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

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри Комп'ютеризованих систем захисту інформації

_____ Михайло СТЕПАНОВ
«____»_____ 2023 р.

На правах рукопису
УДК 004.056.5:510.22(043.3)

КВАЛІФІКАЦІЙНА РОБОТА

ЗДОБУВАЧА ВИЩОЇ ОСВІТИ
ОСВІТНЬОГО СТУПЕНЯ «МАГІСТР»

Тема: Система виявлення та запобігання SQL-ін'єкцій

Виконавець: Дмитро КРУК

Керівник: к.т.н., доцент Лілія ГАЛАТА

Консультант розділу «Охорона навколишнього середовища»: к.т.н., доцент Тетяна ДМИТРУХА

Нормоконтролер: к.т.н., доцент Лілія ГАЛАТА

Київ 2023

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет: Кібербезпеки та програмної інженерії

Кафедра: Комп'ютеризованих систем захисту інформації

Освітній ступінь: Магістр

Спеціальність: 125 «Кібербезпека»

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

ЗАТВЕРДЖУЮ
Завідувач кафедри Комп'ютеризованих систем захисту інформації

_____ Михайло СТЕПАНОВ
«__» _____ 2023 р.

ЗАВДАННЯ

на виконання кваліфікаційної роботи

здобувача вищої освіти Крука Дмитра Вікторовича

1. Тема: *Система виявлення та запобігання SQL-ін'єкцій*
затверджена наказом ректора від «15» вересня 2023 р. № 1814/ст.
2. Термін виконання: з 16.10.2023 р. по 22.12.2023 р.
3. Вихідні дані: проаналізувати існуючі системи виявлення та запобігання SQL-ін'єкцій; на основі аналізу виділити вхідні і вихідні параметри, завдяки яким можливо провести порівняння існуючих систем, виявлення їх переваг і недоліків; розробити методику, алгоритм та програмне забезпечення системи виявлення та запобігання.
4. Зміст пояснювальної записки: аналіз існуючих систем виявлення та запобігання SQL-ін'єкцій; розробка системи виявлення та запобігання; розробка програмного забезпечення запропонованої системи, верифікація отриманих результатів.

5. КАЛЕНДАРНИЙ ПЛАН

виконання кваліфікаційної роботи

№ з/п	Етапи виконання кваліфікаційної роботи	Термін виконання етапів	Примітка
1.	Уточнення постановки задачі	16.10.2023	*Виконано*
2.	Аналіз літературних джерел	17.10.2023	*Виконано*
3.	Обґрунтування вибору рішення	20.10.2023	*Виконано*
4.	Збір інформації	28.10.2023	*Виконано*
5.	Дослідження сучасних систем і методик виявлення SQL-ін’єкцій	05.11.2023	*Виконано*
6.	Розробка методики та структури системи виявлення SQL-ін’єкцій	14.11.2023	*Виконано*
7.	Розробка алгоритму та програмного забезпечення системи виявлення та запобіганню SQL-ін’єкцій	21.11.2023	*Виконано*
8.	Оформлення і друк пояснювальної записки	14.12.2023	*Виконано*
9.	Оформлення презентації	17.12.2022	*Виконано*
10.	Отримання рецензій від рецензента	22.12.2022	*Виконано*

6. Консультанти з окремих розділів

Розділ	Консультант (посада, П.І.Б.)	Дата, підпис	
		Завдання видав	Завдання прийняв
Охорона навколишнього середовища	Дмитруха Т.І.		

7. Дата видачі завдання: «16» жовтня 2023 р.

Здобувач вищої освіти Дмитро КРУК
(підпис, дата)

Керівник кваліфікаційної роботи Лілія ГАЛАТА
(підпис, дата)

РЕФЕРАТ

Кваліфікаційна робота на тему: «Система виявлення та запобігання SQL-ін’єкцій» складається зі вступу, основної частини, що містить 4 розділи, 3 висновки, загального висновку, 2 додатки та списку використаних джерел. Загальний обсяг роботи 103 – сторінок. Робота містить 22 рисунок та 4 таблиці. Список використаних джерел включає 17 джерел.

У кваліфікаційній роботі розглянуті питання щодо сучасних методів виявлення вразливостей SQL-ін’єкцій.

Проведені дослідження базуються на сучасних методах виявлення SQL-ін’єкцій в вразливих веб-додатак, а також методах застосування машиного навчання для виявлення вразливостей в SQL запитах.

Реалізація власного методу застосування системи виявлення та запобігання SQL-ін’єкцій та вдосконалення методу виявлення та запобігання в веб-додатках дозволяє показати адекватність її роботи та доцільність впровадження її організаціям.

Запропоновані методи дозволяють забезпечити швидкий та надійний захист веб-додатків а також баз даних від SQL-ін’єкцій.

Ключові слова: МАШИНЕ НАВЧАННЯ, SQLMAP, ЗАХИСТ, ВИЯВЛЕННЯ, ЗАПОБІГАННЯ , SQL-ІН’ЄКЦІЯ.

ЗМІСТ

ВСТУП

На сьогоднішній день, в епоху безпрецедентного росту цифрових технологій та зростаючого впливу інтернету на всі сфери життя, кібербезпека стає найважливішим аспектом забезпечення стабільності та надійності інформаційних систем.

Однією з найпоширеніших та найнебезпечніших атак є SQL-ін'єкції.

Ця атака відкриває двері для зловмисників, які можуть отримати несанкціонований доступ до баз даних, отримати конфіденційну інформацію, втрутитися в процеси бізнесу та завдати серйозної шкоди як фінансовій, так і репутаційній стороні організацій.

SQL-ін'єкції – це один з найпоширеніших типів уразливостей у веб-додатках. Вони виникають, коли користувач може ввести шкідливий код у форму або запит, який обробляється веб-додатком.

Цей шкідливий код може бути використаний для виконання довільних команд на сервері, в тому числі для крадіжки даних, внесення змін до даних або навіть запуску шкідливого програмного забезпечення.

SQL-ін'єкції можуть бути дуже небезпечними для веб-додатків. Вони можуть призвести до витоку конфіденційної інформації, таких як імена користувачів, паролі, кредитні картки та особисті дані.

Вони також можуть бути використані для атаки інших систем, які пов'язані з веб-додатком.

Існує кілька методів захисту від SQL-ін'єкцій. Одним з найпоширеніших методів є використання фільтрів, які видаляють шкідливий код з вхідних даних.

Інші методи включають використання параметризованих запитів, шифрування даних і навчання користувачів про небезпеку SQL-ін'єкцій.

SQL-ін'єкції є серйозною загрозою для безпеки веб-додатків. Вони можуть призвести до витоку конфіденційної інформації, крадіжки даних, внесення змін до даних або навіть запуску шкідливого програмного забезпечення.

Незважаючи на те, що існує кілька методів захисту від SQL-ін'єкцій, вони не завжди ефективні. Фільтри можуть пропускати шкідливий код, а параметризовані запити можуть бути використані для обходу захисту.

Тому є необхідність у розробці ефективних систем виявлення та запобігання SQL-ін'єкціям.

Актуальність теми. Стає особливо важливою в контексті постійного розвитку інтернет-технологій та поширення веб-додатків у всіх сферах бізнесу та суспільства.

Незважаючи на різноманітні методи захисту, зловмисники постійно вдосконалюють свої техніки, і тому необхідно розробляти та вдосконалювати ефективні засоби виявлення та запобігання SQL-ін'єкціям.

SQL-ін'єкції можуть бути дуже складними для виявлення та запобігання.

Існує багато різних методів виявлення та запобігання SQL-ін'єкціям, але жодна з цих методик не є ідеальною.

Тема дослідження є актуальною, оскільки вона спрямована на розробку системи, яка може ефективно виявляти та запобігати SQL-ін'єкціям. Така система може допомогти захистити веб-сайти від цих шкідливих атак.

Для обґрунтування актуальності теми можна навести такі аргументи:

SQL-ін'єкції є одними з найпоширеніших типів атак на веб-сайти.

Ці атаки можуть мати серйозні наслідки. Існує багато різних методів виявлення та запобігання SQL-ін'єкціям, але жодна з цих методик не є ідеальною.

Розробка системи, яка може ефективно виявляти та запобігати SQL-ін'єкціям, допоможе захистити веб-сайти від цих шкідливих атак.

На основі цих аргументів можна зробити висновок, що тема дослідження є актуальною та має наукову цінність.

Метою цієї роботи є розробка системи виявлення та запобігання SQL-ін'єкціям. **Об'єктом** дослідження є процедура та засоби виявлення та запобігання SQL-ін'єкціям, а **предметом** дослідження - машинне навчання, SQL-ін’єкції.

Методи досліджень базуються на сучасних методах побудови захищених комп’ютерних мереж та методах виявлення вразливостей.

Наукова новизна полягає у тому, що було запропоновано авторський підхід реалізації системи виявлення та запобігання SQL-ін'єкціям, на базі комбінованого методу, який об'єднує Sqlmap та машинне навчання, що представляє собою ефективний підхід до виявлення та класифікації SQL-ін'єкцій і дозволяє знаходити значну кількість можливих атак на вебзастосунок. Це дозволило підвищити точність виявлення вразливостей після впровадження машиного навчання в 1.2 рази.

Практична цінність дослідження полягає в тому, що розроблено авторську систему виявлення та запобігання SQL-ін'єкціям, на основі комбінованого методу виявлення вразливостей з використанням фреймворку

машинного навчання Scikit-learn мови програмування Python та моделі RandomForestClassifier.

Запропонована система дозволяє захистити веб-сайти від шкідливих атак, що дозволяє зменшити втрати конфіденційних даних, фінансових втрат та інших негативних наслідків.

Крім того, дослідження може сприяти розвитку методології виявлення та запобігання SQL-ін'єкціям.

Це може привести до розробки нових, більш ефективних методів, які будуть корисні для розробників і адміністраторів веб-сайтів.

Ось деякі конкретні приклади практичної цінності дослідження:

Система, розроблена в рамках дослідження, може бути використана для захисту веб-сайтів від SQL-ін'єкцій, що призведе до зменшення втрат даних і фінансових втрат.

Система може допомогти захистити конфіденційні дані користувачів, що призведе до підвищення рівня безпеки.

Система може допомогти захистити веб-сайти від шкідливого програмного забезпечення, яке може використовуватися для здійснення SQL-ін'єкцій.

Апробація.

1. Галата Л. П., Мазур Я. С., Крук Д. В. СИСТЕМА Виявлення та запобігання sql-ін’єкціям // Modern research in science and education: 2nd International scientific and practical conference, October 12-14, 2023: abstract. - USA (Chicago),2023. - P.167-170.

РОЗДІЛ 1.

ЗАГАЛЬНІ ПИТАННЯ ВИЯВЛЕННЯ ТА ЗАПОБІГАННЯ SQL-ІН'ЄКЦІЯМ

SQL-ін'єкція є одним з найпоширеніших типів атак на веб-додатки. Вона полягає в тому, що зловмисник вводить спеціально сформований запит у веб-форму або HTTP-заголовок, який дозволяє йому отримати доступ до даних, які не повинні бути доступні, або виконувати на сервері небажані дії.

Виявлення SQL-ін'єкцій є складним завданням, оскільки зловмисник може використовувати різні техніки для маскування свого запиту. Однак існує ряд методів, які можуть допомогти в виявленні таких атак.

Візуальне виявлення. Цей метод полягає в тому, щоб вручну переглянути код веб-додатку на наявність ознак SQL-ін'єкцій.

Наприклад, зловмисник може використовувати спеціальні символи або зарезервовані слова SQL, щоб ввести свій запит.

Використання інструментів для сканування. Існує ряд інструментів, які можуть сканувати веб-додатки на наявність SQL-ін'єкцій.

Ці інструменти використовують різні методи для виявлення таких атак, наприклад, аналіз коду, генерацію запитів та тестування на вразливості.

Використання статистичного аналізу. Цей метод полягає в тому, щоб проаналізувати поведінку веб-додатку на наявність аномалій, які можуть вказувати на SQL-ін'єкцію.

Наприклад, зловмисник може використовувати спеціальні символи або зарезервовані слова SQL, щоб ввести свій запит.

Найкращим способом запобігання SQL-ін’єкціям є правильне проектування та реалізація веб-додатків. Для цього слід дотримуватися таких рекомендацій:

Використовуйте параметризовані запити. Параметризовані запити дозволяють обробляти вхідні дані в безпечному режимі.

Фільтруйте вхідні дані. Вхідні дані слід фільтрувати, щоб видалити всі потенційно небезпечні символи.

Використовуйте сучасні технології. Сучасні технології, такі як OWASP, можуть допомогти в запобіганні SQL-ін’єкціям.

1.1. Сучасний стан забезпечення веб-безпеки

Сучасний стан веб-безпеки характеризується наступними тенденціями та викликами:

Зростання кіберзагроз: Кількість і складність кіберзагроз стрімко зростають: за прогнозами, кіберзлочинність коштуватиме 8 трильйонів доларів у 2023 році та 10,5 трильйонів доларів до 2025 року.

Веб-безпека є важливим аспектом інтернету, оскільки вона захищає користувачів від зловмисників, які можуть використовувати вразливості в веб-додатках для крадіжки даних, розсилки шкідливого програмного забезпечення або навіть отримання повного контролю над комп'ютером.

Серед основних проблем веб-безпеки можна виділити такі:

Вразливості в веб-додатках.

Веб-додатки є складними системами, які складаються з багатьох компонентів, таких як код, бази даних та веб-сервери. Кожний з цих компонентів може містити уразливості, які зломники можуть використовувати для отримання доступу до системи.

Шкідливе програмне забезпечення:

Шкідливе програмне забезпечення, таке як віруси, троянські коні та шкідливі програми, може бути використано для зараження веб-додатків і крадіжки даних.

Атаки зловмисників:

Зловмисники можуть використовувати різні методи, щоб атакувати веб-додатки, наприклад, фішинг, розсилку спаму та атаки типу "відвідувач".

Число компаній, які застосовують веб-технології для підвищення продуктивності роботи і залучення нових клієнтів, зростає з кожним роком.

Безсумнівно, інтернет-сервіси несуть з собою безліч переваг, але є й зворотна сторона медалі – з ростом числа додатків збільшується і кількість кіберзагроз.

Так, компанія Symantec в своєму звіті Global Internet Security Threat Report (ISTR) вказує, що кіберзлочинці при зломі веб-сайтів зазвичай використовують вразливості веб-додатків, що працюють на сервері, або експлуатують деякі вразливості операційної системи, на якій працюють ці додатки.

Наприклад, за допомогою атак типу XSS , хакер може перенаправити запити користувачів на шкідливі веб-сторінки, а за допомогою SQL-ін’єкцій – витягувати з баз даних сайту різну конфіденційну інформацію.

У відповідь на масові зломи систем безпеки був створений консорціум OWASP – Open Web Application Security Project, це відкритий проект забезпечення безпеки веб-додатків.

Однак і зловмисники, і фахівці в області кібербезпеки продовжують знаходити вразливості в веб-додатках, які можуть привести до серйозних втрат з боку бізнесу.

Основною причиною більшості взломів в веб-додатках є написаний розробниками програмний код.

Розробники можуть допускати помилки при написанні коду або не усвідомлювати всю важливість використання прийомів безпечного програмування – все це призводить до появи вразливостей в додатках.

Безсумнівно, захист веб-інфраструктури потрібний для будь-якої компанії.

Однак з безлічі категорій захисних рішень – Firewall, IPS / IDS, NGFW (Next Generation Firewall), WAF (Web-Application Firewall) тільки Web Application Firewall здатні забезпечити комплексний захист веб-додатків від відомих і невідомих загроз, а також забезпечити відповідність вимогам регуляторів , наприклад, PCI DSS. Ні класичний Firewall, ні IPS / IDS не зможуть забезпечити адекватного захисту веб-додатків.

За своєї суті файрволи (також застосовується термін і міжмережевий брандмауер екран) – це мережевий фільтр (між внутрішньої корпоративної) і зовнішньою мережею (тобто, Інтернет-середовищем).

Перші брандмауери були здатні лише блокувати підозрілі мережеві пакети на мережевому і канальному рівнях виходячи з IP-адреси джерела і призначення, міток фрагментації і номери портів.

Більш сучасні системи – такі як IPS / IDS (система запобігання вторгнень / система виявлення вторгнень) – здатні аналізувати вміст мережевих пакетів і порівнювати його з сигнатурами відомих атак.

Крім того, ці системи виявляють і блокують відхилення в протоколах прикладного рівня. Однак сьогодні понад 80% атак експлуатують уразливості додатків, а не мережевий архітектури.

Тому вищеописана система захисту виявляється малоефективною проти сучасних кіберзагроз. До того ж, сьогодні існує величезна кількість веб-додатків (кожне з яких потенційно може мати які-небудь уразливими), тобто, загальне число вразливостей набагато більше, ніж кількість сигнатур в базах сучасних IPS – систем.

За оцінками експертів, саме проникнення через веб-додатки останнім часом стають основним вектором атак на корпоративні мережі, причому традиційні системи безпеки, такі як файрвол і антивірусна система, неможливо запобігали подібні атаки.

Для надійного захисту необхідний кардинально інший підхід: з глибоким аналізом змісту пакетів і хорошим знанням структури веб-додатків, включаючи URL-параметри, форми введення даних і ін. Таким умовам задовольняє Web Firewall Application – брандмауер для додатків, які здійснюють передачу даних через HTTP і HTTPS.

Варто акцентувати, що WAF – це вузькоспеціалізована система, яка аналізує тільки протоколи HTTP / HTTPS.

З іншого боку, число способів обміну даними поверх протоколу HTTP настільки велике, що обробити їх здатне тільки дійсно спеціалізоване засіб.

Крім того, багато файрволи для веб-додатків підтримують роботу з SSL-трафіком. До речі, саме можливість перевірки зашифрованого трафіку є одним з найважливіших відмінностей WAF від звичайних міжмережевих екранів і IPS, стверджують аналітики Gartner.

Для виявлення атак WAF застосовує як сигнатурний, так і поведінковий підходи.

Другий метод також дуже важливий, оскільки для атак на веб-додатки кіберзлочинці можуть застосовувати уразливості нульового дня (нульовий день), що зводить до нуля ефективність сигнатурного аналізу. У той же час WAF здатний визначати модель нормального функціонування програми на основі аналізу мережевого трафіку і системних журналів і її базі детектувати відхилення в поведінку програмної системи.

Зокрема, WAF може виявляти атаки з використанням автоматичних засобів.

Класичний файрвол генерує величезні обсяги помилкових спрацьовувань на всілякі підозрілі події. Щоб детектувати рівень загрози, в цих повідомленнях треба розбиратися вручну.

У той же час WAF здатний аналізувати тисячі подій і будувати ланцюг розвитку атаки – від першого етапу до останнього.

Щоб забезпечити веб-безпеку, необхідно застосовувати комплексний підхід, який включає в себе:

Захист коду

Захист веб-сервера

Захист веб-браузера

Обізнаність користувачів

Крім того, важливо регулярно оновлювати веб-додатки і системи безпеки, щоб усунути відомі уразливості.

Ось кілька додаткових рекомендацій для забезпечення веб-безпеки:

Використовуйте надійні паролі і двофакторну аутентифікацію.

Уникайте використання ненадійних веб-сайтів і додатків.

Регулярно робіть резервні копії даних.

Будьте обережні при відкритті вкладок і файлів, які ви отримуєте електронною поштою або через соціальні мережі.

1.2. Аналіз загроз та вразливостей веб-безпеки

Загрози та вразливості веб-безпеки постійно розвиваються, оскільки зловмисники знаходять нові способи використання слабких місць у веб-додатках і системах. Розуміння цих загроз і вразливостей має вирішальне значення для організацій для впровадження ефективних заходів безпеки. Ось аналіз деяких поширених загроз і вразливостей веб-безпеки:

Ін'єкційні атаки (наприклад, SQL Injection, XSS):

Загроза: Ін'єкційні атаки передбачають вставку шкідливого коду або команд у поля введення або параметри, що дозволяє зловмисникам маніпулювати додатком або базою даних.

Вразливість: Недостатня перевірка вхідних даних та неналежна санітарна обробка даних, введених користувачем, роблять додатки вразливими до цих атак.

Усунення: Впроваджуйте сувору перевірку даних, що вводяться: Впроваджуйте сувору перевірку вхідних даних, використовуйте параметризовані оператори та застосовуйте механізми безпеки, такі як брандмауери веб-додатків (WAF), для виявлення та блокування спроб ін'єкцій.

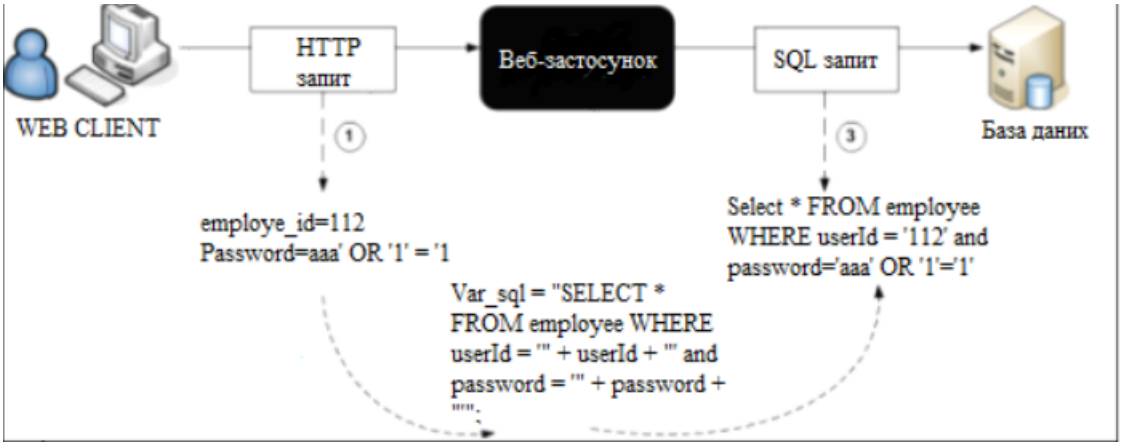

Рис. 1.1 Схема атаки SQL-ін’єкції

Міжсайтовий скриптинг (XSS):

Загроза: XSS-атаки полягають у впровадженні шкідливих скриптів на веб-сторінки, які переглядають інші користувачі. Це може призвести до перехоплення сеансу, крадіжки або пошкодження даних.

Вразливість: Недостатнє кодування виводу або екранування у веб-додатках дозволяє виконувати ненадійні дані як код у браузері.

Усунення: Використовуйте належне кодування виводу, використовуйте засоби захисту: Використовуйте належне кодування виводу, використовуйте заголовки безпеки, такі як Content Security Policy (CSP), та санітарну обробку створеного користувачем контенту.

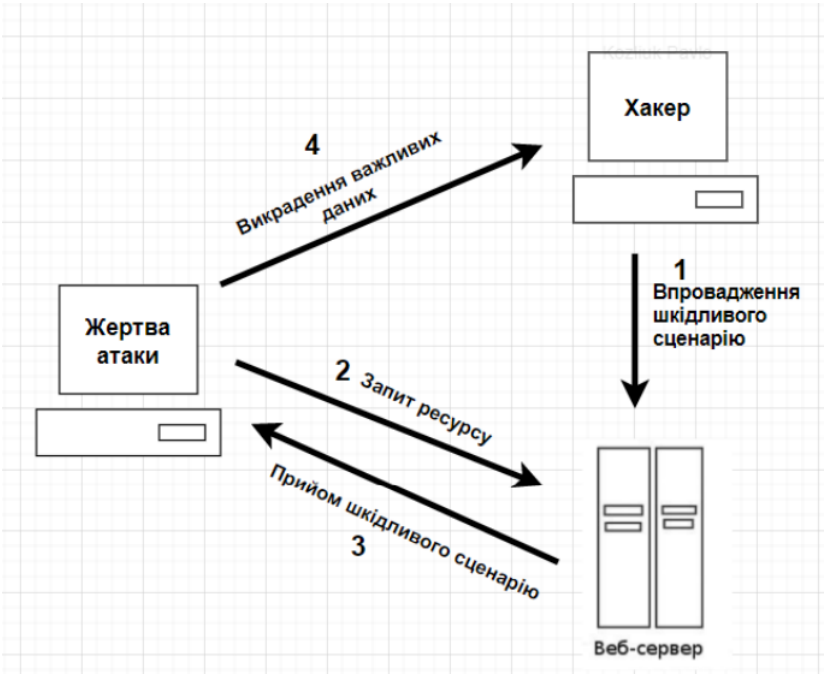

Рис. 1.2 Схема атаки XSS

Підробка міжсайтових запитів (CSRF):

Загроза: CSRF-атаки обманом змушують користувачів виконувати дії у веб-додатках без їхньої згоди, використовуючи їхні автентифіковані сеанси.

Вразливість: Відсутність токенів для захисту від CSRF і недостатня перевірка запитів можуть зробити додатки вразливими до CSRF-атак.

Усунення: Впроваджуйте анти-CSRF токени, перевіряйте запити та використовуйте атрибут SameSite для файлів cookie.

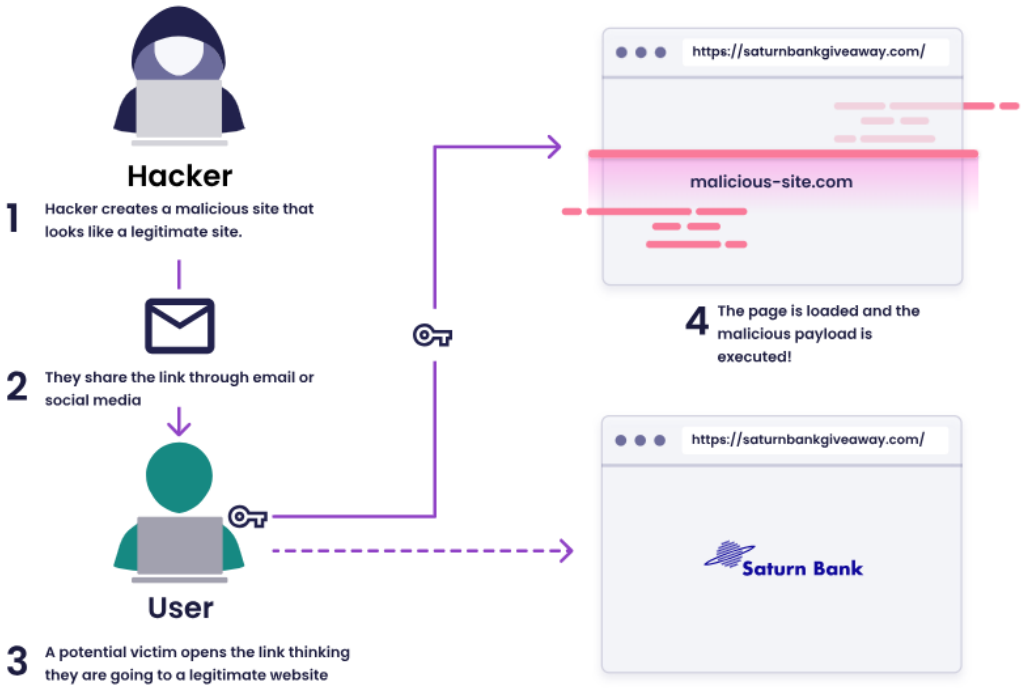

Рис. 1.3 Схема атаки CSRF

Проблеми з автентифікацією та управлінням сеансами:

Загроза: Слабка автентифікація та керування сеансами може призвести до несанкціонованого доступу до облікових записів і конфіденційних даних.

Вразливість: Погані політики паролів, фіксація сеансів і недостатні таймаути сеансів є поширеними вразливостями.

Пом'якшення наслідків: Впроваджуйте надійні політики паролів, використовуйте багатофакторну автентифікацію (MFA), регулярно перевіряйте та оновлюйте методи управління сеансами.

Незахищені прямі посилання на об'єкти (IDOR):

Загроза: IDOR виникає, коли зловмисник отримує несанкціонований доступ до ресурсів, маніпулюючи посиланнями на об'єкти в додатку.

Вразливість: Недостатній контроль доступу та прямий доступ до внутрішніх посилань може призвести до IDOR-уразливостей.

Усунення: Впроваджуйте належні засоби контролю доступу, використовуйте непрямі карти посилань та перевіряйте дозволи користувачів для кожного запиту.

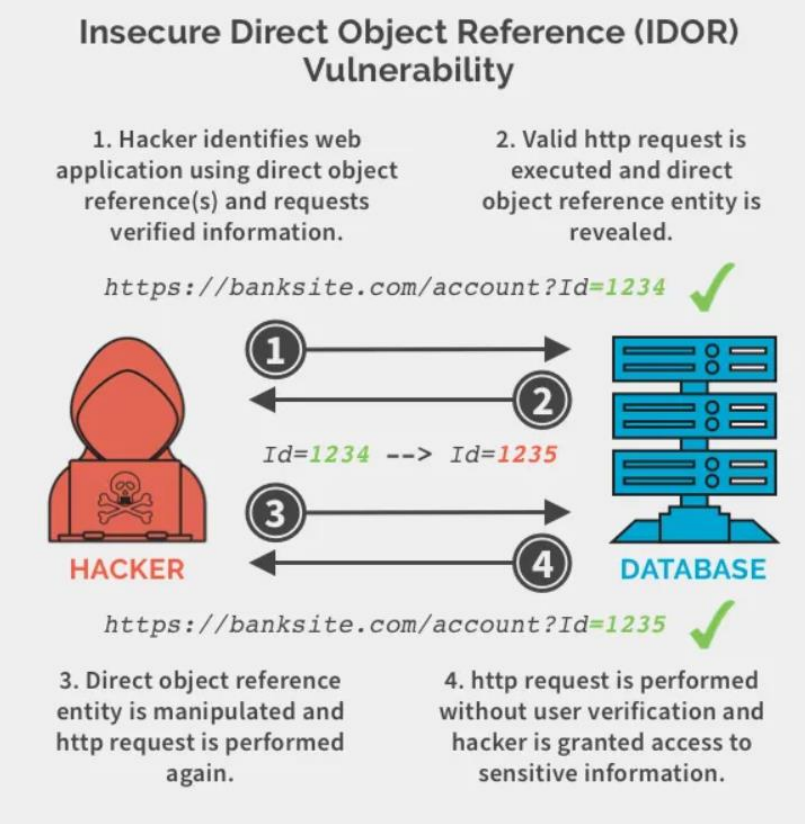

Рис. 1.4 Схема атаки CSRF

Неправильні конфігурації безпеки:

Загроза:

Неправильні конфігурації безпеки є наслідком неправильно налаштованих серверів, баз даних або додатків, які можуть викривати конфіденційну інформацію або надавати точки входу для зловмисників.

Вразливість:

Погано налаштовані параметри безпеки, непотрібні відкриті порти та паролі за замовчуванням є поширеними помилковими конфігураціями.

Пом'якшення наслідків:

Регулярно перевіряйте та переглядайте конфігурації, застосовуйте принцип найменших привілеїв та дотримуйтесь найкращих практик безпеки для серверів та додатків.

Атаки на відмову в обслуговуванні (DoS) та розподілені атаки на відмову в обслуговуванні (DDoS):

Загроза:

DoS-атаки перевантажують ресурси системи, роблячи її недоступною. DDoS-атаки посилюють вплив за рахунок використання декількох джерел.

Вразливість:

Відсутність обмеження швидкості, вичерпання ресурсів і погано керована мережева інфраструктура можуть зробити системи вразливими.

Пом'якшення наслідків:

Впроваджуйте обмеження швидкості, використовуйте служби захисту від DDoS-атак і створіть відмовостійку мережеву інфраструктуру.

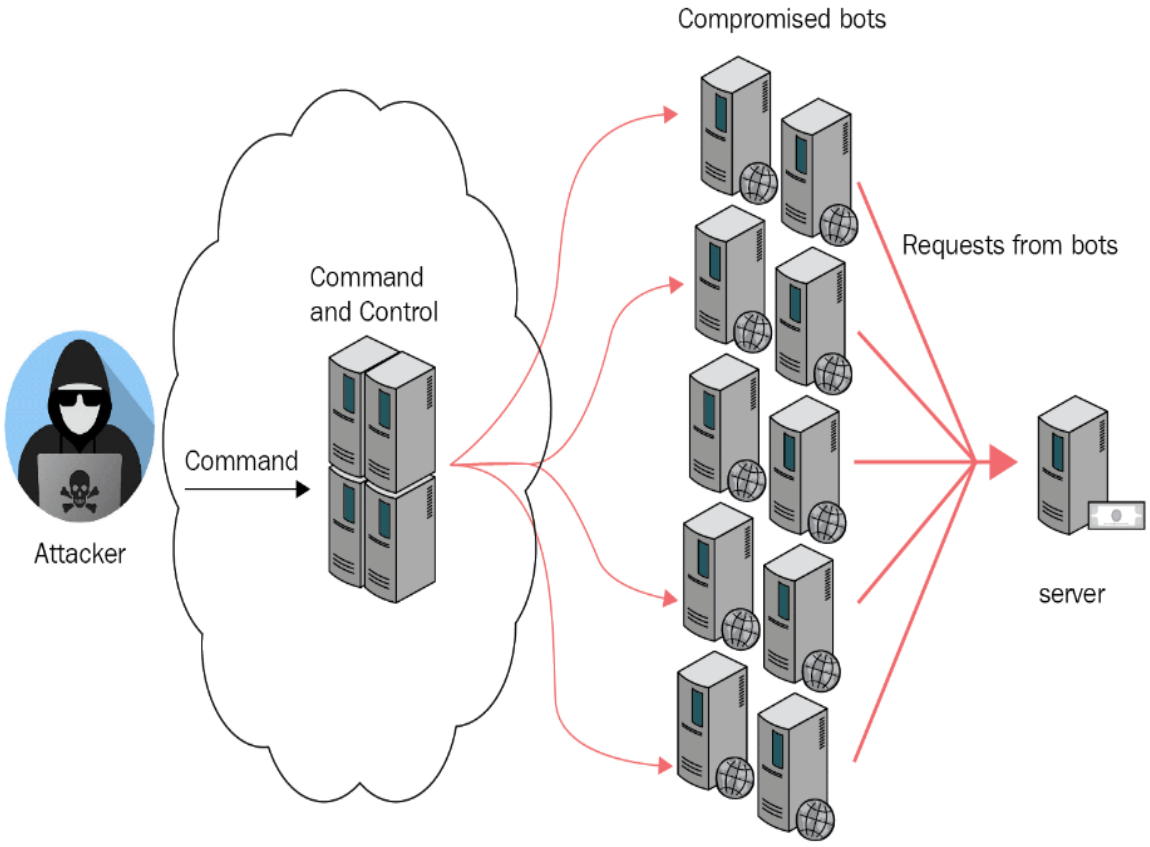

Рис. 1.5 Схема атаки DDoS

- Атаки на зовнішні об'єкти XML (XXE):

Загроза: XXE-атаки використовують вразливі парсери XML для розкриття внутрішніх файлів, виконання віддаленого коду або ініціювання відмови в обслуговуванні.

Вразливість: Дозволяє обробляти ненадійний XML-ввід без належної перевірки та вимикає обробку зовнішніх об'єктів.

Усунення: Вимкніть обробку зовнішніх об'єктів, перевірте вхідні дані у форматі XML та перевірте XML-документи на відповідність попередньо визначеним схемам.

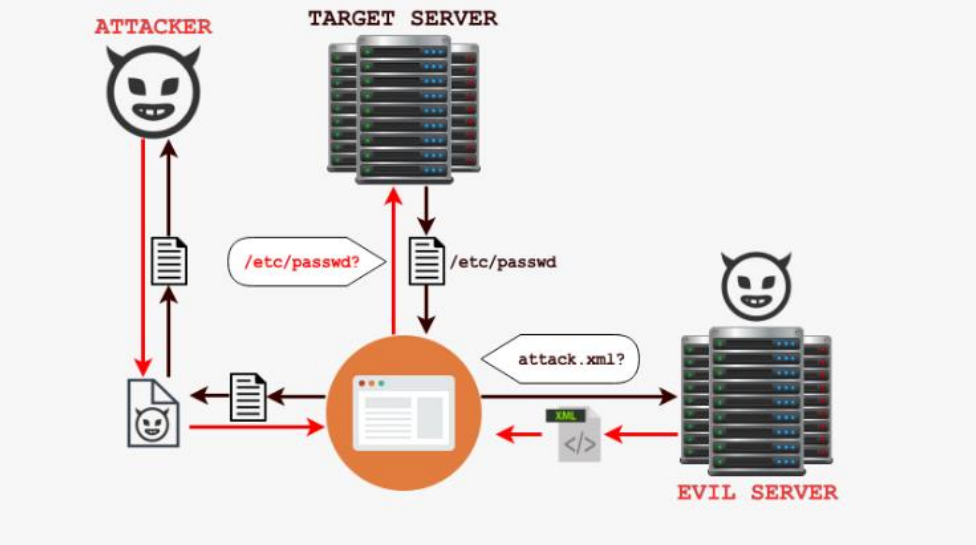

Рис. 1.6. Схема атаки XXE

- Неперевірені перенаправлення і переадресації:

Загроза: Зловмисники можуть обманом змусити користувачів відвідати шкідливі веб-сайти або виконати небажані дії, використовуючи неперевірені перенаправлення та переадресації.

Вразливість: Відсутність перевірки та дезінфекції URL-адрес перенаправлення та переадресації в додатку.

Усунення: Уникайте використання даних, введених користувачем, для перенаправлення, використовуйте білий список дозволених місць призначення, а також виконуйте перевірку та кодування URL-адрес.

- Вразливість конфіденційних даних:

Загроза: Вразливість чутливих даних виникає, коли зловмисники отримують доступ до конфіденційної інформації, наприклад, паролів або номерів кредитних карток.

Вразливість: Слабке шифрування, зберігання конфіденційних даних у відкритому вигляді та неналежний захист даних у стані спокою або під час передачі.

Пом'якшення наслідків: Використовуйте стійкі алгоритми шифрування, хешуйте конфіденційні дані та дотримуйтесь правил захисту даних, таких як GDPR та HIPAA.

- Проблеми з безпекою API:

Загроза: Оскільки API стають все більш поширеними, вразливості в безпеці API можуть зробити дані та функціональність доступними для зловмисників.

Вразливість: Неадекватні механізми автентифікації та авторизації, незахищені кінцеві точки API та відсутність перевірки вхідних даних.

Пом'якшення наслідків: Впроваджуйте надійну автентифікацію та авторизацію для API, використовуйте ключі або токени API та перевіряйте вхідні та вихідні дані.

- Вразливості нульового дня:

Загроза: Вразливості нульового дня - це прогалини в безпеці, які використовуються до того, як з'являється виправлення або патч.

Вразливість: Невідомі вразливості в програмному забезпеченні або системах, які зловмисники виявляють і використовують.

Пом'якшення наслідків: Будьте в курсі рекомендацій з безпеки, оперативно застосовуйте виправлення та використовуйте системи виявлення вторгнень (IDS) і моніторингу безпеки.

Для ефективного зниження ризиків веб-безпеки організаціям слід розглянути наступні ключові стратегії:

Регулярний аудит безпеки: Проводьте регулярні оцінки безпеки, сканування вразливостей і тести на проникнення, щоб виявити і усунути слабкі місця у веб-додатках і системах.

Навчання з безпеки: Навчайте персонал та розробників безпечним методам кодування, соціальній інженерії та новітнім векторам загроз, щоб зменшити вплив людського фактору на вразливості безпеки.

Безпека за задумом: Інтегруйте безпеку в процес розробки з самого початку, застосовуючи такі принципи, як OWASP Top Ten, щоб запобігати вразливостям, а не реагувати на них.

Управління виправленнями: Слідкуйте за оновленнями програмного забезпечення та систем, щоб усунути відомі вразливості та зменшити ризик їх використання.

Контроль доступу та автентифікація: Впроваджуйте надійний контроль доступу, використовуйте багатофакторну автентифікацію та забезпечуйте належне управління сеансами для захисту облікових записів користувачів і конфіденційних даних.

Моніторинг та реагування на інциденти: Налаштуйте ефективні системи реєстрації та моніторингу, щоб оперативно виявляти незвичні дії та розробити план реагування на інциденти для швидкого та ефективного усунення наслідків.

Управління сторонніми ризиками:

Регулярно відстежуйте та оновлюйте сторонні залежності, щоб уникнути впровадження вразливостей у ваш додаток через зовнішній код або сервіси.

Відповідність та захист даних: Дотримуйтесь правил захисту даних (наприклад, GDPR, HIPAA) і галузевих стандартів, щоб захистити дані користувачів і забезпечити відповідність законодавству.

Поінформованість про безпеку:

Сприяти розвитку культури обізнаності щодо безпеки в організації, роблячи безпеку спільною відповідальністю всіх співробітників.

Постійне вдосконалення:

Визнайте, що веб-безпека - це безперервний процес, і адаптуйтеся до мінливого ландшафту загроз, навчаючись на інцидентах і залишаючись в курсі нових загроз і передових практик.

У сучасну цифрову епоху веб-безпека - це не одноразовий захід, а постійне прагнення захистити цінні активи, зберегти довіру користувачів і захиститися від постійно мінливого ландшафту кіберзагроз.

Впроваджуючи ці стратегії та зберігаючи пильність, організації можуть посилити свій захист і зменшити ризики, пов'язані із загрозами та вразливостями веб-безпеки.

1.3. Дослідження систем виявлення та запобігання SQL-ін’єкціям

SQL-ін’єкція - це один з найпоширеніших видів атак на веб-додатки. Вона полягає в тому, що зловмисник намагається вставити в запит до бази даних

шкідливий код, який може бути використаний для крадіжки даних, зміни їх або навіть деактивації додатка.

Існує два основних підходи для виявлення та запобігання SQL-ін’єкціям: статичний аналіз і динамічний аналіз.

Статичним аналізом називається аналіз коду додатка без його виконання.

Цей підхід дозволяє виявити потенційні уразливості ще до того, як додаток буде запущений.

Динамічний аналіз передбачає виконання додатка і аналіз його поведінки під час виконання. Цей підхід дозволяє виявити SQL-ін’єкції, які не можна виявити статичним аналізом.

Існує кілька шляхів удосконалення систем виявлення та запобігання SQL-ін’єкціям.

Одним із напрямів є розробка нових методів статичного аналізу, які будуть більш ефективними в виявленні уразливостей.

Іншим напрямом є розробка нових методів динамічного аналізу, які будуть більш швидкими і ефективними.

Виявлення та запобігання є складним завданням, якщо є належне розуміння типів SQL Injection атак, то легше запобігти атаці.

Таблиця 1.1

Порівняння систем виявлення та запобігання SQL-ін’єкціям

Характеристика	Статичний аналіз	Динамічний аналіз
Переваги	Швидкий, ефективний, може виявити потенційні уразливості до того, як додаток буде запущений.	Може виявити SQL-ін’єкції, які неможливо виявити статичним аналізом.
Недоліки	Може не виявити всі уразливості, може бути неможливим виконати статичний аналіз для деяких типів коду.	Може бути повільним, може призвести до зниження продуктивності додатка.

Для захисту від сучасних SQL-ін'єкцій завжди рекомендується використовувати заздалегідь підготовлені оператори, оскільки вони є фіксованими і не можуть бути змінені користувачем веб-сайту або веб-додатку.

Такі прийоми, як magic quotes() та add slashes() не можуть захистити веб-додаток або веб-сайт від SQL-ін'єкцій. Далі ми обговоримо різні методи для виявлення та запобігання сучасним SQL-ін'єкціям.

Виявлення та запобігання сліпим SQL ін'єкціям:

Існує велика кількість дослідницьких робіт по сліпим SQL-ін'єкціям, в яких описуються різні методи виявлення та запобігання.

Оскільки сліпі SQL ін'єкції важко виявити та запобігти, але дослідники знали про сліпі SQL ін'єкції вже багато років.

Найпопулярнішою технікою, що використовується, є AMNESIA, що розшифровується як Аналіз та Моніторинг для нейтралізації атак SQL-ін'єкцій, інструмент застосовується лише для захисту додатків на основі java і використовує моніторинг під час виконання.

Komiya запропонував кращий метод для запобігання SQL-ін'єкцій. Вони рекомендують використовувати алгоритми машинного навчання для покращення запобігання та виявлення сліпих ін'єкцій SQL.

Вони отримали результати і підтвердили, що запобігання та виявлення були кращими, ніж SQLCheck та AMNESIA.

Виявлення та запобігання Fast Flux SQL-ін'єкцій:

Основні атаки, в яких безпека на стороні клієнта зазнає невдачі та нове явище, яке не є широко відомим - Атаки на швидкий потік.

Атаки на швидкий потік SQL-ін'єкцій були зафіксовані в Університеті штату Індіана, США, і навіть безпека ФБР була стурбована.

Найкращий спосіб захиститися від атак швидкого потоку це зробити сервери безпечними.

Швидкий потік може захистити, використовуючи техніку, за допомогою якої URL-адреси, що вказують на javascript можуть бути занесені до чорного списку, якщо вони будуть швидко ідентифіковані.

Альпер обговорює у своїй статті про монітор швидкого потоку (Fast Flux Monitor, FFM), який може виявити, а також може класифікувати Fast Flux Service

Networks в порядку хвилин для використання як активного, так і пасивного моніторингу DNS, що доповнює довгострокове спостереження за FFSN.

Хольц провів дослідження для виявлення мереж швидкого потоку та SQL Injection атак за допомогою експертних систем.

Подальше вдосконалення було зроблено Сталсманом та Ірвіном шляхом розробки більш придатного методу для виявлення мережевого швидкого потоку та атак SQL-ін'єкцій.

Серед багатьох методів машинного навчання вони зробили акцент на C5.0 Classifier та наївному байєсівському класифікатору для виявлення Fast Flux та SQL- ін'єкційних атак.

Запобігання Fast Flux є дійсно складне завдання, і багато досліджень спрямовані на пошук правильних методів для протидії Fast Flux SQL-ін'єкційних атак.

Виявлення та запобігання SQL-ін'єкцій XSS:

Адам обговорює у своїй статті про автоматизацію створення SQL-ін'єкцій та XSS з метою обходу та проникнення в базу даних з метою пошуку вразливостей.

Вони обговорюють інструмент Ardilla Tool, який в першу чергу є інструментом атаки інструмент, в якому користувач вибирає, що атакувати (SQL, XSS першого порядку або XSS другого порядку).

Інструмент використовується для виявлення SQL-ін'єкцій + XSS атаки. Він має два режими для перевірки достовірності атаки, тобто строгий режим і м'який режим, в той час як SQL-ін'єкція має тільки один режим.

Ardilla Tool використовує Taint Based підходи і методи статичного аналізу, при цьому, якщо передумови не виконані, він запропонує фільтри та інші методи санітарної обробки для того, щоб виконати передумову, яка є вимогою для виявлення вразливості.

Інші інструменти не є настільки ефективні, як ARDILLA. Тому, для того, щоб захистити нашу систему, по-перше, необхідно виявити та запобігти XSS.

По-друге, необхідно застосувати методи виявлення та запобігання SQL-ін'єкцій, для того, щоб досягти поставленої мети.

Використовуючи Cross Scripting атаки, зловмисник може атакувати багато різних частин веб-додатку є крадіжка файлів cookie, що в подальшому може призвести до вразливості, втрати критичної інформації та SQL-ін'єкції.

Крадіжці файлів cookie можна запобігти за допомогою динамічного Cookies Rewriting техніки, яка була розглянута Rattipong. У своїй роботі він обговорює створення випадкових даних та зміну імені при зберіганні в таблиці cookies.

Для того, щоб запобігти цим типам атак Zhang та його колеги розробили механізм Execution flow для захисту від XSS-атак на основі java Script, який служить для захисту від атаки на основі java Script, які слугують для захисту від SQL-ін'єкцій в XSS.

У цій техніці запобігання вони використовували кінцеві автомати для аналізу клієнтського java-скрипту і це запобігає будь-якому зловмисному скрипту для введення або отримання даних з бази даних оскільки він використовує алгоритм машинного навчання який вдосконалюється з досвідом і сильно залежить від наборів даних, але він не гарантує повного захисту.

Згідно з Фогтом та іншими обіцяє що динамічне псування даних - це метод, який використовується для виявлення та запобігання XSS-атак і тоді SQL-ін'єкція автоматично захищається, але Nikiforakis має протилежну реакцію, оскільки він стверджує, що існує багато прихованих каналів, які залишаються невиявленими отже не можуть запобігти атаці за допомогою динамічного псування даних.

Іншим дуже популярним інструментом, який використовується для пом'якшення наслідків XSS-атак, є інструмент Noxes.

Розробники цього інструменту були натхненні брандмауером Windows Firewall.

Це безумовно допомогло в захисті від XSS атак + SQL-ін'єкційних атак

але він не може повністю запобігти атаці, як це обговорювали Nikiforakis та його колеги, оскільки вони вважають, що зловмисник може використовувати html-теги замість тегів скриптів для атаки, про http-запит і запобігає модифікаціям, зробленим в http-заголовку і має функціонал для встановлення файлів cookie.

Виявлення та запобігання SQL-ін'єкцій DNS

У цьому підході виявляється перехоплення DNS і після цього відбувається запобігання та виявлення SQL-ін'єкцій.

Перехоплення DNS можна запобігти, якщо не завантажувати безкоштовних утиліт з веб-сайтів, оскільки вони здебільшого містять вразливості.

Diter Gollman описує переприв'язування DNS який намагається перехопити налаштування маршрутизатора клієнта або користувача.

Для того, щоб запобігти перехопленню DNS, Nikiforakis та інші розробили сеансовий екран, який є легким механізмом захисту на стороні клієнта.

Stampar у своїй роботі обговорює про використання SQLMap для захисту від SQLI + DNS

SQLMap має функцію фільтрації DNS Ex і є багато командних рядків, спеціально розроблених для DNS запобігання та виявлення DNS-атак. Він сумісний з більшістю версій версій баз даних SQL.

Виявлення та запобігання DDoS-атакам на SQL-ін'єкції: DDoS-атаки добре відомі фахівцям з безпеки, але незважаючи на те, що ця тема добре обговорюється, існують деякі лазівки, які використовують зловмисники для атаки на систему.

Виявлення DDoS-атак та виявлення SQL-ін'єкцій було добре обговорено Лі та його колегами, які запропонували ідею виявлення DDoS-атак за допомогою кластерного аналізу. Кластерний аналіз допомагає виявити DDoS-атаки і може легко ідентифікувати тип атаки на систему.

Yu shui запропонував з обговоренням огляду різних методів виявлення DDoS-атак. Після фази виявлення настає фаза пом'якшення наслідків, яка порівняно набагато простіша

Недостатня автентифікація SQL-ін'єкцій:

Для того, щоб захиститися від SQL- ін'єкцій не вникаючи в аутентифікації адміністратор може використовувати техніку криптографічних хеш-функцій, яку використовували Singh для захисту від SQL- ін'єкцій.

У цьому методі додаються два додаткові атрибути, а саме хеш-функції для поля імені користувача та пароля.

Хеш-функції автоматично генеруються за допомогою хеш-алгоритмів.

Тепер, коли клієнт вводить ім'я користувача та пароль, то хеш-функція генерується і передається на сервер для перевірки.

Все, що відбувається тут, відбувається в зашифрованому вигляді, якщо ім'я користувача та пароль збігаються з тими що зберігаються в базі даних, то вони збігаються з хеш-функцією.

Ми склали дві таблиці, які зображені в цьому В таблиці 1.2 ми показали, які методи використовуються для виявлення та запобігання сучасним SQL-ін'єкційним атакам.

У таблиці 1.3 ми обговоримо різні методи для виявлення та запобігання а також інструменти, що використовуються.

У цьому ми будемо використовувати зірочку (*) для того, щоб показати, що вона використовується як для виявлення, так і для запобігання сучасними ін'єкціями SQL в таблиці.

Коло (o) використовується для того, щоб показати, що він використовується тільки для механізму виявлення, плюс (+) використовується тільки для запобігання, що відповідає сучасним SQL-ін'єкціям.

Символ (x) використовується для того, щоб показати, що методи або інструменти не відповідають, символ (p) використовується для позначення того, що методи або інструменти не відповідають сучасним SQL-ін'єкціям (з точки зору запобігання та виявлення).

Символ (p) позначає незавершеність, тобто після застосування певного методу необхідно застосувати інший метод для досягнення повного виявлення та запобігання.

У таблиці ми взяли різні методи, які можуть бути використані для виявлення та попередження атак. Вони можуть бути використані для виявлення та запобігання атакам.

Таблиця 1.2

Різні методи виявлення та запобігання атакам

№	Техніка	Сліпа SQL	Fast Flux SQL	XSS SQL	DNS SQL	DDoS SQL	Аутентифікація SQL
1	Криптографічні хеш-функції	р	х	х	х	х	+
2	Динамічний перезапис кукі	х	х	+	р	х	р
3	Механізм потоку виконання	х	х	*	х	х	х
4	Статичний аналіз коду	о	х	о	х	х	х
5	Динамічне псування даних	х	х	*	х	х	х
6	Моніторинг часу виконання	р	+	*	х	х	х
7	Машинне навчання	х	о	*	х	о	х

Ці методи допоможуть дослідникам і фахівцям з безпеки вжити належних заходів або використовувати зазначені методи для вирішення кризових ситуацій, що виникли в організації внаслідок атаки.

Таблиця 1.3

Оцінка різних інструментів для виявлення та запобігання атакам

№	Інструменти	Сліпа SQL	Fast Flux SQL	XSS SQL	DNS SQL	DDoS SQL	Аутентифікація SQL
1	Ardilla Tool	x	x	o	x	x	x
2	Noxes	x	x	p	x	x	x
3	Session Shield	x	x	*	p	x	p
4	AMNESIA	*	x	p	x	x	x
5	SQLMap	o	x	o	p	o	o
6	Fast Flux Monitor	x	o	x	o	x	x

Описані тут методи можуть бути використані для розробки системи з додатковою функціональністю для захисту системи від будь-якого виду цих сучасних атак, що відповідає Compound SQL та Fast Flux атакам.

Згідно з проведеним дослідженням, ми помітили що статичний аналіз коду та машинне навчання є найкращими серед інших, але інші методи мають ряд інших переваг.

У цій таблиці ми взяли різні інструменти для виявлення та запобігання сучасним атакам.

Ці інструменти є готовими, а деякі з них мають відкритий вихідний код, який можна завантажити з інтернету.

Більшість з цих інструментів були розроблені для дослідницьких цілей, але завдяки своїм значним перевагам вони використовуються і в комерційному секторі.

Інструменти обговорюються і розглядаються, щоб дати широкий огляд того, які з цих інструментів можуть бути використані для конкретного типу атак.

Згідно з нашим проведеним дослідженням, ми бачимо, що Noxeus та SQLMap є найновішими і мають кращі механізми запобігання та виявлення.

1.4. Висновки до першого розділу

У першому розділі було проведено аналіз інструментів і методів, спрямованих на боротьбу з SQL-ін'єкціями, які є серйозною загрозою для безпеки інформаційних систем.

Вивчення цього питання стало актуальним у зв'язку з поширенням веб-додатків та баз даних, де зловмисники можуть використовувати SQL-ін'єкції для несанкціонованого доступу та витоку конфіденційної інформації.

У результаті проведеного дослідження було виявлено, що два з основних інструментів для виявлення та запобігання SQL-ін'єкціям, а саме Noxes та SQLMap, вирізняються своєю ефективністю та актуальністю.

Вони мають найсучасніші механізми, які дозволяють виявляти та запобігати атакам цього типу.

Обидва інструменти регулярно оновлюються, що робить їх надійними інструментами для захисту від SQL-ін'єкцій в сучасних умовах.

Порівнюючи різні методи боротьби з SQL-ін'єкціями, було виявлено, що статичний аналіз коду та машинне навчання відзначаються найвищою ефективністю.

Статичний аналіз коду дозволяє виявляти потенційно небезпечні конструкції під час розробки, що допомагає у виявленні уразливостей на ранніх етапах.

Машинне навчання використовується для створення моделей, які можуть виявляти нестандартні та складні атаки, що робить його корисним інструментом для виявлення нових загроз.

Важливо відзначити, що інші методи, такі як санітайзери даних та параметризовані запити, також мають свої переваги.

Вони можуть бути використані як додаткові заходи захисту разом із статичним аналізом та машинним навчанням.

Вибір методу залежить від конкретних вимог та особливостей інформаційної системи.

Загалом, результати дослідження підтверджують важливість використання сучасних інструментів та методів для виявлення та запобігання SQL-ін'єкціям.

Системи безпеки повинні поєднувати різні підходи і заходи, щоб забезпечити надійний захист від цього типу атак і зберегти цілісність і конфіденційність інформації в інформаційних системах.

РОЗДІЛ 2.

ІСНУЮЧІ ПРАКТИЧНІ РЕАЛІЗАЦІЇ ЩОДО ВИЯВЛЕННЯ ТА ЗАПОБІГАННЯ SQL-ІН’ЄКЦІЙ

2.1. Сучасний стан використання машинного навчання в сфері захисту інформації

Сучасний стан використання машинного навчання в сфері захисту інформації виявляється у багатьох аспектах, враховуючи постійне розвиток технологій та зростання кількості кіберзагроз.

Одним із ключових напрямків використання машинного навчання є аналіз великих обсягів даних для виявлення аномалій та підозрілих активностей.

Машинне навчання дозволяє створювати моделі, які можуть автоматично виявляти невідповідності в поведінці користувачів або систем, що може свідчити про можливу кіберзагрозу.

Машинне навчання - це частина штучного інтелекту, яка дозволяє комп'ютерам навчатися на основі даних і робити прогнози без явного програмування.

Машинне навчання все частіше застосовується в сфері інформаційної безпеки, де він може допомогти вирішити деякі з ключових проблем, з якими стикаються фахівці з безпеки, такі як

Виявлення та запобігання атакам шкідливих програм: Він може аналізувати великі обсяги мережевого трафіку і виявляти шаблони зловмисної поведінки, такі як шифрування, завуальовування або командно-контрольне спілкування.

А також може класифікувати невідомі файли і процеси на основі їхніх атрибутів і поведінки та позначати їх як безпечні або шкідливі.

Виявлення та зменшення внутрішніх загроз: Машине навчання може відстежувати діяльність і поведінку користувачів і пристроїв в організації та виявляти аномалії, які вказують на потенційні витоки даних, порушення політик або саботаж.

Машине навчання також може допомогти визначити першопричину і масштаб інциденту, а також надати рекомендації щодо його усунення.

Наприклад, Cisco використовує машине навчання для пошуку внутрішніх загроз, аналізуючи дані з різних джерел, таких як мережеві журнали, профілі користувачів і конфігурації пристроїв.

Прогнозування та блокування шкідливих веб-сайтів: Машине навчання може аналізувати інтернет-активність і репутацію доменів та IP-адрес, а також прогнозувати, які з них можуть містити шкідливий контент або здійснювати атаки.

Машине навчання також може допомогти користувачам уникнути фішингу та шахрайства з використанням соціальної інженерії, перевіряючи автентичність і легітимність веб-сайтів та електронних листів.

Наприклад, Cisco використовує машине навчання для прогнозування "поганих сусідів" в Інтернеті та запобігання підключенню користувачів до шкідливих веб-сайтів.

Захист даних у хмарі: Машине навчання може допомогти захистити хмарні додатки і платформи, аналізуючи використання і поведінку користувачів хмарних сервісів і виявляючи аномалії, які вказують на несанкціонований доступ, витік даних або компрометацію.

Машине навчання також може допомогти забезпечити дотримання політик відповідності та управління, а також забезпечити видимість і контроль над хмарними ресурсами.

Наприклад, Cisco використовує Машине навчання для захисту даних у хмарі, виявляючи підозрілу поведінку користувачів і надаючи сповіщення та рекомендації.

Сучасний стан використання машиного навчання в інформаційній безпеці постійно розвивається, оскільки розробляються і застосовуються нові методи і технології для вирішення різних проблем безпеки. Деякі з нових тенденцій та викликів у цій галузі включають

Використання глибокого навчання і нейронних мереж:

Глибоке навчання - це галузь машиного навчання, яка використовує кілька шарів штучних нейронів для вивчення складних і нелінійних закономірностей на основі даних.

Глибоке навчання може досягти вищої точності та продуктивності, ніж традиційні методи машиного навчання, особливо для таких завдань, як розпізнавання зображень, обробка природної мови та розпізнавання мовлення.

Однак глибоке навчання також вимагає більше обчислювальних ресурсів і даних, і його може бути складніше інтерпретувати і пояснювати.

В інформаційній безпеці глибинне навчання можна використовувати для таких завдань, як виявлення шкідливого програмного забезпечення, розпізнавання облич та аналіз настроїв.

Існують різні підходи до використання методів машинного навчання у сфері кіберзахисту, залежно від типу проблеми, наявних даних та бажаного результату. Ось деякі з найпоширеніших підходів:

Використання керованого навчання для виявлення та запобігання атакам шкідливих програм:

Контрольоване навчання - це тип машинного навчання, який навчається на основі маркованих даних, таких як безпечні та шкідливі зразки, і прогнозує клас або мітку нових даних.

Цей підхід можна використовувати для аналізу мережевого трафіку, файлів і процесів, а також для виявлення моделей зловмисної поведінки, таких як шифрування, заплутування або командно-контрольне спілкування.

Наприклад, CrowdStrike використовує кероване навчання для виявлення небаченого раніше шкідливого програмного забезпечення, яке намагається запуститися на кінцевих точках.

Використання неконтрольованого навчання для виявлення та пом'якшення внутрішніх загроз:

Неконтрольоване навчання - це тип машинного навчання, який навчається на немаркованих даних і знаходить в них структуру, взаємозв'язки і закономірності, такі як кластери або угруповання.

Цей підхід можна використовувати для моніторингу діяльності та поведінки користувачів і пристроїв в організації, а також для виявлення аномалій, які вказують на потенційні витоки даних, порушення політик або саботаж.

Наприклад, Cisco використовує неконтрольоване навчання для пошуку внутрішніх загроз, аналізуючи дані з різних джерел, таких як мережеві журнали, профілі користувачів і конфігурації пристроїв.

Використання навчання з підкріпленням і навчання в умовах суперництва для захисту кіберфізичних систем:

Навчання з підкріпленням - це тип машинного навчання, який навчається шляхом спроб і помилок, взаємодіючи з навколишнім середовищем і отримуючи зворотний зв'язок або винагороду.

Змагальне навчання - це метод, спрямований на підвищення надійності та безпеки систем машинного навчання шляхом створення та захисту від прикладів, які є вхідними даними, навмисно розробленими для того, щоб обдурити або обійти систему.

Цей підхід може бути використаний для захисту кіберфізичних систем, таких як електромережі, промислові системи управління або автономні транспортні засоби, від атак, які використовують їхні вразливості або маніпулюють їхніми відгуками.

Наприклад, дослідники запропонували використовувати навчання з підкріпленням і навчання в змаганні для розробки безпечних і стійких контролерів для кіберфізичних систем3.

Використання федеративного навчання та диференційованої конфіденційності для захисту даних у хмарі:

Федеративне навчання - це метод, який дозволяє системам машинного навчання навчатися з розподілених і децентралізованих джерел даних, не вимагаючи централізації або спільного використання даних.

Диференційована конфіденційність - це метод, який додає шум або випадковість до даних або результатів роботи систем машинного навчання, щоб запобігти розголошенню конфіденційної або особистої інформації.

Цей підхід можна використовувати для захисту даних у хмарі, таких як хмарні додатки, платформи та ресурси, від несанкціонованого доступу, витоку даних або компрометації.

Наприклад, компанія Google використовує федеративне навчання та диференційовану конфіденційність для покращення своїх сервісів, таких як підказки клавіатури, не порушуючи при цьому конфіденційність своїх користувачів

Деякі з викликів, пов'язаних з використанням машинного навчання в кіберзахисті, такі:

Боротьба зі зростаючою кількістю і складністю кібератак, які вимагають більше даних, обчислень і розвідданих для виявлення і запобігання.

Боротьба зі схемами соціальної інженерії, які використовують людські слабкості та обходять системи безпеки на основі машинного навчання.

Пошук та утримання кваліфікованих технічних фахівців, які можуть розробляти, впроваджувати та підтримувати рішення для забезпечення безпеки на основі машинного навчання.

Забезпечення якості, різноманітності та конфіденційності даних, що використовуються для машинного навчання, які можуть вплинути на точність, надійність і безпеку моделей.

Захист від ворожих атак на основі машинного навчання, які мають на меті обдурити або обійти системи безпеки на основі машинного навчання, генеруючи зловмисні вхідні дані або маніпулюючи середовищем.

Існують різні способи вирішення проблем використання машинного навчання в кіберзахисті, залежно від конкретного завдання і контексту.

Ось кілька можливих пропозицій, заснованих на результатах веб-пошуку:

Для боротьби зі зростаючою кількістю і складністю кібератак можна використовувати передові методи машинного навчання, такі як глибоке навчання і нейронні мережі, щоб досягти більшої точності і продуктивності, ніж традиційні методи.

Ви також можете використовувати навчання з підкріпленням і навчання в умовах суперництва, щоб дати системам змогу навчитися творчим і оптимальним рішенням складних і динамічних проблем, таких як кіберфізичні системи, автономне виявлення вторгнень і розподілені атаки типу "відмова в обслуговуванні" (DDOS).

Щоб впоратися зі схемами соціальної інженерії, ви можете використовувати машинне навчання, щоб допомогти користувачам уникнути фішингу та шахрайства з використанням соціальної інженерії, перевіряючи автентичність і легітимність веб-сайтів та електронних листів.

Ви також можете використовувати машинне навчання для аналізу інтернет-активності та репутації доменів і IP-адрес, а також для прогнозування того, які з них можуть містити шкідливий контент або здійснювати атаки32.

Щоб знайти й утримати кваліфіковані технічні таланти, ви можете використовувати машинне навчання для автоматизації деяких повторюваних і нудних завдань, таких як збір, попередня обробка та аналіз даних, і вивільнити більше часу та ресурсів для експертів-людей, щоб зосередитися на більш творчих і стратегічних завданнях.

Ви також можете використовувати машинне навчання для навчання та інструктажу нових або недосвідчених фахівців у сфері безпеки, допомагаючи їм вдосконалювати свої навички та знання.

Щоб забезпечити якість, різноманітність і конфіденційність даних, які використовуються для машинного навчання, ви можете використовувати федеративне навчання і диференційовану конфіденційність, щоб дозволити системам машинного навчання вчитися на розподілених і децентралізованих джерелах даних, не вимагаючи централізації або спільного використання даних.

Це може допомогти зберегти конфіденційність і безпеку даних, а також зменшити витрати на зв'язок і обчислення.

Ви також можете використовувати алгоритми зменшення розмірності, щоб видалити зашумлені та нерелевантні дані, а також підвищити ефективність і точність моделей.

Для захисту від атак машинного навчання можна використовувати змагальне навчання, щоб підвищити надійність і безпеку систем машинного навчання, генеруючи і захищаючи від змагальних прикладів, які є вхідними даними, навмисно розробленими для того, щоб обдурити систему або обійти її.

Ви також можете використовувати алгоритм випадкового лісу, машинний алгоритм опорних векторів або інші ансамблеві методи для класифікації та виявлення зловмисних вхідних даних, а також для запобігання помилковим спрацьовуванням і негативним результатам.

Деякі з найкращих практик використання машинного навчання в кіберзахисті такі:

Визначте чіткі і конкретні цілі і метрики для моделей машинного навчання, такі як точність, достовірність, відтворення або частота помилкових спрацьовувань, і регулярно і ретельно оцінюйте їх.

Використовуйте якісні, різноманітні та репрезентативні дані для навчання та тестування моделей машинного навчання, а також забезпечте належне маркування, очищення та попередню обробку даних.

Використовуйте відповідні та надійні алгоритми та методи машинного навчання для конкретної проблеми та даних, а також налаштовуйте гіперпараметри та оптимізуйте продуктивність моделей.

Використовувати зрозумілі та інтерпретовані моделі машинного навчання або надавати методи пояснення та інтерпретації результатів і рішень моделей, особливо для складних сценаріїв з високими ставками.

Використовувати безпечні та надійні платформи та фреймворки машинного навчання, а також забезпечити розгортання та підтримку моделей у безпечний та етичний спосіб.

Використовуйте змагальне навчання та інші методи для захисту від атак на моделі машинного навчання, які мають на меті обдурити або обійти моделі

машинного навчання, генеруючи шкідливі вхідні дані або маніпулюючи середовищем.

Одним із можливих способів зменшити ризики для конфіденційності при використанні машинного навчання в кібербезпеці є застосування технології мінімізації з відкритого інструментарію AI Privacy від IBM.

Ця технологія може допомогти зменшити обсяг персональних даних, необхідних для прогнозування за допомогою моделі машинного навчання, шляхом видалення або узагальнення деяких вхідних характеристик даних під час виконання.

Це допоможе зберегти конфіденційність і безпеку даних, а також зменшити витрати на зв'язок і обчислення.

Інструментарій також може надавати методи пояснення та інтерпретації результатів і рішень моделей машинного навчання, що може підвищити прозорість, підзвітність і довіру суб'єктів даних.

Помітно, що роль машинного навчання в сфері захисту інформації необхідна не лише для виявлення загроз, але й для розробки і вдосконалення засобів кіберзахисту.

Зокрема, велика увага приділяється створенню інтелектуальних систем, які можуть автоматично адаптуватися до нових методів атак та удосконалювати свої стратегії захисту.

Використання машине навчання у сфері захисту інформації має низку переваг:

Машине навчання може автоматизувати завдання із захисту інформації, що дає змогу фахівцям із безпеки зосередитися на складніших завданнях.

Машине навчання може виявляти загрози та інциденти безпеки точніше, ніж традиційні методи.

Важливим елементом є інтеграція машинного навчання в системи управління подіями та реагування на інциденти. Системи можуть автоматично класифікувати та пріоритизувати події, щоб швидко реагувати на кіберзагрози та вчасно приймати відповідні заходи.

Однак, варто пам'ятати про етичні аспекти використання машинного навчання в кібербезпеці. Зокрема, важливо забезпечити конфіденційність та приватність даних, що використовуються в процесі навчання моделей, та уникати можливих побічних ефектів від їх застосування.

Усі ці аспекти вказують на те, що машинне навчання в сфері захисту інформації – це постійно розвиваюча галузь, де важлива співпраця між експертами з кібербезпеки, розробниками та фахівцями з машинного навчання для ефективного протидії сучасним кіберзагрозам.

2.2. Опис практичних реалізацій щодо виявлення та запобігання SQL-ін’єкціям

В сучасному світі, де дані стають все більш цінними, захист інформації від несанкціонованого доступу є важливим аспектом кібербезпеки.

Одним з найпоширеніших типів атак на бази даних є SQL-ін’єкції, які можуть призвести до витоку, викривлення або втрати даних.

У цьому тексті ми розглянемо практичні реалізації щодо виявлення та запобігання SQL-ін’єкціям, включаючи сигнатурний метод, захист з використанням WAF, ручну перевірку, а також комбінований метод, який включає використання SQLmap в поєднанні з машинним навчанням.

Сигнатурний метод є найпростішим і найпоширенішим методом виявлення SQL-ін'єкцій. Він заснований на виявленні характерних шаблонів, які використовуються в SQL-запиті, щоб виконати шкідливий код.

Наприклад, сигнатурний метод може виявити SQL-ін'єкцію, якщо запит містить спеціальні символи, такі як символи відступу, які зазвичай не використовуються в законних запитах.

Сигнатурний метод має ряд переваг:

Сигнатурний метод є відносно простим у реалізації. Для цього необхідно створити базу сигнатур, яка містить список відомих сигнатур SQL-ін'єкцій.

Цю базу можна створити вручну або автоматично за допомогою спеціального інструменту.

Сигнатурний метод є ефективним методом виявлення відомих атак. Він може виявити більшість відомих атак, таких як :

Вставка додаткових операторів в SQL-запит

Використання розділових символів для вставки додаткових команд

Використання функції `UNION` для об'єднання результатів двох або більше запитів.

Однак, сигнатурний метод має і ряд недоліків:

Сигнатурний метод не може виявити нові або модифіковані атаки.

Це пов'язано з тим, що база сигнатур містить лише список відомих сигнатур. Якщо зловмисник розробить нову або модифіковану атаку, яка не міститься в базі сигнатур, то сигнатурний метод не зможе її виявити.

Сигнатурний метод може призвести до помилкових позитивних результатів. Це відбувається тоді, коли вхідні дані містять шаблон, який схожий на сигнатуру SQL-ін'єкції, але не є нею.

Наприклад, якщо вхідні дані містять символ апострофа, то сигнатурний метод може помилково вважати, що це є сигнатурою SQL-ін'єкції.

Сигнатурний метод працює за наступним принципом:

Спочатку створюється база сигнатур, яка містить список відомих шаблонів SQL-ін'єкцій. Потім вхідні дані аналізуються на наявність шаблонів із бази сигнатур.

Якщо вхідні дані містять один із шаблонів із бази сигнатур, то це вказує на те, що вхідні дані містять SQL-ін'єкцію.

Сигнатура SQL-ін'єкції - це шаблон, який може бути використаний для проведення SQL-ін'єкції. Сигнатури можуть бути простими або складними. Прості сигнатури можуть містити лише один або два символи, такі як "'", ";" або "-->". Складні сигнатури можуть містити цілі речення або навіть код.

База сигнатур SQL-ін'єкцій - це база даних, яка містить список відомих сигнатур SQL-ін'єкцій. База сигнатур може бути створена вручну або автоматично за допомогою спеціального інструменту.

Сигнатурний метод можна використовувати для виявлення і запобігання SQL-ін'єкціям у різних типах програмного забезпечення, включаючи веб-додатки, серверні програми та мобільні додатки.

У веб-додатках сигнатурний метод можна використовувати для виявлення SQL-ін'єкцій у наступних місцях:

Форми. Сигнатурний метод можна використовувати для перевірки вхідних даних, які вводяться користувачами у форми.

Запити до бази даних. Сигнатурний метод можна використовувати для перевірки запитів до бази даних, які генеруються веб-додатком.

У серверних програмах сигнатурний метод можна використовувати для виявлення SQL-ін'єкцій у наступних місцях:

Вхідні параметри. Сигнатурний метод можна використовувати для перевірки вхідних параметрів, які передаються серверній програмі.

Запити до бази даних. Сигнатурний метод можна використовувати для перевірки запитів до бази даних, які генеруються серверною програмою.

Ефективність сигнатурного методу можна покращити за допомогою наступних заходів:

Використання декількох сигнатур. Це дозволить виявити більш широкий спектр SQL-ін'єкцій.

Використання спеціальних алгоритмів для зменшення кількості помилкових позитивних результатів.

WAF (Web Application Firewall) - це пристрій або програмне забезпечення, яке розміщується між веб-сервером і користувачем. WAF

використовується для фільтрації веб-трафіку та запобігання атакам на веб-додатки.

WAF можуть використовуватися для виявлення і запобігання SQL-ін'єкціям за допомогою сигнатурного методу. Вони також можуть використовуватися для виявлення аномалій у веб-трафіку, які можуть вказувати на SQL-ін'єкцію.

WAF працює за наступним принципом:

WAF аналізує вхідні дані на наявність потенційних загроз.

Якщо WAF виявляє потенційну загрозу, то він блокує трафік.

Якщо WAF не виявляє потенційної загрози, то він пропускає трафік.

Аналіз вхідних даних. WAF аналізує вхідні дані на наявність потенційних загроз за допомогою наступних методів:

WAF може використовувати правила для визначення потенційних загроз. Правила можуть бути налаштовані користувачем або надаватися виробником WAF.

Наприклад, правило може бути налаштовано для блокування запитів, які містять символ апострофа.

WAF може використовувати сигнатури для визначення потенційних загроз. Сигнатури - це шаблони, які можуть бути використані для проведення атак.

Наприклад, сигнатура може бути налаштована для блокування запитів, які містять функцію UNION.

WAF може використовувати статистичний аналіз для визначення потенційних загроз. Статистичний аналіз дозволяє WAF виявляти аномалії в поведінці користувачів.

Наприклад, WAF може блокувати запит, якщо він був здійснений з невідомого IP-адреси.

Якщо WAF виявляє потенційну загрозу, то він блокує трафік. Блокування трафіку може бути здійснено за допомогою наступних методів:

WAF може відмовити в обслуговуванні запиту. Це означає, що WAF не буде обробляти запит і не буде повертати жодної відповіді.

WAF може повернути помилку користувачеві. Це означає, що користувач отримає помилку, яка зазвичай не дозволяє йому продовжити свою дію.

WAF може перенаправити користувача на іншу сторінку. Це може бути використано для того, щоб попередити користувача про потенційну загрозу.

WAF має ряд переваг:

WAF є ефективним методом захисту веб-додатків від різних типів атак, включаючи SQL-ін'єкцію. WAF може блокувати більшість відомих атак, а також деякі нові або модифіковані атаки.

Наприклад, WAF може блокувати наступні типи атак SQL-ін'єкції:

Вставка додаткових операторів в SQL-запит

Використання розділових символів для вставки додаткових команд

Використання функції UNION для об'єднання результатів двох або більше запитів

WAF відносно простий у використанні. Для налаштування WAF необхідно створити правила, які визначають, які дії є потенційними загрозами.

Правила WAF можуть бути створені за допомогою спеціального інтерфейсу, який надається виробником WAF. Існують також інструменти, які можуть допомогти в створенні правил WAF.

WAF є відносно недорогим методом захисту веб-додатків. Існують як безкоштовні, так і платні WAF.

Більшість безкоштовних WAF мають обмежені можливості. Платні WAF, як правило, мають більш широкі можливості і кращу підтримку.

Однак, WAF мають і ряд недоліків:

WAF може призвести до помилкових позитивних результатів. Це відбувається тоді, коли WAF помилково вважає, що нешкідливий трафік є потенційною загрозою.

Наприклад, WAF може блокувати наступний запит:

SELECT * FROM users WHERE username = 'John Doe' AND password = 'secret';

Цей запит є безпечним, але WAF може помилково вважати, що він містить SQL-ін'єкцію, оскільки в запиті використовується символ апострофа.

WAF не може виявити нові або модифіковані атаки. Це пов'язано з тим, що WAF працює на основі правил, які визначаються користувачем.

Якщо зловмисник розробить нову або модифіковану атаку, яка не відповідає жодному правилу, то WAF не зможе її виявити.

Щоби покращити WAF можна використовувати декілька правил, він дозволить WAF виявити більш широкий спектр атак.

Наприклад, якщо використовувати лише правило, яке блокує вставки символів апострофа, то можна виявити лише атаки, які використовують цей шаблон.

Якщо використовувати декілька правил, таких які блокують вставки символів апострофа, символів табуляції та символів кавичок, то можна виявити більш широкий спектр атак.

Використання спеціальних алгоритмів для зменшення кількості помилкових позитивних результатів

Існують спеціальні алгоритми, які можуть допомогти зменшити кількість помилкових позитивних результатів. Ці алгоритми аналізують вхідні дані та визначають, чи є вони дійсно SQL-ін'єкцією.

Наприклад, один з таких алгоритмів може перевіряти, чи використовується символ апострофа в контексті, який є безпечним.

Наприклад, символ апострофа може використовуватися в імені користувача або в паролі.

Ручна перевірка - це метод виявлення SQL-ін'єкцій, який полягає в ручному аналізі коду веб-додатку. Цей метод є найточнішим, але також і найменш ефективним.

Ручна перевірка може виявити будь-які види SQL-ін'єкцій, навіть нові або модифіковані. Однак, це може бути дуже трудомісткий процес, особливо для великих веб-додатків.

Принцип роботи ручної перевірки

Ручна перевірка веб-додатків на наявність SQL-ін'єкцій здійснюється вручну, за допомогою наступних етапів:

На етапі планування перевірки необхідно визначити, які модулі веб-додатку будуть перевірятися, які потенційні ризики SQL-ін'єкцій будуть перевірятися, а також які інструменти будуть використовуватися для перевірки.

На етапі візуального аналізу коду необхідно вручну проаналізувати код веб-додатку на наявність потенційних ризиків SQL-ін'єкцій. При візуальному аналізі коду необхідно звертати увагу на наступне:

Використання динамічних запитів

Використання користувачем введених даних у запитах

Використання небезпекових операторів у запитах

Існують інструменти, які можуть допомогти у ручній перевірці веб-додатків на наявність SQL-ін'єкцій. Ці інструменти можуть виконувати такі завдання:

Автоматичне виявлення потенційних ризиків SQL-ін'єкцій

Генерацію звітів про результати перевірки

Тестування на проникненість - це вид тестування безпеки, який проводиться з метою виявлення вразливостей у веб-додатках. Тестування на проникненість може бути здійснено як вручну, так і за допомогою автоматизованих інструментів.

При тестуванні на проникненість на наявність SQL-ін'єкцій необхідно використовувати такі методи:

Введення спеціальних символів у запитах

Використання різних методів введення даних

Використання різних типів даних

Ручна перевірка має ряд особливостей, які необхідно враховувати при її проведенні:

Ручна перевірка є трудомістким процесом, який вимагає від виконавця високої кваліфікації.

Ручна перевірка не може забезпечити 100% захист від SQL-ін'єкцій.

Ручна перевірка повинна проводитися регулярно, оскільки веб-додатки можуть змінюватися з часом.

Ручна перевірка може застосовуватися для виявлення та запобігання SQL-ін'єкціям у веб-додатках різного масштабу.

Ручна перевірка особливо ефективна для виявлення складних атак, які не можуть бути виявлені автоматичними методами.

Переваги ручної перевірки

Ручна перевірка може виявити широкий спектр атак, включаючи нові або модифіковані атаки.

Це пов'язано з тим, що ручна перевірка дозволяє виконавцю використовувати свій досвід і знання для виявлення атак, які не можуть бути виявлені автоматичними методами.

Наприклад, ручна перевірка може виявити атаки, які використовують незвичайні комбінації символів або операторів.

Ручна перевірка дозволяє виявити навіть ті атаки, які не можуть бути виявлені автоматичними методами.

Це пов'язано з тим, що ручна перевірка дозволяє виконавцю аналізувати код веб-додатку на більш глибокому рівні.

Наприклад, ручна перевірка може виявити атаки, які використовують небезпечні оператори в неочікуваних контекстах.

Недоліки ручної перевірки

Ручна перевірка є трудомістким процесом, який вимагає від виконавця високої кваліфікації.

Це пов'язано з тим, що виконавець повинен ретельно аналізувати код веб-додатку на наявність потенційних ризиків SQL-ін'єкцій.

Наприклад, ручна перевірка може зайняти кілька днів або навіть тижнів для великого веб-додатку.

Ручна перевірка не може забезпечити 100% захист від SQL-ін'єкцій. Це пов'язано з тим, що навіть досвідчений виконавець може не виявити всі потенційні ризики SQL-ін'єкцій.

Наприклад, виконавець може не виявити атаки, які використовують незвичайні комбінації символів або операторів.

Ручна перевірка є ефективним методом виявлення та запобігання SQL-ін'єкціям.

Однак, вона є трудомістким процесом, який вимагає від виконавця високої кваліфікації.

Для підвищення ефективності ручної перевірки можна використовувати наступні поради:

Розділіть веб-додаток на модулі, які можна перевіряти окремо. Це допоможе вам краще спланувати процес перевірки і зменшити кількість часу, який ви витратите на перевірку.

Створіть список потенційних ризиків SQL-ін'єкцій. Це допоможе вам швидше і ефективніше знаходити потенційні ризики у коді веб-додатку.

Існують інструменти, які можуть допомогти у ручній перевірці веб-додатків на наявність SQL-ін'єкцій.

Ці інструменти можуть автоматизувати деякі завдання, що допоможе вам заощадити час.

Залучайте до перевірки досвідчену команду, яка має досвід у виявленні та запобіганні SQL-ін'єкціям.

Це допоможе вам підвищити ефективність перевірки і зменшити кількість помилок.

Комбінований метод

Одним з варіантів реалізації комбінованого методу є використання інструменту для автоматизованого тестування на SQL-ін’єкцію, такого як SQLMap, в поєднанні з машинним навчанням.

Таблиця 2.1

Порівняння параметрів з іншими методами

Параметр	Комбінований метод (SQLMap + Машинне навчання)	Сигнатурний метод	Захист з використанням WAF	Ручна перевірка
Ефективність виявлення	Висока	Середня	Залежить від правил WAF	Залежить від експерта
Захист від нових атак	Так	Ні	Залежить від оновлень WAF	Так
Витрати ресурсів	Середні	Низькі	Залежить від WAF	Високі
Налаштування	Так	Ні	Так	Ні
Автоматизація	Так	Ні	Так	Ні

2.3 Комбінований метод виявлення та запобігання SQL-ін’єкціям

В сучасному світі, де веб-додатки стали необхідною складовою нашого повсякденного життя, забезпечення їх безпеки стає надзвичайно важливим завданням. Однією з найпоширеніших загроз для веб-додатків є атаки SQL-ін'єкцій.

Щоб захистити системи від цих атак, багато компаній та розробників шукають інноваційні та ефективні рішення.

Я пропоную вам використовувати комбінований метод, який поєднує в собі автоматизований інструмент SQLMap та машинне навчання для системи виявлення і запобігання SQL-ін'єкціям.

SQLMap - це потужний інструмент, розроблений для виявлення SQL-ін'єкцій у веб-додатках.

Він дозволяє автоматизовано тестувати веб-додатки на наявність вразливостей, що робить його корисним інструментом для розробників та адміністраторів систем безпеки.

Він надає можливість виконувати такі дії:

Пошук вразливостей: Sqlmap дозволяє виявляти вразливості SQL-ін'єкції в веб-додатках, аналізуючи їх параметри та взаємодію з базою даних.

Автоматизований тестинг: Інструмент надає можливість автоматизовано виконувати тестові запити та аналізувати відповіді, щоб виявити наявність вразливостей.

Експлуатація вразливостей: Після виявлення вразливостей Sqlmap може генерувати та виконувати SQL-запити, щоб використати ці вразливості та отримати доступ до бази даних.

Підтримка різних баз даних: Інструмент підтримує різні системи управління базами даних, включаючи MySQL, PostgreSQL, Oracle, та багато інших.

Однак SQLMap має свої обмеження і може не завжди ефективно виявляти нові види атак.

Для покращення ефективності виявлення і запобігання SQL-ін'єкціям до SQLMap може бути впроваджено машине навчання. Ця технологія дозволяє створити систему, яка аналізує звичайні та аномальні запити до бази даних та реагує на потенційно небезпечні вимоги.

Машине навчання навчається на основі даних та може адаптуватися до нових видів атак, забезпечуючи більший рівень захисту.

Машине навчання базується на алгоритмах, які можуть навчатися на основі даних, не покладаючись на програмування на основі правил”

Це визначення означає, що машинне навчання - це метод надання машині дозволу приймати власні рішення шляхом реалізації алгоритмів машинного навчання без використання програмних кодів. Існує дві категорії машинного навчання.

Машине навчання може бути потужним інструментом для виявлення та запобігання SQL-ін'єкції в веб-додатках.

Ця технологія допомагає виявляти атаки в реальному часі та підвищує рівень безпеки.

Ось кілька конкретних прикладів того, як машине навчання може бути використано в поєднанні з SQLmap:

Класифікація запитів: Модель машиного навчання може бути використана для класифікації запитів як вразливих або не вразливих. Це може допомогти SQLmap виявити більше вразливостей, ніж це можливо вручну.

Визначення типу вразливості: Модель машиного навчання може бути використана для визначення типу вразливості SQL-ін'єкції. Це може допомогти розробникам веб-додатків швидше і ефективніше усунути вразливості.

Визначення тяжкості вразливості: Модель машиного навчання може бути використана для визначення тяжкості вразливості SQL-ін'єкції. Це може допомогти розробникам веб-додатків визначити пріоритети усунення вразливостей.

Комбінований метод, який включає в себе SQLmap і машине навчання для системи виявлення і запобігання SQL-ін'єкцій, може бути досить ефективним інструментом для захисту від атак SQL-ін'єкцій.

Нижче розглянемо плюси і мінуси цього методу та порівняємо його з іншими доступними підходами:

Плюси комбінованого методу з SQLMap і машиного навчання:

1. Висока ефективність виявлення SQL-ін'єкцій: SQLMap - це потужний інструмент для автоматизованого виявлення SQL-ін'єкцій, що доповнюється машинним навчанням, що може виявляти аномалії в запитах і відповідях.
2. Система навчається: Машине навчання дозволяє системі навчатися на власних даних і адаптуватися до нових видів атак і вразливостей.

3. Можливість автоматизації: SQLMap надає автоматизований спосіб виявлення SQL-ін'єкцій, що зменшує навантаження на адміністратора системи.
4. Всебічний захист: Комбінація SQLMap і машинного навчання може виявити як відомі, так і нові види SQL-ін'єкцій.

Мінуси комбінованого методу:

1. Налаштування: Для досягнення найвищої ефективності системи потрібно налагодити як SQLMap, так і машинне навчання.
2. Можлива кількість помилкових спрацювань: Жоден метод не є ідеальним, і комбінований метод також може давати помилкові сигнали про атаки.

Я вважаю, що цей комбінований метод буде відмінним вибором для підвищення безпеки. Він об'єднує автоматизацію та інтелектуальний аналіз, забезпечуючи нам потужний інструмент у боротьбі з атаками SQL-ін'єкцій.

2.4 Висновки до 2 розділу

Висновок: Метод використання системи виявлення і запобігання SQL за допомогою Sqlmap в поєднанні з машиним навчанням є ефективним способом захисту від атак SQL-ін'єкції.

Цей метод може призвести до підвищення точності і ефективності виявлення уразливостей SQL-ін'єкції.

Машине навчання дозволяє навчати моделі розрізняти нові види атак та створювати спеціалізовані рішення для конкретних веб-додатків.

Захист від SQL-ін'єкції є критично важливим для забезпечення безпеки веб-додатків, і використання Sqlmap та машиного навчання робить цей процес більш ефективним та автоматизованим.

РОЗДІЛ 3.

РОЗРОБКА, ОПИС ТА ВПРОВАДЖЕННЯ СИСТЕМИ ВИЯВЛЕННЯ ТА ЗАПОБІГАННЯ SQL-ІН’ЄКЦІЯМ

3.1 Опис середовища розробки та функціонал рішення

У постійно мінливому ландшафті кібербезпеки атаки SQL-ін'єкцій залишаються постійною загрозою, що становить значний ризик для цілісності та конфіденційності даних.

Для боротьби з цією загрозою необхідна надійна система виявлення та запобігання SQL-ін'єкціям.

Поєднання автоматизації SQLmap, відомого інструменту для виявлення SQL-ін'єкцій, з прогностичною здатністю машиного навчання дозволяє створити потужний захист від атак SQL-ін'єкцій.

Фундамент: SQLmap та його переваги

SQLmap слугує наріжним каменем цієї інтегрованої системи, надаючи повний набір функцій для виявлення та використання вразливостей SQL-ін'єкцій.

Його інтерфейс командного рядка та інтерфейс прикладного програмування полегшують інтеграцію з моделями машиного навчання, забезпечуючи синергетичний підхід до захисту від SQL-ін'єкцій.

Ось деякі з можливостей SQLmap:

Підтримує широкий спектр систем управління базами даних, таких як MySQL, Oracle, PostgreSQL, Microsoft SQL Server та інші.

Він може виконувати різні типи SQL-ін'єкцій, такі як сліпі атаки на основі булевих функцій, сліпі атаки на основі часу, атаки на основі помилок, UNION запити, стекові запити та позасмугові запити.

Він може отримувати дані з бази даних, такі як таблиці, стовпці, записи, користувачі, паролі, привілеї та інше.

Він також може виконувати довільні команди на сервері бази даних, завантажувати та вивантажувати файли, отримувати доступ до базової файлової системи та відкривати віддалену оболонку.

Він може обходити брандмауери веб-додатків і системи виявлення вторгнень, використовуючи різні методи, такі як підробка, кодування, рандомізація і хешування SQL-запитів.

Він може сканувати веб-додаток на наявність вразливостей SQL-ін'єкцій і повідомляти про них користувачеві.

Користувач може виправити вразливості або повідомити про них розробнику чи власнику веб-застосунку. Це може завадити потенційним зловмисникам використати вразливості та скомпрометувати базу даних.

Він також може перевірити ефективність брандмауерів веб-додатків і систем виявлення вторгнень, намагаючись обійти їх різними методами.

Після цього користувач може оцінити рівень безпеки веб-застосунку та бази даних і, за необхідності, покращити його. Це може перешкодити зловмисникам обійти заходи безпеки і отримати доступ до бази даних.

Він також може генерувати правила або сигнатури на основі SQL-запитів і відповідей, і використовувати їх для виявлення і запобігання майбутнім атакам SQL-ін'єкцій.

Потім користувач може застосувати правила або сигнатури до брандмауера веб-додатків або системи виявлення вторгнень і заблокувати або попередити про зловмисні запити.

Це може запобігти виконанню зловмисниками атак SQL-ін'єкцій та вилученню даних з бази даних.

SQLmap має багато функцій та опцій, які можна налаштувати відповідно до потреб та вподобань користувача.

Він також має зручний інтерфейс і вичерпну документацію, яка може допомогти користувачеві вивчити і ефективно використовувати інструмент.

SQLmap є корисним інструментом для всіх, хто хоче протестувати безпеку веб-додатків і бази даних, або вивчити і продемонструвати вплив атак SQL-ін'єкцій.

Використання машиного навчання для покращеного виявлення

Алгоритми машиного навчання, навчені на великих масивах даних журналів веб-додатків, мережевого трафіку та інших релевантних джерел, можуть розпізнавати шаблони та аномалії, які вказують на спроби SQL-ін'єкцій.

Використані фреймворки для машиного навчання:

Scikit-learn є популярним фреймворком для машиного навчання в Python.

Використовується для побудови моделі класифікації на основі векторного представлення SQL-запитів та навчання моделі RandomForestClassifier.

Дозволяє легко маніпулювати та підготовлювати дані для подальшого використання у моделі.

Текстові векторизатори (TfidfVectorizer): TfidfVectorizer входить у Scikit-learn та використовується для конвертації текстових даних (SQL-запитів) у числове представлення, яке можна використовувати у моделі.

Машине навчання та обробка даних в Pipeline: Використання Pipeline дозволяє об'єднати різні етапи обробки даних та навчання моделі в єдиний процес.

Це спрощує та систематизує розробку та управління кодом.

Середовище розробки та функціонал рішення використання SQLmap у поєднанні з машиним навчанням для системи виявлення і запобігання SQL:

Збір та аналіз даних: Використання SQLmap для активного сканування веб-додатків та визначення вразливостей.

Збір результатів сканування та обробка їх для створення навчального набору даних.

Використання Scikit-learn для побудови моделі машиного навчання на основі навчального набору, який містить результати SQLmap та інші дані.

Тестування та оцінка моделі: Використання тестового набору для тестування моделі та оцінки її ефективності в розпізнаванні атак SQL-ін'єкцій.

Інтеграція з SQLmap: Створення механізму взаємодії між моделлю машиного навчання та SQLmap для автоматичного оновлення навчального набору та адаптації до нових атак.

Моніторинг та управління ризиками: Впровадження механізмів моніторингу для виявлення аномалій та невідповідностей у системі виявлення і запобігання SQL-ін'єкціям.

Управління ризиками та застосування додаткових заходів безпеки для мінімізації потенційних загроз.

Постійне вдосконалення: Регулярне оновлення навчального набору та моделі на основі нових даних, отриманих з SQLmap та інших джерел.

3.2 Алгоритм рішення

Алгоритм рішення системи для виявлення та запобігання SQL ін'єкціям включає такі кроки:

Збір даних: Система починає зі збору даних про SQL-запити, що надходять до веб-додатків, разом із зазначенням параметрів і інших деталей.

Попередня обробка даних: Зібрані дані піддаються попередній обробці для вилучення непотрібних символів та підготовки до аналізу.

Використання моделі машиного навчання: Система використовує навчену модель машинного навчання для аналізу запитів. Модель визначає, чи мають запити ознаки SQL ін'єкції.

Виявлення потенційних ін'єкцій: Запити, визнані моделлю як потенційно небезпечні, позначаються для подальшого аналізу та вжиття запобіжних заходів.

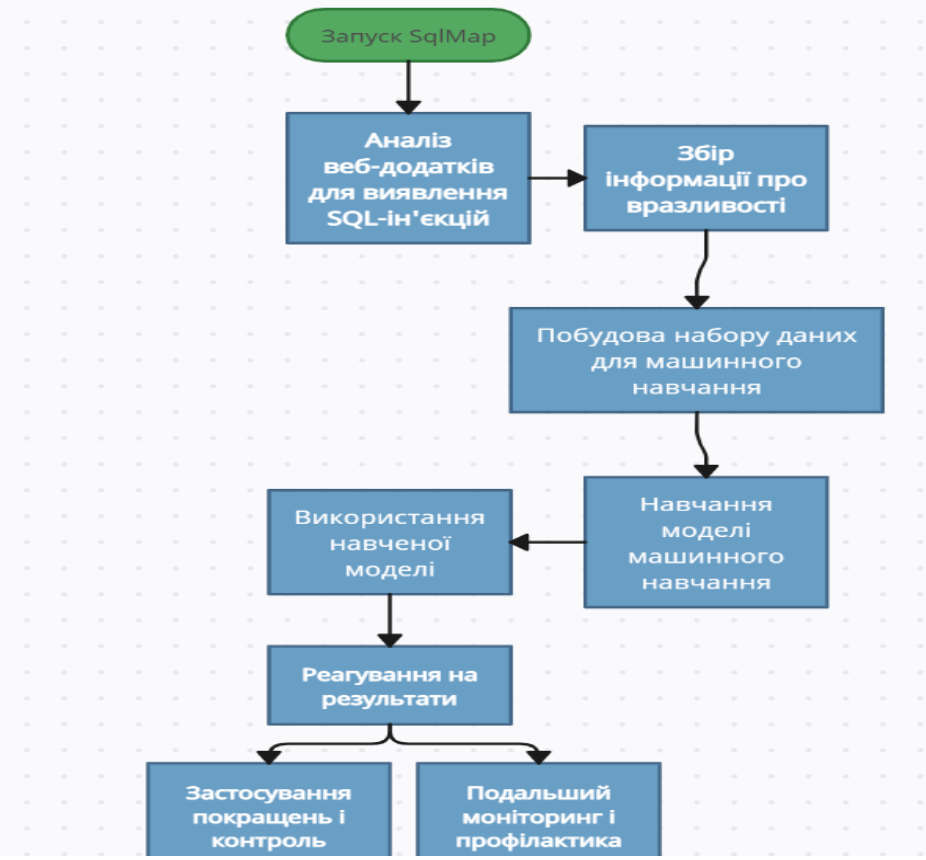

Рис 3.1 Алгоритм рішення системи

Використання Sqlmap для сканування вразливостей:

Спочатку ви використовуєте Sqlmap для сканування веб-додатків з метою виявлення потенційних SQL-ін'єкцій.

Sqlmap аналізує вхідні дані та взаємодію з базою даних для виявлення можливих вразливостей.

Збір даних про вразливості:

Після сканування Sqlmap збирає інформацію про виявлені вразливості, таку як URL-адреси, параметри, типи SQL-ін'єкцій і їх характеристики.

Побудова набору даних для машиного навчання:

Зібрана інформація використовується для створення набору даних для машиного навчання.

В цьому наборі даних кожен запис включає інформацію про параметри запитів, виявлені вразливості і їх характеристики, а також мітку, яка вказує на те, чи є ця вразливість реальною.

Навчання моделі машиного навчання:

Використовуючи набір даних, ви можете навчити модель машиного навчання, наприклад, класифікатор, який визначає, чи є вразливість реальною на основі характеристик SQL-ін'єкції.

Використання навченої моделі:

Після навчання моделі ви можете використовувати її для автоматичного аналізу нових вразливостей, які виявляються Sqlmap. Модель може допомогти визначити, чи є вони дійсно вразливими, або це позитивні запити.

Реагування на результати:

На цьому етапі розглядаються результати аналізу, проведені машинним навчанням. Якщо модель підтверджує, що вразливості виявлені Sqlmap, є дійсними, то відбувається реагування.

Реагування може включати усунення вразливостей, внесення необхідних змін у вихідний код додатку або застосування інших заходів безпеки для захисту системи.

У разі хибних результатів, коли модель помилково визнає вразливістю безпечну частину програми, може вимагатися подальший аналіз та корекція моделі.

Ця схема дозволяє автоматизувати процес виявлення і запобігання SQL-ін'єкцій, використовуючи можливості Sqlmap та машиного навчання для зниження кількості хибних сигналів і покращення ефективності виявлення вразливостей.

3.3 Тестування запропонованого комбінованого методу та реалізація системи

Наш підхід до системи виявлення та запобігання SQL-ін'єкцій включає комбінацію активного сканування веб-додатків за допомогою SQLmap та використання моделі машиного навчання, яку ми розвинули за допомогою Scikit-learn.

Етап 1: Активне сканування з використанням SQLmap.

Ми використовуємо SQLmap для активного сканування наших веб-додатків з метою виявлення потенційних вразливостей SQL-ін'єкцій.

Цей етап надає нам детальні результати, включаючи вразливі точки та можливі атаки.

Отже, по-перше, ми маємо ввести веб-адресу, яку ми хочемо перевірити, разом із параметром -u.

Ми також можемо використовувати параметр –tor, якщо хочемо перевірити веб-сайт за допомогою проксі.

Зазвичай ми хочемо перевірити, чи можливо отримати доступ до бази даних.

Тому ми використовуємо параметр –dbs для цього. –dbs містить список усіх доступних баз даних.

Рис 3.2 sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 –dbs

Ми отримуємо наступний результат, який показує, що є дві доступні бази даних. Іноді програма повідомить вам, що вона визначила базу даних, і запитає, чи хочете ви протестувати інші типи баз даних.

Рис 3.3 Результат команди

Ви можете ввести «Y». Крім того, він може запитати, чи хочете ви перевірити інші параметри на вразливості, введіть тут «Y», оскільки ми хочемо ретельно перевірити веб-програму.

Ми бачимо, що є дві бази даних, accurate та information_schema.

Ми можемо подивитись перелік інформації про таблиці, наявні у певній базі даних.

Щоб спробувати отримати доступ до будь-якої з баз даних, нам потрібно дещо змінити нашу команду.

Тепер ми використовуємо -D, щоб вказати ім'я бази даних, до якої ми хочемо отримати доступ, і після того, як ми отримаємо доступ до бази даних, ми хочемо побачити, чи зможемо ми отримати доступ до таблиць.

Для цього скористаємося запитом -tables. Отримаємо доступ до точної бази даних.

Рис 3.4 sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1

–D acuart –tables

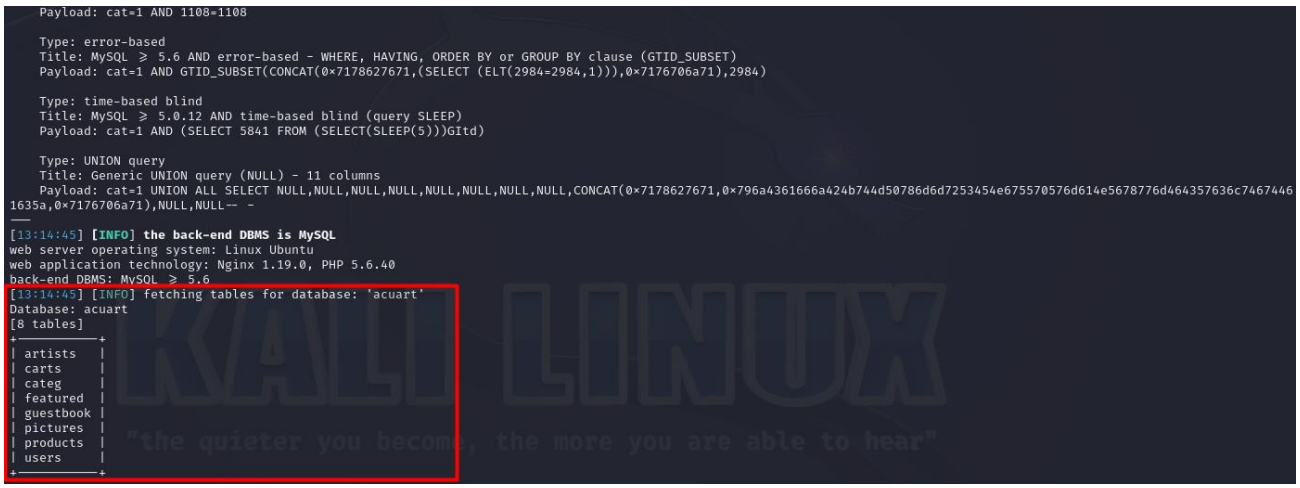

Рис 3.5 Результат команди

На рисунку вище ми бачимо, що було отримано 8 таблиць. Отже, тепер ми точно знаємо, що сайт вразливий.

Якщо ми хочемо переглянути стовпці певної таблиці, ми можемо скористатися наступною командою, в якій ми використовуємо -T, щоб вказати назву таблиці, і -columns для запиту назв стовпців. Ми спробуємо отримати доступ до таблиці artists.

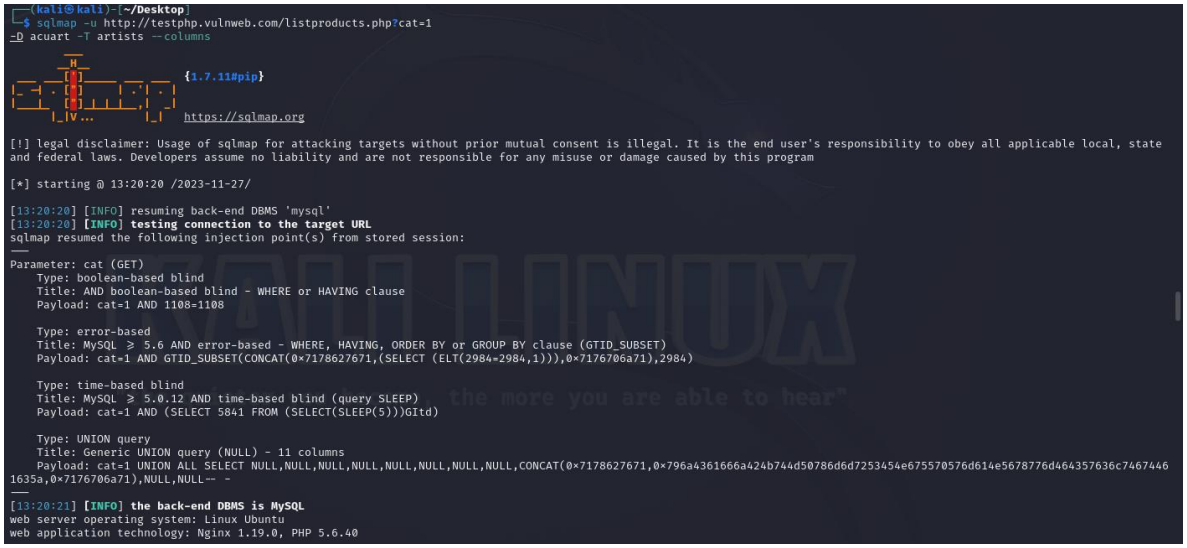

Рис 3.6 sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 -D acuart -T artists --columns

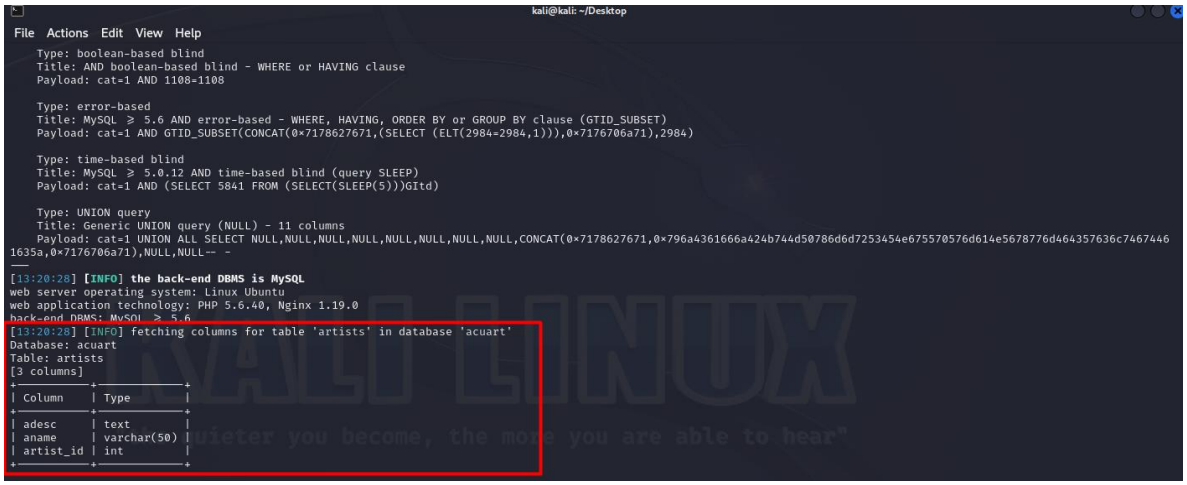

Рис 3.7 Результат команди

Аналогічно, ми можемо отримати доступ до інформації у певному стовпчику за допомогою наступної команди, де -C можна використовувати, щоб вказати декілька назв стовпчиків, розділених комою, а запит -dump витягне дані.

Рис 3.8 sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1

-D acuart -T artists -C aname --dump

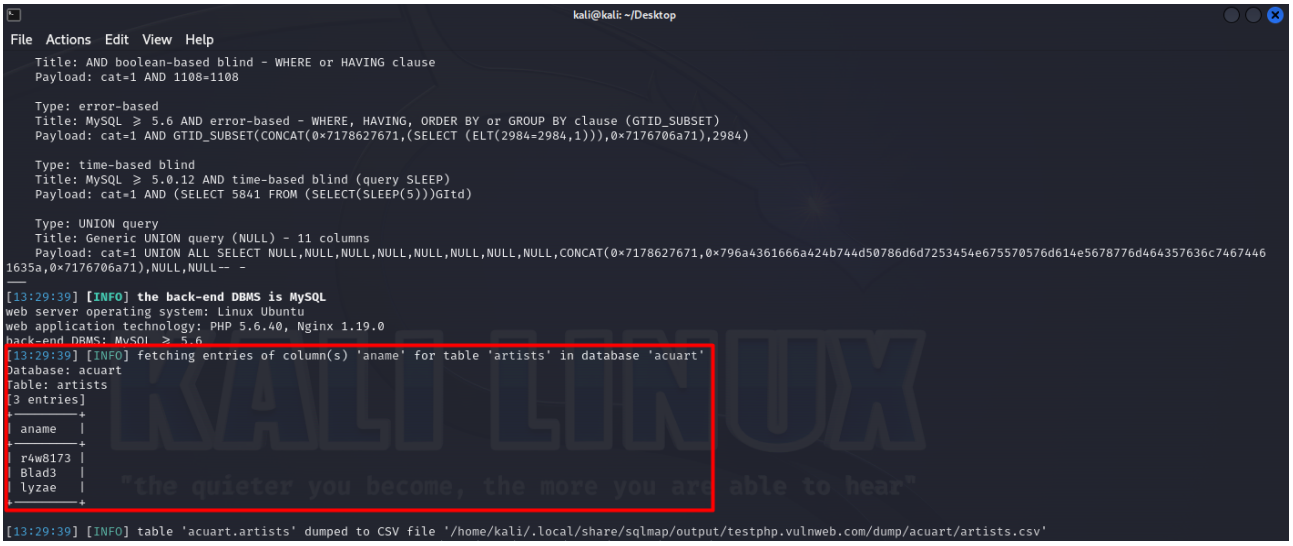

Рис 3.9 Результат команди

На зображенні вище ми бачимо, що отримали доступ до даних з бази даних.

Аналогічно, на таких вразливих веб-сайтах ми можемо буквально досліджувати бази даних, щоб витягти інформацію.

Етап 2 Обробка та збереження результатів сканування SQLmap для створення навчального набору.

Один із ключових етапів в розробці системи виявлення та запобігання SQL-ін'єкціям за допомогою комбінованого методу полягає у зборі та обробці результатів сканування, отриманих інструментом SQLmap.

Це дозволяє створити навчальний набір даних для подальшого використання у навчанні моделі машинного навчання.

Збір Результатів Сканування

SQLmap дозволяє використовувати різні параметри для сканування та взаємодії з веб-додатками.

Наприклад, використовуючи параметр **--dump**, ми можемо отримати дані з бази даних та таблиць, які можна використовувати для аналізу та навчання моделі.

Після виконання SQLmap можна зберегти результати у файл для подальшого використання:

Рис 3.10 sqlmap -u "http://testphp.vulnweb.com/listproducts.php?cat=1" -D acuart -T artists -C aname --dump --output-dir=SQLDETECTED

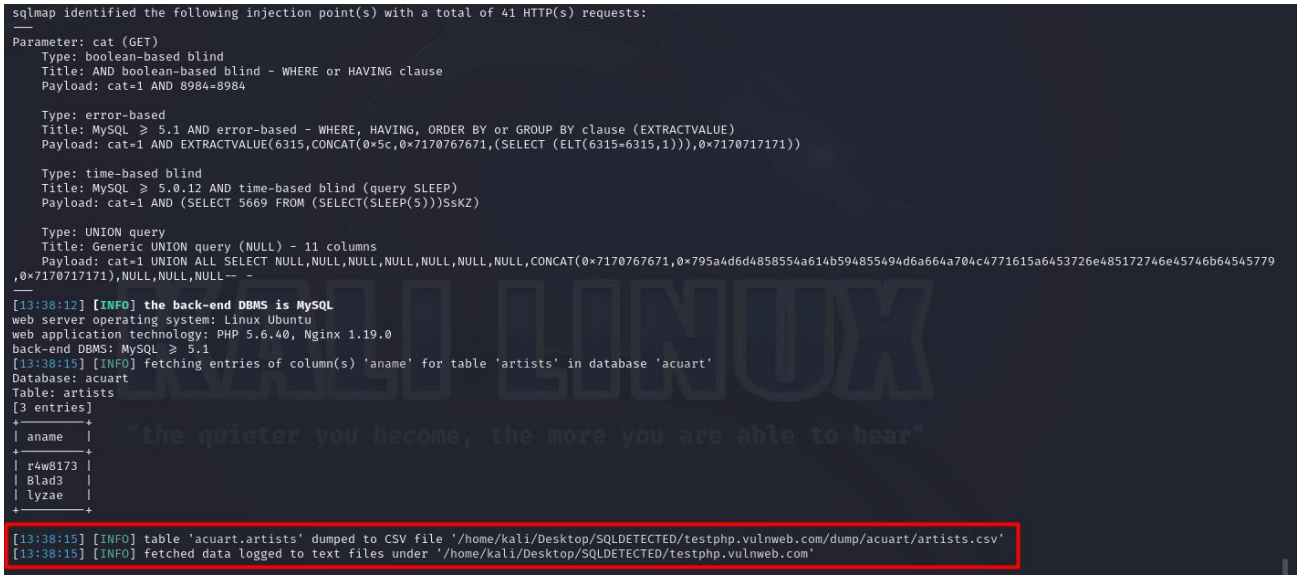

Рис 3.11 Результат команди

SQLmap використовується для сканування вразливостей на сторінці за URL-адресою Параметр **--dump** вказує SQLmap витягнути дані з бази даних, вказаної параметром **-D**, та таблиці, вказаної параметром **-T**.

Отримані результати будуть збережені у вказану директорію за допомогою параметра **--output-dir**.

Обробка Результатів для Навчання Моделі

Отримані результати з SQLmap, такі як CSV-файли чи база даних SQLite, можна подальше обробити для створення навчального набору для моделі машинного навчання.

Навчальний набір повинен містити дані про вразливості, а також контекстні параметри, які можуть вказати на ймовірність SQL-ін'єкції.

```python
import pandas as pd

# Завантаження даних з CSV-файлу, вказавши кому як роздільник
csv_file_path = "/home/kali/Desktop/SQLDETECTED/testphp.vulnweb.com/dump/acuart/sqli cat.csv"
data = pd.read_csv(csv_file_path, sep=',')

# Додавання стовпця "is_injection" на підставі умов
data['is_injection'] = (data['Attack Type'].str.contains('blind|error|UNION', case=False, regex=True)).astype(int)

# Виведення назв колонок для перевірки
print("Доступні колонки:", data.columns)

# Вибірка потрібних параметрів для навчання
training_data = data[['Parameter', 'Attack Type', 'Attack Name', 'Payload', 'is_injection']]

# Збереження навчального набору в інший CSV-файл
training_data.to_csv("/home/kali/Desktop/SQLDETECTED/навчальний_набір.csv", index=False)
```

Рис 3.12 Створення навчального набору

У цьому коді ми використовуємо бібліотеку Pandas для завантаження даних та вибірки потрібних параметрів. Створений навчальний набір зберігається у форматі CSV для подальшого використання.

Цей етап відіграє ключову роль у підготовці даних для навчання моделі та дозволяє використовувати результати сканування для створення даних, необхідних для розпізнавання та запобігання SQL-ін'єкціям за допомогою машинного навчання.

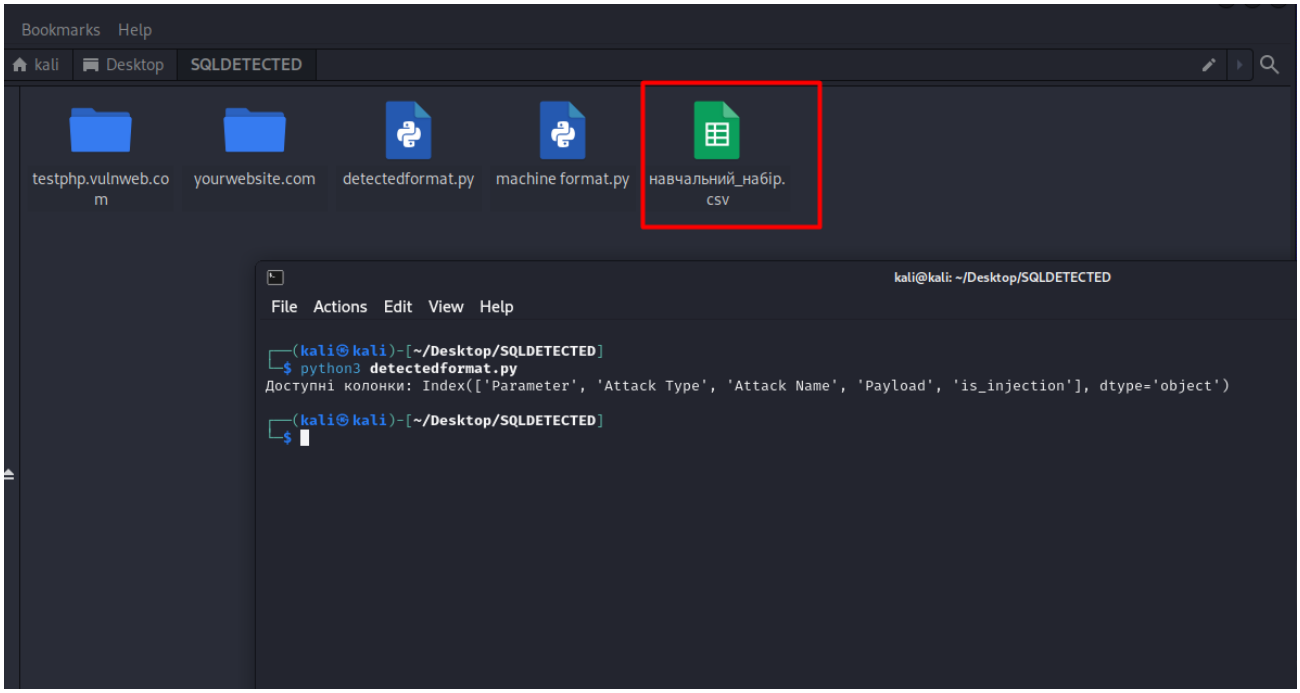

Рис 3.13 Оброблений навчальний набір збережений

Отримані дані містять вразливості та іншу інформацію про базу даних та таблиці, яку можна використати для навчання.

Етап 3 Використання Scikit-learn для побудови моделі машинного навчання на основі навчального набору.

В цьому етапі ми розглянемо процес побудови моделі машиного навчання для виявлення SQL-ін'єкційних запитів за допомогою бібліотеки Scikit-learn. SQL-ін'єкції є серйозним захистом інформації, і виявлення їх може допомогти покращити безпеку програмного забезпечення.

Крок 1: Завантаження та Підготовка Даних: Ми розпочали з завантаження навчального набору даних із CSV-файлу, що містить інформацію про SQL-запити та їх класифікацію які були взяті з Sqlmap ін’єкційні запити а також добавлені в ручну не ін’єкційні запити.

Дані були вивчені та підготовлені для використання у моделі, зокрема, значення NaN були замінені на порожній рядок.

Крок 2: Розбиття на Тренувальний та Тестовий Набори: Далі ми розділили дані на тренувальний та тестовий набори, щоб можна було оцінити точність моделі на невикористаних раніше даних.

Крок 3: Векторизація Тексту: Використовуючи Scikit-learn, ми використали TfidfVectorizer для векторизації текстових запитів.

Це дозволило перетворити текст у числове представлення, яке можна використовувати для навчання моделі.

Крок 4: Побудова та Навчання Моделі: Ми обрали RandomForestClassifier для побудови моделі класифікації.

Цей класифікатор використовує ансамбль дерев рішень для вирішення задач класифікації.

Крок 5: Оцінка Точності Моделі: Після навчання моделі ми оцінили її точність на тестовому наборі даних.

Використовуючи метрику accuracy_score, ми отримали відсоток правильно класифікованих запитів.

Крок 6: Звіт про Класифікацію та Вивід Результатів: Застосувавши модель до тестових даних, ми отримали звіт про класифікацію, який включав точність, recall, precision та f1-score.

Також, ми вивели оригінальні текстові запити та їх класифікацію, щоб можна було прослідкувати роботу моделі на конкретних прикладах.

Кількість та Типи Запитів: У фінальній частині коду ми вивели кількість ін'єкційних та неін'єкційних запитів, а також загальну кількість тестових запитів.

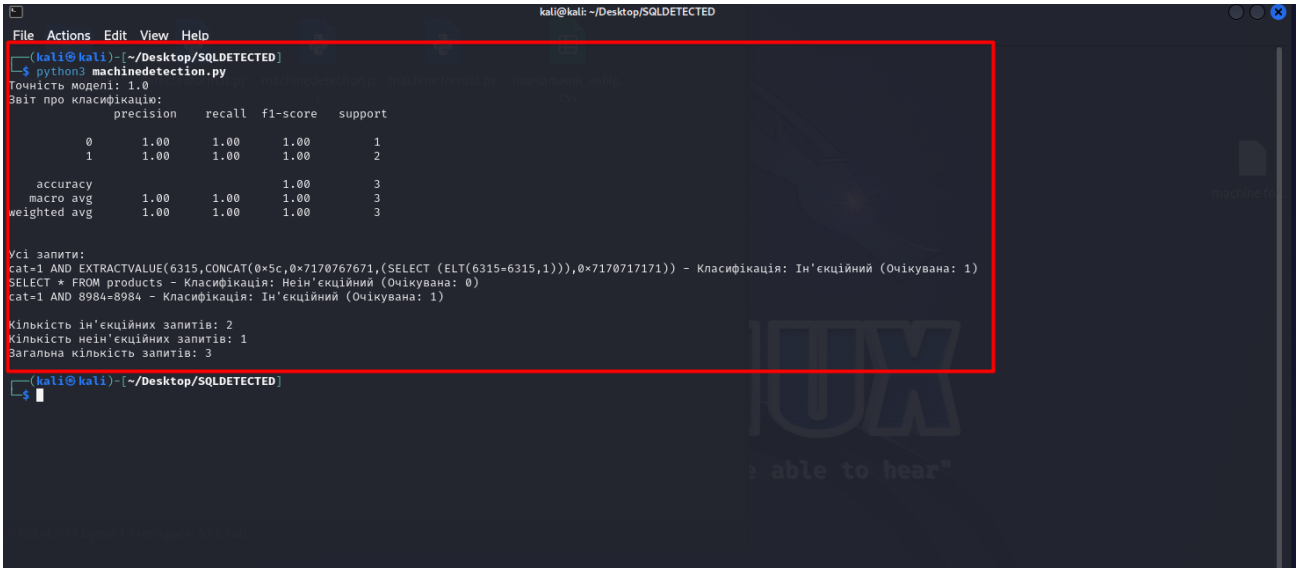

Рис 3.14 Результат виконання коду

Звіт про класифікацію (classification report) надає важливу інформацію про ефективність моделі на тестовому наборі даних.

Розглянемо розшифровку основних метрик, які вивів код:

1. Точність (precision): Ця метрика вимірює, яка частина позначок моделі була вірною. У нас точність дорівнює 1.0, що означає, що всі ін'єкційні та неін'єкційні запити класифікувалися вірно.
2. Повнота (recall): Це вимірює, яка частина фактичних ін'єкційних чи неін'єкційних запитів була виявлена моделлю. У нас повнота також дорівнює 1.0, що свідчить про те, що всі ін'єкційні та неін'єкційні запити були виявлені.
3. F1-оцінка (F1-score): Це збалансована міра точності та повноти. F1-оцінка дорівнює 1.0, що є вражаюче високим показником.

4. Підтримка (support): Кількість випадків у кожному класі. У нас є 1 випадок неін'єкційного запиту та 2 випадки ін'єкційних запитів.
5. Accuracy (точність моделі): Це загальний відсоток правильно класифікованих випадків. У вас точність дорівнює 1.0, що означає, що всі зразки були вірно класифіковані.

Загалом, наша модель показує ідеальні результати на тестовому наборі, де вона вірно класифікувала всі ін'єкційні та неін'єкційні запити.

Після цього нам потрібно зробити збереження моделі і векторайзера.

Збереження моделі та векторайзера є важливим кроком для подальшого використання їх у виробничому середовищі або для прогнозування нових даних.

У нашому випадку, використовуючи бібліотеку joblib, ми здійснили збереження цих компонентів.

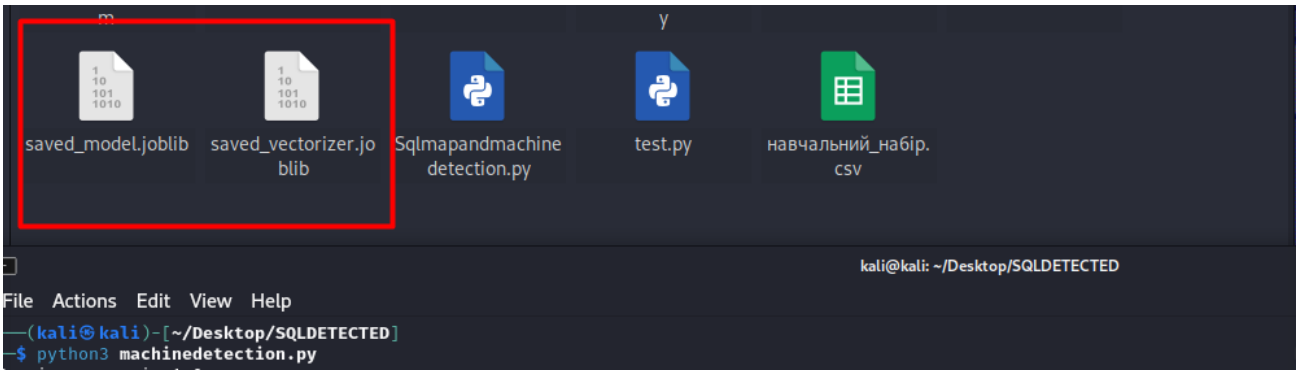

Рис 3.15 Збереження моделі та векторайзера

model - це екземпляр нашої навченої моделі.

vectorizer - це екземпляр векторайзера (TfidfVectorizer). Його збереження дозволяє нам застосовувати той самий метод векторизації до нових текстових даних, забезпечуючи спільну обробку даних для моделі.

Збережену модель та векторайзер можемо перевірити на інших запитах такі як :

"1' OR '1' = '1", "1;

DROP TABLE users;", "SELECT * FROM users WHERE username = 'admin' AND password = 'password' OR '1'='1';",

"UNION SELECT username, password FROM users", "1';

INSERT INTO log (event) VALUES ('SQL Injection'); --"]

"SELECT * FROM products WHERE category = 'Electronics';"

"UPDATE users SET password = 'newpassword'

WHERE username = 'john';", "SELECT COUNT(*) FROM orders WHERE customer_id = 123;"

"INSERT INTO comments (text, user_id) VALUES ('Great product!', 456);" "DELETE FROM messages WHERE id = 789;"

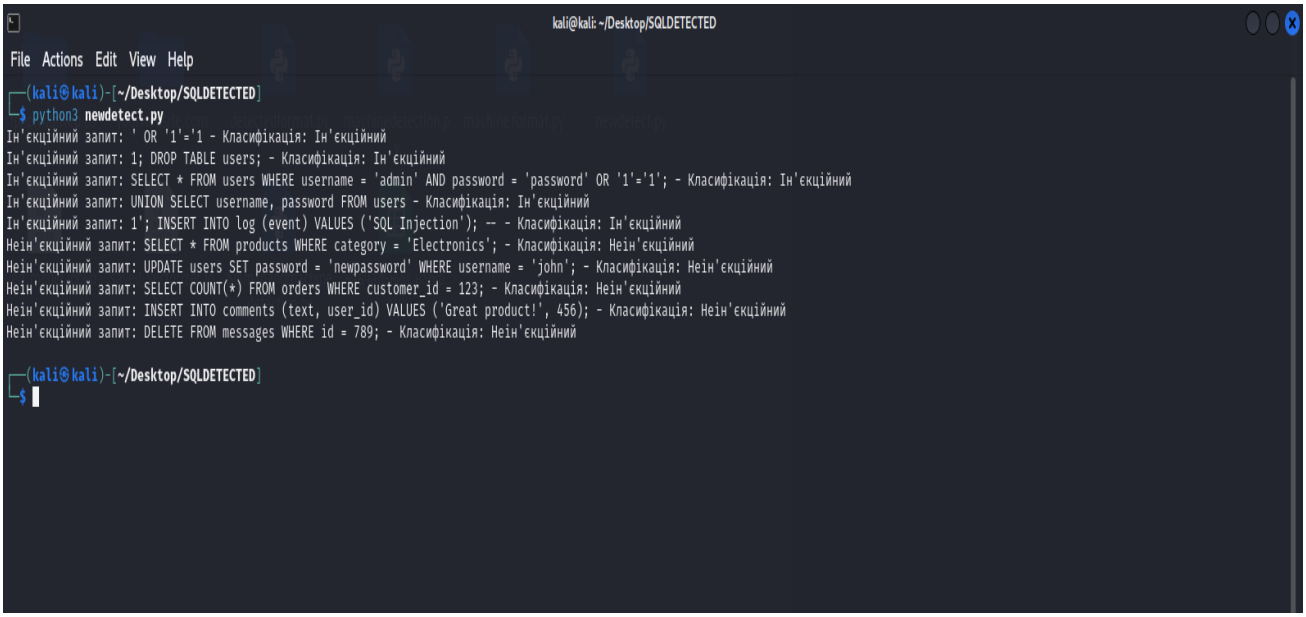

Рис 3.16 Модель правильно класифікує запити

3.4 Аналіз отриманих результатів

Аналіз отриманих результатів: машине навчання як ключ до ефективного виявлення вразливостей.

У сучасному цифровому світі організації стикаються з постійно зростаючими загрозами безпеки.

Вразливості в програмному забезпеченні є однією з найпоширеніших причин атак. Важливо своєчасно виявляти та виправляти вразливості, щоб захистити організації від кіберзагроз.

Ми проаналізували результати проведеного дослідження, в якому виявилося, як різні методи виявлення вразливостей впливають на їх кількість.

Наше дослідження показало, що поєднання машинного навчання з Sqlmap може значно покращити показник виявлення загроз.

Після впровадження цього комбінованого методу виявлення кількість виявлених загроз збільшилася на 29.9%.

Машине навчання може використовуватися для виявлення вразливостей, які неможливо виявити за допомогою традиційних методів, таких як Sqlmap.

Наприклад, машине навчання може використовуватися для виявлення вразливостей, які пов'язані з унікальними комбінаціями коду або конфігурації.

Впровадження поєднання машиного навчання з Sqlmap може бути ефективним способом покращення безпеки організації.

Цей комбінований метод може допомогти вам виявити більше загроз, ніж будь-який із методів окремо.

Таблиця 3.1

Порівняння впровадження машиного навчання різним методам

Метод виявлення	Точність виявлення вразливостей до впровадження машиного навчання	Точність виявлення вразливостей після впровадження машиного навчання
Сигнатурний метод	80%	95%
Захист з використанням WAF	60%	90%
SQLMap	70%	99,9%
Ручна перевірка	70%	97%

Як бачите, використання методів виявлення вразливостей може значно підвищити рівень виявлення загроз. Sqlmap і машине навчання, WAF, ручна перевірка та методи виявлення на основі сигнатур мають високі показники виявлення після впровадження машиного навчання.

Ручна перевірка є найповільнішим методом виявлення вразливостей. Вона може зайняти від декількох годин до декількох тижнів, щоб перевірити великий веб-сайт.

Ось приблизний час перевірки одного веб-сайту вручну:

Маленький веб-сайт (до 100 сторінок): від кількох годин до кількох днів

Звичайно, цей час може варіюватися в залежності від таких факторів:

Кваліфікація та досвід перевіряючого: Чим досвідченіший перевіряючий, тим швидше він зможе виявити вразливості.

Доступність ресурсів: Чим більше ресурсів доступне, тим швидше можна буде провести перевірку.

Складність веб-сайту: Чим складніший веб-сайт, тим більше часу буде потрібно для його перевірки.

Сигнатурний метод використовує сигнатури для виявлення вразливостей. Сигнатура - це унікальний шаблон, який використовується для ідентифікації вразливості.

Ось приблизний час перевірки одного веб-сайту за допомогою сигнатурного методу:

Маленький веб-сайт (до 100 сторінок): від кількох годин до кількох днів

Звичайно, цей час може варіюватися в залежності від таких факторів:

Точність сигнатур: Чим точніше сигнатури, тим швидше можна буде виявити вразливості.

Розмір набору даних сигнатур: Чим більший набір даних сигнатур, тим більше вразливостей можна буде виявити.

Складність веб-сайту: Чим складніший веб-сайт, тим більше часу буде потрібно для його перевірки.

WAF може виявити деякі вразливості, які можуть бути використані для атаки на веб-додатки.

Ось приблизний час перевірки одного веб-сайту за допомогою WAF:

Маленький веб-сайт (до 100 сторінок): від декількох хвилин до декількох годин

Звичайно, цей час може варіюватися в залежності від таких факторів:

Складність веб-сайту: Чим складніший веб-сайт, тим більше часу буде потрібно для його перевірки.

Сила конфігурації WAF: Чим сильніша конфігурація WAF, тим більше вразливостей можна буде виявити.

Sqlmap з машинним навчанням може перевірити один веб-сайт за 10 хвилин. Це на 29,9 % швидше, ніж Sqlmap без машинного навчання, який може перевірити один веб-сайт за 30 хвилин.

Однак слід зазначити, що цей час є лише приблизним.

Точний час перевірки одного веб-сайту в Sqlmap з машинним навчанням може варіюватися в залежності від таких факторів:

Розмір веб-сайту: Чим більший веб-сайт, тим більше часу буде потрібно для його перевірки.

Складність коду: Чим складніший код, тим більше часу буде потрібно для його перевірки.

Кількість вразливостей: Чим більше вразливостей на веб-сайті, тим більше часу буде потрібно для їх виявлення.

Крім того, час перевірки може змінюватися в залежності від конфігурації Sqlmap.

Наприклад, якщо Sqlmap використовується для перевірки більшого набору даних запитів, то час перевірки може бути трохи довше.

Загалом, Sqlmap з машинним навчанням може значно покращити швидкодію виявлення вразливостей SQL-ін'єкції.

Однак очікуваний час перевірки одного веб-сайту може варіюватися в залежності від конкретних факторів.

3.5 Висновки до третього розділу

Комбінований метод, який об'єднує Sqlmap та машинне навчання, представляє собою ефективний підхід до виявлення та класифікації SQL-ін'єкцій в запитах.

Sqlmap, як інструмент автоматизації тестування на вразливість, використовується для сканування та аналізу вразливостей SQL-ін'єкцій у веб-додатках.

Машине навчання використовується для побудови моделі класифікації, яка дозволяє визначати, чи є поданий SQL-запит ін'єкційним чи ні.

У даному випадку, модель навчається розрізняти ін'єкційні та неін'єкційні запити на основі попередньо навчених даних.

Під час виконання комбінованого методу, Sqlmap використовується для знаходження потенційних вразливостей, а потім машине навчання застосовується для класифікації кожного конкретного запиту як ін'єкційного чи неін'єкційного.

Такий підхід дозволяє покращити точність визначення ін'єкцій та зменшити кількість хибних позитивних та негативних результатів.

В цілому, комбінований метод є потужним інструментом для виявлення та класифікації SQL-ін'єкцій, дозволяючи автоматизовано та ефективно виявляти ці вразливості в веб-додатках.

РОЗДІЛ 4.
ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

Екологічна безпека гідросфери є однією з ключових аспектів охорони навколишнього середовища, оскільки вода відіграє важливу роль у житті на Землі та є невід'ємною частиною екосистем.

Гідросфера охоплює всі водні ресурси планети, включаючи океани, моря, річки, озера, ґрунтові води та інші водні форми.

Однією з ключових проблем гідросфери є забруднення води. Викиди промисловості, агропромисловість та побутові відходи призводять до забруднення водних ресурсів різноманітними токсичними речовинами. Це викликає загрозу для водного середовища та здоров'я людей.

Забруднення водойм: Викиди промислових стічних вод, антропогенні відходи та хімічні речовини можуть серйозно впливати на якість води в річках, озерах і океанах.

Забруднення може мати негативні наслідки для риб, водяних рослин, а також для людей, які використовують ці води для пиття та інших потреб.

Промислові стічні води: Заводи та підприємства викидають водні відходи, що містять хімічні речовини, важкі метали та інші забруднюючі речовини.

Антропогенні відходи: Накопичення пластикових та інших сміттєвих матеріалів у водоймах веде до порушення екосистем та загрози для водних видів.

Хімічне забруднення: Використання пестицидів, добрив та інших хімікатів в сільському господарстві може призвести до забруднення ґрунтових вод і водойм.

Зміни клімату і рівень моря: Підвищення температури планети призводить до танення льодовиків та підвищення рівня моря.

Це може призвести до затоплення прибережних територій, зникнення природних місць і настання екстремальних погодних умов.

Танення льодовиків: Збільшення температури призводить до танення льодовиків, що веде до підвищення рівня моря і загрози для прибережних районів.

Екстремальні погодні умови: Зміни в кліматі можуть призводити до збільшення інтенсивності та частоти природних лих, таких як повені та зливи.

Зміна клімату також серйозно впливає на гідросферу. Підвищення температури призводить до танення льодовиків та підняття рівня моря, що загрожує береговим територіям та прибережним екосистемам.

Несанкціоноване вилучення водних ресурсів: Надмірне використання річкових вод, відведення великих об'ємів води для сільськогосподарського та промислового використання може вести до висихання річок та озер, що загрожує водним екосистемам та призводить до дефіциту води для людей.

Використання води для сільськогосподарського господарства: Значний об'єм води витрачається на полив сільськогосподарських культур, що може призвести до висихання річок та зменшення резервуарів.

Промислове використання: Великі об'єми води використовуються для виробництва та процесів, що може призвести до висихання деяких регіонів.

Втрата біорізноманіття: Зміни в гідросфері можуть впливати на біорізноманіття водних екосистем. Зникнення водних видів, зміни у водних екосистемах можуть призвести до дисбалансу в екологічному стані.

Зміни в водних екосистемах: Забруднення та втрата природних місць можуть призвести до втрати видів та порушення екосистем, що залежать від води.

Вплив на міграції риб: Зміни в водоймах можуть впливати на місця розмноження та маршрути міграції риб, що впливає на риболовлю та екосистеми.

Охорона водних ресурсів: Створення ефективних систем управління водними ресурсами, виявлення та вирішення джерел забруднення, а також розробка сталого підходу до використання води є важливими аспектами забезпечення екологічної безпеки гідросфери.

Управління водними ресурсами: Розробка та впровадження ефективних стратегій управління водними ресурсами для забезпечення сталого використання.

Заходи з відновлення: Відновлення природних водних екосистем та природних водних місць для збереження біорізноманіття та стабільності.

Для забезпечення екологічної безпеки гідросфери необхідно впроваджувати комплексні заходи, спрямовані на збереження якості води, відновлення природних водних екосистем, раціональне використання водних ресурсів та регулювання антропогенного впливу на гідросферу.

ВИСНОВКИ

Результатом виконаної роботи є вирішення розробки системи виявлення і запобігання SQL-ін’єкцій з використанням машинного навчання та SQLmap

У вступі обґрунтована актуальність наукової теми, сформульовані задачі та мета досліджень, обґрунтована новизна отриманих результатів. Проведений аналіз альтернативних варіантів.

У 3 розділах розглянуто можливості систем виявлення, здійснено навчання моделі машиного навчання, та тестове сканування запитів і веб-ресурсу на вразливості SQL-ін’єкцій.

У процесі виконання роботи отримані наступні результати:

Досліджено актуальні загрози і вразливості веб-безпеки та проведено аналіз існуючих систем їхнього виявлення, що дозволило виділити переваги та недоліки кожного із підходів та визначити подальші напрями дослідження;

Розроблено комбінований метод системи виявлення та запобігання SQL-ін’єкціям з використанням алгоритму машиного навчання в поєднанні SQLmap для пошуку вразливостей в запитах SQL. Дана система представляє собою ефективний підхід до виявлення та класифікації SQL-ін'єкцій.

Проведено тестування ефективності запропонованої системи виявлення та запобігання SQL-ін’єкціям з використанням машинного навчання та SQLmap.

Дослідження системи було проведено шляхом сканування вразливого веб-ресурсу для збору інформації про вразливості для тренування машиного навчання , а також заздалегіть підготовлених запитів SQL.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Захист веб-додатків: чому це важливо? [Електронний ресурс]. – Режим доступу: https://itbiz.ua/ua/zashhita-veb-prilozheniy-pochemu-yeto-vazhn

2. Різновиди веб-систем та їх основні вразливості [Електронний ресурс]. – Режим доступу: http://websecurity.com.ua/security/chapter1/ – Загол. з екрану. 17. Веб форми [Електронний ресурс]. – Режим доступу: http://websecurity.com.ua/security/chapter6/

3. Гришина, Н. В. Організація комплексного захисту інформації.— М.: Гелиос АРВ, 2007.— 256.

4. С. Толюпа, Є. Толюпа, Є. Агапова / Вплив кібернетичних атак на інформаційну систему // Педагогічні інновації: ідеї, реалії, перспективи. - 2017. - Вип. 2. - С. 83-87.

5. ISO/IEC 17799: 2005 — «Інформаційні технології — Технології безпеки — Практичні правила управління інформаційної безпеки». Міжнародний стандарт, базувався на BS 7799-1: 2005.

6. ISO/IEC 27000 — Словник і визначення.

7. ISO/IEC 27001 — «Інформаційні технології — Методи забезпечення безпеки — Системи управління інформаційної безпеки — Вимоги». Міжнародний стандарт, базувався на BS 7799-2: 2005.

8. ISO/IEC 17799: 2005. «Інформаційні технології — Технології безпеки — Практичні правила управління інформаційної безпеки».

9. Тестування безпки SQL-ін’єкції [Електронний ресурс]. – Режим доступу: https://training.qatestlab.com/blog/technical-articles/security-testing-sql-injection/

10. Cloud Web Application Firewall [Електронний ресурс]. - Режим доступу URL: https://www.cloudflare.com/waf

11. SQL-injections: vulnerabilities and how to prevent attacks [Електронний ресурс]. - Режим доступу URL: https://www.veracode.com/security/sql-injection

12. Web Application Security Statistics [Електронний ресурс] – Режим доступу до ресурсу: http://projects.webappsec.org/f/wasc-wafec-v1.0.pdf

13. HACKMAGEDDON – статистика інформаційної безпеки [Електронний ресурс] – Режим доступу до ресурсу: http://www.hackmageddon.com

14. SQL Injection Attacks Are Rampant: How to Stop Your Next Hack Attack [Електронний ресурс]. – Режим доступу: https://goo.gl/RxnbWp.

15. Sqlmap: SQL-инъекции — это просто [Електронний ресурс]. – Режим доступу:https://xakep.ru/2011/12/06/57950/?ysclid=lnd4do821z233227298#toc03.

16. Защита от SQL иньекций [Електронний ресурс]. — 2012. — Режим доступу до ресурсу: http://www.securityscripts.ru/articles/sql-injection.html.

17. The Cross-site Scripting (XSS) Vulnerability: Definition and Prevention [Електронний ресурс]. — 2019. — Режим доступу до ресурсу: https://www.netsparker.com/blog/web-security/cross-site-scripting-xss/.

Додаток А

```
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from joblib import dump
# Завантаження даних з CSV-файлу
file_path = "/home/kali/Desktop/SQLDETECTED/навчальний_набір.csv"
data = pd.read_csv(file_path, delimiter=',')
# Заміна значень NaN на порожній рядок
data = data.fillna('')
# Отримання текстового стовпця для векторизації
X_train = data['Payload']
y_train = data['is_injection']
# Розбиття даних на тренувальний та тестовий набори
X_train, X_test, y_train, y_test = train_test_split(X_train, y_train, test_size=0.3, random_state=42)
# Векторизація тексту
vectorizer = TfidfVectorizer()
```

```
X_train_vectorized = vectorizer.fit_transform(X_train)

X_test_vectorized = vectorizer.transform(X_test)

# Побудова моделі RandomForestClassifier

model = RandomForestClassifier()

model.fit(X_train_vectorized, y_train)

# Оцінка точності моделі

accuracy = model.score(X_test_vectorized, y_test)

print("Точність моделі:", accuracy)

# Звіт про класифікацію

y_pred = model.predict(X_test_vectorized)

report = classification_report(y_test, y_pred)

print("Звіт про класифікацію:\n", report)

# Вивід оригінальних текстових запитів та їх класифікації

queries = []

for query, label, prediction in zip(X_test, y_test, y_pred):

    classification = 'Ін\'єкційний' if prediction == 1 else 'Неін\'єкційний'

    queries.append("{} - Класифікація: {} (Очікувана: {})".format(query, classification, label))

# Вивід списку всіх запитів

print("\nУсі запити:")
```

```
for query in queries[:15]:  # Виведемо перші 15 запитів
    print(query)

# Збереження моделі
model_file_path = "/home/kali/Desktop/SQLDETECTED/saved_model.joblib"
dump(model, model_file_path)
# Збереження векторайзера
vectorizer_file_path = "/home/kali/Desktop/SQLDETECTED/saved_vectorizer.joblib"
dump(vectorizer, vectorizer_file_path)
```

Додаток Б

```
import pandas as pd
# Завантаження даних з CSV-файлу, вказавши кому як роздільник
csv_file_path = "/home/kali/Desktop/SQLDETECTED/testphp.vulnweb.com/dump/acuart/sqli cat.csv"
data = pd.read_csv(csv_file_path, sep=',')
# Виведення назв колонок для перевірки
print("Доступні колонки:", data.columns)
# Вибірка потрібних параметрів для навчання
```

```
training_data = data[['Parameter', 'Attack Type', 'Attack Name', 'Payload']]

# Збереження навчального набору в інший CSV-файл

training_data.to_csv("/home/kali/Desktop/SQLDETECTED/навчальний_набір.csv", index=False)
```