УДК 004.056/.57:004.72REST API
DOI: https://doi.org/10.17721/ISTS.2024.8.60-65

Юрій ЩЕБЛАНІН, канд. техн. наук, ст. наук. співроб.
ORCID ID: 0000-0002-3231-6750
e-mail: shcheblanin.yurii@knu.ua
Київський національний університет імені Тараса Шевченка, Київ, Україна

Богдан СИДОРЕНКО, студ.
ORCID ID: 0009-0006-5646-333X
e-mail: sidorenko2002bogdan@gmail.com
Київський національний університет імені Тараса Шевченка, Київ, Україна

Інна МИХАЛЬЧУК, канд. техн. наук,
ORCID ID: 0000-0002-1802-7653
e-mail: inna.mykhalchuk@knu.ua
Київський національний університет імені Тараса Шевченка, Київ, Україна

БЕЗПЕКА REST API: ЗАГРОЗИ ТА МЕТОДИ ЗАХИСТУ

Вступ. *Зростання зловмисної активності в інформаційному просторі створює додаткові виклики для організацій, які використовують REST API в процесі передачі даних та організації взаємодії з клієнтами і партнерами. Згідно зі статистикою, більше 80 % сучасного вебтрафіка проходить через веб-API, що робить його привабливою мішенню для кіберзлочинців. Вразливість у механізмах автентифікації та авторизації REST API може призвести до витоку конфіденційної інформації, фінансових втрат і загроз репутації. Тому забезпечення безпеки REST API є критично важливим завданням для сучасних компаній, особливо тих, що працюють у галузях із високим рівнем ризику.*

Методи. *Застосовано методи аналізу загроз безпеці й оцінювання ризиків, що виникають у процесі використання REST API.*

Результати. *Організації інвестують значні ресурси у розвиток технологій захисту REST API, впроваджують токени для контролю доступу, шифрують передачу даних за допомогою TLS/SSL та інтегрують сучасні засоби захисту в свої додатки. Проте дослідження показує, що основні загрози безпеці все ще залишаються актуальними через недостатній рівень захищеності процесу валідації вхідних даних, слабкі паролі та відсутність багатофакторної автентифікації. Також встановлено, що значна кількість API не мають обмежень на частоту запитів, що робить їх вразливими до атак на виснаження ресурсів (DoS- і DDoS-атаки).*

Висновки. *Одним із ключових напрямів розв'язання проблеми безпеки REST API є впровадження системи управління безпекою API, до якої належить використання багаторівневого підходу до захисту. Це включає контроль доступу, застосування токенів для авторизації, регулярну перевірку систем на наявність вразливостей та обмеження швидкості запитів для зменшення ризику атак на відмову в обслуговуванні. До того ж упровадження сучасних практик безпеки, таких як багатофакторна автентифікація, допоможе мінімізувати ризики несанкціонованого доступу. Результати дослідження можуть бути використані для вдосконалення існуючих політик безпеки REST API й оптимізації підходів до управління загрозами в компаніях різного масштабу.*

Ключові слова: *REST API, інформаційна безпека, автентифікація, авторизація, загрози, інформаційна безпека.*

Вступ

Зростання зловмисної активності в інформаційному та кібернетичному просторах ставить перед керівниками підприємств і власниками компаній нові завдання щодо захисту своїх цифрових активів. Особливо вразливими є системи, що використовують REST API (Representational State Transfer Application Programming Interface) для обміну даними, оскільки вони стають мішенню для атак, спрямованих на отримання несанкціонованого доступу до конфіденційної інформації або порушення нормального функціонування систем.

Компрометація таких активів, як конфіденційні дані клієнтів, фінансова або інтелектуальна власність, може призвести до порушення безперервності бізнесу, фінансових втрат і втрати репутації компанії. У відповідь на ці виклики організації мають упроваджувати нові загрози та вразливості, характерні для архітектури REST.

З огляду на ці ризики підприємства та компанії мають оцінювати рівень зрілості процесів забезпечення безпеки своїх API та впроваджувати відповідні системи захисту, які враховують специфіку їхньої діяльності. Це питання стає критичним для залучення інвестицій, збереження довіри клієнтів і підтримки безперервної діяльності компанії в умовах сучасної кіберзагрози.

Зловмисники можуть використовувати різні методи атак, включаючи SQL-ін'єкції, атаки на автентифікацію та авторизацію, атаки на відмову в обслуговуванні (DoS, DDoS) та перехоплення даних. Ці атаки можуть призвести до несанкціонованого доступу до конфіденційної інформації, втрат фінансових ресурсів або порушення роботи системи.

Для захисту REST API необхідно впроваджувати комплексні заходи безпеки, які включають використання надійних механізмів автентифікації та авторизації, шифрування даних на транспортному рівні, валідацію вхідних даних та обмеження доступу до ресурсів. Відсутність цих заходів може призвести до серйозних наслідків для організації, включаючи втрату даних, фінансові збитки та порушення репутації.

Ця робота спрямована на аналіз сучасних загроз для безпеки REST API та розроблення підходів до підвищення рівня його захисту на основі сучасних технологій безпеки.

Мета статті – дослідити основні загрози безпеці REST API та проаналізувати методи захисту, які здатні забезпечити надійний захист даних, що передаються через ці інтерфейси.

Методи

Використано метод аналізу загроз безпеці й оцінювання ризиків, що виникають під час застосування REST API, а також системний підхід для комплексного дослідження архітектури REST API.

Результати

У сучасному цифровому середовищі REST API є ключовою складовою для забезпечення зв'язку між додатками, особливо у вебсервісах та мікросервісних архітектурах. Оскільки більшість інтернет-трафіка проходить через API, їхня безпека стає критично важливою для захисту даних і забезпечення стабільної роботи вебдодатків.

ISSN 2707-1758

REST API – це архітектурний стиль для створення легких і гнучких вебсервісів, що використовує стандартний протокол HTTP для передачі даних між клієнтом і сервером. REST API надає клієнтам доступ до певних ресурсів на сервері, дозволяючи отримувати або змінювати дані через стандартизовані запити. На відміну від інших архітектурних стилів, REST спирається на принципи простоти, ефективності та незалежності компонентів, що робить його ідеальним вибором для розподілених систем і мобільних додатків, де важлива швидка і стабільна передача даних.

Однією з ключових концепцій REST API є організація ресурсів за унікальними адресами. Кожен ресурс, наприклад, дані про користувача або продукт, доступний через визначений URL. Клієнт звертається до конкретного ресурсу через запит на сервер, який відповідає, обробляючи цей запит відповідно до правил REST. Щоб досягти цього, REST API використовує стандартні HTTP-методи, що описують дії, які клієнт може виконувати над ресурсами. Метод GET застосовують для отримання інформації про ресурси, POST – для створення нових записів, PUT – для оновлення існуючих даних, а DELETE – для видалення ресурсів (рис. 1).

Архітектурний стиль REST також базується на принципі динамічної взаємодії, що означає відсутність збереження стану між запитами клієнта. Це означає, що кожен запит обробляється незалежно від попередніх, що знижує навантаження на сервер, оскільки йому не потрібно зберігати інформацію про стан сеансу користувача. Завдяки цьому REST API добре масштабується і підходить для оброблення великої кількості запитів одночасно, що є важливим аспектом для вебдодатків і хмарних сервісів (Salva et al., 2024; Laptiev et al., 2022).

REST API часто передає дані у форматах JSON або XML (рис. 1), які легко обробляються більшістю мов програмування та забезпечують зручність взаємодії між різними типами клієнтів, такими як веббраузери, мобільні додатки або навіть інші сервери. Це робить REST API універсальним для різних платформ і значно спрощує інтеграцію нових компонентів у систему. Крім того, використання стандартних форматів передачі даних дозволяє швидко обробляти відповіді сервера і зменшує затримки під час передачі інформації (Zahynei et al., 2024; Syrotynskyi et al., 2024).

Щоб краще зрозуміти принцип роботи REST API, доцільно представити процес взаємодії між клієнтом і сервером у вигляді схеми. Клієнт, що може бути веб-додатком або мобільним пристроєм, формує HTTP – запит для отримання або зміни певних даних на сервері. Сервер приймає запит, перевіряє права доступу, виконує валідацію введених даних і, якщо запит коректний, звертається до бази даних або інших ресурсів для виконання необхідної операції. Після цього сервер формує відповідь із результатом і повертає її клієнту. Відповідь може містити дані у форматі JSON або XML, а також статус виконання запиту, що сигналізує клієнту про успішність операції або наявність помилок.

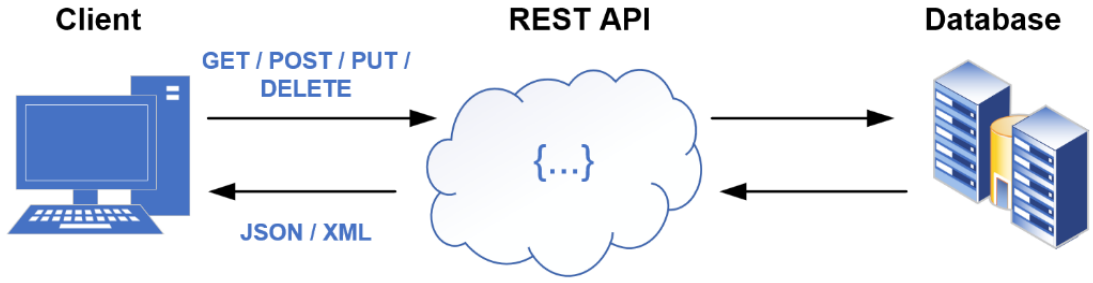

Рис. 1. Схема роботи REST API

Однак REST API схильні до численних загроз, що можуть поставити під загрозу конфіденційність і цілісність даних, а також загальну стабільність систем. Розуміння цих загроз й упровадження ефективних методів захисту є невід'ємною частиною побудови безпечних систем.

Основні загрози для REST API можна поділити на кілька категорій (OWASP API Security Top 10, 2023).

Перша категорія – це загрози пов'язані з недостатньою автентифікацією та авторизацією користувача. Якщо REST API не використовує належні механізми перевірки особи користувача, або процесу, це може дозволити несанкціонованим особам отримати доступ до конфіденційної інформації або навіть здійснювати небажані операції. Наприклад, зловмисники можуть використати слабкі або неправильно налаштовані механізми автентифікації, щоб отримати доступ до захищених ресурсів. Особливо небезпечною є ситуація, коли REST API не належно розмежовує доступ до різних рівнів даних для користувачів із різними ролями або рівнями доступу.

Другою загрозою є атаки типу Denial of Service (DoS, DDoS), які спрямовані на перевантаження серверів API численними запитами, що в кінцевому результаті може призвести до збоїв у роботі або зробити сервіс недоступним для легітимних користувачів. Атаки DoS і DDoS можуть бути надзвичайно руйнівними, оскільки їхня мета – повністю вивести з ладу систему, змушуючи сервер обробляти надмірну кількість запитів або споживати всі доступні ресурси.

Третьою загрозою є SQL-ін'єкції, коли зловмисники вводять шкідливий код у запити, що надсилаються до бази даних через API. Якщо REST API не фільтрує або належно не обробляє вхідні дані, це може призвести до виконання шкідливих команд на рівні бази даних, що відкриває доступ до конфіденційної інформації або дозволяє змінювати дані.

Четвертою загрозою є перехоплення даних. Якщо інформація, яка передається між клієнтом і сервером через REST API, не зашифрована, то зловмисники можуть перехоплювати її. Це особливо небезпечно, коли передають конфіденційні дані, такі як паролі,

особиста інформація або інші дані, які можуть бути використані для викрадення конфіденційної інформації або проведення фінансових махінацій.

До того ж REST API можуть бути вразливими до атак, які експлуатують вразливості програмного забезпечення, наприклад: неправильна обробка вхідних даних, використання застарілих бібліотек або фреймворків, що містять вразливості. Зловмисники можуть скористатися цими недоліками для компрометації системи та виконання шкідливих дій.

Основні методи захисту REST API спрямовано на мінімізацію вказаних загроз і забезпечення безпеки систем.

Одним із ключових методів захисту є використання токенів для автентифікації, зокрема і токенів на основі JSON Web Token (JWT). Цей підхід дає можливість забезпечити перевірку користувача у кожному запиті без необхідності зберігання стану сесії на сервері. Токени JWT включають зашифровану інформацію про користувача, що дозволяє серверу перевіряти права доступу і підтверджувати автентичність кожного запиту. Важливо також упроваджувати механізми оновлення та відкликання токенів, щоб уникнути їх несанкціонованого використання (Shcheblanin et al., 2023, pp. 266–271).

Другим важливим методом є шифрування даних за допомогою TLS/SSL. Це забезпечує захист даних під час передачі між клієнтом і сервером, знижуючи ризик перехоплення конфіденційної інформації зловмисниками. Використання HTTPS для всіх запитів REST API є обов'язковою практикою для забезпечення безпеки. Ще один важливий аспект захисту – валідація та фільтрація вхідних даних. REST API повинні ретельно перевіряти всі вхідні дані, щоб запобігти SQL-ін'єкціям та іншим типам атак на введення. Для цього використовують спеціальні анотації та механізми перевірки даних, що допомагають автоматизувати процес і запобігати поширеним вразливостям.

Обмеження доступу до ресурсів API є ще одним важливим заходом. Необхідно впроваджувати чіткі правила авторизації, що дозволяють обмежити доступ до певних частин API для різних груп користувачів. Це може бути реалізовано за допомогою ролей і привілеїв, що дозволяє контролювати, хто має доступ до певних функцій або даних.

Для захисту від атак типу DoS та DDoS важливо впроваджувати обмеження кількості запитів або використовувати механізми хешування, які дозволяють зменшити навантаження на сервер. Це допомагає знизити ймовірність перевантаження сервера та забезпечити стабільну роботу системи навіть під час великих обсягів запитів.

Отже, забезпечення безпеки REST API є комплексним процесом, який вимагає впровадження різних методів захисту для мінімізації загроз.

Інциденти з компрометацією API. Відомі інциденти, що сталися у великих компаніях, чітко ілюструють небезпеку недостатнього захисту REST API. Вони показують, до яких наслідків може призвести нехтування базовими принципами безпеки (OWASP API Security Project, 2021).

2019 р. через вразливість в API Facebook було викрадено особисті дані близько 50 мільйонів користувачів. Реалізація атаки стала можливою через вразливість у системі автентифікації, яка дозволяла зловмисникам отримати доступ до токенів користувачів і використовувати їх для входу в облікові записи без введення паролю.

Атака на API T-Mobile призвела до витоку даних близько 2 мільйонів користувачів. Ця атака була результатом недостатнього захисту інтерфейсів для сторонніх розробників, що дозволило зловмисникам отримати доступ до конфіденційної інформації користувачів, включаючи номери телефонів та адреси.

Компанія Uber зазнала атаки на свій REST API, що призвело до викрадення даних водіїв і пасажирів. Хакери скористалися помилками в API для отримання доступу до приватних інформаційних ресурсів, оскільки в API не було достатніх обмежень на доступ.

Рекомендації OWASP для підвищення рівня захищеності API. Організація OWASP (Open Web Application Security Project) створила список найпоширеніших загроз для API та рекомендацій щодо захисту від них (табл. 1). OWASP API Security Top 10 є основним орієнтиром для розробників, які прагнуть забезпечити безпеку своїх API.

Розглянемо детальніше захисні заходи для кожної загрози з OWASP API Security Top 10.

Broken Object Level Authorization (Порушення авторизації на рівні об'єктів). Однією з найбільших загроз для API є порушення авторизації на рівні об'єктів. Ця загроза виникає, коли користувачі отримують доступ до об'єктів, на які вони не мають прав. Щоб запобігти цьому, необхідно впроваджувати посилену авторизацію для кожного запиту до API. Кожен користувач повинен мати доступ лише до тих об'єктів, для яких йому надано дозволи. Важливим є використання контролю доступу на основі ролей (RBAC) або атрибутів (ABAC), що дозволяє точніше визначати права доступу. Журналювання всіх запитів до об'єктів допомагає відстежувати можливі порушення та забезпечувати належний рівень захисту (Schmidt, & Meier, 2020).

Broken Authentication (Недостатня автентифікація). Ненадійні або неправильно налаштовані механізми автентифікації можуть дозволити зловмисникам отримати доступ до API, видаючи себе за легітимних користувачів. Для захисту від цього рекомендується впроваджувати багатофакторну автентифікацію (2FA), яка забезпечує додатковий рівень безпеки. Використання токенів на основі JWT або OAuth2 допомагає уникнути проблем із зберіганням сесій і спрощує процес автентифікації. Варто також обмежити кількість спроб входу для запобігання атакам типу "груба сила". Надійне хешування паролів за допомогою алгоритмів є ще одним критично важливим заходом для захисту паролів від підбору (Stallings, 2020; Subhadeep et al., 2024).

Excessive Data Exposure (Надмірна передача даних). Однією з поширених проблем є надмірне розкриття даних через API, коли відповідь містить більше інформації, ніж потрібно клієнту. Для уникнення цього необхідно фільтрувати дані на рівні відповіді, щоб віддавати клієнту лише ту інформацію, яка йому необхідна для виконання запиту. Використання механізмів форматування відповіді на стороні сервера дозволяє контролювати, які дані можуть бути відправлені клієнту, запобігаючи надмірному розкриттю.

Lack of Resources & Rate Limiting (Відсутність обмежень на використання ресурсів). Відсутність обмежень на кількість запитів може призвести до атак на виснаження ресурсів сервера, таких як DoS- та DDoS-атаки. Для захисту від цього рекомендують упроваджувати обмеження швидкості запитів (Rate Limiting), що дозволить обмежити кількість запитів від одного клієнта за одиницю часу. Крім того,

ISSN 2707-1758

використання хешування може допомогти зменшити навантаження на сервер, особливо для повторюваних запитів. Важливо також упроваджувати моніторинг аномальної активності, щоб своєчасно виявляти підозрілу поведінку і вживати відповідних заходів.

Таблиця 1

Топ-10 загроз OWASP для безпеки API

Номер	Загроза	Опис
API 1	Broken Object Level Authorization	Недостатній контроль доступу до об'єктів, що дозволяє зловмисникам отримати доступ до чужих ресурсів
API 2	Broken Authentication	Недостатньо захищені або неправильно налаштовані механізми автентифікації можуть дозволити компрометацію
API 3	Excessive Data Exposure	API може віддавати занадто багато інформації через недостатнє обмеження на рівні відповіді
API 4	Lack of Resources & Rate Limiting	Відсутність обмежень на кількість запитів може призвести до атак на виснаження ресурсів або DoS- та DDoS-атак
API 5	Broken Function Level Authorization	Недостатній контроль доступу до певних функцій API може дозволити користувачам виконувати заборонені дії
API 6	Mass Assignment	Зловмисники можуть змінювати атрибути об'єктів, які не повинні бути доступні через API
API 7	Security Misconfiguration	Неправильна конфігурація безпеки API, включаючи відсутність шифрування або слабкі налаштування доступу
API 8	Injection	Вразливості до ін'єкцій дозволяють упроваджувати шкідливий код через вхідні дані
API 9	Improper Assets Management	Недостатнє управління ресурсами API може призвести до розкриття конфіденційної інформації
API 10	Insufficient Logging & Monitoring	Відсутність належного журналювання та моніторингу ускладнює виявлення атак або реагування на них

Broken Function Level Authorization (Недостатня авторизація на рівні функцій). Недостатня авторизація на рівні функцій може дозволити користувачам отримувати доступ до функцій, які для них не призначені. Щоб запобігти цьому, потрібно впроваджувати строгий контроль доступу до кожної функції. Використання правил авторизації, заснованих на ролях користувачів, дозволяє ефективно розмежовувати доступ до різних функцій API. Також важливо перевіряти права доступу до кожної функції перед її виконанням (Rzaieva et al., 2024; Sobchuk, Zelenska, & Laptiev, 2023).

Mass Assignment (Масове призначення). Масове призначення відбувається, коли API дозволяє клієнтам передавати більше даних, ніж потрібно, і в такий спосіб змінювати критичні атрибути. Щоб цього уникнути, необхідно явно визначати поля, які можуть бути змінені через API-запити. Крім того, важливо перевіряти дозволи на зміну кожного окремого поля, щоб уникнути несанкціонованих змін важливих даних.

Security Misconfiguration (Неправильна конфігурація безпеки). Неправильна конфігурація безпеки API, така як відсутність шифрування або використання застарілих бібліотек, може стати серйозною загрозою. Для уникнення цього важливо регулярно оновлювати компоненти API та використовувати актуальні версії бібліотек із виправленими вразливостями. Стандартизація конфігурацій безпеки допомагає уникнути помилок у налаштуваннях. Проводьте періодичні аудити безпеки, щоб виявляти можливі слабкі місця та вразливості (Rzaieva et al., 2024, pp. 27–38).

Injection (Ін'єкція). Атаки ін'єкції, наприклад SQL-ін'єкції, виникають, коли шкідливий код впроваджується у запити API. Щоб запобігти цьому, слід використовувати параметризовані запити або ORM (Object-Relational Mapping) для безпечної взаємодії з базами даних. Фільтрація і екранування вхідних даних також допомагають запобігти впровадженню небезпечних символів у запити.

Improper Assets Management (Неправильне керування активами). Недостатнє керування ресурсами API, такими як застарілі версії API, може створити додаткові вразливості. Для уникнення цього необхідно регулярно проводити інвентаризацію API і вимикати застарілі версії, які більше не використовуються. Важливо переконатися, що доступ до застарілих API обмежений або повністю вимкнений (Atlidakis, Godefroid, & Polishchuk, 2019).

Insufficient Logging & Monitoring (Недостатнє журналювання та моніторинг). Відсутність належного журналювання та моніторингу може ускладнити виявлення атак або реагування на них. Щоб уникнути цього, рекомендують упроваджувати повне журналювання критичних подій, таких як спроби автентифікації або доступ до конфіденційних ресурсів. Налаштування сповіщень дозволяє оперативно реагувати на підозрілі дії або порушення безпеки. Регулярний аналіз журналів подій допомагає вчасно виявляти підозрілу активність і запобігати можливим атакам (Barabash et al., 2023, pp. 177–192).

Отже, упровадження захисних заходів, рекомендованих OWASP, дозволить суттєво знизити ризик компрометації API і підвищити загальний рівень безпеки додатків. Їх дотримання забезпечить захист від найпоширеніших загроз та атак на рівні API, що критично важливо для сучасних вебдодатків і мікросервісних архітектур.

Дискусія і висновки

Безпека REST API є однією з найважливіших складових розроблень сучасних вебдодатків і мікросервісних

архітектур. Використання API для передачі даних між різними сервісами та системами значно підвищує ефективність розроблення, але також відкриває нові вектори для потенційних атак. В роботі розглянуто найпоширеніші загрози безпеці API, визначені в OWASP API Security Top 10, і запропоновано заходи для їхньої мінімізації.

Ключовим аспектом захисту API є впровадження багаторівневої системи безпеки, яка включає не лише базові механізми шифрування й автентифікації, але й контроль доступу на рівні об'єктів і функцій, захист від ін'єкцій, а також моніторинг активності. Одним із найважливіших кроків для забезпечення безпеки є впровадження посиленої авторизації, що дозволяє запобігати доступу до даних користувачів, яким ці дані не призначені. Це також включає використання механізмів багатофакторної автентифікації (2FA) та токенів, таких як JWT, для забезпечення надійного процесу автентифікації.

Досвід великих компаній, таких як Facebook, Uber та T-Mobile, демонструє серйозність наслідків, до яких може призвести нехтування належними заходами безпеки. Компрометація API у цих випадках призвела до витікання конфіденційної інформації мільйонів користувачів, що підкреслює важливість дотримання найкращих практик захисту API. Сучасні реалії вимагають від розробників не тільки впровадження традиційних методів безпеки, але й застосування таких інструментів, як автоматизоване тестування безпеки в межах підходу DevSecOps.

Згідно з рекомендаціями OWASP, основними захисними заходами є обмеження кількості запитів (Rate Limiting), належне журналювання подій, шифрування трафіка та використання сучасних методів захисту від ін'єкцій, таких як параметризовані запити. Регулярні аудити безпеки й оновлення компонентів системи також відіграють ключову роль у підтриманні надійної безпеки.

Зазначимо, що комплексне впровадження захисних заходів, визначених OWASP, допоможе значно знизити ризик компрометації REST API. Дотримання цих рекомендацій гарантуватиме, що система буде захищена від найпоширеніших загроз, а її функціональність залишатиметься стабільною та безпечною. Зростання популярності REST API вимагає від розробників постійного вдосконалення підходів до безпеки, впровадження автоматизованих рішень і готовності швидко реагувати на нові виклики у сфері кібербезпеки.

Внесок авторів: Богдан Сидоренко – концептуалізація, методологія, аналіз джерел, підготування огляду літератури, розробка захисних заходів для REST API; Юрій Щебланін – збір емпіричних даних, їхня валідація, участь у розробці методології дослідження; Інна Михальчук – огляд літератури, редагування рукопису, участь у розробці методології дослідження.

Список використаних джерел

Atlidakis, V., Godefroid, P. & Polishchuk M (2019). RESTler: Stateful REST API Fuzzing. *41st ACM/IEEE International Conference on Software Engineering (ICSE'2019)*. Montreal, QC, Canada. https://ieeexplore.ieee.org/document/8811961

Barabash, O., Sobchuk, V., Musienko, A., Laptiev, O., Bohomia, V., & Kopytko, S. (2023). *System Analysis and Method of Ensuring Functional Sustainability of the Information System of a Critical Infrastructure Object* (pp. 177–192). https://doi.org/10.1007/978-3-031-37450-0_11

Laptiev, O.,Sobchuk, V., Subach, I., Barabash, A. & Salanda, I. (2022). The Method of Detecting Radio Signals Using the Approximation of Spectral Function. *CEUR Workshop Proceedings*, 3384, 52–61.

OWASP API Security Project (2021). OWASP Foundation. Retrieved from: https://owasp.org/www-project-api-security/.

OWASP API Security Top 10. (2023). OWASP Foundation. https://owasp.org/www-project-api-security/.

Rzaieva, S., Rzaiev D., Kostyuk Y., Hulak H., & Shcheblanin O. (2024). *Methods of Modeling Database System Security* (short paper). CPITS 2024: 384–390.

Schmidt, T., & Meier, M. (2020). Secure API Design and Development: Practices for Building Robust APIs. *API Security Journal*, 5(2), 43–57.

Shcheblanin, Y., Oliinyk, B., Kurchenko, O., Toroshanko O., Korshun, & N. (2023). Research of Authentication Methods in Mobile Applications. *CPITS-2023*, 3421, 266–271.

Sobchuk, V., Zelenska, I., & Laptiev, O. (2023). Algorithm for solution of systems of singularly perturbed differential equations with a differential turning point. *Bulletin of the Polish Academy of Sciences Technical Sciences*, 71(3), Article number: e145682. https://doi.org/10.24425/bpasts.2023.145682 WoS.

Stallings, W. (2020*). Cryptography and Network Security. Principles and Practice.* Pearson Education.

Subhadeep C., Sainath C., Pinnarwar S., & Sandosh S. (2024). Real-Time Threat Detection and Mitigation in Web API Development. *International Conference on Electrical Electronics and Computing Technologies (ICEECT)*, 1 (pp. 1–9). Greater Noida, India.

Zahynei A., Shcheblanin Y, Kurchenko O., Anosov A., & Kruglyk V. (2024). *Method for Calculating the Residual Resource of Fog Node Elements of Distributed Information Systems of Critical Infrastructure Facilities (short paper).* CPITS 2024: p. 432-439.

References

Atlidakis, V., Godefroid, P. & Polishchuk M (2019). RESTler: Stateful REST API Fuzzing. *41st ACM/IEEE International Conference on Software Engineering (ICSE'2019)*. Montreal, QC, Canada. https://ieeexplore.ieee.org/document/8811961

Barabash, O., Sobchuk, V., Musienko, A., Laptiev, O., Bohomia, V., & Kopytko, S. (2023). *System Analysis and Method of Ensuring Functional Sustainability of the Information System of a Critical Infrastructure Object* (pp. 177–192) https://doi.org/10.1007/978-3-031-37450-0_11

Laptiev, O.,Sobchuk, V., Subach, I., Barabash, A. & Salanda, I. (2022). The Method of Detecting Radio Signals Using the Approximation of Spectral Function. *CEUR Workshop Proceedings*, 3384, 52–61.

OWASP API Security Project (2021). OWASP Foundation. Retrieved from: https://owasp.org/www-project-api-security/.

OWASP API Security Top 10. (2023). OWASP Foundation. https://owasp.org/www-project-api-security/.

Rzaieva, S., Rzaiev D., Kostyuk Y., Hulak H., & Shcheblanin O. (2024). *Methods of Modeling Database System Security* (short paper). CPITS 2024: 384–390.

Schmidt, T., & Meier, M. (2020). Secure API Design and Development: Practices for Building Robust APIs. *API Security Journal*, 5(2), 43–57.

Shcheblanin, Y., Oliinyk, B., Kurchenko, O., Toroshanko O., Korshun, & N. (2023). Research of Authentication Methods in Mobile Applications. *CPITS-2023*, 3421, 266–271.

Sobchuk, V., Zelenska, I., & Laptiev, O. (2023). Algorithm for solution of systems of singularly perturbed differential equations with a differential turning point. *Bulletin of the Polish Academy of Sciences Technical Sciences*, 71(3), Article number: e145682. https://doi.org/10.24425/bpasts.2023.145682 WoS.

Stallings, W. (2020*). Cryptography and Network Security. Principles and Practice.* Pearson Education.

Subhadeep C., Sainath C., Pinnarwar S., & Sandosh S. (2024). Real-Time Threat Detection and Mitigation in Web API Development. *International Conference on Electrical Electronics and Computing Technologies (ICEECT)*, 1 (pp. 1–9). Greater Noida, India.

Zahynei A., Shcheblanin Y, Kurchenko O., Anosov A., & Kruglyk V. (2024). *Method for Calculating the Residual Resource of Fog Node Elements of Distributed Information Systems of Critical Infrastructure Facilities (short paper).* CPITS 2024: p. 432-439.

Отримано редакцією журналу / Received: 05.11.24
Прорецензовано / Revised : 27.11.24
Схвалено до друку / Accepted : 01.12.24

ISSN 2707-1758

Yurii SHCHEBLANIN, PhD (Engin.), Senior Researcher
ORCID ID: 0000-0002-3231-6750
e-mail: shcheblanin.yurii@knu.ua
Taras Shevchenko National University of Kyiv, Kyiv, Ukraine

Bohdan SYDORENKO, Student
ORCID ID: 0009-0006-5646-333X
e-mail: sidorenko2002bogdan@gmail.com
Taras Shevchenko National University of Kyiv, Kyiv, Ukraine

Inna MYKHALCHUK, PhD (Engin.), Assist.
ORCID ID: 0000-0002-1802-7653
e-mail: inna.mykhalchuk@knu.ua
Taras Shevchenko National University of Kyiv, Kyiv, Ukraine

SECURITY OF REST API: THREATS AND PROTECTION METHODS

Background. ***The increase in malicious activity in the information space creates additional challenges for organizations that use REST APIs to transfer data and facilitate interactions with clients and partners. According to statistics, over 80% of modern web traffic goes through web APIs, making them an attractive target for cybercriminals. Vulnerabilities in REST API authentication and authorization mechanisms can lead to data breaches, financial losses, and reputational risks. Therefore, ensuring REST API security is a critical task for modern companies, especially those operating in high-risk industries.***

Methods. ***Threat analysis and risk assessment methods were used to evaluate the security challenges associated with REST APIs.***

Results. ***Organizations are investing significant resources in the development of REST API security technologies, implementing tokens for access control, encrypting data transmission via TLS/SSL, and integrating modern security measures into their applications. However, research shows that major security threats remain relevant due to insufficient input validation processes, weak passwords, and the lack of multi-factor authentication. It was also found that a significant number of APIs lack rate limiting, making them vulnerable to resource exhaustion attacks (DoS/DDoS attacks).***

Conclusions. ***One of the key approaches to addressing REST API security issues is the implementation of an API security management system that uses a multi-layered approach to protection. This includes access control, token-based authorization, regular system vulnerability checks, and rate limiting to reduce the risk of denial-of-service attacks. In addition, implementing modern security practices, such as multi-factor authentication, will help minimize the risk of unauthorized access. The research findings can be used to improve existing REST API security policies and optimize threat management approaches in companies of various sizes.***

Keywords: ***REST API, information security, authentication, authorization, threats, cybersecurity.***

Автори заявляють про відсутність конфлікту інтересів. Спонсори не брали участі в розробленні дослідження; у зборі, аналізі чи інтерпретації даних; у написанні рукопису; в рішенні про публікацію результатів.

The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses or interpretation of data; in the writing of the manuscript; in the decision to publish the results.

ISSN 2707-1758