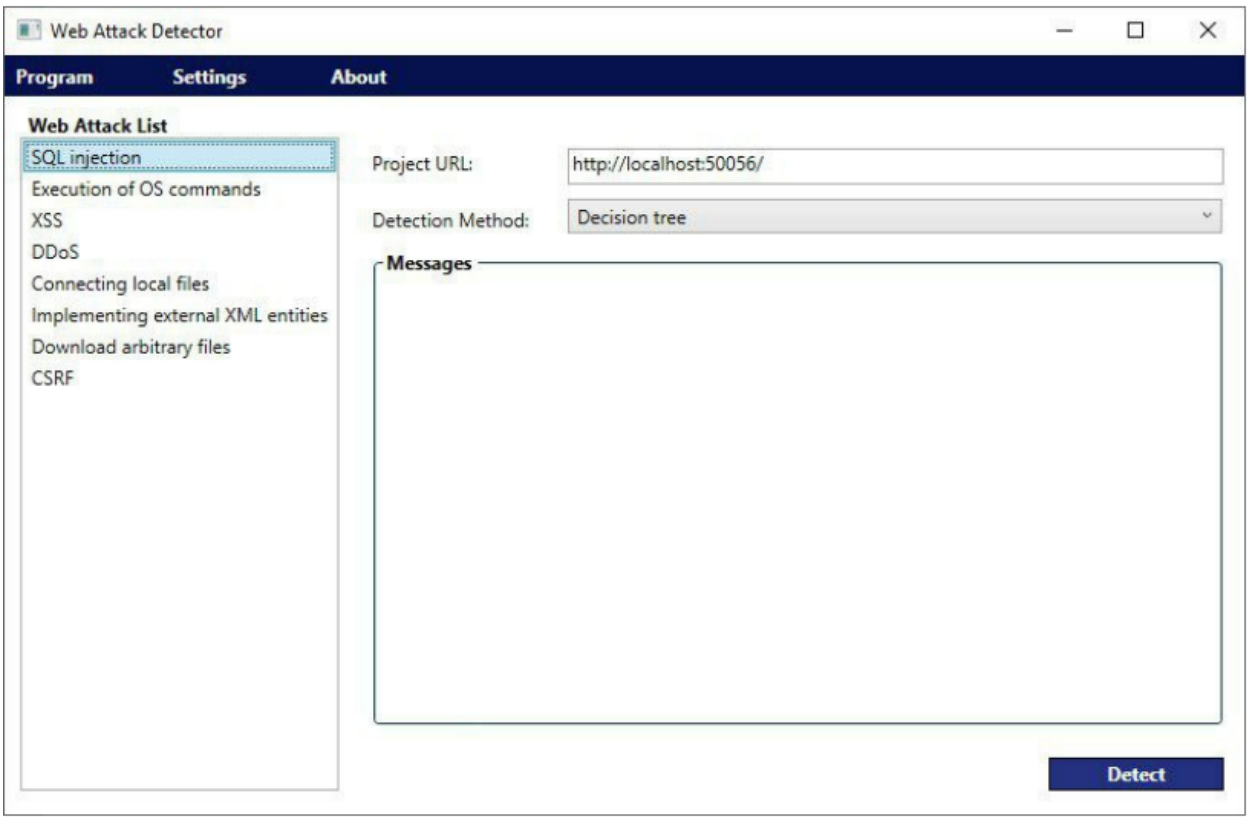


Рис. 3.20 Головне вікно програми

При натисненні на вкладення «Program» відкривається вікно із вкладенням «Web Attack List», яка має наступні можливості: поле для вибору посилання на веб-ресурс, який підлягає дослідженню, вікно із переліком веб-атак, які можуть дослджуватися, вікно із вибором методу, який буде використовуватися для вичвлення атак та вікно для результатів виявлення атак.

При переході на вкладку «Setting» можна переглянути збереження результатів, можна налаштувати тривалість аналізу в секундах тощо (рис. 3.21).

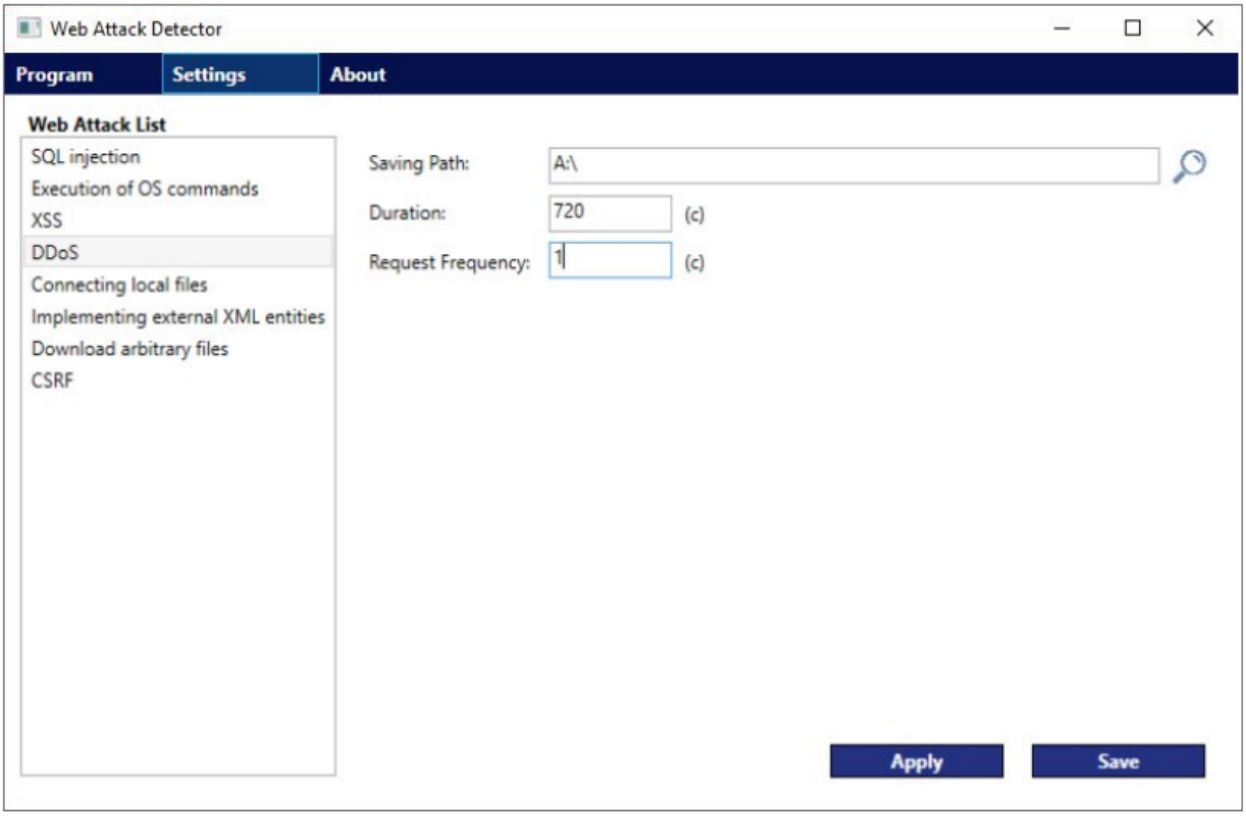


Рис. 3.21. Вікно «Settings»

При переході на вкладку «About» можна знайти інформацію, необхідну для користувача щодо видів веб-атак та методів, які використовуються при знаходження веб-атак.

Висновки по розділу

В даному розділі описана типова структура веб-додатку, а також розглянуті методи виявлення атак на веб-додатки на основі нейронних мереж, в результаті чого був розроблений комбінований метод для виявлення атак.

ВИСНОВКИ

Метою даної дипломної роботи була розробка програмного модуля для виявлення атак на веб-додатки. Для досягнення поставленої мети було розв’язано наступні задачі: дослідити предметну область, що включає в себе аналіз видів атак на веб-додатки, дослідження проблем виявлення та захисту веб-додатків, розгляд стандартних засобів захисту веб-додатків, а також аналіз програмних засобів для виявлення атак на веб-додатки; дослідити існуючі методи захисту веб-додатків; розробити метод виявлення атак на веб-додатки, що є комбінацією двох вже відомих методів; розробити програмний модуль, який реалізує запропонований метод. Розроблений програмний модуль базується на використанні комбінованого алгоритму виявлення мережевих атак на основі комбінації методу dropuot та back-propagation. Був реалізован веб-додаток, який дає змогу тестувати модуль для захисту інформації. Були розглянуті нормативно-правові документи із захисту інформації, було проведено аналіз вразливостей вебдодатків. Проаналізувавши існуючі методи шифрування, можна зробити висновок , що для реалізації шифрування модулю підходить метод RSA. Для розробки програмного продукту був проведений аналіз технологій , які найбільше підійдуть для розробки та захисту повідомлень в веб-додатку. Розробка відбувалась на основі технологій JavaScript, NodeJS та на платформі VueJS , HTML, CSS, Express, PHP.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Sankar R. Burpsuite – A Beginner’s Guide For Web Application Security or Penetration Testing [Електронний ресурс] / Ravi Sankar. – 2018. – Режим доступа: World Wide Web. — URL: https://kalilinuxtutorials.com/burpsuite/

2. 2. Ricca F. https://dl.acm.org/citation.cfm?id=381476 [Електронний ресурс] / F. Ricca, P. Tonella. – 2001. – Режим доступа: World Wide Web. — URL: https://dl.acm.org/citation.cfm?id=3814763.

3. K. Fu, E. Sit, K. Smith, and N. Feamster. Dos and Don’ts of Client Authentication on the Web. – 2011. - Режим доступа: World Wide Web. — URL: https://pdos.csail.mit.edu/papers/webauth:sec10.pdf.

4. J. Bercegay. Double Choco Latte Vulnerabilities. http://www.gulftech.org/ ?node=research&article id=00066-04082005, April 2005.

5. N. Jovanovic. TxtForum: Script Injection Vulnerability. http://www.seclab. tuwien.ac.at/advisories/TUVSA-0603-004.txt, March 2006.

6. A. Klein. Cross Site Scripting Explained. Technical report, Sanctum Inc., 2002

7. Скляров, Д.В. Искусство защиты и взлома информации / Д.В. Скляров. — СПб. : БХВ-Петербург, 2004. — 36 с.

8. Іванов М. А., Зенін О. С., Стандарт криптографічного захисту – AES. Кінцеві поля. М.: КУДИЦ – ОБРАЗ, 2003. 171 с.

9. New types of cryptanalytic attacks using related keys./ Biham E. // Advances in Criptology, Proceedings Eurocrypt’93, LNCS 765, T. Helleseth, Ed., Springer – Verlag, 1993 – P. 398 – 409.

10. Столінгс В. криптографія і захист мереж: принципи і практика. М.: видавничий дім «Вильямс», 2001. 677 с.

11. Web Application Risk – the Threat of and Solution to Sensitive Data Exposure [Електронний ресурс] – Режим доступу до ресурсу: https://www.immuniweb.com/blog/OWASP-sensitive-data-exposure.html.

12. Top 10-2017 A6-Security Misconfiguration [Електронний ресурс]. – 2017. – Режим доступу до ресурсу: https://www.owasp.org/index.php/Top_10-2017_A6- Security_Misconfiguration.

13. Authentication Hacking: What are Authentication Hacking Attacks? [Електронний ресурс]. – 2014. – Режим доступу до ресурсу: https://www.acunetix.com/websitesecurity/authentication/.

14. Charan H. Broken Authentication and Session Management—part Ⅰ [Електронний ресурс] / Hari Charan. – 2017. – Режим доступу до ресурсу: 75 https://medium.com/@grep_security/broken-authentication-and-sessionmanagementpart- ⅰ -50e760c9f599.

15. Testing for CSRF (OTG-SESS-005) [Електронний ресурс]. – 2019. – Режим доступу до ресурсу: https://www.owasp.org/index.php/Testing_for_CSRF_(OTG-SESS-005).

16. Cross Site Scripting (XSS) Attack Tutorial With Examples, Types & Prevention [Електронний ресурс]. – 2019. – Режим доступу до ресурсу: https://www.softwaretestinghelp.com/cross-site-scripting-xss-attack-test/.

17. Методы защиты от CSRF-атаки [Електронний ресурс]. – 2016. – Режим доступу до ресурсу: https://habr.com/ru/post/318748/.

18. Cross-Site Request Forgery (CSRF) [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://www.owasp.org/index.php/CrossSite_Request_Forgery_(CSRF).

19. SQL_Injection_Prevention_Cheat_Sheet [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/SQL_Injection _Prevention_Cheat_Sheet.md.

20. How to Prevent SQL Injection Attacks [Електронний ресурс] // eSecurityPlanet. – 2018. – Режим доступу до ресурсу: https://www.esecurityplanet.com/threats/how-to-prevent-sql-injection-attacks.html.

21. Уязвимости веб-приложений [Електронний ресурс]. – 2019. – Режим доступу до ресурсу: https://www.ptsecurity.com/upload/corporate/ruru/analytics/Web-Vulnerabilities2019-rus.pdf.

22. Web Server Vulnerabilities Attacks: How to Protect Your Organization [Електронний ресурс] // Tech Funnel. – 2018. – Режим доступу до ресурсу: https://www.techfunnel.com/information-technology/web-servervulnerabilitiesattacks-how-to-protect-your-organization/

ДОДАТОК А

Реалізація алгоритму dropout

```
public void Train(double[][] trainData, int maxEpochs, double learnRate)
      {
         // train a back-prop style NN classifier using dropout
         // no momentum or weight decay
         int epoch = 0;
         double[] xValues = new double[numInput]; // inputs
         double[] tValues = new double[numOutput]; // target values

         int[] sequence = new int[trainData.Length];
         for (int i = 0; i < sequence.Length; ++i)
            sequence[i] = i;

         while (epoch < maxEpochs)
         {
            // MSE early exit
            //double mse = MeanSquaredError(trainData); // expensive! consider
only every k epochs
            //if (mse < 0.010) break; // consider passing value in as parameter

            Shuffle(sequence); // visit each training data in random order
            for (int i = 0; i < trainData.Length; ++i)
            {
               int idx = sequence[i];
               Array.Copy(trainData[idx], xValues, numInput); // more flexible
might be a 'GetInputsAndTargets()'
               Array.Copy(trainData[idx], numInput, tValues, 0, numOutput);
               int[] dropNodes = MakeDropNodes();
               ComputeOutputs(xValues, dropNodes); // copy xValues in, compute
outputs (and store them internally)
            } // each training tuple
            ++epoch;
         }

         // divide hidden-output weights by 2.0 to account for dropout
         for (int j = 0; j < numHidden; ++j)
            for (int k = 0; k < numOutput; ++k)
               hoWeights[j][k] /= 2.0;
          }
```

Додаток Б

Реалізація алгоритму back-propagation:

```
    public void Train(double[][] trainData, int maxEpochs, double learnRate)
{
   // train a back-prop style NN classifier using dropout
   // no momentum or weight decay

   double[] xValues = new double[numInput]; // inputs
  double[] tValues = new double[numOutput]; // target values

      for (int i = 0; i < trainData.Length; ++i)
      {
         int[] dropNodes = MakeDropNodes();
         ComputeOutputs(xValues, dropNodes); /
         UpdateWeights(tValues, learnRate, dropNodes);
         }
}
```

ДОДАТОК В

Реалізація синтезованого алгоритму:

```
  private void UpdateWeights(double[] tValues, double learnRate, int[]
dropNodes)
  {
    // update the weights and biases using back-propagation
    // assumes that SetWeights and ComputeOutputs have been called
    if (tValues.Length != numOutput)
      throw new Exception("target values not same Length as output in
UpdateWeights");

    // back-prop related arrays. could be class members to avoid millions of
allocations
    double[] hGrads = new double[numHidden];
    double[] oGrads = new double[numOutput];

    // 1. compute output gradients
    for (int k = 0; k < numOutput; ++k)
    {
      // implicit MSE
      double derivative = (1 - outputs[k]) * outputs[k]; // derivative of softmax =
(1 - y) * y (same as log-sigmoid)
      oGrads[k] = derivative * (tValues[k] - outputs[k]); // 'mean squared error
version' includes (1-y)(y) derivative
    }

    // 2. compute hidden gradients
    for (int j = 0; j < numHidden; ++j)
```

```
    {
      if (IsDropNode(j, dropNodes) == true) continue;
      double derivative = (1 - hOutputs[j]) * (1 + hOutputs[j]); // derivative of
tanh = (1 - y) * (1 + y)
      double sum = 0.0;
      for (int k = 0; k < numOutput; ++k) // each hidden delta is the sum of
numOutput terms
      {
        double x = oGrads[k] * hoWeights[j][k];
        sum += x;
      }
      hGrads[j] = derivative * sum;
    }

    // 3. update input-hidden weights and hidden biases
    // combined for processing efficiency at expense of clarity
    for (int j = 0; j < numHidden; ++j)
    {
      if (IsDropNode(j, dropNodes) == true) continue;
      for (int i = 0; i < numInput; ++i)
      {
        double delta = learnRate * hGrads[j] * inputs[i]; // compute the new delta
        ihWeights[i][j] += delta; // update. note we use '+' instead of '-'. this can
be very tricky.
      }
      double biasDelta = learnRate * hGrads[j] * 1.0; // the 1.0 is the constant
input for any bias; could leave out
      hBiases[j] += biasDelta;
    }
```

```
      //// 4b. update output biases
      //for (int k = 0; k < numOutput; ++k)
      //{
      //  double delta = learnRate * oGrads[k] * 1.0;
      //  oBiases[k] += delta;
      //}

      // 4. update hidden-output weights and output biases
      // combined for processing efficiency at expense of clarity
      for (int k = 0; k < numOutput; ++k)
      {
        for (int j = 0; j < numHidden; ++j)
        {
          if (IsDropNode(j, dropNodes) == true) continue;
          double delta = learnRate * oGrads[k] * hOutputs[j];  // see above:
hOutputs are inputs to the nn outputs
          hoWeights[j][k] += delta;
        }
        double biasDelta = learnRate * oGrads[k] * 1.0;
        oBiases[k] += biasDelta;
      }

    } // UpdateWeights

    // ------------------------------------------------------------------------------

    public void Train(double[][] trainData, int maxEpochs, double learnRate)
    {
```

```
    // train a back-prop style NN classifier using dropout
    // no momentum or weight decay
    int epoch = 0;
    double[] xValues = new double[numInput]; // inputs
    double[] tValues = new double[numOutput]; // target values

    int[] sequence = new int[trainData.Length];
    for (int i = 0; i < sequence.Length; ++i)
      sequence[i] = i;

    while (epoch < maxEpochs)
    {
      // MSE early exit
      //double mse = MeanSquaredError(trainData); // expensive! consider only every k epochs
      //if (mse < 0.010) break; // consider passing value in as parameter

      Shuffle(sequence); // visit each training data in random order
      for (int i = 0; i < trainData.Length; ++i)
      {
        int idx = sequence[i];
        Array.Copy(trainData[idx], xValues, numInput); // more flexible might be a 'GetInputsAndTargets()'
        Array.Copy(trainData[idx], numInput, tValues, 0, numOutput);
        int[] dropNodes = MakeDropNodes();
        ComputeOutputs(xValues, dropNodes); // copy xValues in, compute outputs (and store them internally)
        UpdateWeights(tValues, learnRate, dropNodes);
      } // each training tuple
```

```
      ++epoch;
    }

    // divide hidden-output weights by 2.0 to account for dropout
    for (int j = 0; j < numHidden; ++j)
      for (int k = 0; k < numOutput; ++k)
        hoWeights[j][k] /= 2.0;
  } // Train

  private static void Shuffle(int[] sequence)
  {
    for (int i = 0; i < sequence.Length; ++i)
    {
      int r = rnd.Next(i, sequence.Length);
      int tmp = sequence[r];
      sequence[r] = sequence[i];
      sequence[i] = tmp;
    }
  }

  private double MeanSquaredError(double[][] trainData) // used as a training
stopping condition
  {
    // average squared error per training tuple
    double sumSquaredError = 0.0;
    double[] xValues = new double[numInput]; // first numInput values in
trainData
    double[] tValues = new double[numOutput]; // last numOutput values
```

```
    for (int i = 0; i < trainData.Length; ++i) // looks like (6.9 3.2 5.7 2.3) (0 0 1) (no parens)
    {
      Array.Copy(trainData[i], xValues, numInput); // get xValues. assumes in first columns!
      Array.Copy(trainData[i], numInput, tValues, 0, numOutput); // get target values

      double[] yValues = this.ComputeOutputs(xValues, null); // using current weights (no drop-nodes)
      for (int j = 0; j < numOutput; ++j)
      {
        double err = tValues[j] - yValues[j];
        sumSquaredError += err * err;
      }
    }
    return sumSquaredError / trainData.Length;
  }

  public double Accuracy(double[][] testData)
  {
    // percentage correct using winner-takes all
    int numCorrect = 0;
    int numWrong = 0;
    double[] xValues = new double[numInput]; // inputs
    double[] tValues = new double[numOutput]; // targets
    double[] yValues; // computed Y

    for (int i = 0; i < testData.Length; ++i)
```

```
    {
      Array.Copy(testData[i], xValues, numInput); // parse test data into x-values and t-values
      Array.Copy(testData[i], numInput, tValues, 0, numOutput);
      yValues = this.ComputeOutputs(xValues, null);  // null == don't use any drop-nodes
      int maxIndex = MaxIndex(yValues); // which cell in yValues has largest value?

      if (tValues[maxIndex] == 1.0) // ugly. consider AreEqual(double x, double y)
        ++numCorrect;
      else
        ++numWrong;
    }
    return (numCorrect * 1.0) / (numCorrect + numWrong); // ugly 2 - check for divide by zero
  }
  private static int MaxIndex(double[] vector) // helper for Accuracy()
  {
    // index of largest value
    int bigIndex = 0;
    double biggestVal = vector[0];
    for (int i = 0; i < vector.Length; ++i)
    {
      if (vector[i] > biggestVal)
      {
        biggestVal = vector[i]; bigIndex = i;
      }
```

```
      }  return bigIndex;
    }
} // class NeuralNetwork
}
```