УДК 681.12

І. В. Богач[1]
В. С. Збитківський[1]

ОГЛЯД ШИФРУВАННЯ ІНФОРМАЦІЇ КОРИСТУВАЧА СТРИМІНГОВОГО ВІДЕО СЕРВІСУ

[1] Вінницький національний технічний університет;

Анотація

Зроблено огляд способів та інструментів шифрування інформації та пов'язаних метаданих користувача відео сервісу для її безпечної передачі та збереження в базі даних.

Ключові слова: JWT, веб-додатки, Angular, NodeJS, MongoDB, ExpresJS, BCrypt, програмування веб-додатків.

Abstract

An overview of ways and tools to encrypt information and related metadata of the video service's user for its secure transmission and storage in the database.

Keywords: JWT, web-applications, Angular, NodeJS, MongoDB, ExpresJS, BCrypt, web-applications programming.

Вступ

Сьогодні шифрування даних є невід'ємною частиною будь-якого веб застосунку, що має справу з конфенденційною інформацією користувача. Часто для цього використовується хешування. По суті після хешування практично неможливо виявити, як виглядали вихідні дані. Алгоритм хешування BCrypt спочатку солить фрагмент тексту, а потім хешує його до рядка довжиною 60 символів. Бібліотека BCryptJS пропонує метод matches, який перевіряє, чи відповідає рядок хешу. Наприклад, пароль p@55w0Rd, хешований за допомогою BCrypt, може мати значення:

$2b$10$Qrc6rGzIGaHpbgPM5kVXdeNZ9NiyRWC69Wk/17mttHKnDR2lW49KS.

При виклику методу matches BCrypt для незашифрованого та хешованого пароля ми отримаємо значення true.

Результати дослідження

Наразі ні одна система не може гарантувати поний захист від злому. Хоча все одно завжди варто змінювати паролі після зломів даних, хешування паролів надзвичайно ускладнює пошук реального пароля користувача, оскільки воно є одностороннім алгоритмом. Фактично, можуть знадобитися роки, щоб зламати складний хешований пароль належним чином. Це дає додатковий рівень захисту від крадіжки паролів [1].

Токени API - це також облікові дані. Вони так само важливі, як паролі та токени скидання паролів. Практично всі розробники знають це і намагаються дуже надійно зберігати свої ключі AWS, коди доступу до Twitter та інші подібні речі, проте до програм, які вони пишуть, це часто не стосується. Тому варто скористатись системою JSON Web Tokens (JWT) для створення облікових даних доступу до API. Застосування токенів без стану, які можна додавати до чорних списків і потрібно запитувати, це краще, ніж старий шаблон API key/secret, який використовувався останніми роками.

Система використовує JWT для перевірки аутентифікації користувача таким чином:

1. Спочатку користувач заходить на сервер аутентифікації за допомогою аутентифікаційного ключа (це може бути пара логін/пароль, або Facebook ключ, або Google ключ, або ключ від іншого обліку).
2. Потім сервер автентифікації створює JWT та відправляє його користувачеві.
3. Коли користувач робить запит до програми API, він додає до нього отриманий раніше JWT.
4. Коли користувач робить API запит, програма може перевірити за переданим із запитом JWT чи

є користувач тим, за кого себе видає. У цій схемі сервер програми налаштований так, що зможе перевірити, чи є вхідний JWT саме тим, що був створений сервером аутентифікації (процес перевірки буде пояснений пізніше більш детально).

Схема роботи з JSON Web Tokens зображена на рис. 1.

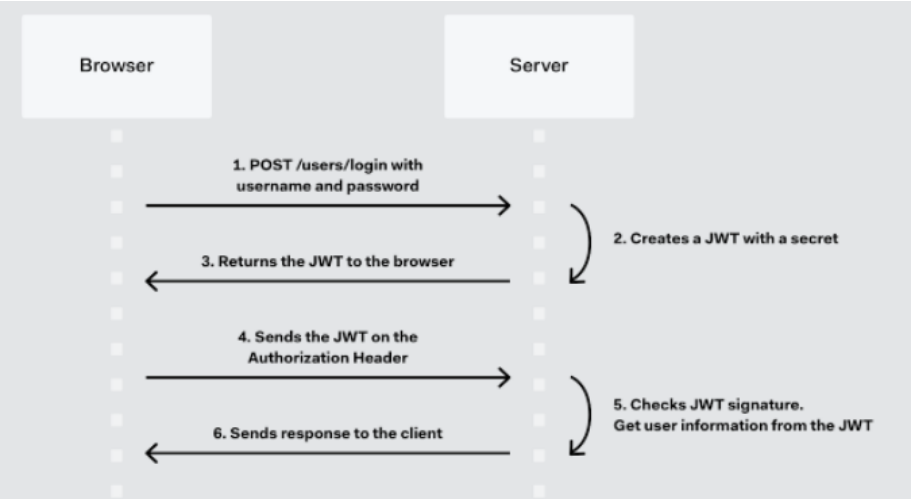

Рис. 1. Схема роботи JSON Web Tokens

Дуже важливо розуміти, що використання JWT не приховує та не маскує дані автоматично. Причина, чому JWT використовуються - це перевірка, що надіслані дані були дійсно відправлені авторизованим джерелом. Мета кодування даних – перетворення структури. Підписані дані дозволяють одержувачу даних перевірити аутентифікацію джерела даних.

Таким чином, закодування та підпис даних не захищає їх. З іншого боку, головною метою шифрування є захист даних від неавторизованого доступу. Оскільки JWT лише закодована і підписана, і оскільки JWT не зашифрована, JWT не гарантує жодної безпеки для чутливих (sensitive) даних [2].

Хеш функція - (згортка) функція однозначного відображення рядка на кінцеве безліч (рядок заданої довжини). Саме число, хеш – результат обчислення хеш-функції над даними. Існують криптографічні та некриптографічні (класифікуються окремо, до них відносяться, наприклад, контрольні суми) хеш-функції. Для криптографічних хешів є три додаткові умови, які відрізняють їх від усіх інших:

1. Необоротність: для заданого значення хеш-функції m має бути обчислювально неможливо знайти блок даних X, для якого H(X)=m.
2. Стійкість до колізій першого роду: для заданого повідомлення M має бути обчислювально неможливо підібрати інше повідомлення N, для якого H(N)=H(M).
3. Стійкість до колізій другого роду: повинно бути обчислювально неможливо підібрати пару повідомлень ~(M, M'), що мають однаковий хеш.

Пароль таблиці користувачів зазвичай зашифровується з використанням незворотного алгоритму, такого як MD5, а потім зберігається. Щоб запобігти злому райдужної таблиці, для шифрування використовується певний рядок (наприклад, ім'я домену), а потім випадкова сіль (значення солі) використовується для шифрування. Конкретний рядок фіксується в програмному коді, а сіль є окремим випадковим чином кожного пароля. Як правило, складно додати поле в таблицю для окремого зберігання. Алгоритм BCrypt рандомізує сіль та змішує її з остаточним зашифрованим паролем, і немає потреби надавати попередню сіль окремо під час перевірки, тому немає потреби окремо вирішувати проблему солі [3].

Висновки

Встановлено, що шифрування інформації користувача є невід’ємною частиною системи і дозволяє гарантувати певний відсоток безпеки щодо злому і розсекречення конфеденційних даних користувача. Ймовірність злому зашифрованої інформації залежить від вибраного алгоритму шифрування, його складності і захищеності алгоритму, який використовується для створення спеціального випадкового ключа що шифрує дані.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Angular, AngularIO [Електронний ресурс]. [Веб-сайт].– 2021. – Режим доступу до ресурсу: https://angular.io/docs.
2. JSON Web Tokens [Електронний ресурс]: [Веб-сайт] .– 2021. – Режим доступу: https://jwt.io/introduction.
3. BCrypt [Електронний ресурс]: [Веб-сайт] .– 2021. – Режим доступу:https://www.npmjs.com/package/bcrypt.

Збитківський Владислав Сергійович – студент групи 2КН-18Б, кафедра комп'ютерних наук, факультет інтелектуальних інформаційних технологій та автоматизації, Вінницький національний технічний університет, м.Вінниця, e-mail: vladz15@ukr.net

Науковий керівник: ***Богач Ілона Віталіївна*** – к.т.н., доцент кафедри автоматизації та інтелектуальних інформаційних технологій, факультет інтелектуальних інформаційних технологій та автоматизації, Вінницький національний технічний університет, м.Вінниця, e-mail: ilona.bogach@gmail.com

Zbytkivskyi Vladislav Sergiyvich – Student of 2CS-18b group, Department of Computer Science, Faculty of Intelligent Information Technology and Automation, Vinnytsia National Technical University, Vinnytsia, e-mail: vladz15@ukr.net

Supervisor: ***Bogach Ilona Vitaliivna*** – Associate Professor of Automation and Intelligent Information Technologies, Faculty of Intelligent Information Technology and Automation, Vinnytsia National Technical University, Vinnytsia, e-mail: ilona.bogach@gmail.com