ISSN 2522-9818 (print)
ISSN 2524-2296 (online) *Innovative technologies and scientific solutions for industries. 2024. No. 3 (29)*

УДК 004.056.5 DOI: https://doi.org/10.30837/2522-9818.2024.3.076

Т. Радівілова, Л. Кіріченко, В. Пантєлєєв, А. Мазепа, В. Білодід

АНАЛІЗ МЕТОДІВ АВТЕНТИФІКАЦІЇ ДЛЯ ВЕБЗАСТОСУНКІВ ТА РЕАЛІЗАЦІЯ ВЕБЗАСТОСУНКУ З ІНТЕГРОВАНОЮ СИСТЕМОЮ АВТЕНТИФІКАЦІЇ

Предметом дослідження є методи й засоби безпечної автентифікації користувачів у вебзастосунках. **Мета роботи** – аналіз методів автентифікації та реалізація вебзастосунку із системою автентифікації з інтеграцією *JWT*-токенів і стандарту *OAuth v2.0*. У статті розв'язуються такі **завдання**: аналіз основних протоколів і методів автентифікації користувачів у вебзастосунках, реалізація автентифікації на основі стандарту *OAuth 2.0* та *JWT Access/Refresh*-токена; дослідження ризиків вразливостей і атак для реалізованих вебзастосунків. Упроваджено такі **методи**: порівняння, емпіричний аналіз, розрахункові методи. **Досягнуті результати**: проаналізовано протоколи та методи автентифікації користувачів у вебзастосунках; обрано методи автентифікації *JWT*-токенів і стандарту *OAuth v2.0* для побудови сучасних вебпрограм; створено вебзастосунок на основі обраних методів автентифікації у вебзастосунках; описано ризики реалізації вразливостей і атак у вебпрограмах. **Висновки**. Проаналізовано найбільш відомі методи автентифікації у вебзастосунках, що показало наявність багатьох недоліків, які спричиняють значні ризики в процесі використання вебпрограм. Створено стандартний вебзастосунок на основі конструктора вебпрограм *shopify* з автентифікацією на основі *Hypertext Transfer Protocol cookie session* та проаналізовано ризики вразливостей і атак на нього, унаслідок чого встановлено, що ризики є дуже високими. Обрано два найбезпечніших методи автентифікації для реалізації вебзастосунку: *JWT Access/Refresh*-токена з *browser fingerprints* та стандарту *OAuth 2.0*, на основі яких реалізовано вдосконалений вебзастосунок. Проаналізовано ризики вразливостей і атак на вдосконалений вебзастосунок, який показав, що ризики реалізації вразливостей і атак на нього є дуже низькими. Реалізація, налаштування та аналіз упровадження методу автентифікації *JWT Access/Refresh*-токена в поєднанні з *browser fingerprints* показали, що ця комбінація забезпечує найбільш надійне запобігання викрадення токенів та застосування їх з іншого комп'ютера. Також реалізовано автентифікацію за допомогою *OAuth 2.0*. У дослідженні було виявлено, що делегування автентифікації на сервіси *Facebook* або *Google* можуть забезпечити низький рівень ризику реалізації атак і вразливостей на вебзастосунок.

Ключові слова: автентифікація; комплексна система автентифікації; *OAuth*; *JWT*; токен; вебзастосунок.

Вступ

Інтернет став одним з основних стандартів сучасної людини, він пронизує практично всі аспекти нашого життя – від акаунтів у соцмережах до банківських рахунків. Усю цю важливу для людини інформацію, втрата якої може спричинити чимало проблем, захищають багатьма різними методами [1–3].

Завжди існує питання вибору між необхідним рівнем захисту та ефективністю роботи в мережі [4]. У деяких випадках заходи щодо забезпечення безпеки є занадто складними, що може викликати неможливість отримання потрібної інформації навіть для авторизованих для цього користувачів [5, 6]. Однак засоби захисту даних дають змогу забезпечити конфіденційність, цілісність та доступність інформації за умов впливу на неї різного виду загроз та зменшити ризики втрати чи викрадення конфіденційних даних.

Зважаючи на перелічені чинники, можна зробити висновок, що пріоритет безпеки інформації у вебсередовищі є найвищим у нинішніх реаліях. Особливо це стосується вебзастосунків, що є одними із найбільш вразливих об'єктів у мережі. Одним із ключових аспектів безпеки вебзастосунків є автентифікація, яка передбачає перевірку особи користувачів, що отримують доступ до вебзастосунків. Автентифікація необхідна для перевірки особи користувача або системи, які хочуть отримати доступ до сервісів та інформації. Через неправильну реалізацію автентифікації зловмисники можуть скористатися вразливостями автентифікації, реалізувати атаки на вебзастосунок для отримання несанкційного доступу. Чимало вчених досліджували безпеку методів автентифікації та авторизації та їх впровадження у вебзастосунках, з чого випливає необхідність аналізу наукових робіт і та іншої літератури.

Сучасний стан наукових досліджень та технологій в промисловості. 2024. №3 (29)
ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

Аналіз публікацій та постановка завдання

Нині поширеним є використання рішення єдиного входу *Single Sign-On* (SSO), тобто одного облікового запису для автентифікації на багатьох сайтах, як це зазначено в роботах [7, 8]. Це рішення основане на впровадженні методів автентифікації *OAuth, OpenID*, токенів та інших, що, відповідно, мають бути захищені. Крім того, важливо, щоб системи *SSO* також задовольняли високі вимоги безпеки та конфіденційності. Однак науковець *Schmitz Guido* [9] виявив серйозні вразливості в системах *SSO*, що призводять до критичних атак на їх безпеку та конфіденційність.

Безпекою протоколу *OAuth 2.0*, який використовується в *SSO*, зацікавились автори роботи [10], які за допомогою тестування 75 вебсайтів виявили вразливості протоколу за неналежної конфігурації сайтів. Також вони розробили розширення для браузера, що успішно ідентифікує та попереджає користувача про неналежні налаштування *OAuth 2.0*. Але в роботі не наведено способи розв'язання проблем безпеки на стороні сервера. Для забезпечення кращої безпеки автори роботи [11] провели симуляцію протоколу *OAuth 2.0* і запропонували додаткові функції, основані на використанні токенів, які можуть покращити архітектурний дизайн і підвищити загальну ефективність безпеки протоколу, але не подали конкретних рішень. У праці [12] для продовження часу доступу до захищених ресурсів для автентифікованих користувачів запропоновано унікальний підхід, оснований на технологіях використання *OAuth* і токенів. Але це є рішенням вузькоспрямованої проблеми.

У роботі [13] автори спроєктували спрощений сервіс для автентифікації, авторизації користувачів та управління ними у веб- та мобільних програмах на основі *OAuth* та *OpenID*. Однак розроблений сервіс є небезпечним, особливо для застосунків, що зберігають вразливу інформацію. Над аналізом цієї проблеми працювали автори праці [14]. Вони розробили метод відкликання токена за допомогою запиту на відкликання до сервера авторизації, коли сервер ресурсів виконує аномальну поведінку з використанням токена, наприклад вихід із системи або зміна ідентичності власника ресурсу. Але фахівці не запропонували способи розв'язання інших проблем безпеки.

У роботі [15] проаналізовано програмні продукти, що реалізують підтримку технології єдиного входу *SSO* на основі *Security Assertion Markup Language* (*SAML*) або *OpenID*, подано рекомендації щодо конкретного вибору. Автори дослідження [16] описали протокол *OpenID Connect*, який використовується в стандарті SSO. На основі аналізу визначено ризики для конфіденційності, пов'язані з доступом користувачів до систем SSO за допомогою протоколу *OpenID Connect*. Але в студіях [15, 16] не запропоновано способи розв'язання проблем безпеки. Питання безпеки використання токенів та *OAuth* також досліджували автори роботи [17]. Вони провели *SSO*-тестування між системами, проаналізували валідацію токенів, перевірили структуру *JSON Web Token* (*JWT*) та мережну атаку на перехоплення даних *(Network Sniffing Attack)*. Однак системи *SSO* вразливі не тільки до цієї атаки.

Отже, проблема забезпечення безпеки вебзастосунків є актуальною. З огляду літератури зрозуміло, що потрібно проаналізувати методи та стандарти автентифікації для вебпрограм, спираючись на результати вивчення, розробити вебзастосунок із безпечною системою автентифікації та порівняти його якість із стандартним вебзастосунком.

Метою цієї роботи є аналіз методів автентифікації та реалізація вебзастосунку із системою автентифікації з інтеграцією *JWT*-токенів і стандарту *OAuth v2.0*.

Огляд методів автентифікації

Процесу автентифікації користувача передує його ідентифікація [18]. Під час ідентифікації інформація про особу надається у формі ідентифікатора для системи безпеки. Система безпеки шукатиме всі абстрактні об'єкти в наборі ідентифікаторів і знайде конкретний об'єкт, що застосовує реальна людина. Як тільки це буде зроблено, користувач буде ідентифікований. Наприклад, об'єктом може бути *id* акаунта користувача. Той факт, що особу ідентифіковано, не обов'язково означає, що вона справжня. Користувач має надати докази, що підтверджують його особу в системі, наприклад пароль, який називається обліковими даними. Їх перевірка є процесом автентифікації. Якщо автентифікація була успішною, то користувача буде авторизовано до системи з наданням назначених йому прав

ISSN 2522-9818 (print)
ISSN 2524-2296 (online) Innovative technologies and scientific solutions for industries. 2024. No. 3 (29)

доступу. Автентифікація необхідна для захищеного вебзастосунку.

Найбільш використовуваними є такі методи автентифікації [19]:

- автентифікація на основі файлів *Cookie*;
- автентифікація за допомогою токенів;
- сторонній доступ (*OAuth*, токен *Application Programming Interface* (*API*));
- *OpenID*;
- *SAML*.

Автентифікація на основі *Hypertext Transfer Protocol Cookie session*

Hypertext Transfer Protocol (HTTP) *Cookie session* – це крок до більш надійних і складних методів автентифікації. Усі запити HTTP не мають статусу [19], тобто неможливо зберігати інформацію про взаємодію, що відбулася між сервером і користувачем. Для того, щоб розв'язати цю проблему, була створена функція HTTP-сесії, що дає змогу вебсерверам зберігати дані (запити / відповіді) взаємодії між користувачами та серверами. Вони зберігають інформацію, основану на конкретній сесії (ідентифікатор сесії, час створення, останній доступ тощо), а також відомості про користувача (стан входу в систему та інші дані, які можуть знадобитися застосунку від користувача). Сесії можуть бути реалізовані за допомогою файлів *Cookie*, тобто невеликого фрагменту даних, який сервер надсилає веббраузеру користувача. Браузер може зберігати його і відправляти назад із запитами на той самий сервер. Файли *Cookie* застосовуються для автентифікації особи та містять конфіденційну інформацію, мають невеликий обсяг, але за умови значної кількості сесій і користувачів обсяг даних для зберігання стає проблемою. У табл. 1 наведено переваги та недоліки автентифікації на основі HTTP *Cookie session*.

Таблиця 1. *Переваги та недоліки автентифікації на основі HTTP Cookie session*

Переваги	Недоліки
Файли *Cookie* займають дуже мало місця.	Файли *Cookie* вразливі до *XSS*- та *CSRF*-атак.
Файли *Cookie* прості у використанні.	Масштабування стає проблемою, коли багато користувачів входять у систему.
Файли зберігаються на сервері та їх набагато важче вкрасти або підмінити.	Містять конфіденційну інформацію про користувача, що робить його мішенню для зловмисників.
Інформацію, яку записують у *Cookie*, зашифровують перед відправленням, а самі *Cookie* передають за протоколом HTTPS.	Налаштування використання та відправлення *Cookie* залежить від розробника сайту, який може зробити некоректні налаштування.

Основні принципи автентифікації на основі технології *token*

Як і для попереднього методу автентифікації за допомогою *Cookie session* для цієї стратегії немає конкретного зразка, унаслідок цього всі реалізації специфічні для певних систем. Автентифікація на основі технології *token* найчастіше застосовується в побудові розподілених систем *SSO*, де один застосунок (*service provider* або *relying party*) делегує функцію автентифікації користувачів іншому застосунку (*identity provider* або *authentication service*) [19]. Типовим прикладом цього способу є вхід у програму через обліковий запис у соціальних мережах. Тут соціальні мережі є сервісами автентифікації, а застосунок довіряє функцію автентифікації користувачів обраній соціальній мережі.

Реалізація цього способу полягає в тому, що постачальник ідентифікаційної інформації надає достовірні відомості про користувача у вигляді токена, а застосунок постачальника послуг використовує цей токен для ідентифікації, автентифікації та авторизації користувача. Постачальник ідентифікаційних даних здебільшого використовується в стандартах на зразок *OAuth 2.0*, які застосовують делегування автентифікації. Постачальника ідентифікаційних даних можна буде оминути в деяких реалізаціях автентифікації за токеном доступу. Наприклад, деякі реалізації з *JWT*-токеном не потребують додаткового постачальника ідентифікаційної інформації для генерації токена. Сам сервер створює токен, відправляє та приймає його з клієнта.

Автентифікація на основі *JSON Web Token (JWT)*

JWT є механізмом перевірки власника деяких даних *JSON*. Це зашифрований рядок, що може містити необмежену кількість інформації (на відміну від файлу *Cookie*) і має криптографічний підпис [20].

Коли сервер отримує *JWT*, він може гарантувати, що даним у цьому рядку можна довіряти, оскільки він підписаний криптографічним підписом. Жоден

Сучасний стан наукових досліджень та технологій в промисловості. 2024. №3 (29)
ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

посередник не може змінити *JWT* після його відправлення. Важливо зауважити, що *JWT* гарантує володіння інформацією, але не шифрування. Тобто дані *JSON*, які зберігаються в *JWT*, не зашифровані, і їх можна побачити в процесі перехоплення токена. З цієї причини настійно рекомендується використовувати HTTPS із *JWT*. Одним із найскладніших рішень щодо реалізації *JWT* є місце зберігання токена, таким чином безпека автентифікації на основі *JWT* залежатиме від налаштувань розробника сайту. Токен має зберігатися в безпечному місці в браузері користувача: *local storage/ session storage; Cookie*-файл; у пам'яті застосунку.

У першому випадку, якщо токен зберігається всередині *LocalStorage* або *Session Storage*, він доступний кожному скрипту на вебсторінці. Атака *Cross-site scripting* (*XSS*) може дозволити зловмисникові отримати доступ до токенів. Не рекомендується зберігати токен у локальному сховищі або сховищі сеансів.

У другому випадку токени, що зберігаються у *Cookie*-файлах, вразливі до *CSRF*-атак (*Cross-site request forgery*). Тому також не рекомендовано зберігати їх у *Cookie*-файлах.

Якщо будь-який із скриптів, доданих у вебсторінку, буде скомпрометовано, то зловмисник зможе отримати доступ до всіх токенів, які зберігаються в браузерах. Саме тому найправильнішим рішенням буде зберігання токена безпосередньо в пам'яті застосунку, до якої не буде можливості дістатися звичайними методами. У табл. 2 наведені переваги та недоліки використання *JWT*-токена.

Таблиця 2. *Переваги та недоліки використання JWT-токена*

Переваги	Недоліки
Масштабованість сервера не змінюється в разі збільшення кількості користувачів.	Розмір *JWT* значно більший за ідентифікатор сеансу *Cookie*.
Не має стану – вебзастосунок не зобов'язаний зберігати інформацію сесії на сервері, що зменшує навантаження на сервер, покращує продуктивність та масштабованість.	*JWT* не може відкликати доступ до користувача.
JWT містить більше інформації про користувача.	Токен зберігається на стороні клієнта, що робить його вразливим для зловмисників.
Переносимість – дає змогу міждоменну та міжплатформну автентифікацію та авторизацію.	Схильність до крадіжок, тобто якщо хтось викраде *JWT* у користувача або системи, він може застосувати його, щоб привласнити їхню особистість і отримати доступ до їхніх ресурсів.
Адаптованість – містить будь-яку інформацію щодо вебзастосунку, яка дає змогу контролювати доступ у точний і персоналізований спосіб.	

Автентифікація на основі *Access/Refresh*-токенів

Для підвищення безпеки автентифікації на основі *JWT*-токена було запропоновано розмежування між видами токенів на *Access* і *Refresh* [21, 22]. *Access*-токен використовується для авторизації запитів і зберігання додаткової інформації про користувача. *Refresh*-токен видається сервером за результатами успішної автентифікації та застосовується для отримання нової пари *Access*/*Refresh*-токенів. Частіше за все зберігається в базі даних сервера. Кожен токен має свій термін життя, наприклад *Access* – 30 хв, а *Refresh* – 60 днів. *Refresh*-токен зберігається на сервері для обліку доступу та інвалідації крадених токенів. Таким чином сервер точно визначає, кому дозволено авторизуватися.

Для використання можливості автентифікації на більш ніж одному девайсі необхідно зберігати всі *Refresh*-токени щодо кожного користувача.

У момент *Refresh*, тобто оновлення *Access*-токена, оновлюються обидва токени – *Access* та *Refresh*. *Refresh*-токен у момент *Refresh* порівнює себе з тим *Refresh*-токеном, що лежить в базі даних, і в разі успіху, а також якщо в нього не завершився термін, система оновлює токени. *Refresh*-токен має термін життя в разі, якщо користувач буде в офлайні понад 60 днів, тоді доведеться заново вводити логін / пароль.

Fingerprint браузера – це інструмент ідентифікації браузера користувача. Це геш, згенерований на базі деяких унікальних параметрів браузера. Перевага *fingerprint* полягає в тому, що в процесі генерації це значення буде унікальним для конкретного браузера користувача та не змінюватиметься в майбутньому. Тому скомпрометувати *fingerprint* браузера дуже важко.

ISSN 2522-9818 (print)
ISSN 2524-2296 (online) Innovative technologies and scientific solutions for industries. 2024. No. 3 (29)

Автентифікація на основі мови розмітки тверджень безпеки *SAML*

SAML – це стандарт автентифікації та авторизації, варіант розширюваної мови розмітки (*Extensible Markup Language*, *XML*) для обміну інформацією про безпеку в інтернеті.

Обмін *SAML* відбувається між системними об'єктами, що називаються стороною, яка підтверджує (також називається органом *SAML*), і стороною, яка довіряє та обробляє отримані нею підтвердження безпеки [19, 21]. Підтвердження безпеки – це стандартизовані підтвердження мовою розмітки, які визначають рішення щодо контролю доступу.

Існує два типи єдиного входу (*Single Sign-On*, *SSO*) *SAML*: з ініціативи постачальника ідентифікаційних даних (*Identity Provider*, *IdP*) та з ініціативи постачальника послуг (*Service Provider*, *SP*). Обидва використовують *IdP* для автентифікації користувача, основна різниця полягає у відправній точці. *SAML SSO* з ініціативою *IdP*: користувач намагається увійти до *IdP*, а *IdP* може безпосередньо перевірити особу користувача за допомогою *SAML*-відповіді. *SAML SSO*, ініційоване *SP*: користувач намагається увійти до *SP*, і *SP* маю згенерувати *SAML*-запит. Запит і користувач надсилаються *IdP* для автентифікації, а потім повертаються до *SP* для завершення входу.

Автентифікація на основі протоколу автентифікації *OpenID Connect*

OpenID Connect – це протокол автентифікації, побудований на основі *OAuth 2.0*. Він дозволяє користувачам автентифікуватися та обмінюватися своєю ідентифікаційною інформацією із застосунками та сервісами в стандартизований спосіб [19, 21].

Однак основна різниця між *OAuth 2.0* і *OpenID Connect* полягає в типі токенів, що видаються. *OAuth 2.0* випускає токени доступу, що зазвичай мають обмежений термін дії та видаються для певного набору дозволів. *OpenID Connect* випускає токени ідентифікації, що застосовуються для автентифікації користувача та надання ідентифікаційної інформації застосунку. Хоча *OAuth 2.0* і *OpenID Connect* використовують схожі потоки й механізми, різниця між токенами доступу та ідентифікаційними токенами є важливою. Токени доступу використовуються для авторизації, дозволяючи програмі отримати доступ до захищених ресурсів, тоді як токени ідентифікації застосовуються для автентифікації, підтверджуючи особу користувача.

Переваги та недоліки різних методів автентифікації

Для автентифікації вебзастосунків використовуються кілька методів, зокрема файли *Cookie*, *JWT*, *OAuth*, *API Token*, *SAML* та *OpenID*, описані вище. Залежно від сценарію застосування та вимог безпеки вони мають різні переваги та недоліки (табл. 3).

Таблиця 3. *Переваги та недоліки найпоширеніших методів автентифікації*

Метод	Переваги	Недоліки
Cookies	Прості в реалізації та широко підтримуються, зберігають інформацію про сеанс, налаштування або інші дані для ідентифікації користувача або налаштування користувацького досвіду.	Вразливі до *CSRF*-атак, додають накладні витрати до кожного запиту, обмежують розмір і кількість.
JWT	Бездержавний, може переносити більше даних, може підтримувати кілька доменів або служб, може бути перевірений будь-ким, хто має доступ до ключа.	Вразливий до *XSS*-атак, має фіксований термін дії, не може бути легко відкликаний або оновлений.
OAuth	Дозволяє користувачеві надавати доступ до своїх ресурсів або інформації з одного сайту іншому сайту без обміну обліковими даними, може використовувати різні типи токенів, такі як *JWT*-, *API*-токени або *SAML*-твердження.	Складний і вимагає участі багатьох сторін та взаємодії, створює певні ризики для безпеки, такі як фішингові атаки, витік токенів або їх повторне відтворення.
SAML	Дозволяє користувачеві входити на один сайт і отримувати доступ до іншого сайту без повторного введення облікових даних, використовує твердження, що містять інформацію про особу користувача, атрибути або рішення про авторизацію, може бути підписана й зашифрована постачальником ідентифікації та перевірена постачальником послуг.	Складний і вимагає оброблення та розбору *XML*, має деякі проблеми з продуктивністю через розмір і кількість повідомлень, що беруть участь.
OpenID	Дозволяє користувачеві входити на один сайт і застосовувати свій ідентифікатор для доступу до іншого сайту без створення облікового запису або повторного введення облікових даних, може використовувати різні типи токенів для подання ідентифікаційної інформації.	Складний, вимагає участі багатьох сторін і взаємодії, створює певні ризики для безпеки, такі як фішингові атаки, підроблення ідентифікаторів або витік токенів.

Сучасний стан наукових досліджень та технологій в промисловості. 2024. №3 (29)
ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

Оскільки із загального порівняння складно зробити висновок, то більш детально порівняємо методи автентифікації *Cookie* та *JWT token, OAuth* і *SAML*.

Порівняння використання *Cookie* та *JWT token*

На цей час найбільш поширеними методами автентифікації є *Cookie sessions* та *JWT token* [21]. У табл. 4 наведено результати порівняння цих двох методів за такими параметрами: масштабованість, безпека, сервіси *API RESTful*, продуктивність. Необхідно зауважити, що розглядається стандартна реалізація алгоритму *JWT token* без специфічних змін. Прикладом є *Refresh*-токен.

Зважаючи на результати аналізу, можна зробити висновок, що метод автентифікації *Cookie sessions* відходить на другий план порівняно з *JWT*-токеном. У контексті використання *JWT*-токена із програмами невеликого розміру можна застосовувати звичайну *JWT*-автентифікацію без *Refresh*-токена, яка повністю замінить сесійний варіант із *Cookie sessions*. У разі потреби застосування *JWT* із програмами середнього та великого розміру можна використовувати комбінацію *Access/Refresh*-токена та *fingerprints* браузера.

Таблиця 4. *Параметри використання методів автентифікації Cookie та JWT token*

Параметри	*Cookie sessions*	*JWT token*
Масштабованість	Інформація сеансу зберігається в пам'яті на сервері або в базі даних. У сценарії горизонтального масштабування створюється окрема центральна система зберігання сеансів, до якої мають доступ усі сервери застосунків.	Легко масштабується, оскільки можна використовувати маркери для доступу до ресурсів з різних серверів, що економить витрати, оскільки не потрібен виділений сервер для зберігання сесій споживачів.
Безпека	У стратегії *Cookies session* усі дані зберігаються на сервері, тому вважаються відносно захищеними.	Завдяки підпису за допомогою секретного ключа на сервері токен не можна змінити непомічено без доступу до секретного ключа. Зберігання токена в будь-якому місці, окрім пам'яті застосунку, є дуже ризиковим.
Сервіси *API RESTful*	Збереження стану застосунку. *API*-інтерфейс досить часто обслуговується з одного сервера, а в реальності застосунок використовує його з іншого.	*RESTful API* не має стану, тож не важливо, звідки *API* або застосунок обслуговується.
Продуктивність	Накладні витрати не значні, оскільки в процесі кодування розмір *JWT* буде в кілька разів перевищувати розмір ідентифікатора *SESSION*.	Значне навантаження під час кожного HTTP-запиту. Однак *JWT* обмінюють розмір на можливість зберігати інформацію на стороні клієнта. Наприклад, привілеї в самому токені або *id* користувача.

Порівняння методів автентифікації та авторизації *OAuth* і *SAML*

Для досягнення мети роботи та реалізації автентифікації у вебзастосунку необхідне використання облікового запису єдиного входу. Перевага облікового запису єдиного входу (*single sign-on – SSO*) полягає в тому, що користувачі входять у систему один раз, застосовуючи один набір облікових даних, у цьому разі вони можуть отримувати доступ до декількох сервісів і застосунків протягом одного сеансу [21]. Єдиний вхід використовується для управління автентифікацією та безпечним доступом до корпоративних мереж, вебзастосунках тощо. Для впровадження *SSO* найчастіше впроваджуються стандарти *SAML (Security Assertion Markup Language, SAML), OAuth*, рідше – *OpenID Connect*. Результати аналізу використання методів *SAML* та *OAuth* для реалізації єдиного входу наведено в табл. 5.

Беручи до уваги подане порівняння, для реалізації було обрано два методи автентифікації у вебзастосунку: *JWT*-токени та *OAuth 2.0*. Тож розглянемо технології захисту вебпрограми, які було використано для реалізації захищеного вебзастосунку із впровадженням зазначених підходів автентифікації.

ISSN 2522-9818 (print)
ISSN 2524-2296 (online) Innovative technologies and scientific solutions for industries. 2024. No. 3 (29)

Таблиця 5. *Параметри впровадження методів автентифікації Security Assertion Markup Language та Open Authorization*

Параметри	*SAML*	*OAuth*
Робочий процес	Передбачає обмін інформацією між підтверджувальною стороною (органом *SAML*) і стороною, що покладається на нього, використовуючи підтвердження безпеки у форматі *XML*.	Оснований на взаємодії між власником ресурсу, сервером ресурсу, клієнтом (застосунком) і сервером авторизації.
SSO (Single Sign-On)	Підтримує *SSO* за допомогою передачі інформації про автентифікацію та авторизацію між вебсерверами.	Вмикає *SSO*, дозволяючи автентифікованим користувачам в одному застосунку авторизувати сторонні програми для доступу до даних користувача.
Безпека	Вважається більш безпечним завдяки можливості шифрування тверджень, що підходить для оброблення конфіденційної інформації.	Зосереджується на делегуванні доступу через токени, що не передбачає шифрування тверджень.
Інтероперабельність	Використовується організаціями з великими інвестиціями в *XML* і тими, що потребують безпечного, федеративного управління ідентичностями.	Підтримує різні типи клієнтів, зокрема веб-, мобільні та десктопні застосунки, що робить його цінним для розробників мобільних програм і сервісів у відкритому інтернеті.
Основні чинники	Еволюція *SSO*, розширення федеративного управління ідентичностями, зміна галузевих стандартів.	Простота для розробників, потреба в обмеженому обміні інформацією про користувачів з іншими застосунками, спільна рекламна діяльність між великими інтернет-компаніями.
Шифрування та безпека	Дозволяє шифрувати твердження, забезпечуючи вищий рівень безпеки для обміну конфіденційними даними.	Переважно зосереджується на доступі на основі токенів, без деталізації стандартів шифрування для самих токенів.
Сумісність	Може використовуватися разом з *OAuth* у середовищах, що вимагають як автентифікації, так і авторизації для різних аспектів управління доступом.	Може інтегруватися із *SAML* для сценаріїв, де необхідна первинна перевірка ідентичності користувача перед наданням токенів доступу для доступу до ресурсів.
Варіанти використання	Багатодоменне *SSO*, ідеально підходить для великих організацій та корпоративних застосунків (наприклад, *Salesforce, Marketo*).	Конфіденційність користувачів, надання доступу до приватних ресурсів на різних вебсайтах або в різних застосунках без передачі ідентифікаційних даних користувача.
Рекомендації щодо впровадження	Найкраще підходить для організацій, які вкладають значні кошти в ресурси *XML* і потребують надійних методів автентифікації.	Рекомендується для сучасних мобільних застосунків, а також у разі, коли доступ до *API* є пріоритетним, особливо у відкритому інтернеті.

Опис технологій, що використані для реалізації вебзастосунку

Проаналізуємо безпеку автентифікації вебзастосунку, запропонованого стандартним інтернет-магазином із продажу електротоварів, який розробляється за допомогою конструктора вебпрограм *shopify* [23]. Вебзастосунок є розділенням серверного та клієнтського модулів. Це розділення та технології, використані в процесі розроблення, дають змогу назвати клієнтську частину вебпрограми *Single page application* (SPA). SPA – це вебзастосунок, що взаємодіє з веббраузером, динамічно переписуючи поточну вебсторінку новими даними, отриманими способом відправлення на кінцеву точку вебсервера запиту, замість того щоб за замовчуванням браузер завантажував цілі нові сторінки [21, 24]. У вебзастосунку реалізовано базовий стандартний функціонал сучасної вебпрограми: автентифікація на основі *Hypertext Transfer Protocol Cookie session*; контроль доступу з розподіленням на типи користувача та адміністратора; механізм роботи з базою даних *SQL*; механізм роботи із сервісом *paypal*. Було використано стандартні бібліотеки в розробленні клієнтської частини програми. Серверна частина вебзастосунку створена в середовищі розробки *Nodejs* з допоміжною бібліотекою *express* та стандартними бібліотеками, оскільки серверна частина не містить складного функціоналу в роботі з інформацією, що приходить на сервер. Проведемо якісний аналіз ризиків атак і вразливостей стандартного запропонованого вебзастосунку на основі методології *OWASP* та методу експертного оцінювання [25], результати якого подані в табл. 6.

Для кількісного оцінювання ризику результуючому ризику надано значення від 1 до 5, де дуже низькому ризику відповідає значення 1, низькому – 2, середньому – 3, високому – 4, дуже високому – 5. Відповідно до цього значення результуючого ризику для стандартного вебзастосунку інтернет-магазину

ISSN 2522-9818 (print)
Сучасний стан наукових досліджень та технологій в промисловості. 2024. №3 (29) ISSN 2524-2296 (online)

дорівнює 49, за максимального значення ризику 55, а мінімального – 11. Зрозуміло, що стандартна реалізація вебзастосунку містить значну кількість серйозних вразливостей, ризики реалізації яких є дуже високими. Саме тому цей вебзастосунок буде вдосконалено та побудовано систему безпеки, що використовує провідні методи запобігання вразливостям і кращі методи автентифікації. Така система зведе ризик нападу зловмисників до мінімуму та зменшить імовірність реалізації вразливостей.

Таблиця 6. *Якісна оцінка ризику стандартного вебзастосунку інтернет-магазину*

Назва ризику	Тип вразливості	Збиток	Вірогідність виникнення	Результуючий ризик
Відправлення шкідливого коду на сервер з клієнта через незахищені поля введення	*SQL injection*	дуже високий	висока	високий
Отримання та оброблення сервером шкідливого коду	*SQL injection*	дуже високий	дуже висока	дуже високий
Злам паролю	порушення автентифікації	високий	дуже висока	дуже високий
Викрадення токена сесії	порушення автентифікації	високий	дуже висока	дуже високий
Розкриття паролів під час отримання несанкційного доступу до бази даних	розкриття конфіденційних даних	дуже високий	дуже висока	дуже високий
Викрадення конфіденційної інформації в процесі перехоплення трафіку	розкриття конфіденційних даних	дуже високий	дуже висока	дуже високий
Отримання несанкційного доступу до облікового запису адміністратора	порушення контролю доступу	високий	висока	високий
Отримання несанкційного доступу до функцій адміністратора	порушення контролю доступу	високий	висока	високий
Введення шкідливого коду в клієнтську частину застосунку	*XSS*	високий	низька	середній
Вразливість відправлення конфіденційної інформації за допомогою вебсайту зловмисника	*CSRF*	високий	середня	високий
Невчасне знаходження вразливостей у системі	Недостатнє логування та моніторинг	дуже високий	дуже висока	дуже високий

Технології серверної частини вдосконаленого вебзастосунку

У процесі вдосконалення вебпрограми та реалізації безпечних методів автентифікації необхідно розробити сервер вебзастосунку за допомогою безпечних, швидких і ефективних технологій. За основу серверної частини обрано технологію *Nodejs*, яка впроваджується з фреймоворком *Express*, як і в стандартній реалізації вебзастосунку. Для звичайної автентифікації взято бібліотеку *jsonwebtoken*, яка уможливлює генерацію та перевірку *JWT*. Як автентифікацію на основі стандарту *OAuth 2.0* було взято бібліотеку автентифікації *passportjs* і доповнювальну стратегію автентифікації, що називається *passport-facebook-token*. Хостинг та сертифікування сервера надається хмарним сервісом *Heroku*. Базою даних обрано *PostgreSQL*, що також розміщена на хмарному сховищі *Heroku*. Детальніший перелік бібліотек, використаних у розробленні серверної частини застосунку, наведено в табл. 7.

Таблиця 7. *Детальний перелік використаних бібліотек на сервері*

Назва	Опис бібліотеки
bcryptjs	Бібліотека для шифрування даних на основі секрету.
body-parser	Бібліотека, необхідна для сприйняття сервером інформації, що надходить від клієнтської частини вебзастосунку.
cookie-parser	Бібліотека, потрібна для сприйняття сервером *Cookie*-файлів, що надходять від клієнтської частини вебзастосунку.
cors	Бібліотека, необхідна для сприйняття сервером запитів, що надходять від клієнтської частини вебзастосунку.
pg	Бібліотека, необхідна для взаємодії з базою даних *PostgreSQL*.
express-promise-router	Бібліотека, що допомагає серверу визначити *endpoints*, до яких звертатиметься клієнт.
express-dynamic-middleware	Допоміжна бібліотека для реалізації автентифікації.

Тепер необхідно обрати технології, що використовуються на стороні клієнта.

ISSN 2522-9818 (print)
ISSN 2524-2296 (online) Innovative technologies and scientific solutions for industries. 2024. No. 3 (29)

Технології клієнтської частини вдосконаленого вебзастосунку

За основу клієнтської частини взято технологію *JavaScript*, яка використовується з фреймоворком *React*. Для ініціювання *OAuth 2.0* автентифікації взято бібліотеку *react-facebook-login*, що дає змогу почати процес автентифікації за допомогою сервісу *Facebook*. Хостинг і сертифікування клієнта надається хмарним сервісом *Heroku*. Детальніший перелік бібліотек, використаних у розробленні клієнтської частини вебзастосунку, наведено в табл. 8.

Таблиця 8. *Детальний перелік використаних бібліотек клієнтської частини вебзастосунку*

Назва	Опис бібліотеки
React-router	Бібліотека, що допомагає переходити між сторінками на клієнтській частині та забезпечує безпечний роутинг.
Axios	Бібліотека, що дає змогу клієнтській частині робити запити на сервер і додавати всю необхідну інформацію до цих запитів.
Redux	Бібліотека, що забезпечує клієнтську частину централізованим сховищем даних. Від цієї технології унаслідується низка інших допоміжних технологій.
Redux-thunk	Бібліотека, що допомагає клієнтській частині працювати із запитами на сервер.
Redux-form	Бібліотека, що допомагає клієнтській частині працювати з формами та перевіряти валідність інформації, яка прийшла на цю форму.
material-ui	Бібліотека для полегшення роботи з візуальною частиною клієнтської сторони.
Js-cookie	Бібліотека для роботи з *Cookie*-файлами на клієнтській частині.

Автентифікація застосунку буде реалізована двома способами:

– *JWT*-токен автентифікація для користувачів, які не застосовують сервіс *Facebook*;

– автентифікація, що використовує стандарт *OAuth 2.0* та сервіс *Facebook*.

Реалізація *JWT*-токен автентифікації

Для реалізації *JWT*-токен автентифікації в прикладній програмі застосована стратегія, що передбачає використання *Access/Refresh*-токенів. Для ідентифікації користувача застосовується *browser fingerprints*, що дає змогу відстежити другий браузер, з якого зроблено запит на сервер.

Після відправлення даних користувача (*e-mail*, логін, пароль, *browser fingerprint*) на сервер, він реєструє особу та додає цю інформацію в базу даних *PostgreSQL*. Усі паролі в базі даних *PostgreSQL* зберігаються в гешованому вигляді з додаванням псевдовипадкових чисел для цієї реалізації застосунку. Це зроблено для того, щоб запобігти викраденню паролів користувачів, якщо базу даних буде скомпрометовано зловмисником. Після реєстрації користувач може ввести свої дані знову та пройти автентифікацію для подальшої авторизації на сайті.

Автентифікація до сайту відбувається таким чином, що сервер порівнює інформацію, надану користувачем. Якщо особа була знайдена в базі даних, то їй будуть видані *Access/Refresh*-токени, які вона використовуватиме для доступу до захищених *API* кінцевих точок. Для запобігання фальсифікації нових *Access*-токенів *Refresh*-токен додатково порівнює *fingerprints* браузера, та, якщо дані збігаються, видає новий *Access*-токен. В іншому разі *Refresh*-токен знищується, внаслідок чого ініціюється повторна автентифікація.

Реалізація автентифікації *OAuth 2.0*

Окрім звичайної автентифікації в застосунку також реалізовано автентифікацію за стандартом *OAuth 2.0*. На рис. 1 подано схематичний вигляд автентифікації *OAuth 2.0* у цьому застосунку.

Рис. 1. Схема *Oauth 2.0* у реалізації застосунку

У цій реалізації стратегія автентифікації дозволяє користувачу зайти на сайт від імені *Facebook*. Клієнт, який автентифікується через *Facebook*, довіряє доступ до частини своїх даних. Сервер отримує доступ до сервісу *Facebook* з відомостями користувача та надає повне право

Сучасний стан наукових досліджень та технологій в промисловості. 2024. №3 (29)
ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

особі, яка пройшла автентифікацію через *Facebook*, на доступ та використання захищеної інформації із захищених *API* кінцевих точок.

Процес автентифікації починається на клієнтській частині, коли особа за допомогою *OAuth* надає свої дані від *Facebook*-акаунта. Це досягається за допомогою спливаючого вікна, яке подане сервісом *Facebook*. Передача даних у цьому вікні відбувається за протоколом HTTPS, тому можливість крадіжки інформації у процесі відправлення мінімальна. Після надання дозволу клієнт отримує токен, що буде відправлений до сервера. Після відправлення сервер обробляє токен і передає його назад до сервісу *Facebook* для обміну на дані клієнта, який відправив цей токен.

Список даних, що приходять із *Facebook*, користувач може налаштовувати. Після того, як особа підтвердила список даних, сервер зберігає відомості нового користувача (*browser fingerprints, facebook refresh token, facebook id*) в окрему колекцію в базі даних, яка призначена спеціально для тих, хто автентифікувався за допомогою *Facebook*. Після збереження особи в базі даних сервер відправляє їх *Refresh*- та *Access*-токени, згенеровані сервісом *Facebook*, та інформацію профіля з *Facebook*. Під час наступного входу сервер зможе запросити ці відомості з бази даних та перевірити достовірність користувача.

Аналіз ефективності запропонованих методів автентифікації

Для визначення якості реалізованих методів автентифікації вебзастосунку було проведено його якісне оцінювання в процесі реалізації автентифікації за допомогою *JWT Access/Refresh*-токенів та *OAuth 2.0* з використанням *browser fingerprints*. У цьому разі вебзастосунок та функції захисту було налаштовано, як вказано вище. Було оцінено ризики атак і вразливостей розробленої вебпрограми на основі методології *OWASP* і методу експертного оцінювання [25] з упровадженими підходами автентифікації *JWT Refresh*- та *Access*-токенів і *OAuth 2.0*. Результати оцінювання наведені в табл. 9.

Таблиця 9. *Якісна оцінка ризику вебзастосунку з реалізацією функцій безпеки*

Назва ризику	**Тип вразливості**	**Збиток**	**Вірогідність виникнення**		**Результуючий ризик**	
			JWT	***OAuth***	***JWT***	***OAuth***
Відправлення шкідливого коду на сервер з клієнта через незахищені поля введення	*SQL injection*	низький	дуже низька	дуже низька	дуже низький	дуже низький
Отримання та оброблення сервером шкідливого коду	*SQL injection*	низький	дуже низька	дуже низька	дуже низький	дуже низький
Злам паролю	порушення автентифікації	низький	дуже низька	дуже низька	дуже низький	дуже низький
Викрадення токена сесії	порушення автентифікації	середній	середня	низька	середній	низький
Розкриття паролів у процесі отримання несанкційного доступу до бази даних	розкриття конфіденційної інформації	низький	дуже низька	дуже низька	дуже низький	дуже низький
Викрадення конфіденційної інформації під час перехоплення трафіку	розкриття конфіденційної інформації	дуже низький	дуже низька	дуже низька	дуже низький	дуже низький
Отримання несанкційного доступу до облікового запису адміністратора	порушення контролю доступу	середній	дуже низька	низька	низький	низький
Введення шкідливого коду в клієнтську частину застосунку	*XSS*	дуже низький	дуже низька	дуже низька	дуже низький	дуже низький
Вразливість відправлення конфіденційних даних через вебсайт зловмисника	*CSRF*	низький	дуже низька	дуже низька	дуже низький	дуже низький
Невчасне знаходження вразливостей у системі	недостатнє логування та моніторинг	низький	середня	середня	середній	середній

ISSN 2522-9818 (print)
ISSN 2524-2296 (online) Innovative technologies and scientific solutions for industries. 2024. No. 3 (29)

Для кількісного оцінювання ризику результуючому ризику надано значення від 1 до 5, де дуже низькому ризику відповідає значення 1, низькому – 2, середньому – 3, високому – 4, дуже високому – 5. Максимальне значення ризику дорівнює 55, а мінімальне – 11. Згідно з цим значення результуючого ризику для автентифікації на основі *JWT Refresh*- та *Access*-токенів дорівнює 15, а для *OAuth 2.0* – 14 відповідно, що є досить низьким значенням. Як можна побачити з табл. 9, два пункти мають середній ступінь ризику – це невчасне знаходження вразливостей та викрадення токена сесії. Щодо невчасного знаходження вразливостей, то це залежить від людського фактора. Яка б надійна не була система логування, людина все одно може не побачити загрози, що вже була записана в лог. Щодо вразливості викрадення токена сесії: оскільки роль сесійних токенів відіграють *Cookie* з *JWT*, вони все ще можуть бути викрадені через шкідливе програмне забезпечення (ПЗ) *Stealer*. У цьому разі крадіжка може бути проведена напряму з файлової системи жертви внаслідок потрапляння ПЗ *Stealer* до комп'ютера жертви, а це залежить від його захищеності. Отже, використання *OAuth 2.0* разом із *browser fingerprints* є більш безпечним, ніж у разі автентифікації з *JWT Refresh*- та *Access*-токенами, але різниця в ризику є мінімальною, тож можна використовувати будь-який із реалізованих протоколів.

Висновки

У роботі розв'язано завдання створення вебзастосунку з безпечною автентифікацією. Для цього проаналізовано найбільш відомі методи автентифікації до вебзастосунків: *Hypertext Transfer Protocol Cookie sessions; JWT*-токени; стандарт *OAuth 2.0; OpenID; SAML*. Установлено, що кожен із цих методів автентифікації має недоліки, а це означає наявність ризиків у процесі їх використання та необхідність налаштувань зазначених методів під час їх програмної реалізації. Створено стандартний вебзастосунок електронного магазину на основі конструктора вебпрограм *shopify* з автентифікацією на основі *Hypertext Transfer Protocol Cookie session*. Проаналізовано ризики вразливостей і атак на цей вебзастосунок, який показав, що ризики реалізації вразливостей і атак на програму є дуже високими, з чого випливає необхідність удосконалення автентифікації, налаштувань та функцій безпеки вебзастосунку. З огляду на аналіз ризиків вразливостей і атак створеного стандартного вебзастосунку, переваги та недоліки методів автентифікації, було вдосконалено вебпрограму: методи автентифікації, налаштування застосунку, функції безпеки. Для автентифікації обрано застосування комбінації *JWT Access/Refresh*-токена з використанням *browser fingerprints* та *OAuth 2.0* із делегуванням автентифікації на сервіс *Facebook*. Проаналізовано ризики вразливостей і атак на цей вебзастосунок, який показав, що ризик реалізації вразливостей і атак на вдосконалену програму є дуже низьким. Показано, що одним із найбільш ефективних методів забезпечення безпеки даних користувача вебзастосунку виявилося використання комбінації *JWT Access/Refresh*-токена разом із *browser fingerprints*. Другим безпечним методом автентифікації, який було реалізовано у вебзастосунку, є метод автентифікації *OAuth 2.0* із делегуванням автентифікації на сервіс *Facebook*. Налаштування цього методу показало, що за умови його використання ризик реалізації вразливостей і атак на вебзастосунок є дуже низьким. Зазначено, що такі методи автентифікації можна скомпрометувати тільки під час першого посилання клієнтом початкової інформації з *browser fingerprints*, але ймовірність цього дуже незначна, оскільки вся інформація завжди передається за HTTPS безпечним з'єднанням.

Результати роботи доцільно використовувати для запровадження автентифікації до вебзастосунків від маленького до середнього розмірів.

Надалі необхідно вдосконалити рівень захисту вебзастосунку й автентифікації, а також підходи щодо оцінювання якості розробленої вебпрограми.

Список літератури

1. Радівілова Т.А., Кіріченко Л.О., Тавалбех М.Х., Ільков А.А. Виявлення аномалій в телекомунікаційному трафіку статистичними методами. *Електронне фахове наукове видання "Кібербезпека: освіта, наука, техніка"*. 2021. № 3(11). С.183–194. DOI: 10.28925/2663-4023.2021.11.183194

Сучасний стан наукових досліджень та технологій в промисловості. 2024. №3 (29)
ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

2. Radivilova T. at al. Analysis of Approaches of Monitoring, Intrusion Detection and Identification of Network Attacks. *Proceedings of 2021 IEEE 8th International Conference on Problems of Infocommunications, Science and Technology (PIC S&T)*, Kharkiv, Ukraine. 2021. P. 631–634.DOI: 10.1109/PICST54195.2021.9772226
3. Radivilova T. at al. The Complex Method of Intrusion Detection Based on Anomaly Detection and Misuse Detection. *Proceedings of 2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, Kyiv, Ukraine. 2020. P. 133–137. DOI: 10.1109/DESSERT50317.2020.9125051
4. Радівілова Т.А. та ін. Балансування самоподібного трафіку в мережних системах виявлення вторгнень. *Електронне фахове наукове видання "Кібербезпека: освіта, наука, техніка"*. 2020. № 3(7). С. 17–30. DOI: https://doi.org/10.28925/2663-4023.2020.7.1730
5. Пшеничних С.В., Добринін І.С., Клочкова Д.Ю. Математична модель оптимального вибору засобів захисту інформації при проектуванні комплексної системи захисту на об'єкті інформатизації. *Електронне наукове фахове видання – журнал "Проблеми телекомунікацій"*. 2023. № 1(32). С. 45–58. URL: https://pt.nure.ua/wp-content/uploads/2023/12/123_Pshenychnyh_security_.pdf
6. Добринін І.С., Борова М.П. Оптимізація вибору варіанту побудови системи захисту інформації від атак при антагоністичній грі. *Системи озброєння і військова техніка.* 2018. № 2 (54). С. 89–93. DOI: 10.30748/soivt.2018.54.12
7. Ardi C., Calder M. The Prevalence of Single Sign-On on the Web: Towards the Next Generation of Web Content Measurement. *Proceedings of the 2023 ACM on Internet Measurement Conference.* 2023. DOI: 10.1145/3618257.3624841
8. Shaikh N., Kasat K., Jadhav S. Secured Authentication by Single Sign On (SSO): A Big Picture. *Proceedings of the 2022 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS).* 2022. P. 951–955. DOI: 10.1109/ICCCIS56430.2022.10037708
9. Schmitz G. Privacy-preserving Web single sign-on: Formal security analysis and design. *It-Information Technology*. 2022. № 64.1-2. P. 43–48. DOI: https://doi.org/10.1515/itit-2022-0003
10. Sharma S., Jevitha K. Security Analysis of Oauth 2.0 Implementation. *Proceedings of the 2023 Innovations in Power and Advanced Computing Technologies, i-PACT*, Kuala Lumpur, Malaysia. 2023. P. 1–8. DOI: 10.1109/i-PACT58649.2023.10434479
11. Singh J., Chaudhary N. Oauth 2.0: Architectural design augmentation for mitigation of common security vulnerabilities. *Journal of Information Security and Applications.* 2022. № 65. Article 103091. DOI: https://doi.org/10.1016/j.jisa.2021.103091
12. Al-Husari F. Designating a Leader Browser Tab to Perform Refreshing of Access Token in Oauth 2.0. *Proceedings of the 2023 11th International Scientific Conference on Computer Science, COMSCI 2023*, Sozopol, Bulgaria. 2023. P. 1–4. DOI: 10.1109/COMSCI59259.2023.10315906
13. Shevchuk D., Harasymchuk O., Partyka A., Korshun N. Designing Secured Services for Authentication, Authorization, and Accounting of Users. *CEUR Workshop Proceedings*. 2023 Cybersecurity Providing in Information and Telecommunication Systems II, CPITS-II 2023, Kyiv. 2023. №3550. P. 217–225. URL: https://ceur-ws.org/Vol-3550/short4.pdf
14. Park J., Kim J., Park M., Jung S. A. Study of Oauth 2.0 Risk Notification and Token Revocation from Resource Server. In: Kim, Hw., Choi, D. (eds) *Information Security Applications. WISA 2015. Lecture Notes in Computer Science*, Vol. 9503. Springer, Cham. 2016. DOI: https://doi.org/10.1007/978-3-319-31875-2_23
15. Shaikh N., Kasat K. and Jadhav S. Secured Authentication by Single Sign On (SSO): A Big Picture. *Proceedings of the 2022 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, Greater Noida, India. 2022. P. 951–955. DOI: 10.1109/ICCCIS56430.2022.10037708
16. Al Shabi M., Rashiq R. M. Analyzing Privacy Implications and Security Vulnerabilities in Single Sign-On Systems: A Case Study on OpenID Connect. *International Journal of Advanced Computer Science & Applications*. 2024. №15.4. DOI: 10.14569/IJACSA.2024.0150465
17. Mansur E.S., Rahmatulloh A., Shofa R., Darmawan I. AMAN: Token-based Authentication to Improved Single Sign-On Security Between Systems. *Proceedings of the 2023 International Conference on Advancement in Data Science,*
18. *E-learning and Information System (ICADEIS).* 2023. P. 1–6. DOI:10.1109/ICADEIS58666.2023.10270904
19. Grassi, P.A. at al. NIST SP 800-63B. Digital identity guidelines: Authentication and lifecycle management. *NIST Special Publication (SP).* 2020. DOI: https://doi.org/10.6028/NIST.SP.800-63b
20. Sambit K.D. Ultimate Web Authentication Handbook. Orange Education Pvt Limited, 2023. 340 p. URL: https://github.com/OrangeAVA/Ultimate-Web-Authentication-Handbook
21. RFC 7519. JSON Web Token. 2015. URL: https://tools.ietf.org/html/rfc7519 (дата звернення: 05.08.2024).
22. Alsmadi I. at al. Practical Information Security. *A Competency-Based Education Course.·Springer International Publishing*, 2018. 317 p. DOI: https://doi.org/10.1007/978-3-319-72119-4
23. RFC 6749. OAuth 2.0 Refresh Token. 2020. URL: https://oauth.net/2/grant-types/refresh-token (дата звернення: 01.08.2024).

24. 10 Best E-Commerce Website Builders Compared in 2024. 2024. URL: https://www.websiteplanet.com/blog/best-website-builders-ecommerce-websites/ (дата звернення: 15.07.2024).
25. Single page apps in depth. 2013. URL: http://singlepageappbook.com/goal.html (дата звернення: 03.08.2024).
26. Добринін І. С., Мальцева Н. О. Вдосконалення методики факторного аналізу інформаційних ризиків. *Системи обробки інформації*. 2017. № 3(149). С. 146–150. DOI: 10.30748/soi.2017.149.29

References

1. Radivilova, T.A., Kirichenko, L.O., Tavalbeh, M.H., Ilkov, A.A. (2021), "Detection of Anomalies in Telecommunication Traffic by Statistical Methods" ["Vyyavlennya anomaliy v telekomunikatsiynomu trafiku statystychnymy metodamy"], *Electronic professional scientific publication "Cybersecurity: Education, Science, Technology"*, № 3(11), P. 183–194. DOI: 10.28925/2663-4023.2021.11.183194
2. Radivilova, T. at al. (2021), "Analysis of Approaches of Monitoring, Intrusion Detection and Identification of Network Attacks", *Proceedings of 2021 IEEE 8th International Conference on Problems of Infocommunications, Science and Technology (PIC S&T)*, Kharkiv, Ukraine, P. 631-634. DOI: 10.1109/PICST54195.2021.9772226
3. Radivilova, T. at al. (2020), "The Complex Method of Intrusion Detection Based on Anomaly Detection and Misuse Detection", *Proceedings of 2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, Kyiv, Ukraine, P. 133–137. DOI: 10.1109/DESSERT50317.2020.9125051
4. Radivilova, T.A. at al. (2020), "Balancing self-similar traffic in network intrusion detection systems" ["Balansuvannya samopodibnoho trafiku v merezhnykh systemakh vyyavlennya vtorhnen"], *Electronic professional scientific publication "Cybersecurity: education, science, technology"*, №3 (7), С. 17–30. DOI: https://doi.org/10.28925/2663-4023.2020.7.1730
5. Pshenychnych, S.V., Dobrynin, I.S., Klochkova, D.Yu. (2023), "Mathematical model of the optimal choice of means of information protection when designing a complex system of protection at the object of informatization" ["Matematychna model′ optymal′noho vyboru zasobiv zakhystu informatsiyi pry proektuvanni kompleksnoyi systemy zakhystu na ob″yekti informatyzatsiyi"], *Electronic scientific publication – the journal "Telecommunications Problems"*, № 1(32), С. 45–58. available at: https://pt.nure.ua/wp-content/uploads/2023/12/123_Pshenychnyh_security_.pdf
6. Dobrynin, I.S., Borova, M.P. (2018), "Optimization of the choice of the option of building an information protection system against attacks during an antagonistic game" ["Optymizatsiya vyboru variantu pobudovy systemy zakhystu informatsiyi vid atak pry antahonistychniy hri"], *Weapon systems and military equipment*, № 2 (54), С. 89–93. DOI: 10.30748/soivt.2018.54.12
7. Ardi, C., Calder, M. (2023), "The Prevalence of Single Sign-On on the Web: Towards the Next Generation of Web Content Measurement", *Proceedings of the 2023 ACM on Internet Measurement Conference*. DOI: 10.1145/3618257.3624841
8. Shaikh, N., Kasat, K., Jadhav, S. (2022), "Secured Authentication by Single Sign On (SSO): A Big Picture", *Proceedings of the 2022 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, P. 951–955. DOI: 10.1109/ICCCIS56430.2022.10037708
9. Schmitz, G. (2022), "Privacy-preserving Web single sign-on: Formal security analysis and design", *It-Information Technology*, №64.1-2, P. 43–48. DOI: https://doi.org/10.1515/itit-2022-0003
10. Sharma, S., Jevitha, K. (2023), "Security Analysis of Oauth 2.0 Implementation", *Proceedings of the 2023 Innovations in Power and Advanced Computing Technologies*, i-PACT, Kuala Lumpur, Malaysia, P. 1–8. DOI: 10.1109/i-PACT58649.2023.10434479
11. Singh, J., Chaudhary, N. (2022), "Oauth 2.0: Architectural design augmentation for mitigation of common security vulnerabilities", *Journal of Information Security and Applications*, №65, Article 103091. DOI: https://doi.org/10.1016/j.jisa.2021.103091
12. Al-Husari, F. (2023), "Designating a Leader Browser Tab to Perform Refreshing of Access Token in Oauth 2.0", *Proceedings of the 2023 11th International Scientific Conference on Computer Science*, COMSCI 2023, Sozopol, Bulgaria, P. 1–4. DOI: 10.1109/COMSCI59259.2023.10315906
13. Shevchuk, D., Harasymchuk, O., Partyka, A., Korshun, N. (2023), "Designing Secured Services for Authentication, Authorization, and Accounting of Users", *CEUR Workshop Proceedings*, 2023 Cybersecurity Providing in Information and Telecommunication Systems II, CPITS-II 2023, Kyiv, №3550, P. 217–225. available at: https://ceur-ws.org/Vol-3550/short4.pdf
14. Park, J., Kim, J., Park, M., Jung, S. (2016), "A Study of Oauth 2.0 Risk Notification and Token Revocation from Resource Server". In: Kim, Hw., Choi, D. (eds) *Information Security Applications, WISA 2015. Lecture Notes in Computer Science*, Vol 9503, Springer, Cham. DOI: https://doi.org/10.1007/978-3-319-31875-2_23

ISSN 2522-9818 (print)
Сучасний стан наукових досліджень та технологій в промисловості. 2024. №3 (29) *ISSN 2524-2296 (online)*

15. Shaikh, N., Kasat, K. Jadhav, S. (2022), "Secured Authentication by Single Sign On (SSO): A Big Picture", *Proceedings of the 2022 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, Greater Noida, India, P. 951–955. DOI: 10.1109/ICCCIS56430.2022.10037708
16. Al Shabi, M., Rashiq, R. M. (2024), "Analyzing Privacy Implications and Security Vulnerabilities in Single Sign-On Systems: A Case Study on OpenID Connect", *International Journal of Advanced Computer Science & Applications*, №15.4. DOI: 10.14569/IJACSA.2024.0150465
17. Mansur, E.S., Rahmatulloh, A., Shofa, R., Darmawan, I. (2023), "AMAN: Token-based Authentication to Improved Single Sign-On Security Between Systems", *Proceedings of the 2023 International Conference on Advancement in Data Science, E-learning and Information System (ICADEIS)*, P. 1–6. DOI: 10.1109/ICADEIS58666.2023.10270904
18. Grassi, P.A. at al. (2020), "NIST SP 800-63B. Digital identity guidelines: Authentication and lifecycle management". *NIST Special Publication (SP).* DOI: https://doi.org/10.6028/NIST.SP.800-63b
19. Sambit, K.D. (2023), "Ultimate Web Authentication Handbook", Orange Education Pvt Limited, 340 p. available at: https://github.com/OrangeAVA/Ultimate-Web-Authentication-Handbook.
20. "RFC 7519. JSON Web Token", available at: https://tools.ietf.org/html/rfc7519 (last accessed: 05.08.2024).
21. Alsmadi, I., at al. (2018), "Practical Information Security". *A Competency-Based Education Course,*·Springer *International Publishing*, 317 p. DOI: https://doi.org/10.1007/978-3-319-72119-4
22. "RFC 6749. OAuth 2.0 Refresh Token", available at: https://oauth.net/2/grant-types/refresh-token (last accessed: 01.08.2024).
23. "10 Best E-Commerce Website Builders Compared in 2024", available at: https://www.websiteplanet.com/blog/best-website-builders-ecommerce-websites/ (last accessed: 15.07.2024).
24. "Single page apps in depth", available at: http://singlepageappbook.com/goal.html (last accessed: 03.08.2024).
25. Dobrynin I. S., Maltseva N. O. (2017), "Improving the method of factor analysis of information risks" ["Vdoskonalennya metodyky faktornoho analizu informatsiynykh ryzykiv"], *Information processing systems*, №3(149), С. 146–150. DOI: 10.30748/soi.2017.149.29

Надійшла (Received) 21.08.2024

Відомості про авторів / About the Authors

Радівілова Тамара Анатоліївна – доктор технічних наук, професор, Харківський національний університет радіоелектроніки, професор кафедри інфокомунікаційної інженерії імені В.В. Поповського, Харків, Україна; e-mail: tamara.radivilova@nure.ua; ORCID ID: https://orcid.org/ 0000-0001-5975-0269

Кіріченко Людмила Олегівна – доктор технічних наук, професор, Харківський національний університет радіоелектроніки, професор кафедри штучного інтелекту, Харків, Україна; e-mail: ludmila.kirichenko@gmail.com; ORCID ID: https://orcid.org/ 0000-0002-2780-7993

Пантєлєєв Вадим Олегович – Харківський національний університет радіоелектроніки, аспірант кафедри інфокомунікаційної інженерії імені В.В. Поповського, Харків, Україна; e-mail: vadym.pantelieiev@nure.ua; ORCID ID: https://orcid.org/0009-0008-7824-8782

Мазепа Артем Дмитрович – компанія *Geniox*, фулстек-розробник, Дніпро, Україна; e-mail: yertom500@gmail.com; ORCID ID: 0009-0002-9977-5932

Білодід Володимир Григорович – Харківський національний університет Повітряних Сил ім. І. Кожедуба, науковий співробітник, Харків, Україна; e-mail: vibos111@gmail.com; ORCID ID: https://orcid.org/0009-0002-8976-2310

Radivilova Tamara – Doctor of Sciences (Engineering), Professor, Kharkiv National University of Radio Electronics, Professor at the V.V. Popovskyy Department of Infocommunication Engineering, Kharkiv, Ukraine.

Kirichenko Lyudmyla – Doctor of Sciences (Engineering), Professor, Kharkiv National University of Radio Electronics, Professor at the Department of Artificial Intelligence, Kharkiv, Ukraine.

Pantelieiev Vadym – Kharkiv National University of Radio Electronics, Postgraduate Student at the V.V. Popovskyy Department of Infocommunication Engineering, Kharkiv, Ukraine.

Mazepa Artem – Geniox company, Full Stack Developer, Dnipro, Ukraine.

Bilodid Volodymyr – Kharkiv National University of the Air Force "Chief Marshal of Aviation Ivan Kozhedub", Researcher, Kharkiv, Ukraine.

ISSN 2522-9818 (print)
ISSN 2524-2296 (online) *Innovative technologies and scientific solutions for industries. 2024. No. 3 (29)*

ANALYSIS OF AUTHENTICATION METHODS FOR FULL-STACK APPLICATIONS AND IMPLEMENTATION OF A WEB APPLICATION WITH AN INTEGRATED AUTHENTICATION SYSTEM

The **subject** of research is methods and techniques for secure user authentication in web applications. The **goal** of the work is to analyse authentication methods and implement a web application with an authentication system integrating JWT tokens and the OAuth v2.0 standard. The article solves the **tasks** of analysis of the main protocols and methods of user authentication in web applications, implementation of authentication based on the OAuth 2.0 standard and JWT access/refresh token, and analysis of the risks of vulnerabilities and attacks for the implemented web applications. **Methods** used: comparison, empirical analysis, calculation methods. The next **results** have been obtained: analysed the protocols and methods of user authentication in web applications; selected authentication methods of JWT token and OAuth v2.0 standard for building modern web applications; created a web application based on the selected authentication methods in web applications; analysed the risks of vulnerabilities and attacks in web applications. **Conclusions:** The most well-known authentication methods for web applications are analyzed. It is established that most modern authentication methods have many disadvantages, which leads to increased risks when using these authentication methods. It is shown that one of the most reliable methods of web application user data security is the use of a combination of JWT Access/Refresh token and browser fingerprints. The implementation, configuration, and analysis of this methodology have shown that this combination provides the most reliable prevention of token theft and use from another computer. OAuth 2.0 authentication was also implemented. The study found that delegating authentication to services such as Facebook or Google can provide a low risk of attacks and vulnerabilities for a web application. It is noted that authentication using OAuth 2.0 can be compromised only at the beginning of the connection between the client and the server, or rather when the client first sends initial information from the browser fingerprints. This information is sent over the secure HTTPS (Hypertext Transfer Protocol Secure) protocol, so the risk of compromising OAuth 2.0 authentication is low.

Keywords: authentication; integrated authentication system; OAuth; JWT; token; web application.

Бібліографічні описи / Bibliographic descriptions

Радівілова Т. А., Кіріченко Л. О., Пантєлєєв В. О., Мазепа А. Д., Білодід В. Г. Аналіз методів автентифікації для вебзастосунків та реалізація вебзастосунку з інтегрованою системою автентифікації. *Сучасний стан наукових досліджень та технологій в промисловості*. 2024. № 3 (29). С. 76–90. DOI: https://doi.org/10.30837/2522-9818.2024.3.076

Radivilova, T., Kirichenko, L., Pantelieiev, V., Mazepa, A., Bilodid, V. (2024), "Analysis of authentication methods for full-stack applications and implementation of a web application with an integrated authentication system", *Innovative Technologies and Scientific Solutions for Industries*, No. 3 (29), P. 76–90. DOI: https://doi.org/10.30837/2522-9818.2024.3.076