УДК 004.056.5:004.738.5

М. В. Вербіцький
А. А. Шиян

УДОСКОНАЛЕННЯ МЕТОДУ КОНТРОЛЮ ДОСТУПУ ТА РОЗПОДІЛУ РЕСУРСІВ З УРАХУВАННЯМ ОБМЕЖЕННЯ ТРАФІКУ

Вінницький національний технічний університет

Анотація

У роботі розглянуто удосконалений метод контролю доступу та розподілу ресурсів у веб-додатках. Запропоновано поєднання рольової моделі доступу RBAC, алгоритму Token Bucket, адаптивного коефіцієнта ризику та журналювання подій. Метод передбачає послідовну перевірку автентифікації, ролі користувача, дозволу на дію, активності ресурсу та поточного стану ліміту трафіку. Такий підхід дозволяє розмежувати ситуації, коли користувач не має права на операцію, і коли він має право, але тимчасово перевищив допустиму інтенсивність запитів.

Ключові слова: рольовий доступ, RBAC, Token Bucket, обмеження трафіку, адаптивний ризик, аудит подій, веб-додаток.

Abstract

The paper considers an improved method of access control and resource allocation in web applications. The proposed approach combines role-based access control, the Token Bucket algorithm, an adaptive risk coefficient, and event logging. The method provides a sequential verification of authentication, user role, permission for the requested action, resource activity, and the current state of the traffic limit. This approach makes it possible to distinguish between cases when a user has no permission for an operation and cases when the user has permission but has temporarily exceeded the allowed request intensity.

Keywords: role-based access control, RBAC, Token Bucket, traffic limitation, adaptive risk, event audit, web application.

Вступ

Сучасні веб-додатки та API обслуговують користувачів з різними правами, різними сценаріями роботи та різним рівнем навантаження на сервер. За таких умов класичної перевірки логіна і пароля недостатньо: система повинна визначати, чи має користувач право виконувати конкретну дію, чи активний відповідний ресурс і чи не перевищено допустиму інтенсивність запитів. Це особливо важливо для ресурсів, що мають обмежену пропускну здатність або можуть впливати на доступність сервісу [1, 7].

Рольова модель доступу RBAC є зручною для централізованого керування дозволами, оскільки користувач отримує права через роль, а не через окреме ручне налаштування кожної операції [2]. Водночас RBAC не визначає, скільки разів користувач може виконати дозволену дію за певний проміжок часу. Тому для захисту програмних ресурсів доцільно доповнити рольову перевірку механізмами квотування, обмеження трафіку та журналювання подій.

Метою роботи є обґрунтування удосконаленого методу контролю доступу та розподілу ресурсів, який поєднує RBAC, Token Bucket, адаптивний коефіцієнт ризику і аудит подій. Такий метод орієнтований на веб-додатки, API-шлюзи, навчальні платформи, файлові сервіси та корпоративні системи, де необхідно одночасно контролювати права користувачів і навантаження на ресурси.

Результати дослідження

Удосконалення методу полягає у тому, що рішення про доступ приймається не одним модулем, а послідовністю взаємопов’язаних перевірок. Спочатку виконується автентифікація користувача, потім визначаються його активні ролі, після цього перевіряється наявність дозволу у форматі resource:action, активність ресурсу та стан ліміту трафіку. Результат кожного критичного етапу фіксується в журналі подій, що підвищує контрольованість системи та спрощує подальший аудит [8].

Формально рішення про доступ можна подати як предикат $Access(u, r, a) = Auth(u) \wedge Permission(role(u), r, a) \wedge Limit(u, r, a)$, де u - користувач, r - ресурс, a - дія. Такий запис показує, що позитивне рішення можливе лише тоді, коли користувач успішно автентифікований, його роль має

потрібний дозвіл, а встановлений ліміт використання ресурсу не перевищено. Окремі контекстні умови, наприклад активність ресурсу, IP-адреса, час запиту або ризикова поведінка, можуть розглядатися як спрощені елементи ABAC [3].

Для врахування навантаження запропоновано використовувати ефективну вартість операції c_eff = c_base · k_role · k_resource · k_risk. Базова вартість c_base задається адміністратором залежно від ресурсоємності дії, k_role враховує роль користувача, k_resource описує критичність ресурсу, а k_risk підвищується при частих відмовах 403 або 429. Завдяки цьому дозволена, але потенційно ризикова або ресурсоємна дія може отримувати більшу вартість і швидше вичерпувати доступний ліміт.

Адаптивний коефіцієнт ризику доцільно визначати за формулою k_risk = min(2; 1 + 0,05·N_403 + 0,03·N_429), де N_403 - кількість відмов через нестачу прав, а N_429 - кількість відмов через перевищення ліміту за останній контрольний інтервал. Верхня межа 2 не дозволяє коефіцієнту зростати безконтрольно. Така адаптивність не є складною системою штучного інтелекту, але дозволяє враховувати поведінку користувача під час прийняття рішення.

Для контролю інтенсивності запитів використано алгоритм Token Bucket. Кількість токенів у відрі оновлюється за правилом T(t) = min(C, T(t0) + r·(t - t0)), де C - місткість відра, r - швидкість поповнення, а t - поточний час. Запит дозволяється лише тоді, коли T(t) ≥ c_eff. Якщо токенів недостатньо, сервер повертає відповідь 429 Too Many Requests, а причина відмови зберігається в журналі [4]. Для швидкого зберігання стану лічильників у промисловому варіанті доцільно використовувати Redis, а для навчального прототипу - локальну таблицю лічильників [5].

Архітектура запропонованої системи передбачає клієнтський інтерфейс, API-сервер, модуль автентифікації, RBAC-модуль, модуль ресурсів, лімітер трафіку, базу даних і журнал подій. Основні сутності бази даних охоплюють користувачів, ролі, дозволи, ресурси, правила трафіку, лічильники використання та audit_logs. Для промислового застосування таку структуру можна реалізувати в PostgreSQL, а для локальної демонстрації достатньо SQLite або іншої легкої СУБД [6].

У порівнянні з окремими готовими рішеннями, запропонований підхід має прикладну перевагу: він об’єднує рішення про право доступу і рішення про допустиму інтенсивність використання ресурсу в одній логіці. IAM-системи, наприклад Keycloak, зручні для ідентифікації та ролей, але не завжди враховують вартість конкретної бізнес-операції. Policy engine, такі як Casbin або Oso, дозволяють описувати правила авторизації, однак потребують додаткової інтеграції з механізмом лімітування трафіку [9-11].

Для практичної реалізації методу доцільно використовувати веб-фреймворк FastAPI, оскільки він дозволяє послідовно викликати залежності автентифікації, рольової перевірки і лімітування перед виконанням основної операції [12]. У такій реалізації користувач із роллю user не може виконати адміністративну дію, адміністратор має доступ до політик і журналу, а повторні ресурсоємні запити блокуються після вичерпання токенів.

Таблиця 1 - Узагальнення компонентів удосконаленого методу

Компонент	Основне призначення	Очікуваний результат
RBAC	Перевірка ролей і дозволів у форматі resource:action	Централізоване керування правами користувачів
Token Bucket	Облік токенів, їх поповнення та списання за вартістю операції	Обмеження інтенсивності запитів без повного блокування користувача
Адаптивний ризик	Підвищення ефективної вартості операції за частих відмов 403 і 429	Гнучкіше реагування на підозрілу або надмірну активність
Журналювання	Фіксація відмов автентифікації, заборон доступу, перевищень ліміту та успішних дій	Можливість аудиту, аналізу інцидентів і коригування політик
Адміністрування	Налаштування ролей, ресурсів, вартості операцій і правил трафіку	Кероване оновлення політик без зміни програмного коду

Висновки

У результаті дослідження обґрунтовано доцільність поєднання RBAC, Token Bucket, адаптивного коефіцієнта ризику та журналювання подій у межах єдиного методу контролю доступу. Запропонований підхід дозволяє враховувати не лише факт наявності дозволу, а й поточну можливість виконання дії з урахуванням ліміту трафіку.

Архітектура методу передбачає послідовність перевірок: автентифікація, рольова авторизація, перевірка активності ресурсу, розрахунок ефективної вартості операції, контроль Token Bucket і запис результату в audit_logs. Це дає змогу прозоро відокремити помилки автентифікації, відмови за правами та відмови через перевищення ліміту.

Практичне значення запропонованого рішення полягає у можливості використання такого підходу в корпоративних веб-сервісах, API-шлюзах, файлових сховищах і навчальних платформах. Подальший розвиток може передбачати інтеграцію із зовнішнім провайдером ідентифікації, використання Redis для розподілених лічильників і розширення політик доступу контекстними ABAC-умовами.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. ISO/IEC 27001:2022. Information security, cybersecurity and privacy protection - Information security management systems [Електронний ресурс]. - Режим доступу: https://www.iso.org/standard/27001
2. NIST. Role Based Access Control and Role Based Security [Електронний ресурс]. - Режим доступу: https://csrc.nist.gov/projects/role-based-access-control
3. NIST Special Publication 800-162. Guide to Attribute Based Access Control Definition and Considerations [Електронний ресурс]. - Режим доступу: https://csrc.nist.gov/publications/detail/sp/800-162/final
4. Nottingham M., Fielding R. Additional HTTP Status Codes. RFC 6585. 2012 [Електронний ресурс]. - Режим доступу: https://www.rfc-editor.org/rfc/rfc6585
5. Redis Documentation. Redis data structures and key expiration [Електронний ресурс]. - Режим доступу: https://redis.io/docs/latest/
6. PostgreSQL Documentation. The PostgreSQL Global Development Group [Електронний ресурс]. - Режим доступу: https://www.postgresql.org/docs/
7. OWASP Foundation. API Security Top 10 [Електронний ресурс]. - Режим доступу: https://owasp.org/www-project-api-security/
8. OWASP Foundation. Authorization Cheat Sheet [Електронний ресурс]. - Режим доступу: https://cheatsheetseries.owasp.org/cheatsheets/Authorization_Cheat_Sheet.html
9. Keycloak Documentation. Server Administration Guide [Електронний ресурс]. - Режим доступу: https://www.keycloak.org/documentation
10. Casbin Documentation. Authorization library overview [Електронний ресурс]. - Режим доступу: https://casbin.org/docs/overview
11. Oso Documentation. Authorization as code [Електронний ресурс]. - Режим доступу: https://www.osohq.com/docs
12. FastAPI Documentation. FastAPI framework [Електронний ресурс]. - Режим доступу: https://fastapi.tiangolo.com/

Вербіцький Максим Вікторович - студент групи 2КІТС-22б, факультет менеджменту та інформаційної безпеки, Вінницький національний технічний університет, м. Вінниця, e-mail: mverbitskii@gmail.com.

Шиян Анатолій Антонович - доцент кафедри менеджменту та безпеки інформаційних систем, Вінницький національний технічний університет, м. Вінниця, e-mail: anatoliy.a.shiyan@gmail.com, ORCID: https://orcid.org/0000-0002-5418-1498, Scopus ID: 57219057801; 6603792203.

Verbitskyi Maksym V. - student of group 2KITS-22b, Faculty of Management and Information Security, Vinnytsia National Technical University, Vinnytsia, e-mail: mverbitskii@gmail.com.

Shyian Anatolii A. - Associate Professor, Department of Management and Information Systems Security, Vinnytsia National Technical University, Vinnytsia, e-mail: anatoliy.a.shiyan@gmail.com, ORCID: https://orcid.org/0000-0002-5418-1498, Scopus ID: 57219057801; 6603792203.