ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Комп'ютерної інженерії

Пояснювальна записка

до магістерської роботи
на ступінь вищої освіти магістр

на тему: «**БЕЗПЕКА СУЧАСНОЇ ХМАРНОЇ ІНФРАСТРУКТУРИ ДЛЯ ВЕБ ДОДАТКІВ»**

Виконав: студент 6 курсу, групи КСДМ-62
спеціальності 123 Комп`ютерна інженерія
(шифр і назва спеціальності)
Пахомов М. В._____
(прізвище та ініціали)
Керівник ___Антоненко А. В.
(прізвище та ініціали)
Рецензент___
(прізвище та ініціали)

Київ – 2023

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Комп'ютерної інженерії

Ступінь вищої освіти - «Магістр»

Спеціальність підготовки 123 Комп`ютерна інженерія

ЗАТВЕРДЖУЮ
Завідувач кафедри КІ
_____ Н. О. Лащевська
"____" ___2023 року

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Пахомову Михайлу Васильовичу
(прізвище, ім'я, по батькові)

1.Тема роботи: «БЕЗПЕКА СУЧАСНОЇ ХМАРНОЇ ІНФРАСТРУКТУРИ ДЛЯ ВЕБ ДОДАТКІВ»

Керівник роботи: Антоненко Артем Вікторович доктор філософії, доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від "19" 10 2022 року №145

2. Строк подання студентом роботи 29.12.2023

3. Вихідні дані до кваліфікаційної роботи: науково-технічна література, відомості про елементи сучасної хмарної інфраструктури для веб додатків.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

Дослідження принципів сучасної хмарної інфраструктури

Аналіз технологій машинного навчання та можливості застосування в безпеці сучасної хмарної інфраструктури

Розробка вимог до хмари та моделювання домену

5. Перелік графічного матеріалу: *презентація*

6. Дата видачі завдання «19» жовтня 2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз наявної науково-технічної літератури	19.10-05.11.23	
2	Вивчення матеріалів для аналізу хмарної інфраструктури	05.11-12.11.23	
3	Дослідження хмарної інфраструктури	13.11-19.11.23	
4	Аналіз особливостей впливу хмарної інфраструктури на веб додатки	20.11-25.11.23	
5	Дослідження технологій розвитку хмарної інфраструктури	27.11-03.12.23	
6	Застосування машинного навчання для безпеки хмарної інфраструктури	04.12-10.12.23	
7	Оформлення роботи: вступ, висновки, реферат	11.12-20.12.23	
8	Розробка демонстраційних матеріалів	21.12-29.12.23	

Студент _____ Пахомов М. В.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Антоненко А. В.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня магістра: 89с., 21 рис, 4 табл., 37 джерел.

Мета роботи – дослідження нових методів та технологій, спрямованих на підвищення безпеки веб-інфраструктури. Це включає створення ефективних механізмів автентифікації, авторизації, мережевої безпеки та інших заходів для запобігання можливим загрозам.

Об'єкт дослідження – розробка та впровадження нових методів та технологій, спрямованих на підвищення безпеки веб-інфраструктури.

Предмет дослідження –веб-інфраструктура

Короткий зміст роботи: У роботі проведено дослідження хмарних технологій що можуть бути використані для забезпечення безпеки веб додатку. Проаналізовано основні принципи побудови безпеки додатків.

КЛЮЧОВІ СЛОВА: КІБЕРБЕЗПЕКА МЕРЕЖІ, ХМАРНІ ТЕХНОЛОГІЇ, ВЕБ ДОДАТКИ.

ABSTRACT

The text part of the qualification work for the master's degree: 89 p., 21 figures, 4 tables, 37 sources

The purpose of the work is to research new methods and technologies aimed at increasing the security of the web infrastructure. This includes creating effective mechanisms for authentication, authorization, network security and other measures to prevent possible threats.

The object of the research is the development and implementation of new methods and technologies aimed at improving the security of the web infrastructure.

The subject of the study is to increase the security of the web infrastructure.

Summary of the work: The work carried out a study of cloud technologies that can be used to ensure the security of the web application. The main principles of building application security are analyzed.

KEY WORDS: NETWORK CYBER SECURITY, CLOUD TECHNOLOGIES, WEB APPLICATIONS.

ЗМІСТ

ВСТУП.....11
1. ВИЗНАЧЕННЯ АКТУАЛЬНОСТІ ПРОБЛЕМИ ЗАХИСТУ ВЕБ-ІНФРАСТРУКТУРИ.....13
1.1. Огляд сучасних загроз, з якими стикаються веб-додатки.....14
1.2. Роль криптографії в захисті веб-інфраструктури.....19
1.2.1. Шифрування даних.....20
1.2.2. Цілісність даних.....21
1.2.3. Аутентифікація з використанням криптографії.....22
2. АНАЛІЗ НАЯВНИХ РІШЕНЬ У СФЕРІ БЕЗПЕКИ ВЕБ-ДОДАТКІВ.25
2.1. Набір засобів захисту Web-сервісів.....26
2.1.1. Вимоги до сучасного Web Application Firewall.....27
2.1.2. Використання Firewall та IPS систем.....29
2.2. Застосування машинного навчання та штучного інтелекту у забезпеченні безпеки веб-додатків.....32
2.2.1. Вивчення переваг систем виявлення вторгнень на базі ШІ.....33
2.2.2. Вплив ШІ та машинного навчання на хмарну безпеку.....35
2.2.3. Рішення щодо запобігання втрати даних на основі ШІ для підвищення кібербезпеки.....36
2.3. Інтеграція DevSecOps у безпеку веб-додатків.....37
3. АРХІТЕКТУРА ТА ПРОЕКТУВАННЯ ЗАХИЩЕНОЇ ВЕБ-ІНФРАСТРУКТУРИ.....40
3.1. Компоненти архітектури веб-застосунків.....40
3.1.1. Інтерфейс користувача (UI).....41
3.1.2. Веб-сервер.....42

3.1.3. Сервер бази даних 43
3.1.4. DNS 44
3.1.5. Проміжне програмне забезпечення для обміну повідомленнями ... 44
3.1.6. Load balancer 45
3.1.7. Cache 46
3.1.8. CDN 46
3.2. Вибір хмарного провайдеру 50
3.2.1. AWS 50
3.2.2. Azure 51
3.2.3. Google Cloud 52
3.2.4. AWS vs. Azure vs. Google Cloud: ключові відмінності 52
3.2.5. Переваги та недоліки 55
4. РОЗРОБЛЕННЯ БЕЗПЕКИ ВЕБ-ДОДАТКУ 57
4.1. Розроблення та реалізація механізмів автентифікації та авторизації на базі AWS 57
4.1.1. Патерн 1 58
4.1.2. Патерн 2 60
4.1.3. Патерн 3 61
4.2. Впровадження WAF у веб-додаток 63
4.2.1. Кроки інтеграції AWS WAF 66
4.3. Реалізація системи моніторингу та реагування на інциденти 69
4.3.1. Аналіз логів системи 69
4.3.2. Аналіз метрик системи 73
5. ТЕСТУВАННЯ ЕФЕКТИВНОСТІ ІМПЛЕМЕНТОВАНИХ РІШЕНЬ. 76

5.1. Тестування на проникнення 76
5.1.1. Принцип «білого ящика» 77
5.1.2. Принцип «чорного ящика» 78
5.1.3. Принцип «сірого ящика» 79
5.2. Інструменти для тестування додатків 80
5.3. Додавання AWS Config як елементу постійного тестування ситеми 82
ВИСНОВКИ 85
СПИСОК ВИКОРАСТИНОЇ ЛІТЕРАТУРИ 86

ВСТУП

На сьогоднішній день "Підвищення захищеності інфраструктури для веб додатків" стає все більш актуальною в контексті сучасного світу інформаційних технологій. Існує низка ключових чинників, які роблять цю тему невід'ємною частиною обговорення і досліджень.

У наш час, ми стикаємося зі зростанням кількості кібератак, і веб-додатки являють собою пріоритетний об'єкт для зловмисників. Ефективний захист стає необхідністю, враховуючи потенційні загрози для конфіденційності, цілісності даних і доступності сервісів.

Загрози у сфері кібербезпеки постійно еволюціонують, а зловмисники стають дедалі винахідливими у своїх атаках. Це вимагає постійної адаптації методів захисту і розробки нових підходів до забезпечення безпеки.

Складність веб-додатків також висуває свої вимоги до безпеки. Інтегровані та складні архітектури вимагають глибокого розуміння і комплексного підходу до забезпечення захисту на всіх рівнях.

Зі збільшенням обсягу цифрових даних і їхньої важливості для бізнесу та кінцевих користувачів стає критично важливим забезпечувати конфіденційність. Витік даних може призвести до серйозних наслідків, роблячи забезпечення їх захисту проблемою першорядної важливості.

Регулюючі органи посилюють вимоги до кібербезпеки, і компанії змушені відповідати стандартам, таким як GDPR. Це створює додаткові стимули для підвищення захищеності веб-інфраструктури.

Зі зростанням використання хмарних технологій, де багато веб-додатків розгортаються, виникають нові виклики та ризики. Забезпечення безпеки в хмарі стає вкрай важливим аспектом, який необхідно враховувати під час розроблення та експлуатації веб-додатків. Таким чином, дослідження щодо

поліпшення захищеності веб-інфраструктури стає невід'ємним елементом забезпечення стійкої кібербезпеки в сучасному інформаційному полі

Об'єкт дослідження – веб-інфраструктура для веб-додатків. У цьому контексті, об'єктом є комплекс технічних, програмних та мережевих компонентів, що є базою для функціонування веб-додатків.

Предмет дослідження – підвищення захищеності цієї веб-інфраструктури. Тобто, основною метою є ідентифікація, аналіз та розробка заходів, які можуть підвищити рівень безпеки веб-додатків та їхньої інфраструктури.

Мета – розробка та впровадження нових методів та технологій, спрямованих на підвищення безпеки веб-інфраструктури. Це включає створення ефективних механізмів автентифікації, авторизації, мережевої безпеки та інших заходів для запобігання можливим загрозам.

Методика дослідження – комбінації аналітичних, експериментальних та практичних методів. Включатиме в себе аналіз літератури, проектування безпечних архітектур, програмування та впровадження заходів безпеки, а також експериментальне тестування.

Наукова новизна – полягає в розробці нових архітектурних рішень та методів захисту, що враховують сучасні тенденції в галузі кібербезпеки та веб-розробки.

1. ВИЗНАЧЕННЯ АКТУАЛЬНОСТІ ПРОБЛЕМИ ЗАХИСТУ ВЕБ-ІНФРАСТРУКТУРИ

Актуальність проблеми захисту веб-інфраструктури обумовлена динамічним розвитком інформаційних технологій, які стають невід'ємною частиною повсякденного життя та ділової галузі. У сучасному цифровому світі веб-інфраструктура є ключовим елементом, що забезпечує функціонування веб-застосунків та взаємодію користувачів з інформацією. Проте цей рівень цифрової інтеграції також робить веб-інфраструктуру об'єктом підвищеної уваги з боку кіберзлочинців та зловмисників.

Зі збільшенням обсягу даних, що передаються та обробляються у веб-додатках, загрози для конфіденційності, цілісності та доступності інформації значно посилюються. Кібератаки на веб-інфраструктуру, такі як SQL-ін'єкції, крос-сайтовий скриптинг та DDoS-атаки, стають більш витонченими та масштабними[1].

Проблема захисту веб-інфраструктури набуває також нових аспектів у зв'язку з розширенням використання хмарних технологій та мікросервісної архітектури. Це створює додаткові вразливості, що потребують поглибленого дослідження та ефективних рішень для захисту від нових видів атак.

З урахуванням суворих нормативів у сфері кібербезпеки та зростаючих вимог до дотримання законодавства, забезпечення безпеки веб-інфраструктури стає невід'ємною частиною корпоративної відповідальності та довгострокової стійкості бізнесу.

Таким чином, актуальність проблеми захисту веб-інфраструктури обумовлена не лише динамікою технологічного розвитку, а й посиленням загроз кібербезпеці, які можуть мати серйозні наслідки для бізнесу, особистого життя та суспільства загалом.

1.1.Огляд сучасних загроз, з якими стикаються веб-додатки

У сучасному цифровому світі, веб-програми відіграють ключову роль у повсякденному житті, бізнесі та суспільстві в цілому. Однак зі зростанням їх популярності та функціональності виникає низка серйозних кібербезпекових загроз. Цей реферат присвячений огляду сучасних загроз, з якими стикаються веб-додатки, з акцентом на нові види атак та тренди в галузі кібербезпеки веб-інфраструктури.

Огляд сучасних загроз веб додатків:

`Веб-програми стають об'єктами все більш витончених і різноманітних кібератак. Розглянемо кілька ключових аспектів сучасних загроз:

– XSS (Cross Site Scripting) – це атака з використанням міжсайтових сценаріїв скриптів, які являють собою шкідливий код, котрий зловмисник вводить на сайті жертви. У висновку, якщо сайт не перевіряє вхідні дані, спрацьовують різні типи XSS: 1) код потрапляє в базу даних сайту і виконується на стороні сервера (Stored XSS); 2) код виконується у структурі вихідного коду сайту (DOM Based XSS); 3) код виконується і відображається у браузері користувача (Reflected XSS). Атака спричиняє зловмисні маніпуляції з даними: перехоплення, підміна, викрадення;

– XML External Entities (XXE) – це атака на XML-файли і документи сайту, що призводить до розкриття конфіденційної, службової інформації, а також більш складних атак: XSS, CSRF, RCE[2].

– SQL-injection – це атака з використаннями зловмисних запитів до бази даних сайту на основі мови SQL, які можуть викликати помилки конфігурації, відмову в обслуговуванні, несанкціонований доступ до внутрішньої, службової інформації (HTTP SQLi, UNION SQLi), здійснити злам бази даних MySQL (Blind SQLi) і тим самим зламати сайт, вбудувати зловмисний код (adware, spyware), змінити зовнішній вигляд/інформацію на сайті (Deface), викрасти або знищити користувацькі дані;

– PHP-injection – це атака на сайт з використанням PHP-коду, експлуатація помилок у вихідному коді PHP-об’єктів, блоків, форм, елементів сайту (Object Injection). Атака з використання команд та функцій PHP (Command Injection); [3]

– HTML-injection – це атака з експлуатацією вразливостей в HTML-коді сайту, а саме у HTML-тегах. Може спричинити підміну або перехоплення даних;

– CSRF (Cross Site Request Forgery) – це атака з використанням міжсайтової підробки запитів, які виконують певні несанкціоновані дії на сайті від імені його користувачів, наприклад: зміна даних в профілі, зміна паролю, переказ коштів, здійснення покупки, оформлення підписки та інші шахрайські операції;

– SSRF (Server Side Request Forgery) – це атака з використанням підробки запитів на стороні сервера, коли сервер не перевіряє вхідні дані, а зловмисник зловживає довірою і виконує від його імені, користуючись загальнодоступними функціями, зловмисні запити;

– CRLF-injection – це атака з використанням маніпуляції символами “Повернення каретки” і “Переведення рядка” (Carriage Return and Line Feed), які можуть використовуватися у HTTP-заголовках сайту. Таким чином можна спричинити системний збій, відправити шкідливий запит у логи сервера (Log Injection), виконати потенційно небезпечні дії збоку сервера, знищити дані;

– LFI (Local File Inlcusion) – це атака, яка дозволяє зловмиснику експлуатувати вразливості сайту і переглядати внутрішні файли, обходити каталоги, включати локальні файли у URL-посилання або форми, виконувати з ними різноманітні маніпуляції;

– RFI (Remote File Inclusion) – це атака з використанням віддалених файлів для виконання зловмисних операцій на сайті через вразливі компоненти сайту: URL, форми, код, API. Це призводить до віддаленого виконання шкідливого коду Remote Code Execution (RCE). Зловмисник таким чином

розгортає шелли, руткіти, бекдори, підвищує привілеї і бере під контроль увесь сервер;

– BruteForce – це атака з перебором даних авторизації (логіну та паролю) облікового запису, або будь-яких інших з метою отримання несанкціонованого доступу;

– DOS/DDOS (Denial of Service/Distributed Denial of Service) – це атака з метою спричинити відмову в обслуговуванні на стороні сервера;

– Spamming – це атака на сайт з використанням каналів публікації на сайті або електронної пошти з відправкою великої кількості повідомлень. У результаті велика кількість спаму може спричинити перенавантаження, переповнення жорстких дисків і оперативної пам’яті, привести до збою системних служб. Зловмисник також може з допомогою спаму проводити фішингові атаки з використанням соціальної інженерії; [4]

– Clickjakcing – це атака з використанням вразливості, яка дозволяє зловмиснику включати вміст інших сайтів на своїй фішинговій сторінці через iframe, проводити з ними різноманітні маніпуляції;

– DNS-attacks – це атаки на DNS-сервер сайту з використанням вразливостей мережевих сервісів. Зловмисник може отримати записи DNS-зони (Zone Transfer Attack), провести брутфорс піддоменів, дослідити структуру мережі, підмінити IP-адресу сервера та багато іншого.

– Cntent spoofing — атака направлена на підмінну даних через заміну контенту сторінок можлива. Використовуючи цю техніку, зловмисник змушує користувача повірити, що сторінка згенерована веб-сервером, а не передана із зовнішнього джерела.

– Cross-site request forgery — вид атак на відвідувачів веб-сайтів, який використовує недоліки протоколу HTTP. Якщо жертва заходить на сайт, створений зловмисником, браузер таємно відправляється запит на інший сервер (наприклад, на сервер платіжної системи), який здійснює якусь шкідливу операцію (наприклад, переказ грошей на рахунок зловмисника).

– URL redirector abuse — атака через перенаправления на інші сайти через підміну початкових посилань. Цей вид вразливостей, також як і багато інших перерахованих вище, є різновидом помилок перевірки вхідних данних.

– Ще однією популярною атакою є «Predictable resource location» — знаходження прихованого функціоналу та даних.

Аналізуючи сучасні загрози, необхідно також оцінити ризики та розглянути перспективи у сфері кібербезпеки веб-додатків. Запровадження багаторівневих стратегій захисту, впровадження засобів моніторингу та навчання персоналу є невід'ємними кроками для підвищення стійкості веб-інфраструктури до сучасних загроз але нажаль є випадки коли кібератаки проходять успішно.

Приклади успішних атак на веб додатки:

– SQL-ін'єкції в TalkTalk (2015): Зловмисники використовували SQL-ін'єкції для несанкціонованого доступу до бази даних компанії TalkTalk, вкравши чутливу особисту інформацію більш ніж 150 000 клієнтів. Урок: важливість ретельної валідації вхідних даних. [6]

– DDoS-атаки на веб-сайти Канади та Індії (вересень 2023 року): Індійські хактивісти направили DDoS-атаки на військові та парламентські веб-сайти Канади, що сповільнило роботу систем на кілька годин.

– Фішингова атака на залізничну мережу Ізраїлю (вересень 2023 року): Іранські хакери використовували фішингову кампанію для атаки на електричну інфраструктуру залізниці Ізраїлю.

– Приховані модифікації в роутерах (вересень 2023 року): Китайські державні хакери, ймовірно, розмістили модифікуюче програмне забезпечення в роутерах, щоб атакувати компанії та організації в США та Японії.

– Кібератака на департамент планування Бермуди (вересень 2023 року): Державні служби Бермуди були атаковані і деякі сервіси були недоступні протягом декількох тижнів.

– Фішингова атака на Міністерство фінансів Кувейту (вересень 2023): Кіберзлочинці направили рансомварну атаку на Міністерство фінансів Кувейту.

– Кібератаки на українські органи правопорядку (вересень 2023 року): Російські сили посилюють атаки на українські органи правопорядку, зосереджуючись на підрозділах, які збирають та аналізують докази російських військових злочинів.

– Атака на системи Міжнародного кримінального суду (вересень 2023 року): Російські кіберзлочинці зламали IT-системи Міжнародного кримінального суду під час розслідування російських військових злочинів в Україні.

– Збільшення кібероперацій Китаю в Південно-Китайському морі (вересень 2023 року): Звіт Microsoft свідчить про зростання кібератак з боку Китаю проти США та їх критичної інфраструктури.

– Атака на Adobe (2013): Шкідливий код, впроваджений через вразливість в Adobe Flash Player, дозволив зловмисникам отримати доступ до особистої та фінансової інформації мільйонів користувачів. Урок: регулярні оновлення та виправлення вразливостей.

– XSS-атака на eBay (2014): Зловмисник впровадив шкідливий скрипт через вразливість в форумах eBay, в результаті чого були скомпрометовані облікові записи користувачів. Захист від XSS-атак вимагає строгих контрольних заходів.

Аналізуючи ці та інші атаки можна виділити основні уроки що допоможуть захистити веб додатки у майбутньому:

– Безпека завжди пріоритет: Незалежно від масштабу організації або важливості веб-програми, безпека має бути на першому місці. Регулярно аудитуйте свої системи, оновлюйте програмне забезпечення та стежте за вразливістю.

– Навчання співробітників: Фішингові атаки залишаються одним із найефективніших методів злому. Навчайте своїх співробітників розпізнавати підозрілі повідомлення та посилання.

– Багатофакторна автентифікація (MFA): Використання MFA підвищує безпеку, оскільки навіть якщо пароль вкрадено, зловмисникам буде важко отримати доступ до системи без додаткового фактора автентифікації.

– Моніторинг та реагування на інциденти: Швидке виявлення та реагування на атаки скорочує потенційні збитки. Розробте план реагування на інциденти та навчіть персонал його застосовувати.

– Співпраця та обмін інформацією: Організації повинні обмінюватися інформацією про нові загрози та атаки. Співпраця з іншими компаніями та організаціями допомагає покращити спільну безпеку.

– Регулярні резервні копії: Регулярне створення резервних копій даних допомагає мінімізувати втрати у разі успішної атаки.

– Постійне навчання та адаптація: Кіберзагрози постійно змінюються. Організації повинні стежити за новими методами атак та адаптуватися до них.

Аналіз успішних кібератак на веб-застосунки надає цінні уроки для покращення стратегій кібербезпеки. Розуміння загальних елементів атаки дозволяє розробити ефективніші заходи захисту, включаючи покращення управління доступом, регулярне оновлення систем та акцент на навчання персоналу. Безперервне дослідження випадків успішних атак необхідне для адаптації до ландшафту кібербезпеки, що змінюється.

1.2.Роль криптографії в захисті веб-інфраструктури

Одним з елементів безпеки додатків стало впровадження елементів криптографії у структуру додатку. . Криптографія як наука про безпечну

передачу інформації відіграє ключову роль у забезпеченні конфіденційності та цілісності даних у веб-інфраструктурі.

Основні принципи криптографії[7]:

– Конфіденційність (Шифрування): використання математичних алгоритмів для перетворення тексту, що читається, в незрозумілий вигляд (шифр), який може бути зрозумілий тільки тим, у кого є ключ.

– Цілісність даних: забезпечення того, щоб дані не зазнали зміни чи пошкодження в процесі передачі або зберігання.

– Аутентифікація: перевірка справжності сторін у комунікації для встановлення довіри.

– Невідмовність (Незастосовність відмови): забезпечення того, що відправник не може відмовлятися від факту надсилання повідомлення або виконання операції.

Таким чином ці принципи криптографії є фундаментальними елементами забезпечення безпеки у веб-інфраструктурі. Їхнє правильне розуміння та застосування відіграють ключову роль у створенні надійних та захищених систем передачі та обробки даних у сучасному цифровому світі.

Розглянемо докладніше ці принципи та их використання в захисті веб-інфраструктури

1.2.1. Шифрування даних

Шифрування даних за допомогою криптографії відіграє ключову роль у забезпеченні безпеки інформації під час передачі та зберігання. Це процес перетворення інформації на незрозумілий вигляд з використанням математичних алгоритмів, що робить дані доступними лише авторизованим користувачам. Нижче розглянуто методи шифрування даних у транзиті та зберіганні для забезпечення конфіденційності.

При передачі даних по мережі важливо захистити інформацію від несанкціонованого доступу. Для цього використовуються протоколи шифрування, такі як SSL (Secure Sockets Layer) та його послідовність TLS (Transport Layer Security). Ці протоколи забезпечують захищене з'єднання між клієнтом та сервером шляхом шифрування даних, що передаються через Інтернет. Це дозволяє запобігти перехопленню інформації зловмисниками та забезпечити конфіденційність даних.

1.2.2. Цілісність даних

При зберіганні даних на серверах або в хмарних системах також використовуються методи шифрування. Захист інформації забезпечує її збереження у разі витоку даних або несанкціонованого доступу. [9]

Для цього використовуються різні методи шифрування, такі як симетричне та асиметричне шифрування. Симетричне шифрування використовує той самий ключ для шифрування і розшифрування даних, тоді як асиметричне шифрування використовує пару ключів: відкритий і закритий.

Також важливим методом захисту даних при зберіганні є хешування, яке дозволяє перетворити дані на фіксований набір символів і забезпечити цілісність інформації. Це дозволяє контролювати цілісність даних та запобігати їхній зміні зловмисниками.

Цілісність даних є фундаментальним аспектом інформаційної безпеки, спрямованим на забезпечення недоторканності та незмінності даних протягом усього їхнього життєвого циклу. Цей принцип відіграє ключову роль у підтримці довіри до інформації та запобіганні можливим спотворенням даних, які можуть вплинути на їх достовірність. Ось кілька ключових аспектів цілісності даних:

Хешування для перевірки цілісності:

– Хеш-функції перетворюють дані в унікальний рядок символів (хеш), що є своєрідним "відбитком пальця" для цих даних.

– Будь-які зміни в даних, навіть невеликі, повинні спричинити значну зміну хеш-значення.

– Шляхом порівняння поточного хеша з оригінальним можна швидко виявити, чи були внесені зміни до даних.

Цифрові підписи:

– Цифрові підписи забезпечують аутентифікацію та цілісність даних шляхом використання криптографії. [9]

– Відправник створює цифровий підпис із використанням свого приватного ключа, і одержувач може перевірити справжність даних із використанням публічного ключа відправника.

– Це забезпечує гарантії, що дані не було змінено після створення підпису.

Аудит цілісності

– Регулярні аудити цілісності даних дозволяють виявляти зміни в даних та негайно вживати заходів.

– Журнали аудиту можуть включати інформацію про час і деталі змін, що забезпечує можливість швидкого реагування на будь-які порушення.

Цілісність даних є невід'ємною частиною загальної стратегії безпеки інформації. Цей аспект забезпечує впевненість у тому, що дані залишаються недоступними для неправомірної зміни, підтримуючи тим самим їхню довіру та точність у будь-якій інформаційній системі.

1.2.3. Аутентифікація з використанням криптографії

Аутентифікація з використанням криптографії – це процес перевірки особистості користувача чи пристрою із застосуванням математичних методів

та принципів криптографії. Вона є одним з найважливіших механізмів безпеки в інформаційних системах, включаючи веб-додатки.

У контексті автентифікації, що ґрунтується на криптографії, існує кілька ключових аспектів, націлених на забезпечення безпеки, цілісності та достовірності інформації. Процес автентифікації включає застосування криптографічних методів, які гарантують секретність, цілісність та правильність наданих ідентифікаційних даних. Одним з найпоширеніших методів є хешування паролів, де сам пароль користувача не зберігається у відкритому вигляді, а використовується його хеш-значення. Такий підхід робить вкрай складним відновлення оригінального пароля із зашифрованого значення.

Асиметрична криптографія, що використовує публічні та приватні ключі, також є важливим компонентом. Публічний ключ застосовується для шифрування даних, і їхнє розшифрування можливе лише за наявності відповідного приватного ключа. Цей метод широко використовується для забезпечення безпеки під час передачі даних.

Багатофакторна автентифікація, що поєднує паролі, біометричні дані та криптографічні токени, також є важливим аспектом. Цей підхід підвищує рівень безпеки за рахунок комбінації різних факторів та ускладнення процесу несанкціонованого доступу.

Протоколи автентифікації, такі як OAuth та OpenID Connect, що використовують криптографічні принципи, забезпечують безпечну взаємодію між користувачами та різними системами. Ці протоколи дозволяють безпечно авторизуватися в різних програмах, уникаючи при цьому розкриття секретної інформації.

Важливим аспектом є використання криптографії для захисту від різних атак, таких як перехоплення даних, впровадження шкідливого коду та інші

форми кібератак. Криптографічні методи сприяють виявленню та запобіганню таких загроз, забезпечуючи надійність та безпеку в онлайн-середовищі.

2. АНАЛІЗ НАЯВНИХ РІШЕНЬ У СФЕРІ БЕЗПЕКИ ВЕБ-ДОДАТКІВ

Сучасний цифровий ландшафт міцно інтегрував у собі веб-програми, що стали невід'ємною частиною повсякденного життя. Однак, паралельно з їх незаперечними перевагами виникають серйозні виклики, пов'язані з кібербезпекою. Віртуалізація бізнес-процесів і широке використання веб-технологій надають нові можливості, але пов'язані зі збільшенням загроз з боку кіберзлочинців.

Безпека веб-додатків стає пріоритетним питанням, оскільки їх функціональність, великі бази даних та взаємодія з інформацією користувача створюють унікальні точки входу для потенційних кібератак. Витік даних, атаки на автентифікацію, ін'єкції коду – лише частина широкого спектру загроз, з якими веб-програми стикаються щодня. [10]

У цьому контексті необхідно усвідомлене дослідження та аналіз існуючих рішень у сфері безпеки веб-додатків. Розуміння актуальних методів та інструментів, а також їх порівняльний аналіз не лише сприяє виявленню вразливостей, але й формує стратегії ефективного захисту, дозволяючи створювати веб-додатки, стійкі до різних загроз у цифровій епосі.

У сучасному світі, де віртуалізація інформаційних потоків стає невід'ємною частиною нашого повсякденного життя, безпека веб-додатків стає все більш глибоким інтересом і невідкладною потребою. Все більш витончені методи атак з боку зловмисників та різноманітність загроз наголошують на необхідності вдосконалення засобів захисту та контролю у віртуальному просторі.

Ситуація, що склалася, вимагає уважного вивчення існуючих рішень та інструментів, призначених для забезпечення безпеки веб-додатків. У центрі уваги опиняються методи захисту від поширених видів атак, таких як ін'єкції, перехоплення даних та атаки на автентифікацію.

Порівняльний аналіз популярних фреймворків та інструментів для захисту веб-інфраструктури дає можливість виявити їх переваги, недоліки та сфери найкращого застосування. Це не тільки допомагає розробникам створювати більш стійкі та безпечні веб-програми, але й визначає загальні тенденції розвитку в галузі кібербезпеки.

У цьому контексті розділ присвячений аналізу існуючих рішень у забезпеченні безпеки веб-додатків є важливим кроком у розумінні та протистоянні сучасним кіберзагрозам. Він спрямований на виділення найкращих практик, визначення актуальних трендів та забезпечення практичних рекомендацій для забезпечення кібербезпеки у віртуальному середовищі.

2.1. Набір засобів захисту Web-сервісів

Для захисту від WEB-атак класичним пристроєм є Web Application Firewall. Брандмауер веб-додатків застосовує набір правил безпеки до протоколів високого рівня (прикладних) HTTP/HTTPS, FTP/FTPS. Класичне розміщення WAF в мережі - це зворотний проксі перед захищеними веб-серверами[11].

Набір функцій WAF зазвичай включає наступні типові механізми захисту:

- Перевірка протоколу
- Аналіз підписів
- Захист сеансів і файлів cookie
- Блокування витоку даних;
- Розпізнавання атак (з негативної моделі, атак на додатки, мережу,
- Атаки на додатки, мережу, веб-сервер та ОС);
- Можливість створювати власні правила захисту;

– Машинне навчання.

2.1.1. Вимоги до сучасного Web Application Firewall

На сьогоднішній день є визначений список вимог до сучасного Web Application Firewall:

– Компоненти системи WAF повинні відповідати вимогам PCI DSS;

– здатність реагувати на загрози, описані в першій десятці OWASP;

– перевірка повідомлень веб-сервісів, якщо веб-сервіс підключений до Інтернету (SOAP, XML);

– перевірка будь-якого протоколу або конструкції даних, що використовується для передачі даних веб-додатків, незалежно від того, чи є він пропрієтарним або стандартизованим (як для вхідних, так і для вихідних потоків даних);

– застосування як позитивних, так і негативних моделей безпеки; перевірка всього вмісту веб-сторінок, включаючи HTML, DHTML і CSS, а також нижчі протоколи доставки контенту (HTTP/HTTPS);

– запобігання або виявлення підробки ідентифікатора сеансу;

– автоматичне завантаження оновлень сигнатур атак та їх застосування;

– можливість встановлювати режими fail-open та fail-close;

– захист від загроз, спрямованих безпосередньо на WAF;

– підтримка розриву SSL\TLS з'єднання;

– підтримка пристроїв для клієнтських SSL-сертифікатів;

– підтримка апаратного зберігання ключів (FIPS).

– перевірка запитів і відповідей відповідно до політики безпеки, ведення журналу подій; запобігання витоку даних - перевірка відповідей сервера на предмет критично важливих даних[12];

Аналізуючи наведені данні побудемо таблицю (Таблиця 3.1) з висновоками про недоліки системи, а саме про нездатність WAF протистояти деяким видам атак.

Таблиця 2.1 – Ефективність WAF відносно різних типів атак

Типи атак	Чи можуть бути заблоковані WAF
Атаки направлені на викрадання даних	Блокує
Ін’єкції	Блокує
Цілеспрямована атака	Може блокувати, але не у повному обсязі
DDoS	Може бачити активність в мережі і зменшувати кількість запитів до ресурсу за певний час.
Ураження шкідливим ПЗ	Може блокувати, а може і ні, в залежності від атаки
Атаки на компоненти з відомими вразливостями	Частково блокує
XSS	Блокує
SYN-flood	Не блокує
Атаки, спрямовані на викрадання сесії користувача	Блокує
TCP-flood	Не блокує
UDP-flood	Не блокує
CSRF	Блокує

Тому як можемо бачити загальних функцій WAF недостатньо для повноцінного захисту інтернет-ресурсів, оскільки WAF не захищає від низькорівневих атак, таких як DDoS, SYN-flood, TCP-flood, UDP-flood. Таким чином, виникає потреба в додатковому захисті, а саме в створенні комплексу для вдосконалення системи безпеки для захисту веб-ресурсів.

2.1.2. Використання Firewall та IPS систем

Щоб підвищити безпеку веб-ресурсів, потрібно створити захист від низькорівневих атак. Для цього можна використовувати системи IPS і Firewall. Firewall буде формувати доступ до портів і забезпечувати мінімальний захист, в той час як IPS буде відстежувати низькорівневі атаки і реагувати на зміни в потоках трафіку[13].

Система запобігання вторгненням - це система, яка розпізнає ознаки вторгнення, виявляє атаки та запобігає їм. Аналіз використовує різні методи виявлення атак - сигнатурний, поведінковий та виявлення аномалій у протоколах.

Крім того, всі типи технологій IPS зазвичай виконують наступні функції:

– IPS зупиняє саму атаку.

– Блокує шкідливу частину, дозволяючи неураженій частині проникнути в систему в систему.

– Повідомляти адміністраторів безпеки про важливі події, що спостерігаються в системі

– Реагувати на інциденти, змінюючи середовище безпеки, щоб запобігти атаці.

– Формувати звіти.

Загальні вимоги до IPS та типи подій, які найчастіше виявляються:

– Розслідування та атаки на рівні додатків (наприклад, переповнення буфера, підбір пароля, передача шкідливого програмного забезпечення). Більшість мережевих СПП аналізують протоколи додатків.

– Дослідження та атаки на транспортному рівні (наприклад, сканування портів, незвична фрагментація пакетів, переповнення SYN). Найчастіше аналізуються протоколи транспортного рівня - TCP і UDP.

– - Дослідження та атаки на мережевому рівні (наприклад, підміна IP-адрес, аномальні значення IP-заголовків).

– - Неочікувана поведінка додатків (наприклад, хости, що виконують несанкціоновані дії).

– - Порушення політики (наприклад, використання заборонених протоколів)

Firewall - це система мережевої безпеки, яка відстежує і контролює вхідний і вихідний трафік на основі заздалегідь визначених правил безпеки. правила безпеки. Брандмауер зазвичай встановлює бар'єр між захищеною внутрішньою мережею та зовнішньою незахищеною мережею. Його основна мета - захистити внутрішню мережу або її окремі вузли від несанкціонованого доступу. Firewall контролює доступ до мережевих ресурсів, використовуючи модель позитивного контролю (у внутрішню мережу потрапляє тільки дозволений правилами трафік, весь інший трафік заборонений) [14].

Загалом, Firewall поділяються на дві категорії:

– Міжмережні екрани мережного рівня дозволяють чи забороняють трафік, базуючись на адресах джерела IP і адресах чи портах призначення IP. –

– Міжмережні екрани прикладного рівня аналізують протоколи прикладний рівень, відстежуючи активність протоколу щодо певного профілю та дозволяючи або забороняючи трафік, на основі відхилень від профілю.

Типові функції Firewall:

– Контроль доступу до вузлів мережі

– Фільтрація доступу до незахищених сервісів
– Контроль черговості доступу до мережі
– Запобігання спробам доступу із зовнішньої та внутрішньої мережі
– Запобігання отриманню секретної інформації з внутрішньої мережі, що захищається.

Таким чином, в комплексі, системи будуть захищати інтернет-ресурси на всіх рівнях моделі OSI. На рисунку 2.1 зображена схема інтеграції систем. WAF впроваджується в систему в режимі зворотного проксі-сервера перед захищеними веб-серверами. IPS впроваджується в комплекс в режимі Transparent. Отримуючи запити, допущені Firewall, IPS аналізує протоколи і зупиняє певні види атак, надалі дані передаються до WAF, де обробляються і також блокуються атаки функціоналу WAF. [15]

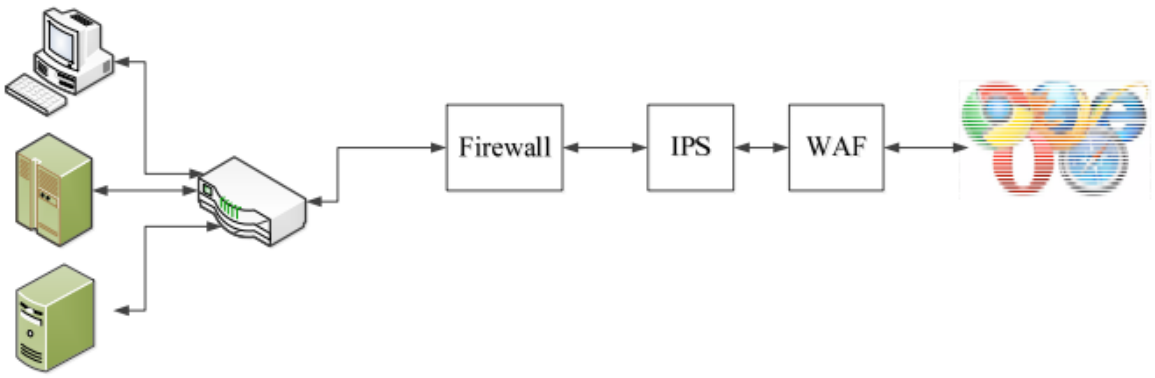


Рисунок 2.1 — Схема інтеграції систем з Web Application Firewall

Відповіді веб-сервера надсилаються назад до WAF, де вони перевіряються на витоку даних. Після перевірки дані надсилаються користувачеві.

Використання такої інтегрованої системи захисту інтернет-ресурсів в що складається з Firewall, IPS і WAF, забезпечує на 30% більшу ефективність, ніж використання звичайного WAF. Використання сучасних високопродуктивних

Firewall та IPS заблокує проникнення шкідливих файлів у внутрішню захищену мережу, забезпечить додатковий захист і знизить ризик цілеспрямованих атак на IT-ресурси. Такий комплекс підвищить безпеку будь-якого інтернет-ресурсу, зменшить навантаження на системних адміністраторів IT-систем і забезпечить більш ефективну обробку запитів від легітимних користувачів інтернет-ресурсів.

2.2. Застосування машинного навчання та штучного інтелекту у забезпеченні безпеки веб-додатків

Штучний інтелект (ШІ) відіграє дедалі важливішу роль у захисті веб-додатків від кіберзагроз. Системи безпеки на основі ШІ використовуються для виявлення, моніторингу та реагування на кібератаки з більшою швидкістю і точністю, ніж будь-коли раніше.

Системи безпеки на основі штучного інтелекту здатні виявляти аномальні дії користувачів, такі як використання грубої сили для входу в систему та інші шкідливі дії, і попереджати адміністраторів про підозрілі дії. ШІ також можна використовувати для аналізу веб-трафіку з метою виявлення шкідливого коду і шкідливих веб-запитів. Це допомагає запобігти проникненню шкідливого коду в систему, а також допомагає виявляти і блокувати відомі шкідливі запити.

ШІ також можна використовувати для виявлення підозрілих моделей поведінки користувачів. Це може допомогти ідентифікувати зловмисників і запобігти їхньому доступу до системи. Системи на основі ШІ також можна використовувати для виявлення фішингових атак та інших форм соціальної інженерії та реагування на них. [16]

Системи безпеки на основі ШІ також можуть використовуватися для виявлення розподілених атак типу "відмова в обслуговуванні" (DDoS) і

реагування на них. Системи, керовані ШІ, можуть допомогти визначити джерело атаки, а також тип атаки і зменшити збиток, заподіяний атакою.

Системи на основі ШІ також можна використовувати для виявлення і зупинки експлойтів нульового дня, які використовують раніше невідомі вразливості програмного забезпечення. Системи на основі ШІ можуть допомогти виявити ці експлойти до того, як ними скористаються зловмисники. [17]

Крім виявлення кіберзагроз і реагування на них, системи на основі ШІ також можуть допомогти підвищити безпеку веб-додатків. Системи на основі ШІ можна використовувати для створення надійніших паролів, виявлення підозрілих дій користувачів і реагування на них, а також для створення якісніших і безпечніших ключів шифрування.

Загалом системи безпеки на основі ШІ відіграють дедалі важливішу роль у захисті веб-додатків від кіберзагроз. У міру того, як витонченість кіберзагроз продовжує зростати, системи безпеки на основі ШІ ставатимуть дедалі важливішими для захисту веб-додатків від зловмисників.

2.2.1. Вивчення переваг систем виявлення вторгнень на базі ШІ

Оскільки підприємства та організації продовжують шукати нові технології для підвищення своєї безпеки та захисту своїх даних, системи виявлення вторгнень (IDS) на базі ШІ стають дедалі популярнішими. IDS - це система, призначена для виявлення шкідливої активності в мережі або системі та оповіщення адміністраторів або фахівців з безпеки про будь-які потенційні загрози. У той час як традиційні IDS покладаються на зумовлені правила і сигнатури для виявлення загроз, IDS на основі ШІ використовують передові алгоритми машинного навчання для виявлення аномальної поведінки, забезпечуючи надійніше і комплексніше рішення для забезпечення безпеки[8].

Переваги IDS на базі ШІ численні. По-перше, вони здатні виявляти загрози, які традиційні IDS не змогли б ідентифікувати, такі як витончені атаки нульового дня, які часто важко виявити традиційним IDS через їхню нову природу. Крім того, IDS на базі ШІ можуть аналізувати величезні обсяги даних у режимі реального часу, що дає змогу точніше і своєчасно виявляти загрози. Це особливо корисно для підприємств і організацій з великими і складними мережами.

Ще одна перевага IDS на базі ШІ полягає в тому, що вони здатні адаптуватися до мінливих загроз. Традиційні IDS вимагають частих оновлень, щоб бути в курсі останніх загроз. Однак IDS на базі ШІ можуть робити висновки зі зібраних ними даних і відповідним чином коригувати свої алгоритми виявлення. Це дає їм змогу залишатися на крок попереду і виявляти нові загрози до того, як вони зможуть заподіяти будь-яку шкоду.

На додаток до цих переваг IDS на основі ШІ також можуть знизити навантаження на фахівців з безпеки. Автоматизуючи ідентифікацію та аналіз загроз, IDS на основі ШІ можуть вивільнити цінні ресурси, які в іншому випадку були б витрачені вручну на моніторинг і аналіз даних. Це дає змогу фахівцям із безпеки зосередитися на інших завданнях, як-от реагування на загрози або вжиття превентивних заходів[19].

Загалом, IDS на базі ШІ надають підприємствам і організаціям більш комплексне рішення для забезпечення безпеки. Використовуючи можливості штучного інтелекту і машинного навчання, ці системи здатні виявляти раніше небачені загрози і адаптуватися до нових загроз. Крім того, вони можуть знизити навантаження на фахівців з безпеки, вивільняючи ресурси для більш важливих завдань. З цих причин IDS на базі ШІ швидко стають кращим рішенням для забезпечення безпеки для багатьох підприємств і організацій.

2.2.2. Вплив ШІ та машинного навчання на хмарну безпеку

Останніми роками штучний інтелект (ШІ) і машинне навчання (МО) стали використовуватися для підвищення безпеки хмарних обчислень. Оскільки хмарні обчислення продовжують поширюватися, потреба в розширених заходах безпеки стає все більш гострою. AI і ML забезпечують вирішення цієї потреби, пропонуючи більш комплексний і надійний підхід до хмарної безпеки.

На найбазовішому рівні AI і ML можна використовувати для автоматизації аналізу та виявлення потенційних загроз безпеці. Використовуючи ці технології, постачальники хмарних послуг можуть відстежувати аномалії в шаблонах даних і трафіку, а також виявляти потенційні загрози, які могли залишитися непоміченими при ручних заходах безпеки. Крім того, AI і ML можна використовувати для прогнозної безпеки. Аналізуючи дані про попередні інциденти безпеки, ШІ та машинне навчання можуть виявляти закономірності та вчитися передбачати потенційні загрози. Це може дозволити хмарним провайдерам вживати попереджувальних заходів для зниження ризиків і вразливостей безпеки.

AI і ML також використовуються для розширення можливостей протоколів управління доступом хмарних провайдерів. Використовуючи штучний інтелект і машинне навчання, постачальники хмарних послуг можуть краще відстежувати й керувати доступом користувачів до даних і ресурсів. Це може допомогти запобігти доступу зловмисників до конфіденційних даних, а також гарантувати, що користувачі мають доступ тільки до тих даних і ресурсів, які їм дозволено використовувати[20].

Нарешті, AI і ML використовуються для підвищення точності та швидкості аудиту безпеки. Ці технології можна використовувати для аналізу даних журналів і отримання відомостей, які можна використовувати для виявлення потенційних проблем безпеки. Це може допомогти постачальникам

хмарних послуг швидко виявляти й усувати потенційні загрози, знижуючи ризик витоку даних.

Використання штучного інтелекту і машинного навчання для хмарної безпеки швидко набирає обертів, і очевидно, що ці технології продовжуватимуть відігравати все більш важливу роль у хмарній безпеці в найближчі роки. Використовуючи ці технології, постачальники хмарних послуг можуть створити більш безпечне середовище для зберігання і доступу до даних.

2.2.3. Рішення щодо запобігання втрати даних на основі ШІ для підвищення кібербезпеки

Оскільки досягнення в галузі технологій продовжують стимулювати цифрову трансформацію, потреба в стратегіях шифрування і захисту даних стає все більш важливою. Штучний інтелект (ШІ) відіграє дедалі важливішу роль, допомагаючи організаціям впроваджувати ефективні заходи безпеки даних, особливо в міру того, як кібератаки стають дедалі витонченішими[22].

Рішення для шифрування з підтримкою ШІ допомагають організаціям захищати свої дані від зловмисників. Використовуючи алгоритми машинного навчання та інші методи штучного інтелекту, організації можуть виявляти шкідливі дії та реагувати на потенційні загрози в режимі реального часу. Методи шифрування на основі ШІ також можна використовувати для захисту даних під час передачі, наприклад, під час надсилання даних між двома різними мережами.

Крім шифрування, ШІ також можна використовувати для виявлення шаблонів у даних, які можуть вказувати на потенційну загрозу. Це можна зробити за допомогою аналізу даних, який може виявити підозрілу активність, яка може свідчити про зловмисну атаку. Використовуючи ШІ для виявлення

цих шаблонів, організації можуть виявляти потенційні загрози до того, як вони стануть проблемою.

ШІ також використовується для розробки більш безпечних методів автентифікації, таких як біометрична автентифікація. Ця форма автентифікації вимагає використання фізичних характеристик користувача, таких як відбитки пальців або розпізнавання облич, для отримання доступу до захищених систем. Використовуючи штучний інтелект для аналізу користувацьких даних, організації можуть гарантувати, що тільки уповноважені особи можуть отримати доступ до конфіденційних систем.

Загалом ШІ відіграє важливу роль, допомагаючи організаціям захищати свої дані від зловмисників. Використовуючи методи шифрування і виявлення на основі ШІ, організації можуть створити безпечне середовище для своїх даних і захистити їх від несанкціонованого доступу. У міру того, як кібератаки стають дедалі витонченішими, використання ШІ в стратегіях шифрування і захисту даних стає дедалі важливішим.

2.3. Інтеграція DevSecOps у безпеку веб-додатків

В наш час, коли розробка та розгортання веб-застосунків стає все більш динамічним процесом, важливо інтегрувати безпеку в кожен етап життєвого циклу розробки. Концепція DevSecOps передбачає впровадження принципів безпеки в процес розробки, що дозволяє запобігати вразливості на ранніх етапах

Інтеграція DevSecOps – це стратегія, спрямована на поєднання процесів розробки (Dev) та операцій (Ops) з упором на безпеку (Sec). Цей підхід ставить перед собою завдання інтегрувати заходи безпеки у кожний етап життєвого циклу розробки, що забезпечує більш надійний та проактивний захист веб-додатків.

Розглянемо основні аспекти інтеграції DevSecOps:

– Раннє впровадження заходів безпеки. Інтеграція DevSecOps передбачає впровадження заходів безпеки на ранніх етапах розробки, починаючи з проектування та кодування. Розробники, оператори та фахівці з безпеки працюють спільно для виявлення та усунення потенційних вразливостей ще до того, як вони опиняться у робочому оточенні.

– Автоматизація тестування на безпеку. Використання DevSecOps включає автоматизоване тестування на вразливості веб-додатків. Це дозволяє виявляти та усувати потенційні проблеми безпеки на ранніх стадіях розробки, прискорюючи процес та забезпечуючи більш високий ступінь захисту.

– Культура безпеки. DevSecOps сприяє формуванню культури безпеки усередині команди розробки. Кожен учасник процесу усвідомлює важливість безпеки та бере участь у забезпеченні її на всіх рівнях розробки та експлуатації.

– Інтеграція інструментів безпеки. Інтеграція DevSecOps передбачає використання спеціалізованих інструментів безпеки, які автоматизують процеси аналізу коду, моніторингу безпеки та реагування на загрози. Ці інструменти інтегруються безпосередньо в інструментарій розробників та операційних команд.

– Безперервна доставка безпеки (Continuous Security). DevSecOps підтримує безперервну доставку безпеки. Це означає, що безпека впроваджується в кожному етапі CI/CD-пайплайну, забезпечуючи автоматичну перевірку безпеки при кожній зміні коду або інфраструктури.

– Навчання та обмін досвідом. Інтеграція DevSecOps включає постійне навчання та обмін досвідом між розробниками та фахівцями з безпеки. Це сприяє підвищенню рівня компетентності всієї команди у сфері кібербезпеки.

– Моніторинг у реальному часі. DevSecOps передбачає наявність систем моніторингу безпеки, що працюють у режимі реального часу. Це дозволяє оперативно реагувати на інциденти та знижує час реакції на погрози.

Інтеграція DevSecOps у забезпечення безпеки веб-додатків забезпечує більш ефективний та надійний захист, враховуючи безпеку на кожному етапі розробки та експлуатації.

3. АРХІТЕКТУРА ТА ПРОЕКТУВАННЯ ЗАХИЩЕНОЇ ВЕБ-ІНФРАСТРУКТУРИ

Враховуючи описані вище загрози та методи їх подолання почнемо проектування захищеного веб-додатку. Це завдання передбачає докладний розгляд кожного елемента, виділення його функцій та важливості у контексті безпеки. Перейдемо до розглядання ключових компонентів веб-інфраструктури[23].

3.1. Компоненти архітектури веб-застосунків

Веб-додаток має два основних елементи - фронтенд і бекенд (Рисунок 3.1).

Frontend - це частина веб-програми, яка є видимою і доступною для користувача і включає в себе елементи користувацького інтерфейсу (UI), такі як кнопки, форми і меню.

Back-end - це частина веб-програми, яка працює на сервері. Зазвичай вона складається з сервера і бази даних.

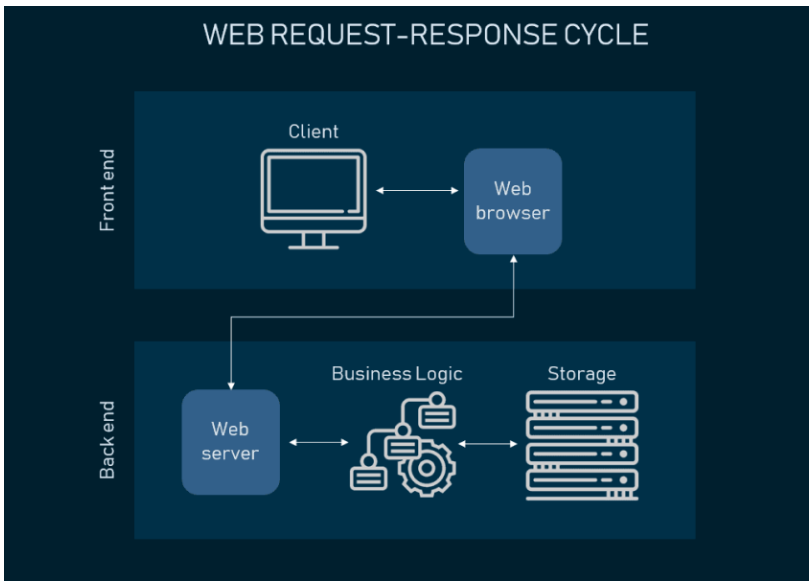


Рисунок 3.1 – Взаємодія фронтенду та бекенду для реалізації клієнтського запиту

Давайте обговоримо компоненти сучасної архітектури веб-додатків та їхні функції.

3.1.1. Інтерфейс користувача (UI)

Інтерфейс користувача (UI) являє собою візуальну частину програмного продукту, вебсайту або цифрового додатка, з якою взаємодіє користувач. Основна мета UI - забезпечити ефективну взаємодію між користувачем і системою. Елементи UI охоплюють різні аспекти:

Візуальний дизайн UI охоплює розробку компонування, кольорової палітри, шрифтів і графічних елементів для досягнення привабливого зовнішнього вигляду.

Навігація надає користувачеві засоби переміщення по додатку за допомогою меню, кнопок і посилань.

Елементи введення контролюють збір інформації від користувача через форми, кнопки та інші інтерактивні елементи.

Зворотний зв'язок містить повідомлення про помилки, підтвердження успішних дій та індикатори завантаження для інформування користувача.

Чуйність гарантує адаптацію інтерфейсу до різних розмірів екранів і пристроїв.

Інтерактивність забезпечується анімаціями, ефектами під час наведення і можливістю перетягування елементів.

Узгодженість досягається завдяки використанню загальних елементів дизайну і стандартних способів навігації

Дизайн інтерфейсу також має враховувати доступність для користувачів з обмеженими можливостями.

Прототипування дає змогу створювати інтерактивні макети для тестування і поліпшення UI перед розробкою.

Ефективний UI забезпечує задоволення користувача і є ключовим фактором успіху програмних продуктів і веб-сайтів.

3.1.2. Веб-сервер

Веб-сервер є програмним забезпеченням, призначеним для обробки запитів від клієнтських браузерів і надання їм веб-сторінок, зображень та іншого контенту. Ось ключові аспекти веб-сервера:

1. обробка запитів: Веб-сервер приймає HTTP-запити від клієнтів (наприклад, веб-браузерів), обробляє їх і повертає відповідну HTTP-відповідь. Цей процес забезпечує передачу веб-сторінок та інших ресурсів.

3. Статичний і динамічний контент: Веб-сервери можуть обслуговувати як статичний, так і динамічний контент. Статичний контент (HTML-сторінки, зображення) зберігається на сервері та надсилається напряму. Динамічний

контент генерується на сервері в процесі виконання запиту, часто з використанням мов програмування, таких як PHP, Python, Ruby та інших[24].

5. Забезпечення безпеки: Багато веб-серверів включають функціональність безпеки, таку як SSL/TLS для шифрування даних і забезпечення безпечної передачі інформації між сервером і клієнтом.

Популярні веб-сервери включають Apache, Nginx, Microsoft IIS, і кожен із них має свої особливості та переваги залежно від вимог веб-додатків.

3.1.3. Сервер бази даних

Сервер баз даних (СУБД) є програмним забезпеченням, спеціально розробленим для зберігання, управління та обробки даних у базах даних. Ось ключові аспекти сервера баз даних:

– Зберігання даних: Основна функція сервера баз даних - збереження структурованих даних. Ці дані організовано у вигляді таблиць, які містять рядки та стовпці, забезпечуючи ефективне зберігання та швидкий доступ;

– Управління даними: Сервер баз даних забезпечує механізми для додавання, оновлення, видалення та запитів даних. Це включає в себе мову SQL (Structured Query Language), яка використовується для взаємодії з базою даних;

– Забезпечення цілісності даних: СУБД надає механізми для підтримання цілісності даних, що означає, що дані в базі залишаються коректними та узгодженими. Це включає в себе обмеження, тригери та інші механізми;

– Транзакційна обробка: СУБД підтримує транзакції, які являють собою логічні одиниці роботи, що гарантують цілісність даних навіть у разі збоїв;

Популярні СУБД включають MySQL, PostgreSQL, Microsoft SQL Server, Oracle Database та інші. Вибір конкретної СУБД залежить від вимог проєкту, обсягу даних, типу застосунку та інших чинників.

3.1.4. DNS

Система доменних імен (DNS) є критично важливим компонентом архітектури веб-додатків. Вона перетворює доменні імена (наприклад, www.example.com), що читаються людиною, в IP-адреси (наприклад, 192.0.2.1), які можуть зрозуміти комп'ютери.

Його основна функція - забезпечити користувачам доступ до веб-ресурсів за допомогою доменних імен, що легко запам'ятовуються, а також полегшити комунікацію між веб-серверами і клієнтами.

3.1.5. Проміжне програмне забезпечення для обміну повідомленнями

Проміжне програмне забезпечення для обміну повідомленнями, відоме також як Message Oriented Middleware (MOM), є ключовим компонентом у системах, де необхідно забезпечити ефективний і надійний обмін повідомленнями між різними компонентами. Ось деякі аспекти цього проміжного програмного забезпечення:

– Черги повідомлень: Однією з основних концепцій MOM є використання черг повідомлень для асинхронного обміну інформацією. Компоненти надсилають повідомлення до черги, а потім інші компоненти витягують їх із черги для обробки.

– Публікація-підписка: MOM підтримує модель "публікація-підписка", де компоненти можуть підписуватися на певні типи повідомлень.

Коли повідомлення цього типу публікується, усі передплатники отримують його для обробки.

– Надійність доставки: Проміжне програмне забезпечення забезпечує механізми надійного доставлення повідомлень, навіть у разі збоїв або тимчасової недоступності компонентів системи.

– Маршрутизація повідомлень: Проміжне програмне забезпечення може надавати служби з маршрутизації повідомлень, спрямовуючи їх від відправника до одержувача з огляду на різні параметри та правила.

Застосування проміжного програмного забезпечення для обміну повідомленнями дає змогу будувати гнучкі та відмовостійкі системи, особливо в розподілених і мікросервісних архітектурах.

Популярні рішення проміжного програмного забезпечення для обміну повідомленнями: Apache Kafka, RabbitMQ та ActiveMQ.

3.1.6. Load balancer

Load Balancer (балансувальник навантаження) являє собою важливий компонент в інфраструктурі мережі, призначений для ефективного розподілу навантаження між різними серверами або ресурсами. Ось кілька ключових аспектів, пов'язаних із роботою Load Balancer:

– Розподіл навантаження: Основне завдання балансувальника навантаження - рівномірний розподіл вхідних запитів від користувачів між кількома серверами. Це сприяє поліпшенню продуктивності та ефективності роботи всієї системи.

– Керування трафіком Load Balancer може забезпечувати більш ефективне управління трафіком, перенаправляючи запити залежно від поточного навантаження на сервери. Це дає змогу запобігати перевантаженням і забезпечувати стабільну роботу системи. [25]

– Обробка збоїв: Балансувальники навантаження можуть виявляти збої в роботі серверів і автоматично перенаправляти трафік на здорові сервери. Це підвищує відмовостійкість системи та забезпечує безперервну доступність сервісів.

– Типи балансування: Існує кілька типів балансування, включно з Round Robin (циклічне), Least Connections (вибір сервера з найменшим числом активних з'єднань), і IP Hash (заснована на хеші IP-адреси клієнта).

– Глобальне балансування навантаження: Для розподілених систем використовується глобальне балансування навантаження, яке може враховувати географічне положення клієнтів і серверів для оптимізації трафіку.

– Масштабування горизонтально: Load Balancer забезпечує можливість горизонтального масштабування, додаючи або видаляючи сервери залежно від обсягу трафіку та вимог системи. [26]

Застосування балансувальників навантаження є невід'ємною частиною високодоступних і масштабованих веб-додатків, забезпечуючи ефективне використання ресурсів і запобігаючи проблемам із продуктивністю.

Популярні Load balancer навантаження: HAProxy, NGINX і F5.

3.1.7. Cache

Cache - це компонент інфраструктури, який зберігає дані або ресурси, до яких часто звертаються, у швидкодоступній пам'яті або сховищі. Його основна мета - підвищити продуктивність і масштабованість.

Популярні рішення для кешування: Redis, Memcached та Varnish.

3.1.8. CDN

Мережа доставки контенту (CDN) - це мережа глобально розподілених серверів, яка доставляє контент користувачам з найближчого до них сервера.

Це покращує продуктивність і доступність веб-додатків за рахунок зменшення затримок і перевантаження мережі.

Популярні CDN: Cloudflare, Akamai та Amazon CloudFront.

Об’єднавши усі написані вище компоненти отримаймо діаграму сучасного веб-додатку (Рисунок 3.2). Також згрупуймо ці по рівнях та їх призначенню (Таблиця 3.1) та розглянемо їх більш детально.

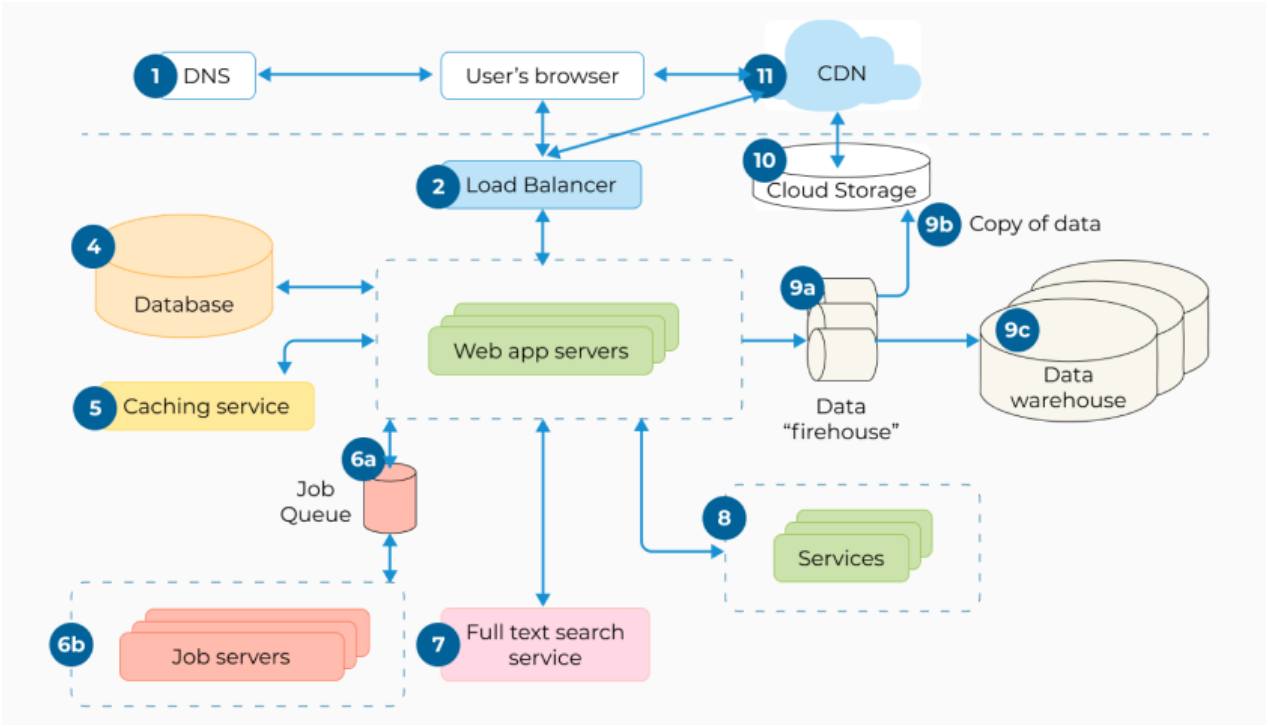


Рисунок 3.2 – Архітектура сучасного веб-застосунку

Таблиця 3.1 – Рівні архітектури веб-додатку

Рівень	**Компоненти**
Рівень подання	Компоненти користувацького інтерфейсу, компоненти обробки користувацького інтерфейсу
Бізнес-рівень	Бізнес-процеси, обробка винятків, обробка записів, допоміжні утиліти
Ріваень доступу до даних	Компоненти доступу до даних, шлюзи сервісів
Рівень бази	Даних База даних, SQL-запити, збережені процедури

Нижче наведено опис чотирьох рівнів архітектури сучасного веб-додатку:

Рівень презентації

Рівень презентації керує користувацьким інтерфейсом додатку, працюючи з HTML, CSS та JavaScript. Він також отримує вхідні дані користувача і надсилає їх на бізнес-рівень для обробки, взаємодіючи з ним через API або інтерфейси. Рівень представлення зазвичай включає веб-компоненти, такі як контролери, подання та шаблони.

Бізнес-рівень

Бізнес-рівень (або рівень додатку) відповідає за бізнес-логіку веб-додатку. Він містить контролери, сервіси та моделі, що відповідають за виконання необхідних дій для виконання запитів користувачів. Бізнес-рівень взаємодіє з рівнем доступу до даних для отримання або маніпулювання даними за потреби.

Рівень доступу до даних

Рівень доступу до даних (або персистентності) переводить дані програми у формат, який можна зберігати та отримувати зі сховища даних. Він містить компоненти, які взаємодіють з базою даних, такі як об'єкти доступу до даних (DAO), об'єктно-реляційні мапери (ORM) та збережені процедури.

Рівень бази даних

Рівень бази даних включає систему управління базами даних (СУБД) і дані, що зберігаються в базі даних. Цей рівень зберігає дані у структурованому форматі, які можна легко запитувати та маніпулювати ними на рівні доступу до даних.

Розглядаючи наведені рівні інфраструктури буде доречним додати алгоритмічне представлення того, як рівні працюють разом:

– Користувач взаємодіє з презентаційним рівнем, вводячи дані через інтерфейс користувача.

– Рівень представлення отримує вхідні дані користувача і надсилає їх на бізнес-рівень.

– Бізнес-рівень обробляє введені користувачем дані, виконує необхідні дії і отримує або оновлює дані через рівень доступу до даних.

– Рівень доступу до даних отримує або оновлює дані з рівня бази даних і надсилає їх назад до бізнес-рівня.

– Бізнес-рівень обробляє отримані дані і генерує відповідь для рівня представлення.

– Рівень представлення отримує відповідь від бізнес-рівня і відповідно оновлює інтерфейс користувача.

– Процес повторюється, коли користувач вводить додаткові дані або здійснює навігацію в додатку.

3.2. Вибір хмарного провайдеру

Важливим етапом побудови безпечної та архітектурно правильної системи є вибір сучасного хмарного провайдера. На сьогоднішній день використання хмарного провайдера має багато переваг:

- Масштабованість;
- Гнучкість та еластичність;
- Доступність і надійність;
- Зниження витрат на інфраструктуру;
- Глобальний розподіл;
- Оновлення та безпека;
- Економія часу.

Саме тому нижче буде розглянуто та проаналізовано найрозвинутіші на найкращі з точки зору безпеки хмарні провайдери а саме AWS Azure та Google Cloud.

Розглянемо основні відомості про хмарні провайдери та побудуємо таблиці для порівняльного аналізу.

3.2.1. AWS

Amazon Web Services (AWS) пропонує комп'ютерні ресурси та послуги, за допомогою яких можна створювати додатки за лічені хвилини за цінами, що залежать від того, як ви платите. Наприклад, ви можете орендувати сервер на AWS, щоб підключатися до нього, налаштовувати, захищати та запускати так само, як і фізичний сервер. Різниця полягає в тому, що віртуальний сервер працює поверх мережі планетарного масштабу, керованої AWS.

Відомі користувачі Amazon Web Services (AWS):

- Coursera

– Expedia
– Netflix
– Coinbase
– Formula 1
– Intuit
– Airbnb
– Lyft
– Управління з санітарного нагляду за якістю харчових продуктів і медикаментів (FDA)
– Coca Cola

3.2.2. Azure

Microsoft Azure - це загальнодоступна хмарна платформа, яка надає рішення інфраструктури як послуги (IaaS), платформи як послуги (PaaS) та програмного забезпечення як послуги (SaaS) для аналітики, віртуальних обчислень, зберігання даних, мереж та інших послуг. Вона може покращити або замінити ваші локальні сервери.

Відомі користувачі Microsoft Azure:

– Bosch
– Audi
– ASOS
– HSBC
– Starbucks
– Walgreens
– 3M
– FedEx
– Walmart
– HP

- Mitsubishi Electric
- Renault

3.2.3. Google Cloud

Google Cloud, спочатку App Engine, - це пакет послуг хмарних обчислень, створений компанією Google у 2008 році. GCP пропонує підприємствам по всьому світу інфраструктуру як послугу (IaaS), платформу як послугу (PaaS) та програмне забезпечення як послугу (SaaS). Наприклад, GCP - це насамперед сервіс для розробки та підтримки оригінальних додатків, які потім можуть бути опубліковані з його гіпермасштабних центрів обробки даних.

Найвідоміші користувачі Google Cloud Platform (GCP):

- Toyota
- Equifax
- Nintendo
- Spotify
- The Home Depot
- Target
- Twitter
- Paypal
- UPS

3.2.4. AWS vs. Azure vs. Google Cloud: ключові відмінності

AWS, Microsoft Azure та Google Cloud Platform охоплюють новий цифровий світ новим шквалом технологій, заснованих на віддалених серверах. На ринку публічних хмар існує жорстка конкуренція, і ось що відрізняє кожну платформу зображено на Таблиці 3.2.

Таблиці 3.2 – Ключові відмінності хмарних провайдерів

Особливості	**Amazon**	**Microsoft Azure**	**Google Cloud**
Час на ринку	11 років	5 років	6 років
Ціноутворення	Ціноутворення за секунду з мінімальним обмеженням у 60 секунд	За хвилину	За хвилину
Обчислення	EC2 (Elastic Compute Cloud) надає всю адміністрацію обчислень. Програма керує віртуальними машинами, які можуть бути створені власником або мати попередньо налаштовані параметри для зручності	За допомогою Microsoft Azure ви можете створювати віртуальні машини та набори масштабування для віртуальних машин	В рамках Google Cloud Platform GCE (Google Compute Engine) виконує схожі функції
Зберігання	AWS надає розподілене, тимчасове (короткострокове) зберігання. На початку сессії сервер, він знищується в кінці сесії	Azure використовує ID-диски (тимчасове зберігання), а також Page Blobs для зберігання обсягів на основі віртуальних машин. Об’єктне зберігання використовує Square Blobs та Files	У порівнянні з іншими платформами Google Cloud Platform пропонує як тимчасове, так і постійне зберігання. Для об’єктного зберігання GCP має Google Cloud Storage

Розуміння історії кожної платформи - це перший крок в оцінці різних постачальників хмарних послуг. Кожна послуга починалася в іншому місці, що впливає на те, як провайдери фокусують свої пропозиції.

Amazon Web Services

AWS стала публічною у 2006 році з такими сервісами, як Elastic Compute Cloud (EC2) та Simple Storage Service (Amazon S3). Elastic Block System (EBS) стала загальнодоступною у 2009 році, а до неї додалися такі сервіси, як Amazon CloudFront і Content Delivery Network (CDN). Це один з перших хмарних провайдерів, який має широку базу користувачів і вищий рівень довіри та надійності.

Microsoft Azure

Microsoft Azure, спочатку відома як Azure, була створена в 2010 році, щоб забезпечити підприємства потужною платформою для хмарних обчислень. У 2014 році Azure було перейменовано на "Microsoft Azure", хоча "Azure" все ще широко використовується. З моменту свого дебюту Microsoft Azure досягла значного прогресу порівняно з конкурентами.

Хмарна платформа Google

Хмарна платформа Google була запущена у 2008 році, і менш ніж за десять років вона зайняла міцні позиції у хмарному бізнесі. Google Cloud посилила продукти Google, включаючи надзвичайно популярну пошукову систему та платформу для обміну відео YouTube. Однак зараз вони запустили корпоративні сервіси, що дозволяють будь-кому отримати доступ до Google Cloud Platform, яка має ту саму інфраструктуру, що й Google Search або YouTube.

Регіони та доступність

При виборі хмарного провайдера перше, на що слід звернути увагу, - це підтримувані регіони та доступність. Адже такі питання, як затримка і дотримання нормативних вимог, особливо при роботі з даними, мають безпосередній вплив на продуктивність вашої хмари.

Ось трійка лідерів станом на вересень 2021 року:

Amazon Web Service поділяється на 22 географічні регіони та 14 центрів обробки даних. Існує понад 114 периферійних локацій та 12 регіональних периферійних кешів.

Microsoft Azure працює в 54 регіонах, кожен з яких має щонайменше три зони доступності та 116 периферійних локацій.

Google Cloud Platform складається з 34 хмарних регіонів, 103 зон та понад 200 периферійних локацій.

3.2.5. Переваги та недоліки

Amazon Web Services

Плюси

– Надає більшість послуг, від мереж до роботів.
– Найдосконаліший
– Вважається найкращим за надійністю та безпекою
– Більше обчислювальних потужностей, ніж у Azure та GC

Мінуси

– Всі основні постачальники програмного забезпечення, які роблять свої додатки доступними на AWS Dev/Enterprise, повинні платити за підтримку.
– Величезна кількість доступних послуг та опцій може бути приголомшливою для новачків.
– Існує відносно мало гібридних хмарних альтернатив.

Microsoft Azure

Переваги

– Інтеграція та міграція поточних служб Microsoft є простими.

– Доступно багато опцій, включаючи найкращий у своєму класі штучний інтелект, машинне навчання та аналітичні сервіси.

– Більшість сервісів є дешевшими у порівнянні з AWS та GCP.

– Широка підтримка гібридних хмарних стратегій.

Мінуси

– Менший вибір сервісів у порівнянні з AWS

– Спеціально розроблений для бізнес-клієнтів

Google Cloud

Переваги

– Добре працює з іншими сервісами та продуктами Google.

– Відмінна підтримка контейнерних робочих навантажень

Недоліки

– Обмежений набір послуг порівняно з AWS та Azure

– Обмежена підтримка корпоративних сценаріїв використання

Кожен бізнес має унікальні потреби, і кожен постачальник послуг реагує на них по-своєму. Наприклад, розробник програмного забезпечення, фінансова установа та компанія, що займається електронною комерцією, використовують хмарні сервіси по-різному.

На них поширюються різні регуляторні зобов'язання. Тим часом, постачальники хмарних послуг можуть пропонувати подібні послуги, як і будь-який інший бізнес, але часто виокремлюють свою нішу, яка добре працює для потенційних покупців. В рамках конкретної задачі, а саме отримання максимальної безпеки додатку, буде використана хмарна платформа AWS.

4. РОЗРОБЛЕННЯ БЕЗПЕКИ ВЕБ-ДОДАТКУ

4.1. Розроблення та реалізація механізмів автентифікації та авторизації на базі AWS

Одним за най перших та най важливіших кроків у створенні безпечного додатку з безпечною інфраструктурою є додавання механізму автентифікації. Контроль доступу дозволяє тільки авторизованим клієнтам отримувати доступ до ресурсів внутрішнього сервера шляхом автентифікації клієнта та надання доступу на гранулярному рівні на основі того, хто є клієнтом.

Розглянемо три архітектурні моделі рішень, які запобігають доступу неавторизованих клієнтів до внутрішніх серверів веб-додатків. У цих архітектурних патернах використовуються сервіси AWS, які відповідають вимогам різних варіантів використання.

Флоу автентифікації OAuth 2.0

Рисунок 4.1 демонструє основи всіх архітектурних патернів, що обговорюються в рамках данного розділу.

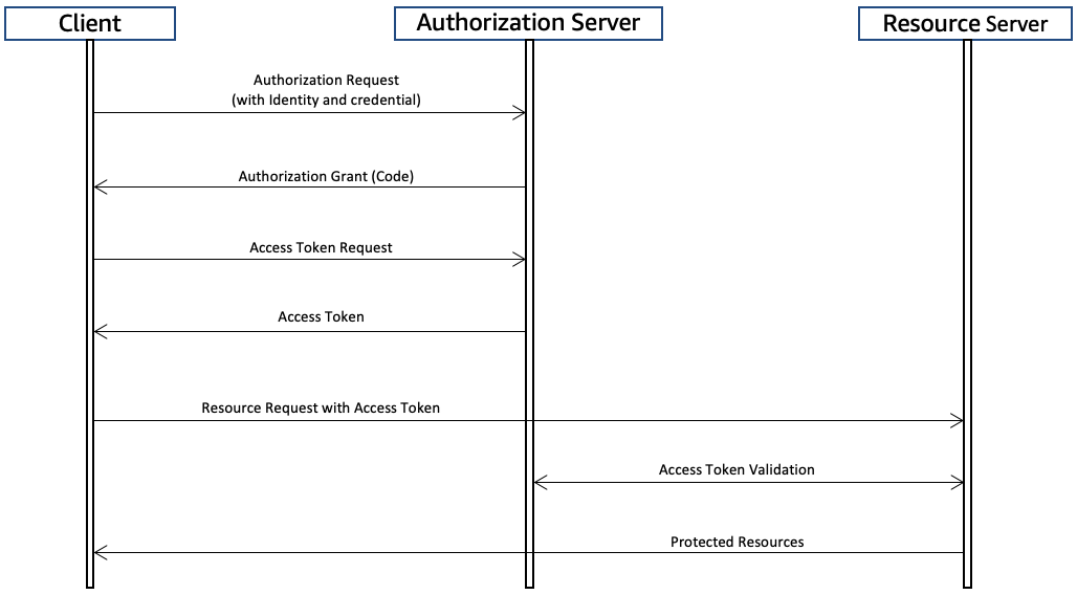


Рисунок 4.1 – Логіка роботи автентифікації OAuth 2.0

Архітектурні моделі, описані в даному розділі, використовують Amazon Cognito як сервер авторизації, та сервер(и) Amazon Elastic Compute Cloud як сервер ресурсів. Клієнтом може бути будь-який інтерфейсний додаток, наприклад, мобільний додаток, який надсилає запит на сервер ресурсів для доступу до захищених ресурсів.

4.1.1. Патерн 1

На рисунку 4.2 зображено архітектурний шаблон, який перекладає роботу з автентифікації клієнтів на Application Load Balance (ALB).

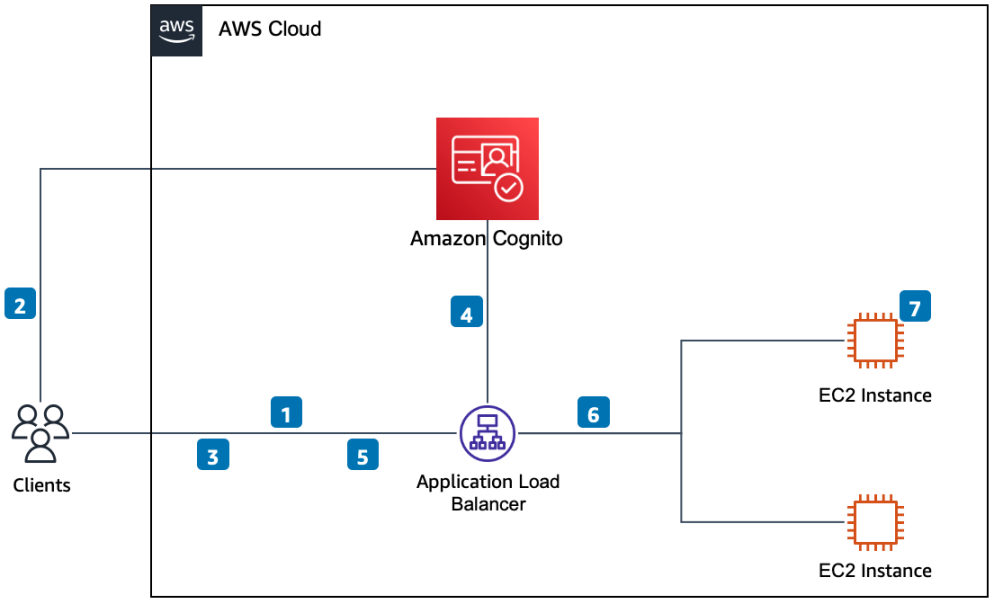


Рисунок 4.2 – Інтеграція Application Load Balance з Amazon Cognito

Application Load Balance можна використовувати для автентифікації клієнтів через пул користувачів Amazon Cognito:

– Клієнт надсилає HTTP-запит на кінцеву точку ALB без файлів cookie сеансу автентифікації.

– ALB перенаправляє запит на кінцеву точку автентифікації Amazon Cognito. Клієнт проходить аутентифікацію на Amazon Cognito.

– Клієнт повертається назад до ALB з кодом автентифікації.

– ALB використовує код автентифікації для отримання токена доступу з кінцевої точки Amazon Cognito, а також використовує токен доступу для отримання вимог клієнта до користувача з кінцевої точки Amazon Cognito UserInfo.

– ALB готує сеансовий файл cookie автентифікації, що містить зашифровані дані, і перенаправляє запит клієнта разом із сеансовим файлом cookie. Клієнт використовує сесійний файл cookie для всіх подальших запитів. ALB перевіряє сеансовий файл cookie і вирішує, чи може запит бути переданий адресату.

– Перевірений запит пересилається до серверних екземплярів з додаванням HTTP-заголовків, які містять дані з токену доступу та інформацію про вимоги користувача.

– Внутрішній сервер може використовувати інформацію в доданих ALB заголовках для управління дозволами на гранулярному рівні.

Основна ідея цього шаблону полягає в тому, що ALB підтримує весь контекст автентифікації, запускаючи автентифікацію клієнта за допомогою Amazon Cognito і готує файл cookie сеансу автентифікації для клієнта. URL-адреса зворотного виклику для входу в Amazon Cognito вказує на ALB, що дозволяє ALB отримати доступ до коду автентифікації.

4.1.2. Патерн 2

Шаблон, показаний на рисунку 4.3, перекладає роботу з автентифікації клієнтів на Amazon API Gateway.

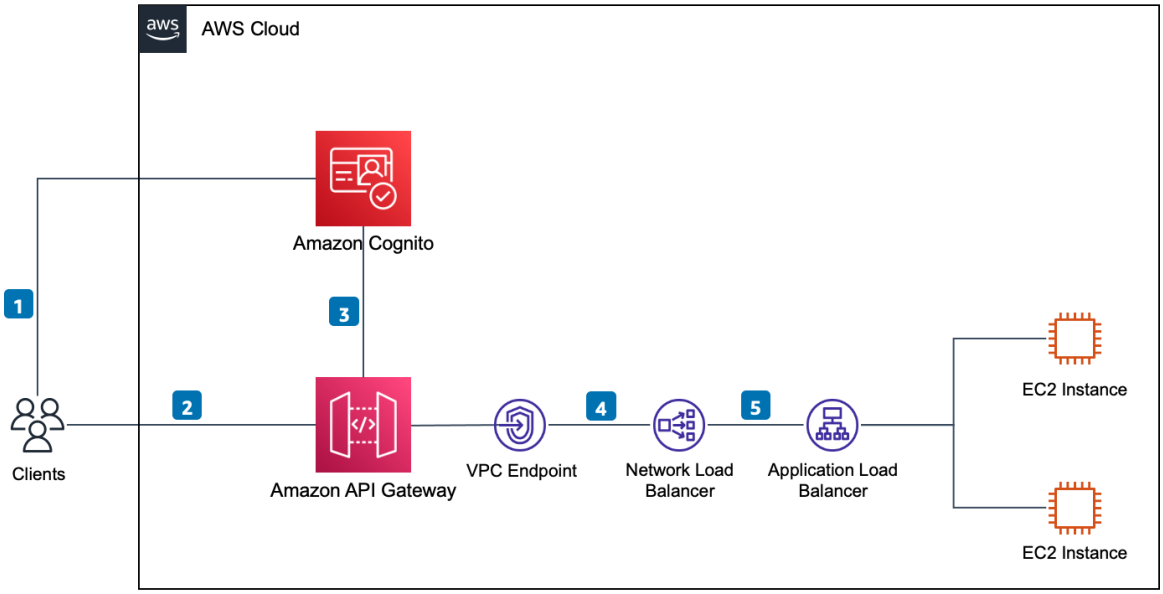


Рисунок 4.3. – Інтеграція Amazon API Gateway з Amazon Cognito

API Gateway може підтримувати як REST, так і HTTP API. Шлюз API інтегрований з Amazon Cognito, а також може мати контрольний доступ до HTTP API за допомогою авторизатора JSON Web Token (JWT), який взаємодіє з Amazon Cognito. ALB може бути інтегрований зі шлюзом API. Клієнт відповідає за автентифікацію в Amazon Cognito для отримання токена доступу.

– Клієнт починає автентифікацію в Amazon Cognito для отримання токена доступу.

– Клієнт надсилає запит REST API або HTTP API із заголовком, що містить токен доступу.

– Шлюз API налаштований так, щоб мати:

– Пул користувачів Amazon Cognito як авторизатор для перевірки токена доступу в запиті REST API, або

– Авторизатор JWT, який взаємодіє з пулом користувачів Amazon Cognito для перевірки токену доступу в HTTP API запиті.

– Після перевірки токена доступу REST або HTTP API запит перенаправляється до ALB, і:

– Шлюз API може спрямовувати HTTP API до приватного ALB через VPC endpoint.

– Якщо використовується загальнодоступний ALB, API-шлюз може спрямовувати як REST API, так і HTTP API до ALB.

– Шлюз API не може безпосередньо спрямовувати REST API до приватного ALB. Він може спрямовувати до приватного Network Load Balancer (NLB) через кінцеву точку VPC. Приватний ALB можна налаштувати як ціль NLB.

Основні висновки до цього шаблону такі:

– Шлюз API має вбудовані функції для інтеграції пулу користувачів Amazon Cognito для авторизації REST та/або HTTP API запитів.

– ALB можна налаштувати так, щоб він приймав лише HTTP API запити від кінцевої точки VPC, визначеної шлюзом API.

4.1.3. Патерн 3

Amazon CloudFront може запускати функції AWS Lambda, розгорнуті на периферії AWS. Цей шаблон (Рисунок 4.4) використовує функцію Lambda@Edge, де він може виступати в ролі авторизатора для перевірки клієнтських запитів, які використовують токен доступу, що зазвичай міститься в заголовку HTTP Authorization.

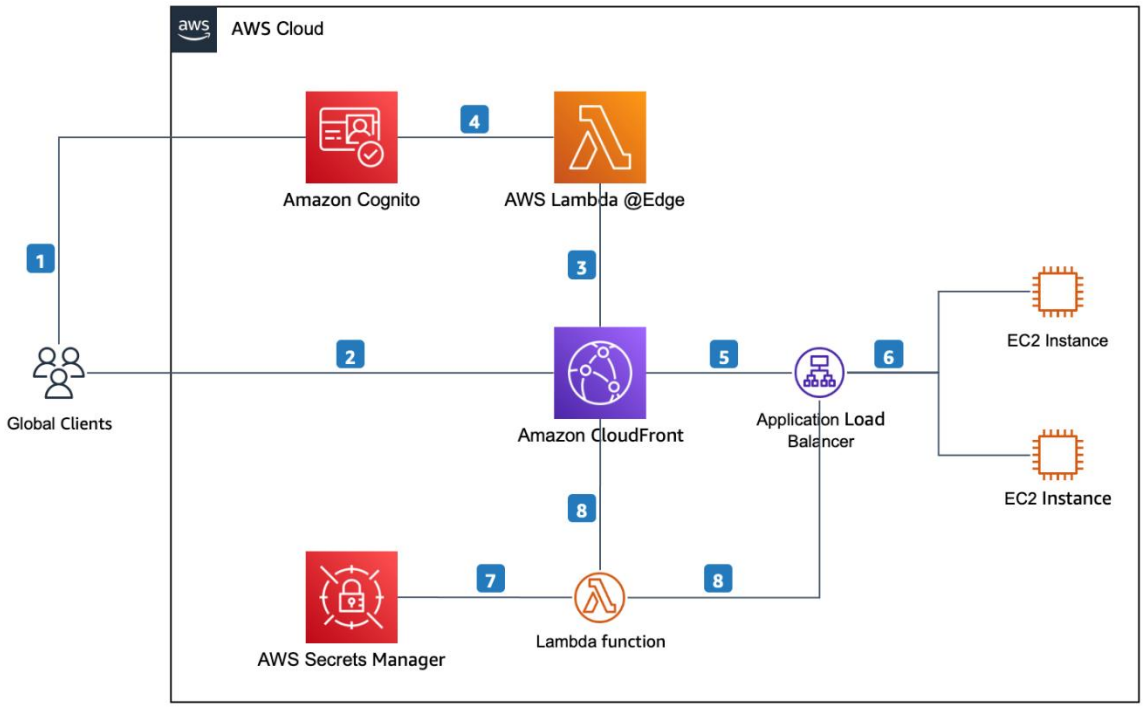


Рисунок 4.4 – Використання Amazon CloudFront та AWS Lambda@Edge з Amazon Cognito

Клієнт може мати індивідуальний потік автентифікації з Amazon Cognito для отримання токену доступу перед відправленням HTTP-запиту.

– Клієнт починає автентифікацію за допомогою Amazon Cognito, щоб отримати токен доступу.

– Клієнт надсилає HTTP-запит із заголовком Authorization, який містить токен доступу, на URL-адресу дистрибутива CloudFront.

– Подія запиту у вьювері CloudFront запускає функцію в Lambda@Edge.

– Лямбда-функція витягує токен доступу з заголовка авторизації та перевіряє його за допомогою Amazon Cognito. Якщо токен доступу не дійсний, запит відхиляється.

– Якщо токен доступу підтверджено, запит авторизується і пересилається CloudFront до ALB. CloudFront налаштовано на додавання спеціального заголовка зі значенням, яке може бути доступне лише для ALB.

– ALB встановлює правило для слухача, щоб перевірити, чи вхідний запит містить спеціальний заголовок зі значенням, до якого надається спільний доступ. Це гарантує, що інтернет-версія ALB приймає лише ті запити, які пересилає CloudFront.

– Для підвищення безпеки спільне значення спеціального заголовка можна зберігати в AWS Secrets Manager. Secrets Manager може запускати пов'язану з ним лямбда-функцію для періодичного обертання секретного значення.

– Лямбда-функція також оновлює CloudFront для доданого користувацького заголовка і ALB для спільного значення в правилі.

Основні висновки з цього шаблону такі:

– За замовчуванням CloudFront видаляє заголовок авторизації перед пересиланням HTTP-запиту до його джерела. CloudFront потрібно налаштувати так, щоб пересилати заголовок авторизації до джерела ALB. Внутрішній сервер використовує токен доступу для застосування гранульованих рівнів дозволу доступу до ресурсів.

– Використання Lambda@Edge вимагає, щоб функція була розташована в регіоні us-east-1.

– Значення кастомного заголовка, доданого CloudFront, зберігається в таємниці, доступ до якої може бути наданий лише ALB.

4.2. Впровадження WAF у веб-додаток

AWS Web Application Firewall (WAF) - це інструмент безпеки, який допомагає захистити додаток від веб-атак. WAF відстежує і контролює незвичайний бот-трафік і блокує поширені шаблони атак, такі як SQL-ін'єкції, міжсайтовий скриптинг тощо. Він також дозволяє відстежувати HTTP- і HTTPS-запити, які перенаправляються до Amazon API Gateway API, Amazon CloudFront або балансувальника навантаження додатків.

– Amazon WAF дозволяє контролювати ваш контент за допомогою IP-адреси, з якої надходить запит.

– Amazon WAF працює завдяки трьом елементам - спискам контролю доступу (ACL), правилам і групам правил.

– Amazon WAF керує одиницями пропускної здатності веб-списку ACL (WCU) для правил, груп правил і веб-списків ACL.

– Amazon WAF включає повнофункціональний API, який ви можете використовувати для автоматизації створення, розгортання та обслуговування правил безпеки.

Брандмауер веб-додатків Amazon пропонує своїм користувачам багато функцій, перелічених нижче.

– Захист від веб-атак: З мінімальним впливом на затримку вхідного трафіку, WAF AWS пропонує безліч правил для перевірки будь-якого елемента веб-запиту. WAF AWS захищає веб-додатки від загроз, фільтруючи трафік відповідно до створених правил.

– Встановіть правила відповідно: WAF AWS є універсальним і цінним інструментом для захисту інфраструктури додатків. І це тому, що він дозволяє користувачам встановлювати правила відповідно до своїх потреб і вразливостей, які вони хочуть зупинити. Ми можемо вважати його чудовим рішенням для захисту будь-якого середовища веб-додатків на рівні підприємства.

– Фільтрація веб-трафіку: WAF дозволяє користувачам створювати правила для фільтрації веб-трафіку. Він фільтрує IP-адреси, заголовки HTTP, тіла HTTP або рядки URI з веб-запиту.

– Гнучка інтеграція з сервісами AWS: AWS Firewall пропонує легку інтеграцію з іншими сервісами AWS, такими як Amazon EC2, CloudFront, Load balancer тощо.

– Правила моніторингу: Брандмауер веб-додатків AWS дозволяє створювати правила, переглядати та налаштовувати їх, щоб запобігти невідомим втручанням.

Брандмауер веб-додатків AWS захищає додатки від зловмисних атак. Робота WAF в AWS описана нижче (Рисунок 4.5).

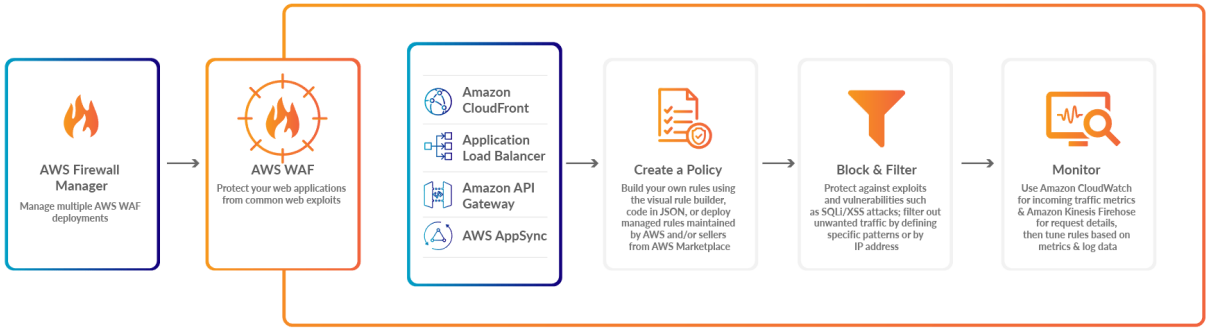


Рисунок 4.5 – Принцип роботи AWS WAF

Управління брандмауером AWS: Керує кількома розгортаннями брандмауера веб-додатків AWS

AWS WAF: захист розгорнутих додатків від поширених веб-експлоітів.

Створення політики: Тепер ви можете створювати власні правила за допомогою візуального конструктора правил.

Блокові фільтри: Блокові фільтри захищають від атак з використанням експлойтів та вразливостей.

Моніторинг: Amazon CloudWatch для збирання метрик вхідного трафіку та Amazon kinesis firehose для деталей запитів, а потім налаштовуйте правила на основі метрик і даних журналів.

4.2.1. Кроки інтеграції AWS WAF

Розглянемо кроки интеграції AWS WAF у інраструктуру.WAF AWS відстежує всі вхідні та вихідні веб-запити, які перенаправляються на API Gateway, Amazon CloudFront і Application Load Balancer. Ми розглянемо, як почати роботу з WAF і створити веб ACL в кілька кроків.

Крок 1: Створення Web ACL: AWS Console і знайдіть Web Application Firewall. Ви потрапите на домашню сторінку WAF (Рисунок 4.6) і оберемо пункт Create Web ACL.

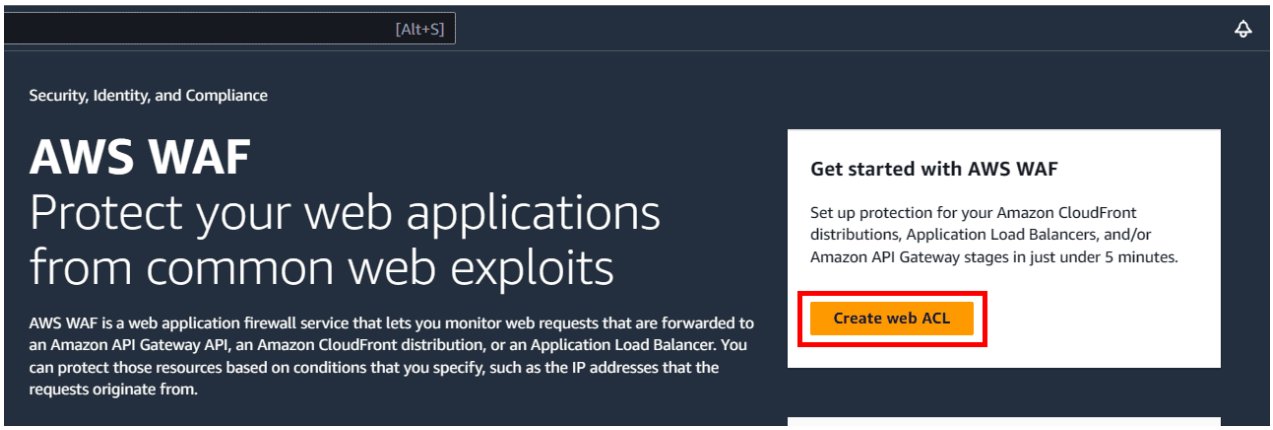


Рисунок 4.6 – Стартова сторінка AWS WAF

Крок 2: Дати назву: Введемо ім'я, обране для ідентифікації цього веб-списку. Після цього введіть опис, якщо потрібно (необов'язково), а потім натисніть Далі.

Крок 3: Додамо групу правил AWS Managed Rules: На наступному кроці потрібно додати правила і групи правил. Натиснемо на Add managed rule groups. Ви потрапите на нову сторінку для керування групою правил.

AWS Managed Rules надає вам колекцію керованих груп правил. Більшість з них є безкоштовними для користувачів Amazon WAF. Після додавання групи керованих правил оберемо збереження правила.

Правила, які створемо, визначатимуть шаблони, які ми хочемо дозволити/заблокувати. Ми додаємо лише 2 правила (Рисунок 4.6).

Regular rule: Це правило захищає програму від атак SQL-ін'єкцій. Воно перевіряє, чи містить шлях до URI ін'єкцію SQL.

Rate-based rule: Це правило блокує запити, зроблені з однієї IP-адреси, якщо вони перевищують певний ліміт за певний проміжок часу.

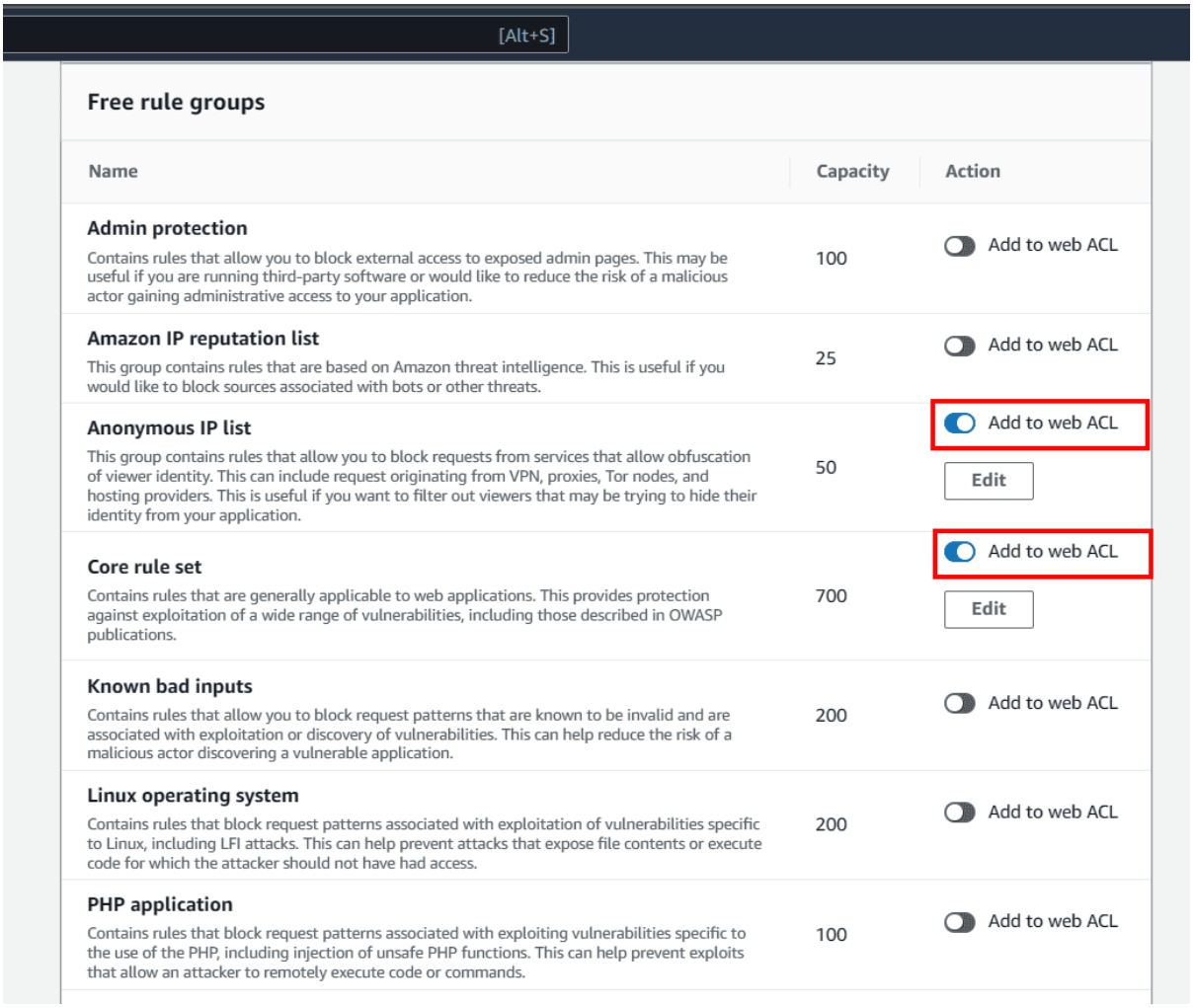


Рисунок 4.6 – Список правил AWS WAF

Після цього перевіримо додані правила і натиснимо Next (Рисунок 4.6).

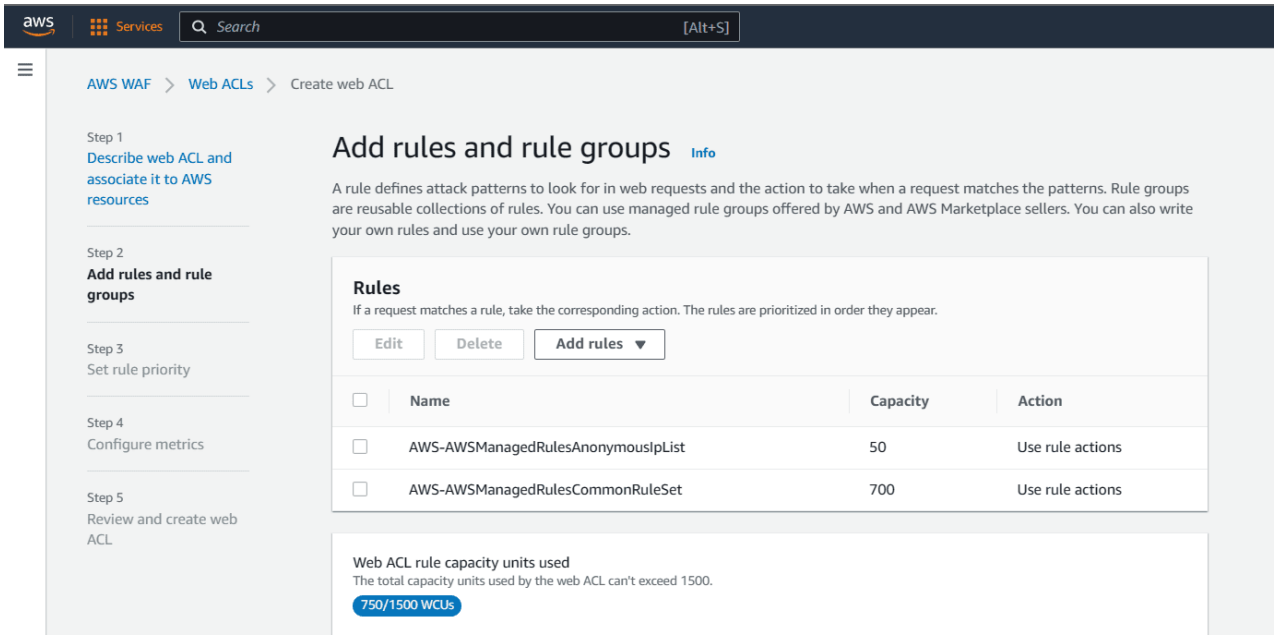
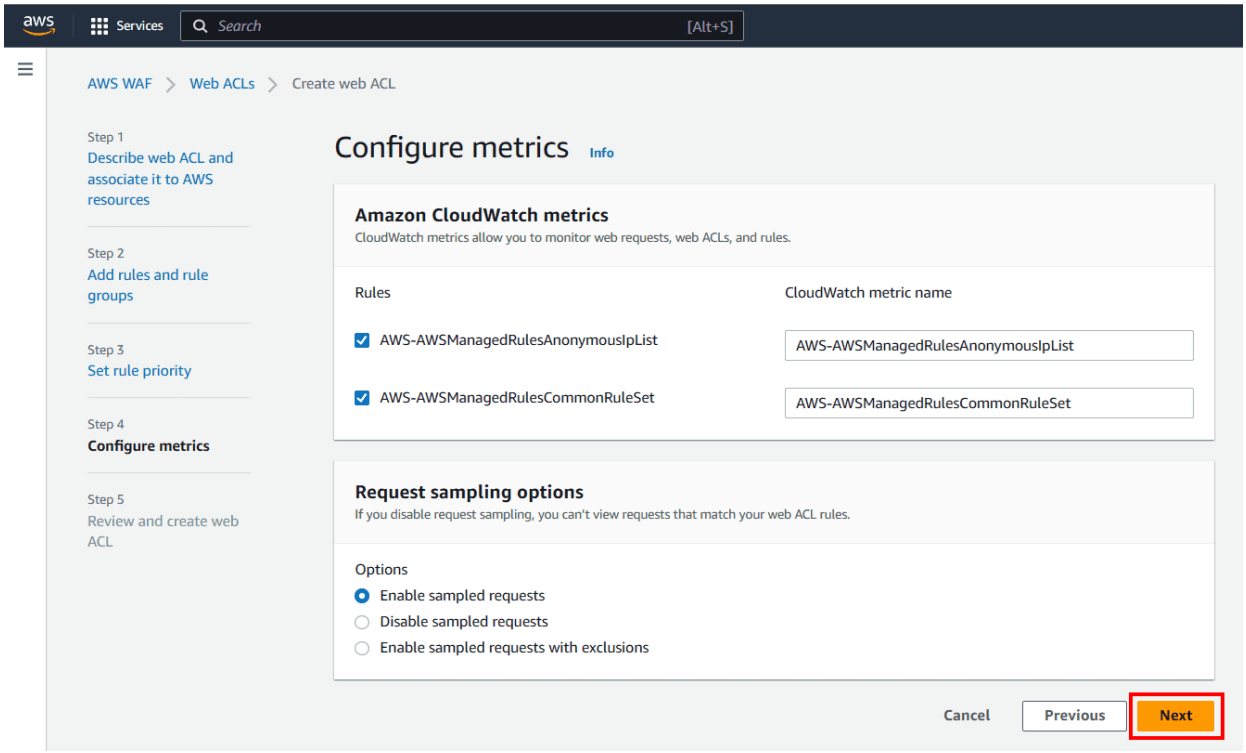


Рисунок 4.7 – Створені правила AWS WAF

Крок 4: Налаштуємо показники моніторингу Cloudwatch (Рисунок 4.7):



Крок 5: Переглянемо конфігурацію веб ACL: На останньому кроці перевірте всі правила і керовані групи і натисніть на кнопку Create web ACL.

Нарешті, з'явиться повідомлення You Successfully created web ACL

4.3. Реалізація системи моніторингу та реагування на інциденти

Хмарний провайдер AWS має сервіс CloudWatch. Amazon CloudWatch - це сервіс, який відстежує додатки, реагує на зміни продуктивності, оптимізує використання ресурсів та надає інформацію про стан роботи. Збираючи дані з усіх ресурсів AWS, CloudWatch дає уявлення про продуктивність всієї системи і дозволяє користувачам встановлювати тривоги, автоматично реагувати на зміни та отримувати уніфіковане уявлення про стан роботи. Розглянемо його інтеграцію у нашу ситему.

4.3.1. Аналіз логів системи

Служба журналів AWS Cloudwatch має можливість зберігати кастомні журнали, згенеровані вашими екземплярами додатків. Журнали доступу до веб-серверів Apache та Nginx або журнали помилок можуть бути перенесені до логів Cloudwatch. Він виступає в якості центрального управління журналами для ваших додатків, що працюють на AWS.

Нижче розглянуто кроки з налаштування агента Cloudwatch на сервері ec2 і налаштуємо його на перенаправлення потрібних логів.

Пересилання журналів програм до Cloudwatch

Можливо надсилати журнали з будь-якої кількості джерел до Cloudwatch. Все, що потрібно - це агент Cloudwatch, запущений на вашому сервері.

Ось які крооки потрібно зробити:

- Створіть власну роль ec2 IAM з доступом на запис до Cloudwatch;

– Встановити агент Cloudwatch logs ec2;
– Налаштувати джерела логів у файлі конфігурації агента Cloudwatch;
– Перевірити журнали в інформаційній панелі Cloudwatch. (Рисунок 4.8);

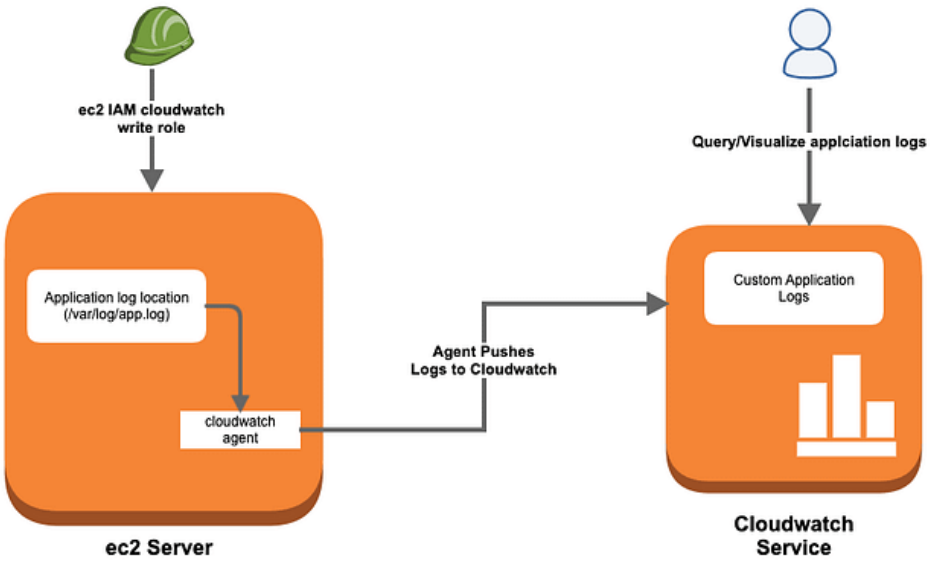


Рисунок 4.8 – Архітектура пересилання логів з S3 до CloudWatch

Створення ролі IAM для Cloudwatch

Щоб налаштувати кастомні журнали AWS, спочатку потрібно створити і додати роль IAM до вашого екземпляра. Ця роль IAM матиме доступ на запис до сервісу Cloudwatch, щоб усі журнали можна було надсилати до Cloudwatch.

Перш ніж створювати роль, вам потрібно створити власну політику.

Крок 1: Перейдіть до AWS IAM -> Політика -> Створити політику

Крок 2: Виберіть опцію JSON

Крок 3: Скопіюйте наступний вміст у блок політики (Рисунок 4.9). На наступній сторінці дайте назву, опис для вашої політики і натисніть опцію "Створити політику".

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogStreams"
    ],
      "Resource": [
        "arn:aws:logs:*:*:*"
    ]
  }
 ]
}
```

Рисунок 4.9 – Плотика для доступу до CloudWatch

Після створення політики потрібно створити роль зі створеною вами кастомною політикою.

Крок 4: Перейдіть до AWS IAM -> Ролі та виберіть опції.

Крок 5: У фільтрі виберіть "Менеджер клієнтів" і виберіть політику, яку створено на кроці 3.

Додавання ролі Cloudwatch до серверу

Тепер перейдимо до ec2 і виберемо сервер, в якому хочемо налаштувати кастомні журнали.

Натиснемо правою кнопкою миші на опціях і виберіть Налаштування екземпляра, а потім виберемо опцію Приєднати/замінити роль IAM.

На наступній сторінці виберемо створену вами роль IAM для хмарного спостереження зі спадного списку і виберіть застосувати.

Налаштування агента журналів Cloudwatch

SSH на екземпляр ec2 і виконаймо наведені нижче дії.

Крок 1: Завантажемо офіційний скрипт налаштування агента Cloudwatch

Крок 2: Запустімо скрипт python, вказавши ваш регіон AWS як параметр.

Перед налаштуванням агента потрібно зрозуміти кілька речей.

Переконаймося, що на вашому сервері встановлено python. Також в останніх версіях серверів за замовчуванням буде доступний python3.

Коли з'явиться запит на введення ключа доступу та секретного ключа, просто натиснимо enter, не вказуючи жодних значень, оскільки ми використовуємо кастомні ролі IAM з дозволами на запис Cloudwatch.

Ви можемо попередньо визначити групу журналів у Cloudwatch і використовувати ту саму назву під час налаштування агента, щоб усі журнали потрапляли до цієї групи журналів.

Надаємо дійсні файли шляху до журналів і власні імена для їх ідентифікації в інформаційній панелі Cloudwatch

- sudo python ./awslogs-agent-setup.py --region us-west-1

Наведений вище скрипт запитає про розташування файлу журналу та інші параметри для керування журналами в Cloudwatch.

Керувати службою агента журналів можна за допомогою наступної команди.

- sudo service awslogs start
- sudo service awslogs stop
- sudo service awslogs restart

Всі конфігураційні файли aws logs та скрипти запуску можна знайти у теці/var/awslogs.

Ви можна додати додаткові конфігурації журналів у файлі /var/awslogs/etc/awslogs.conf. Після внесення змін не забудьте перезапустити агент.

Перевірка користувацьких журналів на інформаційній панелі Cloudwatch

Після завершення налаштування переглянемо всі налаштовані журнали в інформаційній панелі Cloudwatch (в опції "Журнали")

Перейдемо до Logs -> Log Groups та побачимо групу журналів, яку вказали в конфігурації агента.

Оберемо групу журналів, і ви побачимо ідентифікатор екземпляра, який вказали в конфігурації.

Якщо натиснути на ідентифікатор екземпляра, він покаже всі журнали. Можете скористатися фільтром хмарного спостереження, щоб відфільтрувати і запитати необхідні журнали.

4.3.2. Аналіз метрик системи

Панель моніторингу AWS може допомогти візуалізувати продуктивність системи та інтерпретувати метрики для ваших служб і робочих навантажень AWS. Панель моніторингу може надати єдине уявлення про ваші ресурси, агрегуючи інформацію по всьому розгортанню. У нижче наведені інструкція, як створити панель моніторингу за допомогою AWS CloudWatch.

Створення екрану метрик AWS CloudWatch за допомогою консолі AWS показано на Рисунку 4.10.

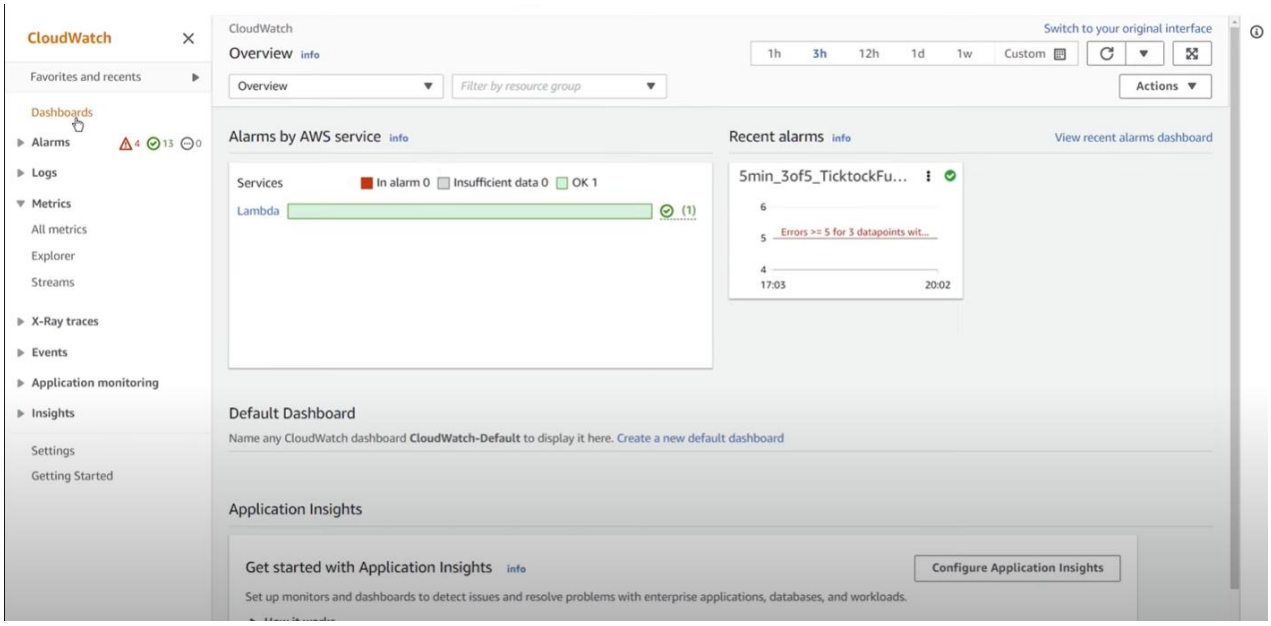


Рисунок 4.10 – Стартова сторінка CloudWatch

Далі наведено короткий посібник, який пояснює, як створити дашборд CloudWatch за допомогою консолі.

Створення дашборду AWS CloudWatch

Перейдімо на сторінку CloudWatch Console і виберемо Dashboards з навігації.

Виберемо Create dashboard і оберемо назву в діалоговому вікні. Після закінчення, натиснемо Create dashboard ще раз.

Назва визначає, де з'явиться ваша дашборд. CloudWatch-Default призведе до того, що вона з'явиться на головній сторінці консолі, тоді як CloudWatch-Default-{ResourceGroupName} призведе до того, що вона з'явиться, коли ви сфокусуєтесь на названій групі.

Додамо свій перший віджет з діалогового вікна Add to this dashboard. Залежно від того, який віджет вам потрібен, ви можете виконати такі дії. Закінчивши, виберіть Create widget..

Додайте текстовий блок - виберемо Текст і Налаштувати. Додайте потрібний текст за допомогою опції Націнка.

Додайте одну метрику - виберемо Номер і налаштуйте. Виберіть метрику, яку ви хочете відобразити.

Додати графік - виберіть тип графіка (Складена область або Лінія) і Налаштувати. Виберіть метрику. Якщо потрібний вам показник не відображається, можна додати його вручну. Після додаймо додаткові віджети на вашу інформаційну панель, вибравши Додати віджет. Повторимо цей крок, поки не додамо всі необхідні віджети, і натискаемо Зберегти дашборд. (Рисунок 4.10)

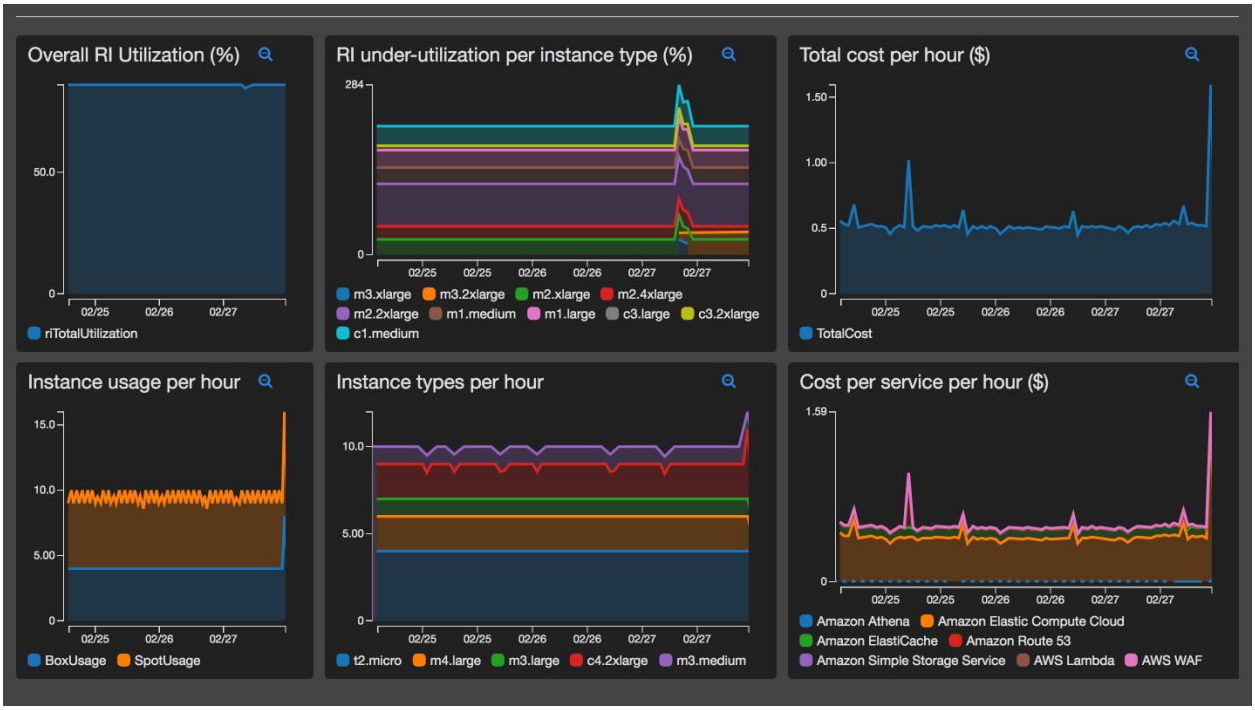


Рисунок 4.11 – Екран метрик CloudWatch

5. ТЕСТУВАННЯ ЕФЕКТИВНОСТІ ІМПЛЕМЕНТОВАНИХ РІШЕНЬ

5.1. Тестування на проникнення

Тестування безпеки-це стратегія тестування, яка використовується для перевірки безпеки системи та аналізу ризиків, пов'язаних із забезпеченням комплексного підходу до захисту додатків від хакерських атак, вірусів та несанкціонованого доступу до конфіденційних даних. Тест призначений для діагностики способів злому системи, оцінки безпеки веб-додатків і веб-сайтів, аналізу веб-сайтів і аналізу ризиків, пов'язаних з підходами до захисту від вторгнень і доступу до конфіденційних даних. На основі принципів конфіденційності, доступності та цілісності тестування безпеки допомагає забезпечити безпеку даних, облікових записів, доступу користувачів та з'єднань.

Загальна стратегія безпеки базується на 3 ключових принципах: конфіденційність, цілісність та доступність. Конфіденційність-це приховування певних ресурсів чи інформації. Під конфіденційністю розуміється обмеження доступу до ресурсу певною категорією користувачів, тобто за умови, що користувач має право доступу до цього ресурсу.

Існує 2 основних критерії для визначення поняття цілісності.

Очікується, що ресурси Trust будуть змінені лише відповідно певною групою користувачів.

Пошкодження та відновлення. Якщо дані пошкоджені або випадково змінені авторизованим користувачем або неавторизованим Користувачем з правами адміністратора. Доступність-це вимога, згідно з якою ресурси повинні бути доступні авторизованим користувачам, внутрішнім об'єктам або

пристроям. Як правило, чим вище важливість ресурсу, тим вище повинен бути рівень доступності.

Існує 3 підходи до виявлення вразливостей у веб-додатках: тестування чорним, сірим і білим ящиком. Різниця між ними визначається ресурсами, доступними під час тестування.

Перший тип, тестування за принципом "білого ящика", в основному передбачає доступ до вихідного коду, технічних специфікацій і всілякої документації.

Навпаки, тести, засновані на принципі "чорного ящика", не вимагають знання внутрішньої структури програми, а тільки вміння взаємодіяти з нею. Прикладом такого тесту є зовнішній аудит веб-програми із закритим кодом.

Нарешті, 3-й тип, тест, заснований на принципі "сірого ящика", означає, що в розпорядженні фахівця є виконуваний файл і, ймовірно, якісь базові документи. Далі ми розглянемо кожен тип більш детально.

5.1.1. Принцип «білого ящика»

Перевірка вихідного коду. Ви можете перевірити вихідний код вручну або за допомогою засобів автоматизації. Враховуючи, що сучасні програми можуть займати сотні тисяч рядків коду, чисто ручний аналіз не здається достатньо ефективним. Інструмент автоматизації позбавляє аналітиків від необхідності ретельно перевіряти кожен рядок програмного коду, але дозволяє їм виявляти лише потенційно вразливі або підозрілі області. Потім кожен із цих розділів потрібно буде вивчити вручну. Звичайно, не існує методу виявлення вразливостей, який був би кращим за інші. Для досягнення найкращих результатів потрібно використовувати всі можливі підходи.

Тестування з використанням методу "білого ящика" має одну велику перевагу - це покриття коду. Оскільки вихідний код доступний, ви можете проаналізувати можливі вразливості.

До недоліків цього методу можна віднести складність, оскільки доступні інструменти відсутні і дають велику кількість помилкових спрацьовувань. Тому звіти, створені в результаті роботи програми, повинні бути ретельно вивчені компетентними експертами. З огляду на обсяг коду, включеного в сучасні додатки, такі звіти можуть мати велику довжину.

5.1.2. Принцип «чорного ящика»

Принцип "чорного ящика" означає, що аналітик може тільки спостерігати за роботою програми, тобто перевіряти надходять в програму отримані дані і аналізувати вихідні дані, але не замислюватися про його внутрішню структуру. Зазвичай це відбувається під час аудиту віддаленої веб-програми.

Ручне тестування. Давайте розглянемо приклад веб-програми. У цьому випадку під час ручного тестування дослідники використовують звичайний Інтернет-браузер для навігації по сторінці програми та модифікації спеціальних символів у полях введення та параметрах запиту; наприклад, одинарні лапки для ідентифікації сценаріїв, потенційно вразливих до ІН'ЄКЦІЙ SQL.

Ручне тестування без використання засобів автоматизації в даний час вважається неефективним.

Автоматичне тестування (розмиття). Незважаючи на те, що оперення засноване на методі грубої сили, недоліки цього підходу компенсуються його простотою і ефективністю. Фактично, розмиття - це передача великої кількості випадкових вхідних даних для обробки досліджуваним додатком і аналізу результатів його роботи.

Варто зазначити, що існують більш просунуті фазери, які генерують відсутні випадкові входи, але залежать від досліджуваних специфікацій протоколу та формату файлу. Такий інструмент також можна класифікувати як метод "сірого ящика". Переваги тестування "чорного ящика" полягають в наступному:

Доступність. Цей метод можна використовувати в будь-якій ситуації і може бути корисним, навіть якщо вихідний код програми доступний.

Універсальність. Оскільки цей підхід не залежить, наприклад, від інформації про певний програмний продукт, 1 інструмент, створений для оцінки безпеки веб-сервера, може використовуватися для інших цілей.

Простота. На самому базовому рівні розмиття не вимагає знання внутрішньої структури програми. Але ясно, що виявити таким чином найскладніші помилки практично неможливо.

Цей метод тестування має багато недоліків.

Охоплення. 1. Однією з найскладніших проблем, яку потрібно вирішити за допомогою процесу набухання, є коли припинити тестування та наскільки воно ефективне.

Примітивний. Розмиття не дуже добре виявляє складні вразливості, такі як вразливості, для усунення яких потрібно кілька кроків, щоб привести програму в певний стан і викликати помилки. Ці вразливості зазвичай виявляються шляхом аналізу вихідного коду.

5.1.3. Принцип «сірого ящика»

Тестування за принципом «сірого ящика» являє собою комбінацію методів, що використовуються при тестуванні за принципом «чорного ящика», а також технологій і прийомів реверс розробки. Цінність вихідного коду в процесі пошуку уразливостей полягає в тому, що він представляє логіку роботи

програми в зрозумілому для дослідника поданні.. Основна мета аналізу – визначення внутрішньої логіки роботи захищеного додатки. Не існує інструмента, що дозволив би отримати оригінальний вихідний код з захищеного файлу (обфускація). Однак, за допомогою засобів реверс розробки можливо представити програму у вигляді, який доступний для сприйняття, хоча це не повноцінний вихідний код.

Даний метод успадковує одну з переваг тестування за принципом «чорного ящика» – доступність. Ще одною істотною перевагою є покриття коду. Інформація, отримана в результаті реверс аналізу, здатна істотно поліпшити якість вхідних даних, які генеруються фаззером.

Великим недоліком даного методу є його складність. Серед розглянутих технологій пошуку вразливостей дана пред'являє найвищі вимоги до кваліфікації аналітика.

5.2. Інструменти для тестування додатків

Нижче розглянемо кілька актуальних на сьогоднішній день інструментів для тестування додатків.

AWS Config - це один з основних інструментів для тестування AWS, який дозволяє оцінювати, перевіряти та оцінювати конфігурацію ваших ресурсів AWS. Відстежуйте історію конфігурації ресурсів і дотримуйтесь стандартів PCI DSS, ISO/IEC 27001:2013, SOC і GDPR, оскільки він постійно відстежує і реєструє зміни конфігурації.

На додаток до того, що він допомагає вам виявити несанкціоновані зміни шляхом тестування на проникнення в AWS, він також визначає і забезпечує бажані конфігурації.

AWS Inspector - це сервіс, який Amazon Web Services (AWS) пропонує для автоматичної оцінки безпеки і тестування на проникнення в AWS. Він

знаходить потенційні недоліки безпеки та порушення найкращих практик у ваших ресурсах AWS. Користувачі отримують доступ до вичерпних звітів, можуть змінювати шаблони оцінки та планувати повторні оцінки.

Цей інструмент пентестування AWS спрощує оцінку безпеки, мінімізує ручну працю та підтримує аудит відповідності. Завдяки інтеграції з іншими сервісами AWS, такими як Amazon CloudWatch Events і AWS Systems Manager, AWS Inspector дозволяє виконувати автоматичні дії.

ScoutSuite - ще один великий гравець серед інших рішень для пентестування AWS. Це програма для аудиту безпеки з відкритим вихідним кодом, яка не обмежується AWS, а також доступна для Microsoft Azure і GCP. Це інструмент пентестування AWS на основі Python, який забезпечує ретельний аудит безпеки та збирає дані про конфігурацію та ресурси з API хмарних провайдерів.

Цей інструмент для тестування AWS розглядає різні сфери хмарної безпеки, включаючи відповідність найкращим практикам, мережеві налаштування, налаштування управління ідентифікацією та доступом (IAM) і права на зберігання даних. ScoutSuite пропонує докладні звіти, в яких висвітлюються можливі вразливості безпеки та помилки конфігурації. Його модульний дизайн дозволяє користувачам адаптувати свої дослідження.

Prowler - один з небагатьох інструментів пентестування AWS з відкритим вихідним кодом для аудиту, реагування на інциденти, безперервного моніторингу, зміцнення і забезпечення готовності до криміналістичних досліджень для середовищ Amazon Web Services (AWS). Він виконує автоматизовану оцінку безпеки для пошуку помилок конфігурації. AWS FTR, ENS, GDPR, HIPAA, FFIEC, SOC2, CIS, PCI-DSS, ISO 27001 та користувацькі фреймворки безпеки є одними з сотень включених засобів контролю.

Prowler можна запустити на вашій робочій станції, екземплярі EC2, Fargate або іншому контейнері, Codebuild, CloudShell, Cloud9 та багатьох інших

платформах. Для створення Prowler використовується мова Python, а також AWS SDK (Boto3), Azure SDK і GCP API Python Client.

CloudSploit - це інструмент для моніторингу та оцінки безпеки хмарних середовищ (AWS), Microsoft Azure та Google Cloud Platform (GCP). Він перевіряє хмарні ресурси на наявність вразливостей безпеки, неправильних налаштувань і порушень нормативних вимог.

Він має гнучкі вихідні формати з консольними таблицями за замовчуванням для безперешкодної інтеграції з іншими кращими інструментами тестування на проникнення в AWS. Він перевіряє конфігурацію стану обслуговування ваших акаунтів AWS IaaS на наявність потенційних вразливостей безпеки та безперервно сканує активність у ваших акаунтах на наявність підозрілих дій та внутрішніх загроз.

Pacu - це безкоштовний фреймворк з відкритим вихідним кодом для експлуатації AWS, призначений для тестування безпеки і тестування на проникнення. Доступна велика колекція інструментів і модулів для оцінки стану безпеки облікових записів AWS і тестування ефективності засобів контролю безпеки. Він підтримує різні сервіси тестування на проникнення в AWS і пропонує гнучкий і розширюваний фреймворк для розширених оцінок безпеки в середовищах AWS.

5.3. Додавання AWS Config як елементу постійного тестування ситеми

AWS Config постійно оцінює, перевіряє та оцінює конфігурації та взаємозв'язки ваших ресурсів у AWS, локальному середовищі та інших хмарах. Увійдемо до AWS Management Console та відкриємо консоль AWS Config. Враховуючи це додаймо його до нашої архітектури.

Розпочнемо налаштування, натиснувши "Почати роботу" на початковій сторінці AWS Config. Потім перейдемо до сторінки налаштувань, оберемо спосіб запису ресурсів і ролей, а також визначимо місце для надсилання історії конфігурації та файлів знімків конфігурації.

Таким чином, ми здійснимо налаштування AWS Config через консоль для ефективного моніторингу та управління конфігурацією ресурсів у вашому обліковому записі AWS.

Проведемо налаштування

В розділі "Метод запису" обирайте стратегію запису, вказавши ресурси AWS, які ви хочете включити в записи AWS Config.

У розділі "Керування даними" обирайте період зберігання даних: встановлюйте період зберігання за замовчуванням на 7 років або визначайте свій власний період для зберігання елементів, записаних AWS Config.

Щодо ролі IAM для AWS Config, обирайте існуючу роль, яка пов'язана зі службою AWS Config, або використовуйте роль IAM зі свого облікового запису. Передвизначені ролі, пов'язані із службами, вже обрані AWS Config та мають всі необхідні дозволи для виклику інших служб AWS.

Зазначайте спосіб доставки в цьому розділі, обираючи відповідні налаштування.

У розділі "Спосіб доставки" обирайте S3-скриньку, до якої AWS Config буде відправляти історію конфігурації та файли знімків конфігурації.

Для створення кошика в полі "Ім'я кошика S3" введіть унікальне ім'я для вашого кошика S3. Це ім'я повинно бути унікальним серед всіх існуючих імен кошиків в Amazon S3. Додавання префіксу, наприклад, назви вашої організації, може забезпечити унікальність. Зверніть увагу, що після створення кошика ви не зможете змінити його ім'я.

Виберіть контейнер зі свого акаунта, обравши потрібний вам контейнер з назви S3.

Для нотифікацій Amazon SNS оберемо Потокове передавання змін конфігурації та сповіщень до Amazon SNS, щоб AWS Config надсилала такі сповіщення, як доставка історії конфігурації, доставка знімків конфігурації та відповідність вимогам.

Обравши потік AWS Config до теми Amazon SNS, виберіть цільову тему:

Створивши топік - У полі Topic Name введіть назву вашої теми SNS.

Обравши топік зі свого облікового запису - У полі Topic Name оберемо бажану тему.

Обравши топік з іншого акаунта - у полі Topic ARN введіть ім'я ресурсу Amazon (ARN) теми. Якщо ви вибираєте топік з іншого акаунта, в ній повинні бути політики, які надають дозволи на доступ до AWS Config.

Після конфігурації маємо систему що аналізує інфраструктуру на реагує у зміни у ний(Рисунок 5.1).

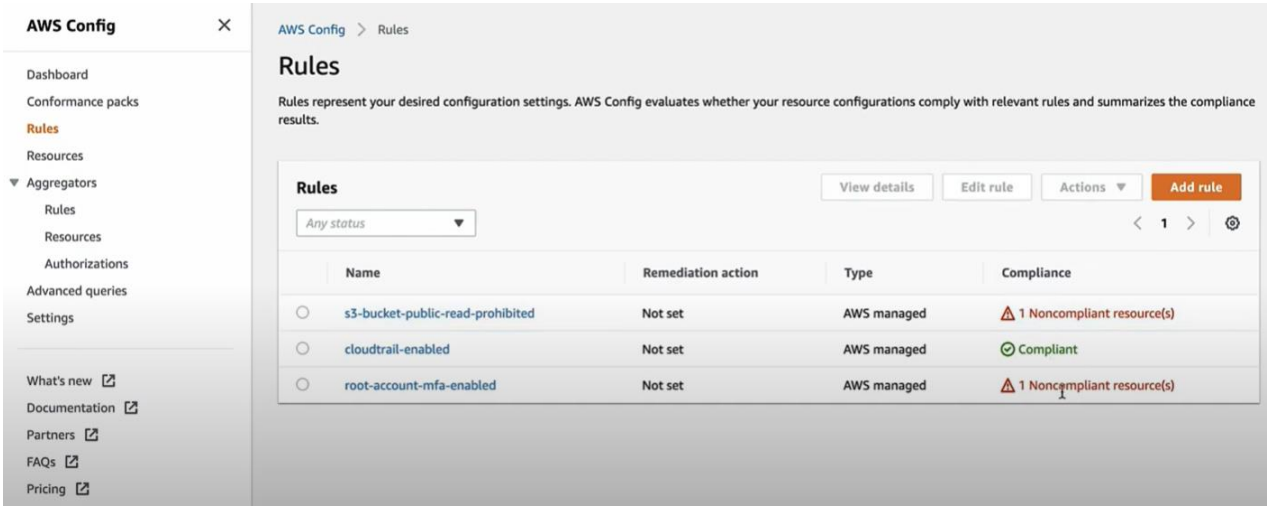


Рисунок 5.1 – Екран метрик AWS Config

ВИСНОВКИ

В даній роботі було вирішено задачу щодо реалізації комплексної системи безпеки для вразливої інфраструктури веб-додатку, використовуючи провідні методи та практики для її побудови.

З цією метою було проведено аналіз поточного стану сучасного веб-додатку. Аналіз показав повну картину розташування веб-додатків як у повсякденному житті користувачів, так і велико системі підприємства різного розміру. Представлені основні переваги і недоліки сучасних веб-додатків. Що стосується термінового збільшення вразливостей у сучасних веб-додатках, особливо що до шкідливих атак, які можуть поставити під загрозу всю програму та її інформацію, провели аналіз цих вразливостей і повністю зрозуміли, як їх уникнути та які небезпеки вони можуть становити.

Грунтуючись на отриманих результатах, було розроблено комплексний підхід до вразливих веб-додатків, використовуючи основні методи впровадження систем безпеки веб-додатків.

Нарешті, процес реєстрації та відстеження був налагоджений, щоб можна було відстежувати не нормальну поведінку програми набагато швидше.

Після впровадження системи безпеки якісна оцінка вже захищеного веб-додатку проводилася знову і знову. Згідно з результатами оцінки, безпека значно підвищилася, а ймовірність використання вразливостей значно знизилася.

Для додаткової перевірки, було використано спеціалізований pin-test додаток, який сканує веб-додаток

СПИСОК ВИКОРАСТИНОЇ ЛІТЕРАТУРИ

1) Мазепа А. Д. Безпечна автентифікація до веб-додатку з використанням JWT та browser fingerprinting / А. Д. Мазепа, А. С. Тарасов. // Харків, ХНУРЕ, Матеріали XXII Міжнародного молодіжного форуму "Радіоелектроніка та молодь у XXI столітті". Том 4. – 2021. – С. 60–61.

2) Мазепа А. Д. Розуміння та захист від атаки DNS rebinding / А. Д. Мазепа, А. С. Тарасов. // Харків, ХНУРЕ, Матеріали XXII Міжнародного молодіжного форуму "Радіоелектроніка та молодь у XXI столітті". Том 4. – 2021. – С. 62–63.

3) Тарасов А. С. Проблеми захисту персональних даних в комп’ютерних системах від шкідливого програмного забезпечення stealer / А. С. Тарасов, А. Д. Мазепа. // Харків, ХНУРЕ, Матеріали XXII Міжнародного молодіжного форуму "Радіоелектроніка та молодь у XXI столітті". Том 4. – 2021. – С. 64–65. 4. Статистика співвідношення кількості веб-додатків до статичних сайтів [Електронний ресурс] – Режим доступу до ресурсу: https://earthweb.com/howmany-websites-are-there/.

5) Статистика атак на компанії за останні три роки [Електронний ресурс] – Режим доступу до ресурсу: https://www.whitehatsec.com/news/new-report-showshalf-of-websites-were-vulnerable-to-exploitation-throughout-2021/.

6) Аналіз причин виникнення вразливостей у веб-додатках [Електронний ресурс] – Режим доступу до ресурсу: https://www.ptsecurity.com/wwen/analytics/web-vulnerabilities-2020/.

7) Визначення поняття зламаного контролю доступу [Електронний ресурс] – Режим доступу до ресурсу: https://owasp.org/Top10/A01_2021-Broken_Access_Control/.

8) Визначення поняття зламаного XSS [Електронний ресурс] – Режим доступу до ресурсу: https://www.whitehatsec.com/glossary/content/cross-site-scripting.

9) Визначення ентропії паролю [Електронний ресурс] – Режим доступу до ресурсу: https://specopssoft.com/blog/password-entropy/.

10) Орієнтир тестування bcrypt [Електронний ресурс] – Режим доступу до ресурсу: https://gist.github.com/epixoip/a83d38f412b4737e99bbef804a270c40

11) Федотов Н.С. Оценка и нейтрализация рисков в информационных системах: метод. пос. / Н.С. Федотов, В.С. Алешин. – М.: МГТУ им. Н.Э.Баумана, 2004. – 52 с.

12) Замула О.А. Аналіз міжнародних стандартів в галузі оцінювання ризиків інформаційної безпеки / О.А. Замула, В.І. Черниш // Системи обробки інформації: зб. наук. пр. – Х.: ХУПС, 2011. – Вип. 2 (92). – С.53-56. 77

13) Попелова И.Г. Применение и развитие современных информационных технологий в системе машиноиспытаний [Текст] /Научно-информационное обеспечение инновационного развития АПК: материалы VII Междунар. науч.-практ. конф. – М.: ФГБНУ «Росинформагротех», 2014 14. НД ТЗІ 2.5-010-03 «Вимоги до захисту інформації WEB-сторінки від несанкціонованого доступу» 15. ISO/IEC 27001:2013. Information technology — Security techniques — Information security management systems — Requirements. [Електрон. ресурс]: – Режим доступу: http://www.itgivernance.co.uk/standards.arx.

16) Гавриленко О.В. Відповідність національної нормативної бази у сфері технічного захисту інформації міжнародним стандартам: зіставлення документів, шляхи гармонізації. Матеріали XVII Міжнародної науково-практичної конференції «Безпека інформації у інформаційно-телекомунікаційних системах», м.Київ, 2015.

17) НД ТЗІ 2.5-004-99 “Критерії оцінки захищеності інформації в комп’ютерних системах від несанкціонованого доступу”. [Електрон. ресурс]: – Режим доступу: http://www.dsszzi.gov.ua/dstszi/control/uk/publish/article ?art_id=40386&cat_id=38835.

18) Закон України “Про захист інформації в інформаційно-телекомунікаційних системах” від 5 липня 1994 року № 80/94-ВР, Відомості Верховної Ради України (ВВР), 1994, № 31. – с.286

19) СОУ Н НБУ 65.1 СУІБ 1.0:2010. Настанова. Методи захисту в банківській діяльності. Система управління інформаційною безпекою. [Електрон. ресурс]: – Режим доступу: http://www.uk.xlibx.com/4yuridicheskie/1354389-1- sou-nbu-651-suib-10-2010-standart-organizacii-ukraini-nastanova-metodi-zahistubankivskiy-diyalnosti-siste.php.

20) НД ТЗІ 3.7-003-05 “Порядок проведення робіт із створення комплексної системи захисту інформації в інформаційно-телекомунікаційній системі”. [Електрон. ресурс]: – Режим доступу: http://www.dsszzi.gov.ua/dstszi/ control/uk/publish/article?art_id=46074&cat_id=38835.

21) Schieler C. Rate-distortion theory forsecrecy systems / C. Schieler, P. Cuff // IEEE Trans. on Inf. Theory. — 2014. — Vol. 66(12). — P.7584-7605. 32. Sahin C.S. General Framework for Evaluating Password Complexity and Strength / C.S. Sahin, R. Lychev, N. Wagner. — 11 p. — Режим доступу в Інтернет: http://arxiv.org/abs/1512.05814

22) Website Security Statistics Report: 2015. — WhiteHat Security, 2015. — 30 p. — Режим дос-тупу в Інтернет: https://info.whitehatsec.com/Website-StatsReport-2015.html

23) Handbook on Ontologies / eds. S. Staab and R. Studer. — International Handbooks on Information Systems. — Berlin: Springer, 2009. — 832 p.

24) Новиков ДА. Теория управления организационными системами / Д.А. Новиков. — М.: Физматлит, 2007. — 584 с.

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

26) HACKMAGEDDON – статистика інформаційної безпеки [Електронний ресурс] – Режим доступу до ресурсу: http://www.hackmageddon.c