Література

1. Konieczna I. Artificial intelligence in transport – will chat GPT replace the forwarding department [Електронний ресурс] / Iwona Konieczna // Prilo. – 2023. – Режим доступу до ресурсу: https://prilo.com/artificial-intelligence-in-transport-will-chat-gpt-replace-the-forwarding- department/.
2. Карпішен Б. С. АНАЛІЗ РОЗРОБКИ І ВИКОРИСТАННЯ СИСТЕМИ ADAS В АВТОМОБІЛІ / Карпішен.Б.С. // Комп'ютерні технології і мехатроніка. Збірник наукових праць за матеріалами III міжнародної науково-методичної конференції. – Харків, ХНАДУ, 2022.. – 2022. – №3. – С. 29–33.
3. Conde, Maria Lopez, and Ian Twinn. "How Artificial Intelligence Is Making Transport Safer, Cleaner, More Reliable and Efficient in Emerging Markets." (2019): n. pag. Web.
4. Badalian V. AI has a finger on the pulse of the transport industry [Електронний ресурс] / Vartan Badalian // GreenBiz Group. – 2023. – Режим доступу до ресурсу: https://www.greenbiz.com/article/ai-has-finger-pulse-transport-industry.
5. W. Huang, G. Song, H. Hong and K. Xie, "Deep Architecture for Traffic Flow Prediction: Deep Belief Networks With Multitask Learning," in IEEE Transactions on Intelligent Transportation Systems, vol. 15, no. 5, pp. 2191-2201, Oct. 2014, doi: 10.1109/TITS.2014.2311123.
6. Junfang Cao, "Mathematical Model and Algorithm of Multi-Index Transportation Problem in the Background of Artificial Intelligence", Journal of Advanced Transportation, vol. 2022, Article ID 3664105, 11 pages, 2022. https://doi.org/10.1155/2022/3664105

JWT АВТЕНТИФІКАЦІЯ ДЛЯ ЗАХИСТУ КЛІЄНТ-СЕРВЕРНОЇ АРХІТЕКТУРИ

Голдчевський М.Р., студент МК 61-23
Науковий керівник – ***Костікова М.В.,*** доц, к.т.н.,
Харківський національний автомобільно-дорожній університет

З розвитком цифрових технологій та збільшенням кількості онлайн-сервісів, питання захисту даних стає дедалі актуальнішим. Одним із наважливіших аспектів безпеки є захист клієнт-серверних архітектур від можливих атак та несанкціонованого доступу до інформації. Ця архітектура використовується у різних додатках, від веб-сайтів і мобільних застосунків до хмарних рішень і мікросервісних платформ.

Основною проблемою безпеки в клієнт-серверних архітектурах є автентифікація користувачів та захист передачі чутливих даних між клієнтом і

сервером. Розробники шукають надійні методи захисту, серед яких популярним є використання JWT (JSON Web Tokens) для автентифікації. Проте, окрім JWT, існують й інші методи та стратегії для забезпечення безпеки клієнт-серверних систем. [1]

1. Використання HTTPS (SSL/TLS): HTTPS (Hypertext Transfer Protocol Secure) забезпечує захищене з'єднання між клієнтом і сервером шляхом шифрування всієї передачі даних. Це забезпечує конфіденційність даних, переданих через мережу, і запобігає їх перехопленню та зміні з боку недобросовісних користувачів.

2. Міждоменна автентифікація (Cross-Origin Resource Sharing - CORS): Механізм, який контролює, як веб-сайти можуть звертатися до ресурсів на інших доменах, що допомагає уникнути атак на основі міждоменної поділки.

3. Політика безпеки веб-додатків (Web Application Security Policy): Встановлення правил та обмежень для веб-додатків, які дозволяють уникнути вразливостей на стороні клієнта, таких як XSS (Cross-Site Scripting) атаки.

4. Механізми обробки паролів (Password Hashing): Використання солей та хеш-функцій для захисту паролів користувачів в базі даних від зловмисних атак.

5. Двофакторна автентифікація (2FA): Введення додаткового рівня автентифікації, наприклад, через SMS-повідомлення або мобільний додаток, для підвищення безпеки входу в систему.

JWT автентифікація в сучасному інтернет-середовищі відіграє ключову роль у забезпеченні безпеки клієнт-серверних архітектур. В порівнянні з іншими методами JWT має свої переваги.

JWT – це стандарт створення токенів доступу, який використовується для автентифікації та передачі даних між двома сторонами. Токен заснований на форматі JSON та підписується або шифрується певним алгоритмом. JWT складається з трьох частин: заголовка, тіла та підпису. Завдяки своїй простоті та ефективності, він здобув популярність як механізм автентифікації в веб-розробці [2].

JWT є компактним і безпечним для використання у веб-запитах, що передаються між сторонами у форматі JSON. Він застосовується для надання доступу до різних ресурсів і сервісів [3].

Авторизація за допомогою JWT – це безстанова система автентифікації та авторизації, що усуває потребу в сеансах або cookies. JWT використовує цифровий підпис, який створюється за допомогою секретного ключа, відомого тільки серверу. Це забезпечує захист від підробок і гарантує, що дані в токені залишаються незмінними під час передачі [4].

Авторизація JWT працює шляхом кодування інформації у веб-токен JSON (JWT), який потім передається між клієнтом і сервером. Етапи типового потоку авторизації JWT такі:

Автентифікація: клієнт надсилає облікові дані користувача на сервер, який автентифікує користувача та генерує JWT, що містить інформацію про користувача.

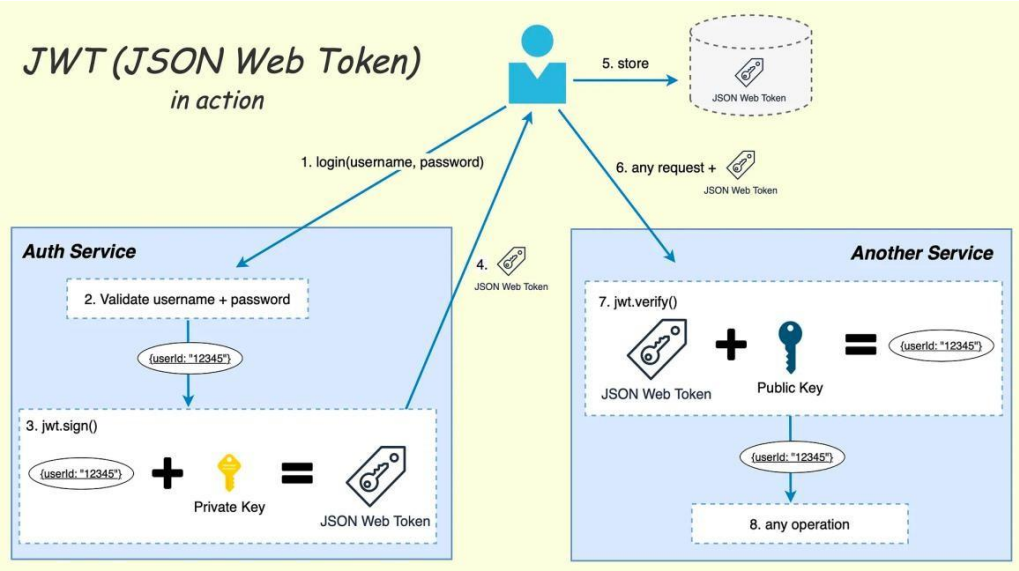

Рисунок 1. Схематичне представлення роботи JWT

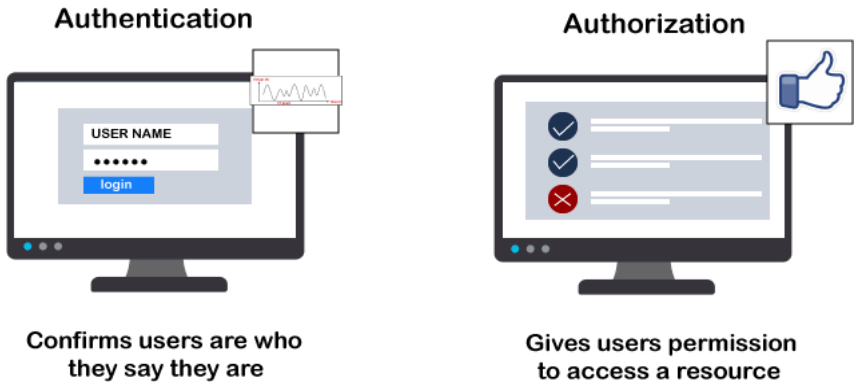

Рисунок 2. Зображення різниці між авторизацією та автинтифікацією.

Видача маркера: сервер надсилає JWT назад клієнту, який зберігає його для майбутнього використання.

Надсилання маркера: коли клієнт хоче отримати доступ до захищеного ресурсу на сервері, він надсилає JWT у заголовку авторизації запиту HTTP.

Перевірка маркера: сервер отримує запит і перевіряє JWT, перевіряючи його підпис за допомогою секретного ключа, який використовувався для його підписання. Якщо JWT дійсний, сервер витягує інформацію, що міститься в ньому, і використовує її для визначення дій, які користувач має право виконувати. Авторизація запиту: якщо користувач авторизований для доступу до ресурсу, сервер повертає запитувані дані. Якщо користувач не авторизований, сервер повертає повідомлення про помилку.

Авторизація за допомогою JWT забезпечує надійний і ефективний обмін даними між клієнтом і сервером, оскільки серверу не потрібно зберігати інформацію про сесію для відстеження статусу автентифікації користувача. Це робить JWT чудовим вибором для мікросервісних архітектур та децентралізмваних систем, де різні компоненти повинні взаємодіяти між собою безпечно.

Порівняно з двофакторною автентифікацією, JWT спрощує інтеграцію для кінцевих користувачів, водночас забезпечуючи високий рівень безпеки. Крім того, JWT знижує навантаження на сервер, оскільки перевірка автентифікаційних даних може відбуватися на стороні клієнта.

У випадку з механізмами обробки паролів, JWT не вимагає зберігання паролів на сервері, що робить його менш вразливим до атак, пов'язаних з витоком паролів чи зламом даних.

Щодо політик безпеки веб-додатків, JWT дозволяє легко впроваджувати стратегії автентифікації та авторизації, використовуючи вбудовані механізми перевірки підписів і токенів.

Міждоменна автентифікація за допомогою JWT може бути реалізована через вбудовані можливості токенів, що гарантує безпечний обмін даними між різними доменами.

На завершення, застосування протоколу HTTPS (SSL/TLS) разом із JWT автентифікацією забезпечує повний рівень шифрування та захист передачі даних між клієнтом і сервером, що гарантує максимальний рівень безпеки.

Використання методу захисту клієнт-серверної архітектури з JWT автентифікацією має кілька переваг, які роблять його кращим вибором порівняно з іншими методами [5]:

1. Простота реалізації. JWT - це стандарт, який досить легко реалізувати в додатках. Більшість сучасних бібліотек і фреймворків надають зручні інструменти для створення та перевірки JWT токенів, що спрощує розробку.

2. Масштабованість. JWT дозволяє легко масштабувати систему, оскільки токени можуть бути валідними на будь-якому сервері або сервісі, який має ключ для їх перевірки. Це дозволяє побудувати розподілені системи без зберігання стану автентифікації на сервері.

3. Зменшення навантаження на сервер. У порівнянні з іншими методами, такими як сеанси або токени, збережені на сервері, JWT переносить навантаження автентифікації на клієнта. Це дозволяє зменшити обсяг даних, які потрібно зберігати та обробляти на сервері.

4. Підтримка для безпеки. JWT можуть бути підписані або зашифровані, що робить їх безпечними для передачі через ненадійні мережі, такі як Інтернет. Підпис дозволяє перевірити цілісність токену, а шифрування забезпечує конфіденційність даних.

5. Універсальність. JWT підтримується багатьма мовами програмування і середовищами, що дозволяє легко інтегрувати їх у різноманітні додатки та сервіси.

6. Можливість розширення. JWT можуть містити будь-яку корисну інформацію у форматі JSON, що робить їх гнучкими і можливими до розширення для вирішення різних завдань автентифікації та авторизації.

JWT автентифікація є ключовим елементом забезпечення безпеки в сучасних клієнт-серверних системах. Вона пропонує ефективний спосіб безпечної автентифікації та авторизації користувачів, дозволяючи обмінюватися компакт-

ними, підписаними токенами між клієнтом і сервером. Використання JWT у веб-розробці забезпечує захист даних та їхню конфіденційність, водночас спрощуючи процеси автентифікації й авторизації. Завдяки цьому методу можна підтримувати мікросервісну архітектуру, де кілька незалежних компонентів безпечно взаємодіють між собою. JWT також підвищує продуктивність, оскільки дозволяє здійснювати швидкий та надійний обмін інформацією між клієнтом і сервером без необхідності зберігати дані про сесію на сервері.

Отже, використання JWT автентифікації є невід'ємною частиною розробки сучасних і безпечних веб-додатків, що дозволяє забезпечити захист важливої інформації та підвищити рівень довіри користувачів.

Література

1. Alkhulaifi A, El-Alfy ESM. Exploring Lattice-based Post-Quantum Signature for JWT Authentication: Review and Case Study. IEEE Xplore. 2020. p.1–5. [Online]: https://ieeexplore.ieee.org/ abstract/document/9129505 https://doi.org/ 10.1109/ VTC2020- Spring48590.2020.9129505

2. Dalimunthe S, Reza J, Marzuki A. Model for Storing Tokens in Local Storage (Cookies) Using JSON Web Token (JWT) with HMAC (Hash-based Message Authentication Code) in E- Learning Systems. Journal of Applied Engineering and Technological Science (JAETS). 2022 Jun 30;3(2):149–55. [Online]: https://www.yrpipku.com/journal/index.php/jaets/article/view/662 https://doi.org/ 10.37385/jaets.v3i2.662

3. Prabath Siriwardena. JWT, JWS, and JWE. Apress eBooks. 2014 Jan 1;201–20. [Online]: https://link.springer.com/chapter/10.1007/978-1-4302-6817-8_13 https://doi.org/10.1007/978-1- 4302-6817-8_13.

4. Bucko A, Vishi K, Krasniqi B, Rexha B. Enhancing JWT Authentication and Authori- zation in Web Applications Based on User Behavior History. Computers, 2023 Apr 1;12(4):78. [Online]: https://www.mdpi.com/2073-431X/12/4/78 https://doi.org/10.3390/computers 12040078

5. Lopez J, Oppliger R, Pernul G. Authentication and authorization infrastructures (AAIs): a comparative survey. Computers & Security. 2004 Oct;23(7): 578–90. [Online]: https://linkinghub. elsevier.com/retrieve/pii/S0167404804001828 https://doi.org/10.1016/j.cose.2004.06.013