ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ЗАХИСТУ ІНФОРМАЦІЇ

КАФЕДРА ІНФОРМАЦІЙНОЇ ТА КІБЕРНЕТИЧНОЇ БЕЗПЕКИ

КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«ТЕХНОЛОГІЇ ЗАСТОСВУВАННЯ МЕТОДІВ АУТЕНТИФІКАЦІЇ ТА АВТОРЗАЦІЇ ДЛЯ ЗАБЕЗПЕЧЕНН БЕЗПЕКИ СУЧАСНИХ ВЕБ ЗАСТОСУНКІВ»

на здобуття освітнього ступеня магістра
зі спеціальності 125 Кібербезпека
(код, найменування спеціальності)
освітньо-професійної програми Інформаційна та кібернетична безпека
(назва)

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело
_____ Анатолій РУДНІК

Виконав: здобувач(ка) вищої освіти групи БСДМ-62
РУДНІК Анатолій
(ПРІЗВИЩЕ, Ім'я)

Керівник: БОРСУКОВСЬКИЙ Юрій
к.т.н, доцент (ПРІЗВИЩЕ, Ім'я)

Рецензент: Туровський О.Л.
(ПРІЗВИЩЕ, Ім'я)

Київ 2024

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ЗАХИСТУ ІНФОРМАЦІЇ

Кафедра Інформаційної та кібернетичної безпеки
Ступінь вищої освіти Магістр
Спеціальність 125 Кібербезпека
Освітньо-професійна програма Інформаційна та кібернетична безпека

ЗАТВЕРДЖУЮ
Завідувач кафедри ІКБ
_____ Галина ГАЙДУР
“___”_____ 2023 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Рудніку Анатолію Андрійовичу
(прізвище, ім’я, по батькові)

1. Тема кваліфікаційної роботи:
«Технології застосування методів аутентифікації та авторизації для забезпечення безпеки сучасних веб застосунків»
керівник кваліфікаційної роботи: БОРСУКОВСЬКИЙ Юрій, к.т.н., доцент,
(ПРІЗВИЩЕ, Ім’я, науковий ступінь, вчене звання)
затверджені наказом Державного університету інформаційно-комунікаційних технологій від «19» жовтня 2023р. №145.

2. Строк подання студентом кваліфікаційної роботи: 15.12.2023 р.

3. Вихідні дані до кваліфікаційної роботи:
рекомендаційні дані;
порівняльна таблиця характеристик методів аутентифікації та авторизації;
наукова та технічна література;
документи, міжнародні стандарти.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Теоретичні аспекти аутентифікації та авторизації;
2. Аналіз сучасних технологій аутентифікації та авторизації;

3. Оцінка ефективності впровадження сучасних технологій аутентифікації та авторизації для забезпечення безпеки бізнесу.

5. Перелік ілюстративного матеріалу:
Презентація PowerPoint

6. Дата видачі завдання 19.10.2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ зп	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Дослідження актуальності проблеми управління привілеями в інформаційній системі організації	19.10.2023 р.	
2.	Аналіз наукової та технічної літератури за темою кваліфікаційної роботи.	22.10.2023 р.	
3.	Аналіз теоретичних матеріалів щодо аутентифікації та авторизації	27.10. 2023р.	
4.	Аналіз сучасних методів та технологій аутентифікації та авторизації	03.11.2023 р.	
5.	Розроблення порівняння сучасних методів, окреслення їх ключових переваг та недоліків	15.11.2023 р.	
6.	Оформлення результатів дослідження. Проходження плагіату	26.11.2023 р.	
7.	Підготовка доповіді до захисту.	15.12.2023 р.	

Здобувач(ка) вищої освіти _____ Анатолій РУДНІК
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Керівник
кваліфікаційної роботи _____ Юрій БОРСУКОВСЬКИЙ
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ЗАХИСТУ ІНФОРМАЦІЇ

ПОДАННЯ
ГОЛОВІ ДЕРЖАВНОЇ ЕКЗАМЕНАЦІЙНОЇ КОМІСІЇ
ЩОДО ЗАХИСТУ КВАЛІФІКАЦІЙНОЇ РОБОТИ
на здобуття освітнього ступеня магістра

Направляється здобувач Руднік А.А. до захисту кваліфікаційної роботи
(прізвище та ініціали)

За спеціальністю 125 Кібербезпека
освітньо-професійної програми Інформаційна та кібернетична безпека
(шифр і назва спеціальності)

на тему: «Технології застосування методів аутентифікації та авторизації для забезпечення безпеки сучасних веб застосунків».

Кваліфікаційна робота і рецензія додаються.

Директор інституту _____ Віталій САВЧЕНКО
(*підпис*) (*Ім'я, ПРІЗВИЩЕ*)

Висновок керівника кваліфікаційної роботи

Здобувач РУДНІК Анатолій обрав тему роботи, метою якої було дослідити зміст технологій аутентифікації та авторизації у сучасних веб застосунках. Перелік використаних джерел свідчить про вміння здобувача розбиратись в наукових питаннях та застосовувати їх при дослідженнях. Під час виконання кваліфікаційної роботи РУДНІК Анатолій показав гарну теоретичну та практичну підготовку, вміння самостійно вирішувати питання і робити висновки. Роботу виконував сумлінно, акуратно та вчасно за планом.

Все це дозволяє оцінити виконану кваліфікаційну роботу здобувача РУДНІК Анатолій на оцінку **«добре»** та присвоїти йому кваліфікацію магістр з кібербезпеки за спеціалізацією інформаційна та кібернетична безпека.

Керівник кваліфікаційної роботи _____ Юрій БОРСУКОВСЬКИЙ
(підпис) *(Ім'я, ПРІЗВИЩЕ)*
"____" _____ 2023 року

Висновок кафедри про кваліфікаційну роботу

Кваліфікаційна робота розглянута. Здобувач(ка) РУДНІК Анатолій. допускається до захисту даної роботи в Державній екзаменаційній комісії.

Завідувач кафедри Інформаційної та кібернетичної безпеки
(назва)

_____ Галина ГАЙДУР
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

ВІДГУК РЕЦЕНЗЕНТА
на кваліфікаційну роботу

здобувача Рудніка Анатолія
на тему: «Технології застосування методів аутентифікації та авторизації для забезпечення безпеки сучасних веб застосунків».

Актуальність:

В наш час веб-застосунки стали фундаментальною частиною нашого щоденного життя. У зв'язку з постійним збільшенням кіберзагроз і залежністю від цих додатків для виконання критично важливих задач, таких як обробка фінансових транзакцій та особисте спілкування, забезпечення безпеки цих застосунків є вирішальним. Надійні стратегії для аутентифікації та авторизації є важливими для захсту цих систем від нелегітимного доступу

Позитивні сторони:

1. Досліджено методи та засоби аутентифікації та авторизації
2. Порівняно ефективність та безпеку різних засобів аутентифікації та авторизації
3. Ідентифіковано проблеми впровадження технологій аутентифікації та авторизації

Недоліки:

1. У кваліфікаційній роботі доцільно було реалізувати повноцінний веб застосунок для демонстративного порівняння ефективності технологій аутентифікації та авторизації

Відзначені зауваження не впливають на загальну позитивну оцінку кваліфікаційної роботи

Висновок: Враховуючи недоліки, кваліфікаційна робота заслуговує оцінку «**добре»**, а здобувач **РУДНІК Анатолій** - присвоєння кваліфікації магістр з кібербезпеки за спеціалізацією інформаційна та кібернетична безпека.

Рецензент: _____ Туровський О.Л.
д.т.н., професор *підпис* *Ім'я, ПРІЗВИЩЕ*

РЕФЕРАТ

Текстова частина кваліфікаційної роботи и на здобуття освітнього ступеня магістра: 67 сторінок, 9 рисунків, 2 таблиці, 47 джерел.

Об'єкт дослідження – процес забезпечення безпеки сучасних веб-застосунків через аутентифікацію та авторизацію.

Предмет дослідження – технології застосування методів аутентифікації та авторизації для забезпечення безпеки сучасних веб-застосунків.

Мета роботи – аналізувати та розробити рекомендації щодо використання різних методів аутентифікації та авторизації у контексті забезпечення безпеки сучасних веб-застосунків.

Методи дослідження – опрацювання літератури, аналіз технічної документації, порівняльний аналіз існуючих рішень, моделювання процесів аутентифікації та авторизації.

У роботі проведено аналіз сучасного стану веб-застосунків та їх потреби в захисті. Визначено важливість та завдання аутентифікації та авторизації для забезпечення безпеки.

Проаналізовано існуючі технології, такі як Множинний фактор аутентифікації (MFA), OAuth, OpenID Connect, SAML, JSON Web Token, HTTP Auth.

Досліджено основні функції, переваги та недоліки кожної технології. На основі цих досліджень розроблено рекомендації щодо застосування цих технологій для забезпечення безпеки веб-застосунків.

Галузь використання – безпека веб-застосунків.

ВЕБ-ЗАСТОСУНКИ, БЕЗПЕКА, АУТЕНТИФІКАЦІЯ, АВТОРИЗАЦІЯ, МНОЖИННИЙ ФАКТОР АУТЕНТИФІКАЦІЇ, OAUTH, OPENID CONNECT, SAML, JSON WEB TOKEN, HTTP AUTH.

ABSTRACT

Text part of the master's qualification work:67 pages, 9 figures, 2 tables, 47 sources.

Object of research - the process of ensuring the security of modern web applications through authentication and authorization.

Subject of research – technologies for applying authentication and authorization methods to ensure the security of modern web applications.

Purpose – to analyze and develop recommendations for the use of various methods of authentication and authorization in the context of ensuring the security of modern web applications.

Research methods – literature review, analysis of technical documentation, comparative analysis of existing solutions, modeling of authentication and authorization processes.

The paper analyzes the current state of web applications and their security needs. The importance and tasks of authentication and authorization for security are determined.

Existing technologies such as Multiple Authentication Factor (MFA), OAuth, OpenID Connect, SAML, JSON Web Token, HTTP Auth are analyzed.

The main functions, advantages, and disadvantages of each technology are investigated. Based on these studies, recommendations for the use of these technologies to ensure the security of web applications have been developed.

The field of application is web application security.

WEB APPLICATIONS, SECURITY, AUTHENTICATION, AUTHORIZATION, MULTIPLE AUTHENTICATION FACTOR, OAUTH, OPENID CONNECT, SAML, JSON WEB TOKEN, HTTP AUTH.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

PCI DSS — Payment Card Industry Data Security Standard — стандарт безпеки даних індустрії платіжних карток;

GDPR — General Data Protection Regulation — загальний регламент захисту персональних даних;

MFA — Multi-Factor Authentication — багатофакторна автентифікація;

JWT — JSON Web Tokens — відкритий стандарт для створення токенів доступу;

GDPR — General Data Protection Regulation — загальний регламент захисту персональних даних;

TOPT — Time-Based One-Time Passwords— одноразові паролі на основі часу;

SAML — Security Assertion Markup Language — Мова розмітки тверджень безпеки;

SSO — Single Sign-On — Служба єдиного входу;

IdP — Identity Providers — Постачальник ідентифікаційних даних;

SP — Service Providers — Постачальник послуг;

ВСТУП

Актуальність дослідження. У сучасному світі веб-додатки стали невід'ємною частиною повсякденного життя. З огляду на зростання кількості кібератак та залежності від цих додатків для важливих завдань, таких як фінансові операції та особисте спілкування, безпека цих систем є критичною. Ефективні методи аутентифікації та авторизації є ключовими для захисту цих систем від несанкціонованого доступу.

У сучасну цифрову епоху веб-додатки стали невід'ємною частиною нашого життя. Від онлайн-банкінгу до соціальних мереж, ці додатки обробляють конфіденційну інформацію користувачів, що робить їхню безпеку головним пріоритетом. Забезпечення конфіденційності, цілісності та доступності даних має вирішальне значення для захисту користувачів від несанкціонованого доступу та потенційних загроз. Метою роботи є вивчення процесу захисту сучасних веб-додатків за допомогою методів аутентифікації та авторизації.

Аутентифікація перевіряє особу користувачів, тоді як авторизація визначає їхні рівні доступу та дозволи в додатку. Ефективно впроваджуючи ці методи, організації можуть зменшити ризик несанкціонованого доступу та захистити дані своїх користувачів.

Важливість цієї теми полягає у зростаючій залежності від веб-додатків для різних видів діяльності, включаючи фінансові транзакції, особисте спілкування та бізнес-операції. З ростом кіберзагроз важливо розуміти і використовувати надійні механізми автентифікації та авторизації, які можуть протистояти новим потенційним загрозам.

Об'єкт дослідження – процес забезпечення безпеки сучасних веб-застосунків через аутентифікацію та авторизацію.

Предмет дослідження – технології застосування методів аутентифікації та авторизації для забезпечення безпеки сучасних веб-застосунків.

Мета роботи – аналізувати та розробити рекомендації щодо використання різних методів аутентифікації та авторизації у контексті забезпечення безпеки сучасних веб-застосунків.

Наукові завдання:

- аналіз існуючих технологій аутентифікації та авторизації;
- оцінка ефективності цих методів у різних веб-додатках;
- розробка вдосконалених рішень для захисту веб-додатків.

Методи дослідження – опрацювання літератури, аналіз технічної документації, порівняльний аналіз існуючих рішень, моделювання процесів аутентифікації та авторизації.

Практичне значення одержаних розроблені рекомендації дозволять організаціям оптимізувати свої системи безпеки веб-додатків, знижуючи ризик кібератак та захищаючи конфіденційність користувацьких даних.

Апробація результатів. Результати даного дослідження доповідались на Всеукраїнській науковій конференції «Актуальні проблеми кібербезпеки».

1 ТЕОРЕТИЧНІ АСПЕКТИ АУТЕНТИФІКАЦІЇ ТА АВТОРИЗАЦІЇ

1.1 Визначення понять «аутентифікація» та «авторизація»

Автентифікація та авторизація є двома фундаментальними поняттями у сфері безпеки, особливо коли мова йде про веб-додатки. Ці поняття відіграють вирішальну роль у забезпеченні того, що тільки авторизовані особи можуть отримати доступ до певних ресурсів або виконувати певні дії в додатку. Хоча ці терміни часто використовуються як взаємозамінні, вони мають різні значення і функції.

Автентифікація це процес перевірки особи або системи, яка намагається отримати доступ до певного ресурсу. По суті, це процес підтвердження того, що користувач є саме тим, за кого себе видає. Методи автентифікації зазвичай передбачають використання облікових даних, таких як імена користувачів, паролі, біометричні дані або криптографічні ключі. З іншого боку, авторизація передбачає надання або відмову в доступі до певних ресурсів або функцій на основі привілеїв автентифікованого користувача. Вона визначає, які дії користувач може виконувати або до яких даних він може отримати доступ у додатку. Механізми авторизації покладаються на заздалегідь визначені правила, політики та списки контролю доступу для впровадження обмежень і забезпечення відповідних рівнів доступу для різних користувачів або груп користувачів [1].

Фахівці та практики з кібербезпеки постійно підкреслюють важливість автентифікації та авторизації для захисту веб-додатків. Вони наголошують на необхідності надійних методів автентифікації для запобігання несанкціонованому доступу та виступають за детальний контроль доступу для забезпечення конфіденційності даних. Ці галузеві перспективи спонукають до впровадження ефективних заходів автентифікації та авторизації. Автентифікація та авторизація відповідають галузевим стандартам і правилам, таким як Стандарт безпеки даних

індустрії платіжних карток (PCI DSS)[2] і Загальний регламент про захист даних (GDPR)[3]. Ці норми вимагають від організацій впроваджувати надійні механізми автентифікації та контролю доступу для захисту конфіденційних даних. Дотримання цих нормативних актів гарантує, що належні заходи автентифікації та авторизації вже впроваджені. Автентифікація та авторизація безпосередньо впливають на загальний рівень безпеки веб-додатків. Впроваджуючи надійні механізми автентифікації, організації можуть гарантувати, що тільки авторизовані особи мають доступ до певних ресурсів. Це зменшує ризик витоку даних, несанкціонованих модифікацій і несанкціонованих дій у додатку, підвищуючи безпеку і захищаючи конфіденційну інформацію.

Таким чином, автентифікація та авторизація є важливими компонентами безпеки веб-додатків. Визнання думок провідних експертів, врахування галузевих перспектив, дотримання найкращих практик, дотримання нормативно-правової бази та розуміння їхнього впливу на безпеку в сукупності підкреслюють важливість автентифікації та авторизації для захисту веб-додатків і забезпечення конфіденційності даних.

1.2 Класифікація методів аутентифікації

Класифікацію методів автентифікації можна краще зрозуміти, якщо розглядати їх у контексті конкретних середовищ або сценаріїв, в яких вони застосовуються [4]. Огляд того, як класифікуються методи автентифікації в різних контекстах наведено нижче:

- Автентифікація на персональних пристроях: На персональних пристроях, таких як смартфони та ноутбуки, поширені такі методи, як біометрія (відбитки пальців, розпізнавання обличчя), пароль/ПІН-код та графічні ключі. Ці методи забезпечують баланс між безпекою та легкістю доступу для користувача;
- Автентифікація в Інтернеті та онлайн-сервісах: Тут широко використовуються парольна автентифікація, багатофакторна автентифікація (MFA) та

методи на основі токенів, такі як OAuth. Вони забезпечують безпечний доступ до онлайн-акаунтів і сервісів;

- Безпека підприємств і мереж: У цьому контексті переважають методи автентифікації на основі сертифікатів, Kerberos [5], LDAP [6] і багатофакторної автентифікації. Вони захищають доступ до корпоративних мереж і конфіденційних ресурсів;
- Фінансові транзакції та онлайн-банкінг: Для захисту фінансових транзакцій і доступу до банківських послуг використовуються такі методи, як OTP (одноразові паролі), біометрична верифікація і системи на основі токенів (наприклад, ті, що використовуються в кредитних і дебетових картах);
- Електронна комерція та роздрібна торгівля: Роздрібні додатки часто використовують паролі та MFA-методи. Крім того, для перевірки транзакцій використовуються методи автентифікації платежів, такі як 3D Secure [7];
- Охорона здоров'я: Через чутливість даних, системи охорони здоров'я часто використовують надійні методи, такі як біометрична автентифікація, смарт-карти та MFA, щоб забезпечити конфіденційність даних пацієнтів;
- Уряд та державний сектор: Тут зазвичай використовуються такі методи, як перевірка електронної ідентифікації, автентифікація документів (для паспортів та ідентифікаційних карток) і цифрові підписи;
- Хмарні сервіси: Для хмарних сховищ і сервісів зазвичай використовуються ключі API, автентифікація на основі токенів (JWT) і MFA для захисту доступу та передачі даних;
- Віддалений доступ: У таких контекстах, як віддалена робота та VPN-доступ, такі методи, як MFA, цифрові сертифікати та безпечні протоколи входу, мають вирішальне значення для захисту віддалених з'єднань;

- IoT та розумні пристрої: В Інтернеті речей такі методи, як автентифікація пристроїв, методи на основі сертифікатів і контроль доступу до мережі, мають важливе значення для захисту взаємопов'язаних пристроїв.

Ця класифікація підкреслює, що вибір методу автентифікації значною мірою залежить від конкретних вимог, моделей загроз і парадигм взаємодії з користувачами в контексті, в якому він використовується. Кожен контекст вимагає унікального балансу між безпекою, зручністю для користувача і ресурсними обмеженнями.

1.3 Принцип роботи основних методів аутентифікації

Автентифікація, наріжний камінь кібербезпеки, це процес, за допомогою якого система перевіряє особу користувача або іншої системи. Розуміння принципу роботи основних методів аутентифікації має вирішальне значення для реалізації ефективних заходів безпеки. Ці методи відрізняються за складністю, рівнями безпеки та програмами, кожен з яких адаптований для задоволення конкретних потреб різних середовищ і сценаріїв. Заглиблюючись у те, як кожен метод перевіряє особу та надає доступ, можливо оцінити тонкощі та нюанси, які визначають їх ефективність у захисті інформації та систем.

Аутентифікація за допомогою токенів є одним з ефективних способів забезпечення безпеки автентифікації (рис. 1.1). Для цього методу використовуються фізичні або цифрові токени, які генерують одноразові паролі (OTP) або криптографічні ключі. Фізичні токени можуть бути у формі апаратних пристроїв, таких як смарт-картки або USB-пристрої. Вони містять вбудований генератор OTP або криптографічний ключ, який використовується для створення унікальних кодів підтвердження. Користувач отримує фізичний токен і під час процесу автентифікації вводить згенерований код або використовує його для розшифрування криптографічного ключа.

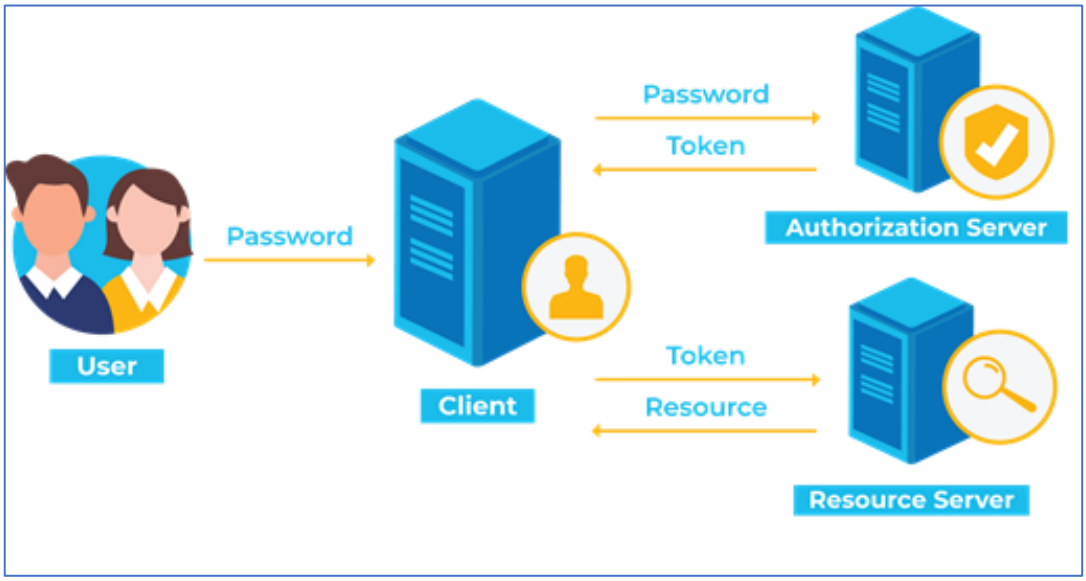


Рисунок 1.1 — Принцип роботи аутентифікації за допомогою токенів

Спочатку користувач реєструє свій токен у системі, що включає передачу аутентифікаційних даних токену на сервер. Після реєстрації при кожній автентифікації токен генерує новий одноразовий пароль або ключ, які використовуються для перевірки користувача. Цей код може бути введений у веб-форму або використаний для розшифрування криптографічного ключа, що порівнюється зі збереженим ключем на сервері [8]. Цифрові токени, такі як мобільні додатки, працюють на тому ж принципі, але не потребують фізичного пристрою. У цьому випадку, програмне забезпечення на мобільному пристрої генерує OTP або криптографічний ключ для автентифікації. Користувач може пред'явити власний мобільний пристрій згенерований код або ключ під час процесу автентифікації. Аутентифікація за допомогою токенів є безпечним методом, оскільки одноразові паролі або криптографічні ключі є унікальними і не можуть бути використані повторно. Цей тип автентифікації використовується у багатьох сферах, включаючи банківські системи, корпоративні мережі та онлайн-сервіси, для забезпечення безпеки доступу до ресурсів та даних.

Автентифікація за допомогою пароля є одним з найпоширеніших і простих методів перевірки особистості користувача. При цьому методі користувач надає унікальний рядок символів (пароль), який порівнюється зі збереженими обліковими даними на сервері.

Процес автентифікації за допомогою пароля включає кілька кроків[9]:

1. Реєстрація пароля: Користувач обирає пароль під час створення облікового запису. Часто вимагаються деякі вимоги до складності пароля, наприклад, мінімальна довжина, наявність букв верхнього і нижнього регістру, цифр або спеціальних символів;

2. Збереження пароля: При реєстрації пароля, зазвичай, його хеш (криптографічний захищений вигляд) зберігається на сервері замість самого пароля. Це забезпечує захист від несанкціонованого доступу до паролів у разі, якщо база даних сервера стає доступною для загроз;

3. Перевірка пароля: Під час процесу автентифікації користувач вводить свій пароль. Сервер обчислює хеш значення введеного пароля і порівнює його зі збереженим хешем у своїй базі даних. Якщо хеші співпадають, користувачу надається доступ;

4. Захист паролів: Для поліпшення безпеки автентифікації за допомогою пароля рекомендується дотримуватися кількох передових практик, таких як використання довгих та складних паролів, унікальних для кожного облікового запису, регулярна зміна паролів, і активування багатофакторної аутентифікації.

Незважаючи на свою поширеність, автентифікація за допомогою пароля має деякі недоліки. Недостатня складність паролів, використання загальнопоширених паролів, і можливість підбору атаками зламу пароля роблять цей метод менш безпечним. Тому рекомендується комбінувати автентифікацію за допомогою пароля з іншими факторами аутентифікації, такими як біометричні дані або використання токенів.

Біометрична автентифікація є передовим методом перевірки особистості користувача, за якого використовуються унікальні фізіологічні або поведінкові характеристики людини.Цей тип автентифікації базується на тому, що кожна людина має унікальні біометричні параметри, які можуть бути використані для підтвердження особи.

Основні види біометричних параметрів, які використовуються в процесі автентифікації, включають(рис. 1.2)[10]:

- Розпізнавання відбитків пальців: Цей метод використовує унікальні лінії і точки на поверхні шкіри пальця, які ідентифікуються і порівнюються з попередньо зареєстрованими шаблонами;
- Розпізнавання обличчя: Цей метод дозволяє ідентифікувати особу за допомогою розпізнавання її обличчя. Використовуються алгоритми, які аналізують геометричні особливості обличчя, такі як розмір очей, форма носа, положення рота і т. д;
- Сканування райдужної оболонки ока: Цей метод вимірює унікальні малі деталі і шаблони на райдужній оболонці ока для ідентифікації особи. Кожний людина має унікальні характеристики райдужки, які використовуються для автентифікації;
- Розпізнавання голосу: Цей метод базується на аналізі унікальних характеристик голосу користувача, таких як тембр, частота, інтонація. Алгоритми порівнюють зразки голосу з попередньо зареєстрованими для встановлення особистості;
- Поведінкова біометрія: Цей метод використовується для аналізу поведінкових характеристик користувача, таких як динаміка натискання клавіш, походження інформації та шаблонів вибору. Алгоритми аналізують ці параметри для перевірки особистості.

Рисунок 1.2 — Основні види біометричної автентифікації

Біометричні дані, що отримуються з цих методів, порівнюються з попередньо зареєстрованими шаблонами, що зберігаються на сервері. Якщо знайдено відповідність, особа отримує доступ до системи або ресурсу.

Багатофакторна автентифікація є методом, що поєднує два або більше фактори автентифікації для забезпечення вищого рівня безпеки (рис. 1.3). Цей підхід використовує комбінацію того, що користувач знає, того, чим користувач володіє, і того, ким він є [11].

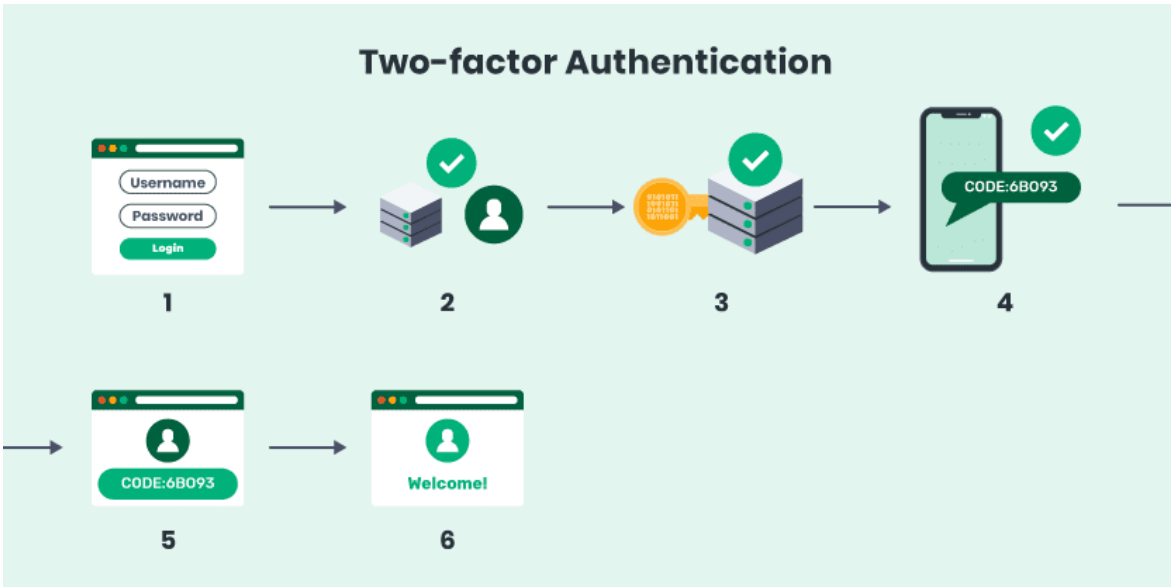


Рисунок 1.3 — Принцип роботи багатофакторної автентифікації

Багатофакторна автентифікація, яка відіграє ключову роль у захисті цифрових систем та даних, базується на використанні декількох незалежних компонентів для верифікації особи користувача. Ці компоненти поділяються на три основні категорії: фактор знання, фактор володіння, та біометричні дані. Перший компонент, фактор знання, передбачає використання інформації, яку знає тільки користувач. Це може бути пароль, пін-код, відповідь на секретне питання безпеки, або інший код, доступ до якого має лише особа, яка підтверджує свою ідентичність.

Другий елемент фактор володіння, вимагає від користувача мати фізичний об'єкт, який підтверджує його особистість. Це може бути фізичний токен, смарт-карта, мобільний телефон, або будь-який інший предмет, що використовується в процесі автентифікації. Третій компонент, біометричні дані, включає унікальні фізіологічні або поведінкові характеристики особи. Це можуть бути відбитки пальців, розпізнавання обличчя, сканування райдужної оболонки ока, або аналіз голосу. Біометричні дані надають високий рівень безпеки, оскільки вони унікальні для кожної особи. Багатофакторна автентифікація забезпечує значно вищий рівень безпеки, оскільки зловмисникам стає набагато складніше отримати доступ до системи. Випадкове викрадення пароля або втрата фізичного токена не дає повного доступу без наявності інших факторів. Цей тип автентифікації широко використовується у банківському секторі, онлайн-сервісах та корпоративних системах для забезпечення високого рівня захисту даних і ресурсів [12].

Також існують менш поширені методи але принцип їх роботи хоча б коротко але варто розглянути. Автентифікація на основі сертифікатів використовує цифрові сертифікати, видані довіреним органом [13]. Користувачі мають приватний ключ і відповідний сертифікат відкритого ключа. Сервер перевіряє особу клієнта, перевіряючи цифровий сертифікат у своєму списку довірених осіб. Цей метод забезпечує безпечний зв'язок і автентифікацію в системах з інфраструктурою відкритих ключів (PKI). Методи автентифікації API спеціально розроблені для автентифікації запитів API (інтерфейс прикладного програмування). До них

відносяться такі методи, як ключі API, OAuth або JSON Web Tokens (JWT). Ці методи забезпечують безпечний доступ до API шляхом перевірки особи програми або користувача, що запитує. Методи автентифікації на основі криптографії використовують криптографічні методи, такі як цифровий підпис або протоколи «виклик-відповідь» для перевірки автентичності користувачів [14]. Ці методи покладаються на математичні алгоритми та ключі для забезпечення безпечного зв'язку та запобігання несанкціонованому доступу.

Це лише кілька прикладів методів автентифікації. Інші категорії включають ідентифікаційну автентифікацію, автентифікацію користувача, автентифікацію веб-додатків, бездротову автентифікацію, автентифікацію баз даних тощо. Кожен метод має свої переваги та особливості, тому важливо вибрати відповідний підхід до автентифікації, виходячи з конкретних вимог безпеки та міркувань зручності для користувача.

1.4 Загальні методи авторизації у веб застосунках

Авторизація у веб-додатках це важливий процес, який визначає, що дозволено робити авторизованому користувачеві. Йдеться про надання або заборону прав на доступ до ресурсів і виконання дій.

Розуміння загальних методів авторизації та принципів їх роботи є важливим для захисту веб-додатків. Проаналізувавши деякі з найпопулярніших видів авторизації а саме [15-17]: контроль доступу на основі ролей (RBAC), контроль доступу на основі атрибутів (ABAC), дискреційний контроль доступу (DAC), обов’язковий контроль доступу (MAC) ,авторизацію на основі токенів, авторизація на основі сеансів, OAuth, OpenID Connect та JWT.

Було створено таблицю 1.1 для подання результатів аналізу в структурованому та зручному вигляді.

Таблиця 1.1.

Загальні методи авторизації

Метод	**Принцип**	**Дія**
RBAC	Користувачі отримують ролі з пов'язаними дозволами.	Система перевіряє дозволи ролі при спробі доступу або дії.
ABAC	Рішення засновані на атрибутах користувача, ресурсу та умовах.	Оцінювання атрибутів в реальному часі для дозволу або заборони доступу.
DAC	Власник ресурсу визначає, хто може отримати доступ і з якими дозволами.	Доступ на основі списку контролю доступу, встановлених власником ресурсу.
MAC	Система визначає політику доступу, засновану на класифікації безпеки.	Контроль доступу заснований на класифікаціях користувача і даних.
Токен	Авторизація через токен, отриманий після автентифікації.	Токен пред'являється при кожному запиті для перевірки дозволу.
OAuth	Дозволяє надавати доступ третім особам до ресурсів без передачі облікових даних.	Надання токенів стороннім додаткам для авторизації.
OpenID Connect	Автентифікаційний рівень поверх OAuth 2.0.	Видача ID токенів після автентифікації для перевірки особи.
Сеанси	Ідентифікаційні дані та дозволи зберігаються в сеансі.	Перевірка сеансової інформації при кожному запиті на дозвіл.
JWT	Компактний спосіб представлення вимог, що передаються між сторонами.	Використання генерованого JWT для авторизації при кожному запиті.

Кожен з цих методів має свої варіанти використання, переваги та обмеження. Вибір того чи іншого методу залежить від конкретних вимог веб-додатку, таких як необхідний рівень безпеки, складність користувацької бази та характер ресурсів, що

захищаються. Належна реалізація цих методів авторизації має вирішальне значення для підтримки безпеки і цілісності веб-додатків.

висновки до розділу 1

У першому розділі було встановлено фундаментальне розуміння «автентифікації» як процесу перевірки особи користувача та «авторизації» як процесу надання або відмови у наданні прав користувачеві на доступ та виконання дій у системі. Ключовим аспектом стала класифікація методів автентифікації, де було проаналізовано різноманіття цих методів від базових, заснованих на паролях, до сучасних біометричних систем. Це підкреслює багатогранність підходів до підтвердження особи користувача та важливість вибору відповідного методу в залежності від контексту застосування та необхідного рівня безпеки.

Далі було розглянуто принципи роботи базових методів автентифікації. Аналіз цих принципів є критично важливим для розуміння того, як саме ці системи захищають цифрові ідентичності та дозволяють нам визначати потенційні вразливості та напрямки для їх удосконалення.

В кінці розділу було описано поширені методи авторизації, які використовуються у веб-додатках. Тут було розглянуто не тільки технічні деталі застосування таких методів, як RBAC, ABAC та авторизація на основі токенів, але й їхні переваги та обмеження. Це дозволило створити повне уявлення про те, як методи авторизації впроваджуються у веб-додатках та як вони впливають на загальну безпеку та функціональність систем. Загалом, розділ надає розуміння концепцій автентифікації та авторизації, які є важливими для захисту цифрових даних та ресурсів. Було розглянуто не тільки класифікацію та принципи роботи різних методів автентифікації, але й зроблено аналіз поширених методів авторизації, що є невід’ємною частиною управління контролем доступу у веб-додатках.

2 АНАЛІЗ СУЧАСНИХ ТЕХНОЛОГІЙ АУТЕНТИФІКАЦІЇ ТА АВТОРИЗАЦІЇ

2.1 Множиний фактор аутентифікації (MFA)

Множиний фактор аутентифікації або багатофакторна автентифікація (MFA) це механізм безпеки, який вимагає від користувачів надання двох або більше факторів перевірки для отримання доступу до ресурсу, наприклад, додатку, онлайн-акаунту або VPN. Це набагато безпечніше, ніж однофакторна, оскільки вимагає поєднання того, що користувач знає (наприклад, пароль), того, що у нього є (наприклад, смартфон), або того, ким він є (наприклад, відбитки пальців). Коли користувач намагається отримати доступ до послуги, йому спочатку пропонується ввести пароль (щось, що він знає). Потім може знадобитися ввести код, надісланий на телефон (те, що він має), або скористатися сканером відбитків пальців (те, що він має)[18].

Розглянемо детальніше програмну реалізацію цього методу аутентифікації на мові програмування Java Script. Для початку створюємо дві форми вводу даних, форму входу (loginForm) та форму верифікації (verificationForm):

```
<form id="loginForm">
  <input type="text" id="username" placeholder="Username">
  <input type="password" id="password" placeholder="Password">
  <button type="submit">Login</button>
</form>
<form id="verificationForm" style="display:none;">
  <input type="text" id="verificationCode" placeholder="Verification Code">
  <button type="submit">Verify</button>
</form>
```

Це стандартні форми які використовуються у веб-орієнтованих системах. Після чого потрібно створити функції які будуть перевіряти уведенні користувачем дані з даними які вже є в системи. Прив’яжемо функцію перевірки даних користувача на натиск кнопки Verify:

```
document.getElementById('loginForm').addEventListener('submit',  function(event)
{
  event.preventDefault();
```

```
  document.getElementById('verificationForm').style.display = 'block';
});
```

Тут буде оброблятись перевірка даних на валідність та відповідність даним у системі після чого буде генеруватись ключ, який потрібно буде ввести у форму verificationForm яка появиться після того як користувач натисне кнопку Verify. Найпопулярнішими методами генерації та надсилання кодів підтвердження для MFA зазвичай є одноразові паролі на основі часу (TOTP)[19] та коди підтвердження на основі SMS-повідомлень[20]. Розглянемо ці методи детальніше.

TOTP являє собою алгоритм, який генерує одноразовий пароль на основі поточного часу та спільного секретного ключа. Він широко використовується в програмах-автентифікаторах, таких як Google Authenticator, Microsoft Authenticator тощо[21]. Пароль змінюється через заданий інтервал (зазвичай 30 секунд). Вважається більш безпечним, ніж SMS-коди, оскільки не залежить від каналів зв’язку, які можуть бути перехоплені. Для реалізації TOTP потрібно використати бібліотеку speakeasy[22] в Node.js:

```
const speakeasy = require('speakeasy');
const secret = speakeasy.generateSecret({length: 20});
const token = speakeasy.totp({
  secret: secret.base32,
  encoding: 'base32'
});
```

Тут використовується метод base32 щоб закодувати бінарні дані в текстовий формат, який складається з 32 можливих символів (літери A-Z та цифри 2-7). Процес кодування base32 дозволяє представити цей бінарний ключ у вигляді текстового рядка з використанням лише 32 символів[23].

Щодо SMS-верифікації, то цей принцип надсилає код на мобільний телефон користувача за допомогою текстового повідомлення. Це поширений метод, який використовується в різних сервісах, оскільки він простий в реалізації і зручний для користувача. Однак він менш безпечний у порівнянні з TOTP, оскільки SMS-повідомлення можуть бути перехоплені або перенаправлені. Для надсилання SMS

часто використовують такі сервіси, як Twilio. Нижче наведено приклад з використанням API Twilio[24] в Node.js:

```
const twilio = require('twilio');
const accountSid = 'YOUR_TWILIO_ACCOUNT_SID'; // Twilio Account SID
const authToken = 'YOUR_TWILIO_AUTH_TOKEN'; // Twilio Auth Token
const client = new twilio(accountSid, authToken);
const verificationCode = Math.floor(100000 + Math.random() * 900000);
client.messages.create({
  body: `Your verification code is: ${verificationCode}`,
  to: '+1234567890',  // User's phone number
  from: '+10987654321' // Twilio phone number
})
.then(message => console.log(message.sid));
```

В цьому коді спочатку іде з’єднання з сервісом twilio вказуючи необхідні accountSid та authToken. Створюємо нового клієнта const client, та генеруємо 6-значний код верифікації verificationCode. Використовуючи метод create() клієнта Twilio, надсилаємо повідомлення SMS з кодом верифікації на вказаний номер телефону. Після створення будь якого вище перерахованих методів, користувачу необхідно ввести код підтвердження у форму verificationForm, після чого форма перевірить валідність коду та пропустить користувача далі або повідомить про помилку.

Після аналізу програмної реалізації та опису важливості багатофакторної автентифікації (MFA), необхідно окреслити переваги та недоліки даного методу. MFA є критично важливим заходом безпеки, вимагаючи кількох форм перевірки, тим самим зменшуючи ризик несанкціонованого доступу та підвищуючи загальну безпеку системи. Однак важливо враховувати як переваги, так і потенційні недоліки впровадження MFA.

Основними перевагами методу можна вважати:

- MFA забезпечує багаторівневий захист, що ускладнює несанкціонований доступ до системи;
- Вимагаючи декількох форм перевірки, MFA може зменшити ймовірність шахрайських дій;

- Багато нормативних баз вимагають MFA, допомагаючи організаціям відповідати вимогам відповідності;
- Підвищує довіру користувачів до системи, демонструючи прихильність до безпеки.

Недоліки:

Впровадження може бути складним і вимагати додаткового обладнання або програмного забезпечення;

Деякі користувачі можуть вважати методи MFA громіздкими, особливо якщо їм часто потрібно отримувати доступ до системи;

Якщо користувачі втрачають доступ до свого другого фактору, наприклад, мобільного пристрою, відновлення може бути складним і тривалим;

Хоча MFA значно покращує безпеку, він не є надійним і може бути вразливим до соціальної інженерії або складних фішингових атак.

2.2 OAuth

OAuth (Open Authorization) відкритий стандарт для автентифікації та авторизації на основі токенів в Інтернеті. Він дозволяє стороннім сервісам, таким як Facebook, Google або Microsoft, використовувати дані облікового запису кінцевого користувача без розкриття пароля користувача. OAuth зазвичай використовується як спосіб, за допомогою якого інтернет-користувачі надають веб-сайтам або додаткам доступ до своєї інформації на інших веб-сайтах, не повідомляючи їм паролі.

Процес починається з того, що програма (відома як клієнт) запитує у користувача дозвіл на доступ до його ресурсів (наприклад, інформації профілю, електронної пошти тощо), розміщених на сервері ресурсів. Якщо користувач надає дозвіл, програма отримує токен від сервера власника ресурсу. Додаток використовує цей токен для доступу до сервера ресурсів і отримання необхідної інформації [25].

Умовно це можна зобразити як ситуацію коли власник машини (користувач) передає ключі паркувальнику (системі) (рис. 2.1). Токени, як правило, обмежені в часі та обсязі. Після закінчення терміну дії токена програма може використовувати токен поновлення (якщо він передбачений) для отримання нового токена доступу.

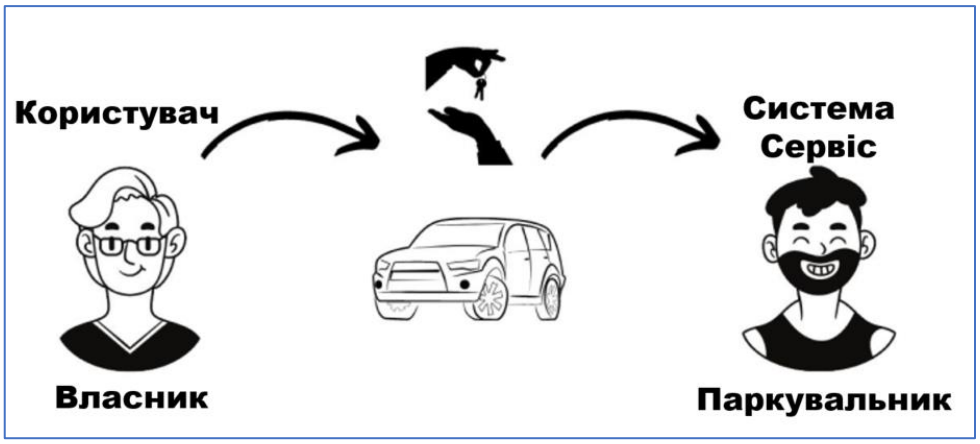


Рисунок 2.1 — Ілюстрація принципу роботи OAuth

Програмна реалізація OAuth може бути такою [26]. Спочатку користувач перенаправляється на сторінку авторизації постачальника послуг:

```
const express = require('express');
const app = express();
const port = 3000;
app.get('/login', (req, res) => {
  const authorizationUrl = 'https://example.com/oauth/authorize';
  const clientId = 'YOUR_CLIENT_ID';
  const redirectUri = 'http://localhost:3000/callback';
  const responseType = 'code';
  const authUrl =
`${authorizationUrl}?response_type=${responseType}&client_id=${clientId}&redirect_uri
=${redirectUri}`;
  res.redirect(authUrl);
});
```

Для початку в коді використовуємо бібліотеку Express, вона дозволяє створити веб-сервер та обробляти HTTP-запити і відповіді. Коли клієнт робить GET-запит на URL /login, сервер надсилає перенаправлення на сторінку авторизації OAuth з вказаними параметрами response_type, client_id та redirect_uri. Після того, як користувач надав дозвіл, він буде перенаправлений назад до системи з кодом

авторизації. Код авторизації обмінюється на токен доступу за допомогою запиту до кінцевої точки токена:

```
const axios = require('axios');
app.get('/callback', async (req, res) => {
  const authorizationCode = req.query.code;
  const tokenUrl = 'https://example.com/oauth/token';
  const clientId = 'YOUR_CLIENT_ID';
  const clientSecret = 'YOUR_CLIENT_SECRET';
  const redirectUri = 'http://localhost:3000/callback';
  try {
    const response = await axios.post(tokenUrl, {
      code: authorizationCode,
      redirect_uri: redirectUri,
      client_id: clientId,
      client_secret: clientSecret,
      grant_type: 'authorization_code'
    });
    const accessToken = response.data.access_token;
    // Use the access token to access the resource server
  } catch (error) {
    // Handle errors
  }
});
```

Тут використовується Axios. Бібліотека Axios дозволяє виконувати HTTP-запити до зовнішніх ресурсів, таких як API. Спочатку в коді визначається маршрут /callback, який обробляє GET-запити. Цей маршрут прив'язаний до URL-шляху /callback. Коли клієнт робить GET-запит на цей URL-шлях, виконується функція-обробник запиту. Потім отримується значення параметра code з запиту, який клієнт відправляє після авторизації користувача на сторінці авторизації OAuth. Далі встановлюється URL для отримання токена доступу. Встановлюються значення clientId, clientSecret та redirectUri. І виконується POST-запит з використанням Axios до URL для отримання токена доступу. І у відповіді сервера отримується токен доступу accessToken.

Після розгляду потужного та поширеного стандарту авторизації OAuth, який забезпечує безпечний і делегований доступ до ресурсів користувача без розголошення паролів, наступним потрібно дослідити переваги та недоліки використання OAuth. Хоча OAuth пропонує численні переваги, важливо враховувати як його переваги, так і потенційні недоліки.

Переваги OAuth:

- Дозволяє користувачам надавати доступ третім особам, не розкриваючи облікові дані користувача;
- OAuth може підтримувати велику кількість користувачів без прямого зв'язку між користувачем, постачальником послуг і споживачем;
- Як широко прийнятий стандарт, OAuth надає розробникам безпечний і спрощений метод реалізації контролю доступу.

Недоліки OAuth:

- Потік OAuth може бути складним для правильної реалізації, що потенційно може призвести до вразливостей у безпеці, якщо не зробити це належним чином;
- Якщо користувачі не будуть обережними, OAuth може бути використаний для фішингу, обманом змушуючи користувачів надавати шкідливим програмам доступ до своїх даних;
- Токени доступу, якщо їх перехопити, можуть дозволити зловмисникам отримати доступ до даних користувача.

2.3 OpenID Connect

OpenID Connect рівень автентифікації, побудований на основі протоколу OAuth 2.0 [27]. Він дозволяє клієнтам перевіряти особу кінцевого користувача та отримувати основну інформацію про профіль у сумісному та REST-подібному режимі. У той час як OAuth 2.0 зосереджений на делегуванні доступу (надання доступу до вашої інформації без розкриття вашої особистості), OpenID Connect використовується для аутентифікації [28].

На офіційному сайті OpenID представлено набір протоколів OpenID Connect, структурований за трьома рівнями складності впровадження (рис. 2.2).

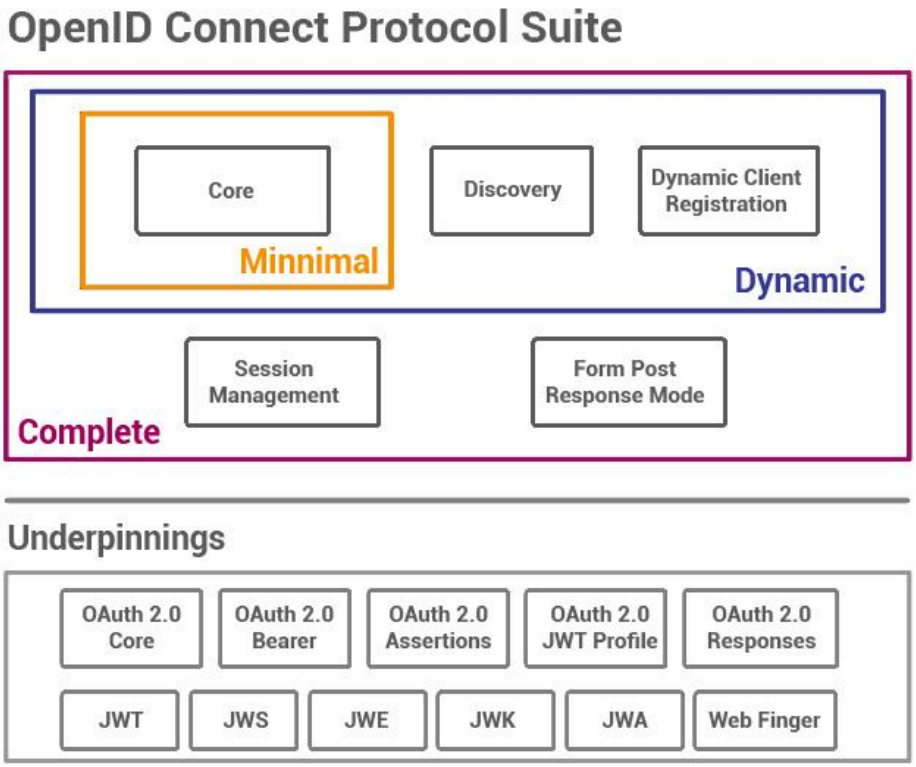


Рисунок 2.2 — Набір протоколів OpenID Connect

Набір протоколів OpenID Connect включає, мінімальний який містить ядро. Ядро представляє важливу частину протоколу OpenID Connect, який включає основні потоки автентифікації та механізми для безпечного входу користувачів і перевірки їх особи. Динамічний що містить в собі виявлення, цей компонент дозволяє клієнтам динамічно знаходити інформацію про постачальника OpenID, наприклад URL-адреси для автентифікації, маркер, інформацію про користувача тощо. Динамічна реєстрація клієнта, дозволяє клієнтам динамічно реєструватися в постачальника OpenID, отримуючи облікові дані клієнта та інші деталі конфігурації, необхідні для участі клієнта в OpenID Connect.

Завершення включає: керування сеансами що описує, як керувати сеансами користувачів, включаючи вхід, вихід із системи та зміни сеансів. режим повідомлення форми який визначає, як дані можуть бути повернуті клієнту через повідомлення форми HTML, метод для надсилання відповіді автентифікації назад клієнту.

Основи, базові технології та специфікації, на які спирається OpenID Connect:

- OAuth 2.0 Core: базовий протокол, який OpenID Connect використовує для авторизації;
- Носій OAuth 2.0: специфікація використання маркерів носія в запитах HTTP для доступу до ресурсів, захищених OAuth 2.0;
- Твердження OAuth 2.0: докладно описано, як твердження можна використовувати як надання авторизації;
- Профіль OAuth 2.0 JWT: визначає, як веб-токени JSON (JWT) можна використовувати як засіб представлення заяв;
- Відповіді OAuth 2.0: описує форматування та обробку відповідей JWT і пов'язані технології;
- JWT (веб-токен JSON): компактний безпечний для URL-адрес спосіб представлення претензій між двома сторонами;
- JWS (JSON Web Signature): стандарт для підпису довільних даних;
- JWE (JSON Web Encryption): надає стандарт для шифрування даних JSON;
- JWK (веб-ключ JSON): стандартний спосіб представлення криптографічних ключів у структурі JSON;
- JWA (веб-алгоритми JSON): визначає алгоритми для підписання та шифрування токенів JWT;
- Web Finger: протокол для виявлення інформації про сутності, визначені URI.

Зображення містить концептуальний огляд набору протоколів OpenID Connect та його основи, підкреслюючи багаторівневий підхід до побудови безпечної, надійної системи перевірки особи в Інтернеті.

Щодо програмної реалізації цього методу, то він відбувається в декілька кроків. Крок 1: Перенаправлення користувача для аутентифікації:

```
const express = require('express');
const app = express();
```

```
const port = 3000;
app.get('/login', (req, res) => {
  const authorizationUrl = 'https://openidprovider.com/auth';
  const clientId = 'YOUR_CLIENT_ID';
  const redirectUri = 'http://localhost:3000/callback';
  const responseType = 'code';
  const scope = 'openid profile email';
  const authUrl =
`${authorizationUrl}?response_type=${responseType}&client_id=${clientId}&redirect_uri
=${redirectUri}&scope=${scope}`;
  res.redirect(authUrl);
});
```

В цьому кроці користувач перенаправляється на URL постачальника OpenID (authorizationUrl) для початку процесу аутентифікації. Створюється URL з параметрами, такими як response_type, client_id, redirect_uri і scope, і виконується направлення користувача за допомогою res.redirect.

Крок 2: Обробка зворотного виклику:

```
app.get('/callback', (req, res) => {
  const authorizationCode = req.query.code;});
```

Отримуємо код авторизації (authorizationCode), який постачальник OpenID надсилає у відповідь на запит аутентифікації.

Крок 3: Обмін коду авторизації на токени:

```
const axios = require('axios');
app.get('/callback', async (req, res) => {
  const authorizationCode = req.query.code;
  const tokenUrl = 'https://openidprovider.com/token';
  const clientId = 'YOUR_CLIENT_ID';
  const clientSecret = 'YOUR_CLIENT_SECRET';
  const redirectUri = 'http://localhost:3000/callback';
  try {
    const response = await axios.post(tokenUrl, {
      code: authorizationCode,
      redirect_uri: redirectUri,
      client_id: clientId,
      client_secret: clientSecret,
      grant_type: 'authorization_code'
    });
    const idToken = response.data.id_token;
    const accessToken = response.data.access_token;
    // Validate idToken and use the information
  } catch (error) {
    // Handle errors
  }
});
```

Тут обмінюється код авторизації на id_token і access_token. Виконується POST-запит до tokenUrl з використанням Axios, передаючи необхідні параметри. Отримані токени зберігаються у змінних idToken і accessToken.

Крок 4: Перевірка ID Token. Цей крок включає перевірку та валідацію id_token. Зазвичай потрібно використовувати бібліотеку для декодування і перевірки JWT-токена, щоб переконатись, що він є законним і не простроченим.

Крок 5: Отримання додаткової інформації про користувача (необов'язково):

```
app.get('/user-info', async (req, res) => {
  const userInfoUrl = 'https://openidprovider.com/userinfo';
  const accessToken = 'YOUR_ACCESS_TOKEN';
  try {
    const response = await axios.get(userInfoUrl, {
      headers: { Authorization: `Bearer ${accessToken}` }
    });
    const userInfo = response.data;
  } catch (error) {
  }
});
```

У цьому кроці використовується access_token для отримання додаткової інформації про користувача з точки доступу UserInfo. Виконується GET-запит до userInfoUrl, передаючи access_token у заголовку авторизації. Отримана інформація про користувача зберігається у змінній userInfo.

Спираючись на обговорення OpenID Connect і його ролі в реалізації автентифікації користувача з достовірністю OAuth 2.0 для авторизації, наступним необхідно описати переваги та недоліки використання OpenID Connect.

Переваги:

- OIDC дозволяє здійснювати надійну та стандартизовану перевірку особи за допомогою токенів;
- Полегшує SSO, покращуючи взаємодію з користувачем, зменшуючи кількість разів, коли йому потрібно входити в систему;
- Як стандартизований протокол, OIDC працює на різних платформах і сервісах, спрощуючи процеси автентифікації для розробників і користувачів.

Недоліки:

- Впровадження OIDC може бути складним і вимагати глибокого розуміння протоколу для забезпечення безпеки;
- Оскільки OIDC залежить від токенів, безпека системи значною мірою залежить від захисту цих токенів;
- Неправильна реалізація може призвести до вразливостей, ризикуючи витоком інформації користувача.

2.4 SAML

Мова розмітки тверджень безпеки (SAML) є важливим стандартом для забезпечення безпеки та зручності в області ідентифікації та авторизації в мережевому середовищі. Вона дозволяє постачальникам ідентифікаційних даних (IdP) ефективно передавати авторизаційну інформацію постачальникам послуг (SP), спрощуючи таким чином процес входу та доступу користувачів до різних додатків і ресурсів . SAML особливо корисний у сфері служб єдиного входу (SSO), де він дозволяє користувачам використовувати один обліковий запис для зручного та безпечного доступу до багатьох різноманітних додатків і сервісів (рис. 2.3), спрощуючи процес управління обліковими даними та забезпечуючи більшу безпеку у використанні онлайн-ресурсів [29].

Алгоритм роботи SAML включає 5 етапів [30]:

1. Запит на доступ користувача: Коли користувач намагається отримати доступ до додатку постачальника послуг, він перенаправляється до постачальника ідентифікаційних даних SAML для автентифікації;

2. Автентифікація: Постачальник автентифікації перевіряє облікові дані користувача і генерує твердження SAML;

3. Передача твердження: IdP надсилає твердження SAML назад до SP;

4. Перевірка постачальника послуг: Постачальник послуг перевіряє твердження SAML і надає доступ користувачеві;

5. Надання доступу користувачеві: Користувач отримує доступ до додатку SP без необхідності надавати облікові дані, специфічні для цього додатку.

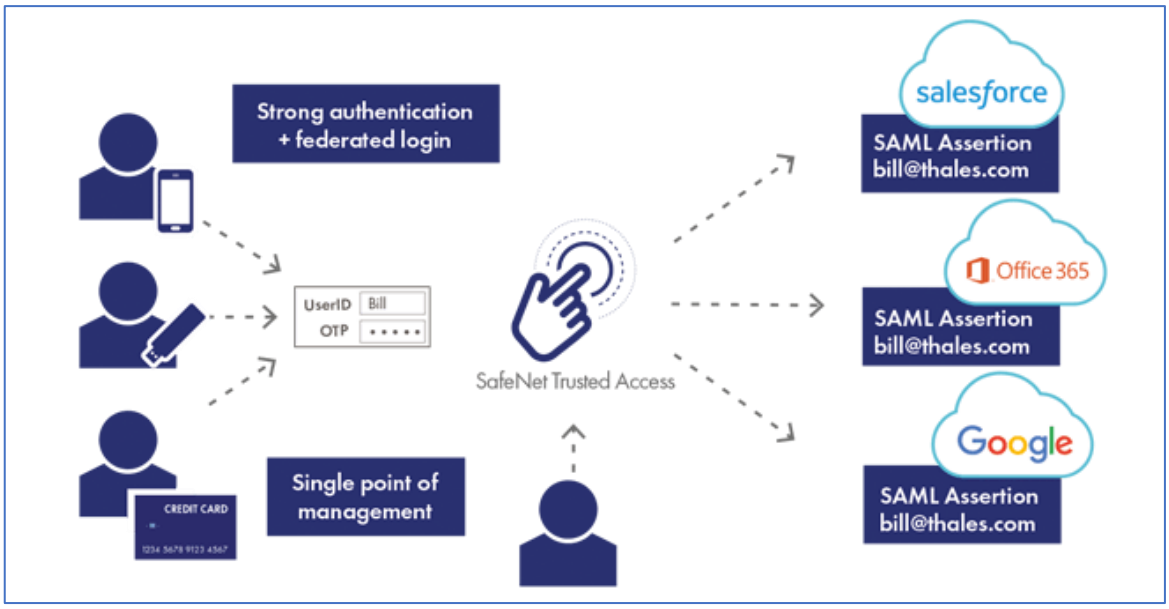


Рисунок 2.3 — Принцип роботи SAML

Також метод SAML містить основні компоненти:

Ствердження: XML-документи, які стверджують щось про користувача (наприклад, ім'я, адресу електронної пошти);

Протокол: Набір правил, які пояснюють, як елементи SAML (наприклад, твердження) упаковуються в SAML-повідомлення;

Прив’язка: Метод, який використовується для обміну SAML-повідомленнями між IdP та SP (наприклад, HTTP POST).

Програмна реалізація SAML на JavaScript зазвичай передбачає використання бібліотек, таких як passport-saml [31] та відбувається в декілька кроків[32].

Крок 1: Налаштування Passport-SAML:

```
const passport = require('passport');
```

```
const SamlStrategy = require('passport-saml').Strategy;
passport.use(new SamlStrategy(
  {
    path: '/login/callback',
    entryPoint: 'https://idp.example.com/saml/login',
    issuer: 'https://sp.example.com/',
    cert: 'IdP public certificate goes here'
  },
  function(profile, done) {
    done(null, profile);
  }
));
```

У цьому кроці налаштовується Passport-SAML, використовуючи стратегію SamlStrategy. Передаються параметри конфігурації, такі як шлях (path), точку входу (entryPoint), видавця (issuer) та сертифікат постачальника SAML (cert). Ці параметри визначають налаштування для з'єднання з постачальником SAML.

Крок 2: Аутентифікація запитів:

```
app.get('/login',
  passport.authenticate('saml', { failureRedirect: '/', failureFlash: true }),
  function(req, res) {
    res.redirect('/');
  }
);
app.post('/login/callback',
  passport.authenticate('saml', { failureRedirect: '/', failureFlash: true }),
  function(req, res) {
    res.redirect('/app');
  });
```

В цьому кроці обробляються маршрути запитів для аутентифікації. При GET-запиті /login використовується Passport-SAML для аутентифікації користувача, перенаправляючи його на URL постачальника SAML. При POST-запиті /login/callback також Passport-SAML для обробки відповіді від постачальника SAML. У випадку успішної аутентифікації, користувач перенаправляється на /app, а в разі невдачі - на /.

Крок 3: Обробка виходу:

```
app.get('/logout', (req, res) => {
});
```

Тут обробляється маршрут /logout для виходу користувача. Для реалізації протоколу SAML виходу, потрібно налаштувати відправку запиту на вихід з даними SAML постачальнику. Сумуючи вище описане можна навести наступні висновки

щодо SAML. Формат XML для передачі інформації про користувача і є ключовим компонентом стандарту SAML. IdP відповідає за перевірку особи користувача, а потім надає твердження SP, де SP довіряє IdP для автентифікації користувачів. SAML значною мірою базується на XML для визначення формату тверджень. Для захисту тверджень SAML використовуються цифрові підписи та шифрування. SAML зазвичай використовується для реалізації SSO, коли користувач входить в систему один раз і отримує доступ до декількох додатків.

На практиці SAML є складним і вимагає ретельного впровадження для забезпечення безпеки. У веб-розробці, особливо на стороні клієнта за допомогою JavaScript, безпосередня робота з твердженнями SAML зустрічається рідко, оскільки через делікатний характер процесу автентифікації вона часто виконується внутрішніми службами або додатками на стороні сервера. Хоча SAML відіграє вирішальну роль у процесі автентифікації, особливо на стороні сервера, важливо враховувати як переваги, які він пропонує, так і будь-які недоліки.

Переваги SAML:

- Використовує формати даних на основі XML для безпечного шифрування та підпису токенів;
- SAML підтримується багатьма постачальниками послуг корпоративного рівня і дозволяє інтегруватися з різними постачальниками ідентифікаційних даних;
- SAML полегшує SSO, покращуючи взаємодію з користувачем, дозволяючи використовувати один набір облікових даних для декількох додатків.

Недоліки SAML:

- SAML є складним у впровадженні і вимагає глибокого знання XML та розуміння концепцій безпеки;
- оскільки SAML базується на XML, він менш придатний для мобільних додатків порівняно з протоколами на основі JSON, такими як OAuth;

- Багатослівність SAML призводить до збільшення розмірів запитів і відповідей, що може вплинути на продуктивність.

2.5 JSON Web Token (JWT)

JSON Web Token (JWT) компактний, безпечний для URL засіб представлення вимог, що передаються між двома сторонами. JWT використовуються для автентифікації та обміну інформацією, де їх компактний розмір полегшує передачу через HTTP [33]. Вони є самодостатніми, тобто мають всю необхідну інформацію про користувача, уникаючи необхідності запитувати базу даних більше одного разу. Принцип роботи JWT можна зобразити потоком аутентифікації (рис. 2.4).

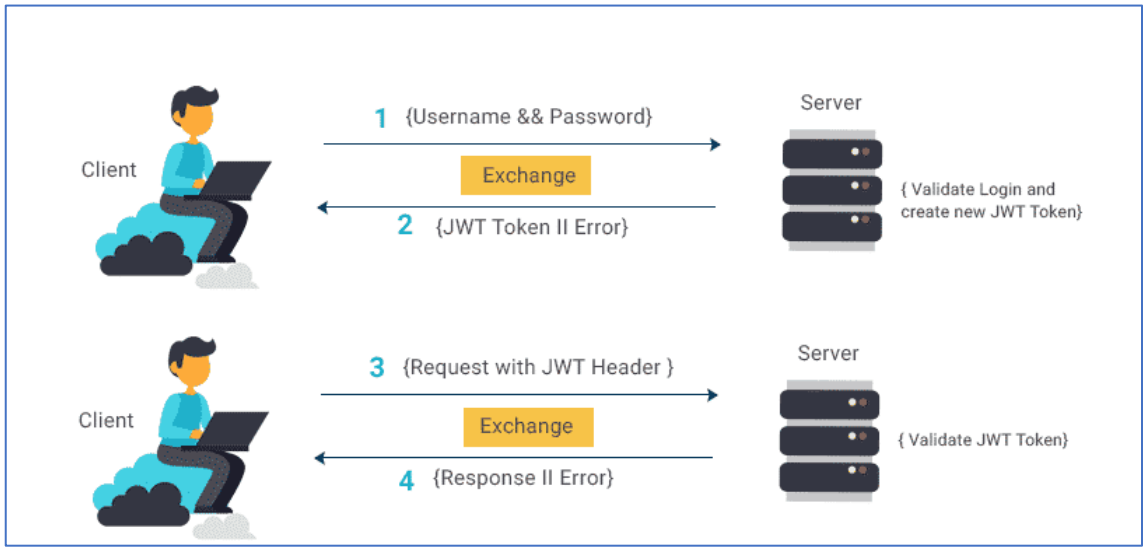


Рисунок 2.4 — Принцип роботи JWT

Клієнт надсилає на сервер запит на вхід із обліковими даними користувача (ім’я користувача та пароль). Після успішної перевірки облікових даних сервер генерує JWT і надсилає його назад клієнту. Якщо під час перевірки сталася помилка, натомість повертається повідомлення про помилку. Клієнт, який тепер володіє JWT, робить запит до захищеного маршруту на сервері та включає JWT у заголовок запиту для

автентифікації. Сервер перевіряє JWT, включений у заголовок запиту. Якщо маркер дійсний, сервер обробляє запит і повертає відповідну відповідь. Якщо JWT недійсний (наприклад, прострочений або підроблений), повертається помилка. Зображений потік JWT гарантує, що після початкового входу наступні запити аутентифікуються за допомогою маркера, усуваючи необхідність багаторазово надсилати облікові дані через мережу. Цей механізм забезпечує безпечний і ефективний метод автентифікації клієнтських запитів і керування сеансами[34].

JWT містить 3 основні компоненти (рис. 2.5) заголовок, підпис та корисне навантаження.

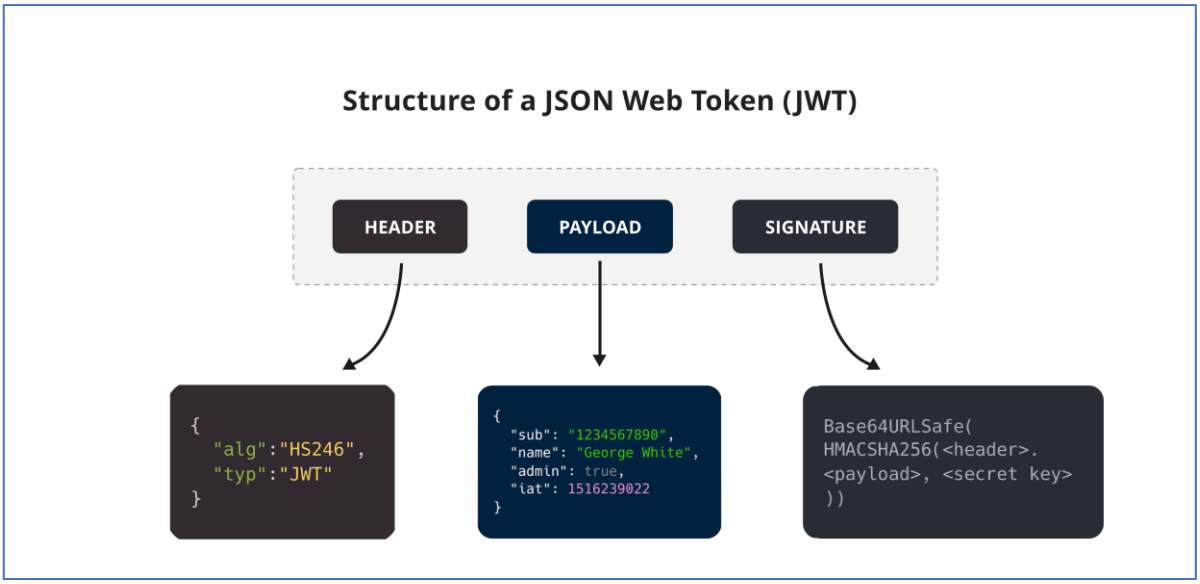


Рисунок 2.4 — Компоненти JWT

Заголовок? зазвичай заголовок складається з двох частин типу маркера, яким є JWT, і використовуваного алгоритму хешування, наприклад HMAC SHA256 або RSA. корисне навантаження містить претензії. Претензії це заяви про сутність (як правило, користувача) і додаткові дані. Корисне навантаження може включати попередньо визначені претензії (такі як тема, термін дії) і спеціальні претензії. Підпис використовується для перевірки того, що повідомлення не було змінено на шляху, і, у

випадку маркерів, підписаних закритим ключем, він також може підтвердити, що відправник JWT є тим, за кого себе видає.

Для програмної реалізації цього методу аутентифікації потрібно використати npm-пакет jsonwebtoken [35]. Створення JWT:

```
const jwt = require('jsonwebtoken');
const user = {
  id: '1234',
  username: 'johndoe',
  email: 'john@example.com'
};
const secret = 'your-256-bit-secret';
const token = jwt.sign(user, secret, { expiresIn: '1h' });
console.log(token);
```

Тут використовується jsonwebtoken для створення JWT. Вказується об’єкт user, який містить інформацію про користувача, яку потрібно включити у JWT. Також передається секретний ключ (secret), який повинен бути триманий в секреті і є ключем для підпису JWT. Параметр { expiresIn: '1h' } вказує термін дії JWT, в даному випадку - 1 година. Функція jwt.sign створює JWT на основі цих даних.

Перевірка JWT:

```
jwt.verify(token, secret, (err, decoded) => {
  if (err) {
    console.log('JWT verification failed:', err);
  } else {
    console.log('JWT decoded:', decoded);
  }
});
```

Використовується функція jwt.verify для перевірки та розкодування JWT. Передається JWT та секретний ключ (secret) для перевірки цілісності та правильності підпису. Функція jwt.verify перевіряє JWT та повертає помилку (err), якщо перевірка не пройшла успішно. У разі успішної перевірки, повертається розкодовані дані JWT (decoded).

JWT це надійний метод безпечної передачі інформації між сторонами у вигляді JSON-об’єктів, особливо в контексті веб-додатків. Він став стандартним методом автентифікації на основі токенів у сучасній веб-розробці. Хоча JWT набув

популярності в сучасній веб-розробці, важливо враховувати його недоліки та переваги.

Переваги:

- JWT є URL-безпечними і можуть надсилатися через URL, POST-параметри або HTTP-заголовки, що полегшує міждоменні запити у веб-додатках;
- Токени містять всю необхідну інформацію, що зменшує потребу в додаткових запитах до бази даних;
- JWT не вимагають зберігання сесій на стороні сервера, що робить їх ідеальними для масштабування додатків;
- Широко використовуються для авторизації та обміну інформацією між веб-сервісами, API та мікросервісами.

Недоліки:

- Оскільки вони є автономними, JWT можуть бути великими, що може збільшити накладні витрати в HTTP-запитах;
- Якщо токени не захищені належним чином, вони можуть бути вразливими до перехоплення та зловживань;
- Анулювання окремих JWT до закінчення терміну їх дії може бути складним завданням без впровадження додаткової інфраструктури.

2.6 HTTP authentication

HTTP-аутентифікація, протокол який використовується для підтвердження особи користувача при спробі отримати доступ до ресурсів сервера. Він вбудований в протокол HTTP і використовує заголовок Authorization в запиті для передачі облікових даних[36]. Базовий процес HTTP-автентифікації між клієнтом і сервером відбувається в декілька етапів (рис. 2.5)

Процес авторизації розпочинається з запиту клієнта на доступ до захищеного ресурсу на сервері. Після отримання запиту, сервер відправляє клієнту відповідь, у якій запитує облікові дані для автентифікації. Цей запит може містити в собі виклик форми або інші методи для отримання інформації від клієнта. Після цього клієнт передає запитувані облікові дані на сервер, часто включаючи їх у заголовок авторизації HTTP та використовуючи, наприклад, метод базової автентифікації для передачі даних. Процес авторизації спрямований на перевірку та підтвердження облікових даних користувача перед наданням доступу до захищеного ресурсу.

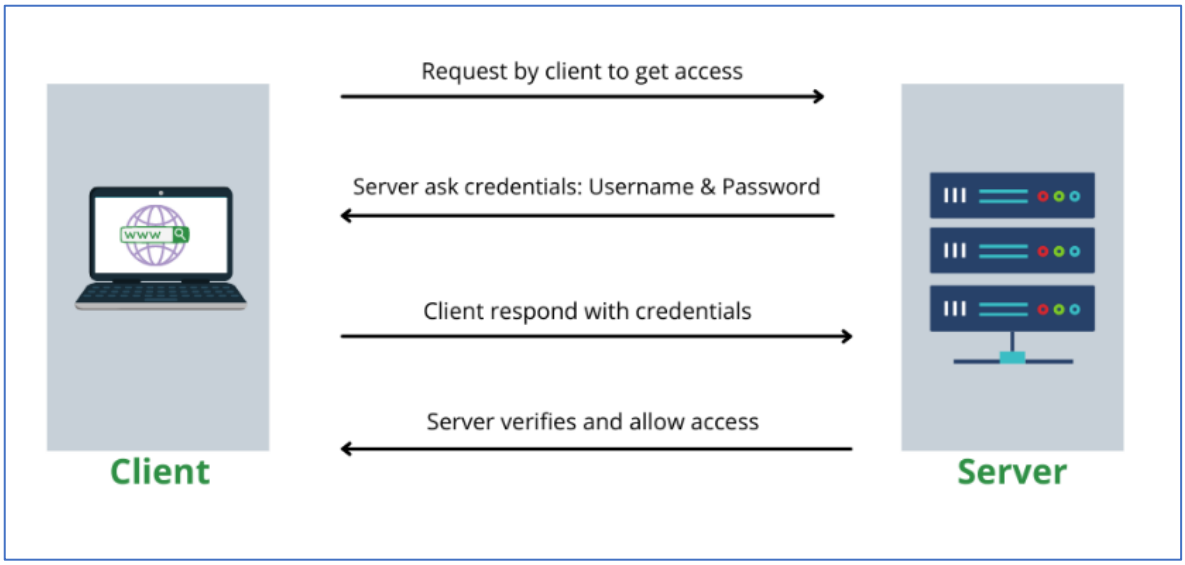


Рисунок 2.5 — Базова HTTP-автентифікація

В кінці процесу сервер перевіряє надані облікові дані зі збереженими даними. Якщо облікові дані дійсні, сервер надає клієнту доступ до запитуваного ресурсу. Якщо ні, він відповідає помилкою, зазвичай кодом статусу 401 Unauthorized, що пропонує клієнту повторити спробу автентифікації.

Програмна реалізація базової HTTP-автентифікації:

```
const http = require('http');
const options = {
  hostname: 'example.com',
```

```
    port: 80,
    path: '/protected',
    method: 'GET',
    headers: {
      'Authorization': 'Basic ' + Buffer.from('username:password').toString('base64')
    }
  };
  const request = http.request(options, (response) => {
    let data = '';
    response.on('data', (chunk) => data += chunk);
    response.on('end', () => console.log(data));
  });
  request.on('error', (e) => {
    console.error(`problem with request: ${e.message}`);
  });request.end();
```

Тут використовується модуль http для створення запиту до захищеної сторінки з використанням основної аутентифікації. Вказуються параметри запиту, такі як хост (hostname), порт (port), шлях (path), метод (method) та заголовок Authorization, який містить дані аутентифікації у форматі Base64. В даному випадку, використовується ім’я користувача (username) та пароль (password), які обгорнуті у формат Base64 за допомогою Buffer.from().toString('base64'). Після відправки запиту, обробляється відповідь.

Методи HTTP-автентифікації є фундаментальними для веб-безпеки, гарантуючи, що тільки авторизовані користувачі можуть отримати доступ до певних ресурсів сервера. Хоча він які всі попередні методи має певний ряд недоліків, також є і переваги які можуть перекрити деякі з недоліків цього методу.

Переваги HTTP-автентифікації:

- Легко реалізувати за допомогою базової автентифікації доступу з використанням стандартних заголовків HTTP;

- Не потрібна сесія на стороні сервера, що спрощує дизайн сервера і покращує масштабованість;
- Підтримується всіма сучасними веб-браузерами та серверами, що полегшує сумісність;
- Підтримує різні методи автентифікації, такі як Basic, Digest і токени на пред'явника.

Недоліки HTTP-автентифікації:

- Базова автентифікація є небезпечною для з'єднань, які не є HTTPS, оскільки облікові дані надсилаються у вигляді відкритого тексту;
- Зазвичай вимагає вбудованого діалогового вікна браузера, яке не можна налаштувати;
- HTTP-автентифікація не має вбудованого механізму виходу, а закриття сеансу часто вимагає закриття браузера;
- Кожен запит повинен містити облікові дані, що збільшує розмір заголовків HTTP.

висновки до розділу 2

В другому розділі роботи було проведено детальний розгляд різних методів автентифікації та авторизації. Зроблено висновки щодо кожного методу, включаючи їх програмну реалізацію та принципи роботи, що лежать в їх основі.

Багатофакторна автентифікація (MFA): Цей підхід підвищує безпеку, вимагаючи від користувачів надання декількох форм верифікації, що ефективно знижує ризик несанкціонованого доступу. OAuth: цей широко розповсюджений протокол забезпечує безпечну авторизацію без необхідності передавати конфіденційні облікові дані між додатками, що робить його надійним вибором у сучасній веб-розробці. OpenID Connect: Поєднуючи автентифікацію користувача з

можливостями довіреної авторизації OAuth 2.0, OpenID Connect пропонує стандартизований і надійний підхід, який забезпечує безпеку і сумісність. SAML: Незважаючи на свою складність, SAML відіграє вирішальну роль у безпечному обміні даними автентифікації та авторизації між сторонами. Через свою чутливість вона зазвичай використовується серверними додатками або внутрішніми службами. JSON Web Token (JWT): Вважається стандартним методом автентифікації на основі токенів у сучасній веб-розробці, JWT надійно передає інформацію між сторонами, зокрема у вигляді JSON-об’єктів. HTTP-аутентифікація: Незважаючи на широку підтримку, HTTP-аутентифікація може мати обмеження з точки зору безпеки і гнучкості в порівнянні з іншими розглянутими методами.

Підсумовуючи розділ надає інформацію про сильні сторони та опис кожного методу автентифікації та авторизації. Розуміючи переваги і недоліки цих методів, розробники можуть приймати обґрунтовані рішення при виборі найбільш підходящого для них конкретного контексту застосування та вимог безпеки.

3 ОЦІНКИ ЕФЕКТИВНОСТІ ВПРОВАДЖЕННЯ СУЧАСНИХ ТЕХНОЛОГІЙ АУТЕНТИФІКАЦІЇ ТА АВТОРИЗАЦІЇ ДЛЯ ЗАБЕЗПЕЧЕННЯ БЕЗПЕКИ БІЗНЕСУ

3.1 Шляхи вирішення актуальних проблем впровадження технологій аутентифікації та авторизації

Оскільки технології продовжують розвиватися, а цифрові системи стають все більш взаємопов’язаними, впровадження технологій автентифікації та авторизації стало критично важливим аспектом забезпечення безпеки та цілісності онлайн-взаємодій. Однак, незважаючи на успіхи, досягнуті в цій галузі, все ще існують виклики і проблеми, які необхідно вирішити для створення надійних і ефективних систем автентифікації та авторизації. Необхідно розглянути поточні проблеми, з якими стикаються при впровадженні цих технологій, та запропонувати способи їх вирішення. Від питань, пов’язаних із зручністю та комфортом користувачів, до проблем конфіденційності та безпеки даних, є кілька ключових сфер, в яких можна досягти покращень.

MFA важливий захід безпеки, який допомагає захистити облікові записи користувачів, вимагаючи декількох форм перевірки перед наданням доступу. Хоча MFA підвищує рівень безпеки, цей метод аутентифікації також може створювати незручності для користувачів. Однак існують рішення для вирішення цих актуальних проблем методу. Однією з проблем традиційних методів MFA є незручності, які вони створюють для користувачів. Запам’ятовування та введення декількох факторів автентифікації може забирати багато часу і викликати роздратування. Щоб вирішити цю проблему, можна впровадити адаптивну MFA [37]. Адаптивна MFA використовує алгоритми машинного навчання для аналізу поведінки користувача і контексту, щоб розумно налаштувати необхідний рівень автентифікації. Це означає, що якщо користувач отримує доступ до свого облікового запису зі знайомого пристрою та

місця, система може вимагати лише один фактор замість кількох, зменшуючи незручності та зберігаючи безпеку.

Ще однією проблемою MFA є обмежені можливості резервного копіювання. Іноді користувачі можуть не мати доступу до свого основного методу автентифікації, наприклад, фізичного токену або пристрою. Щоб вирішити цю проблему, організації можуть розробити ряд резервних методів автентифікації. Наприклад, одноразові коди, надіслані електронною поштою або SMS, можуть слугувати тимчасовими резервними варіантами. Крім того, резервні ключі, які надійно зберігаються на комп'ютері користувача або в менеджері паролів, можуть стати альтернативним засобом автентифікації [38]. Залежність від фізичних пристроїв для MFA також може бути проблематичною. Фізичні токени або апаратні пристрої можуть бути втрачені або пошкоджені, в результаті чого користувачі не зможуть завершити процес автентифікації. Віртуальні варіанти MFA можуть допомогти вирішити цю проблему. Мобільні додатки, які генерують одноразові паролі або використовують біометричну автентифікацію, наприклад, відбитки пальців або розпізнавання обличчя, можуть стати зручною і безпечною альтернативою фізичним токенам. MFA на основі SMS, ще один віртуальний варіант, який надсилає коди автентифікації на мобільний телефон користувача.

Хоча MFA має вирішальне значення для безпеки облікових записів, важливо враховувати незручності, які вона може спричинити для користувачів. Впровадження адаптивної MFA, розробка різних резервних методів автентифікації та використання віртуальних опцій MFA може значно покращити користувацький досвід без шкоди для безпеки.

OAuth це широко розповсюджений метод для авторизації та автентифікації, що дозволяє користувачам надавати обмежений доступ до своїх ресурсів стороннім додаткам. Хоча OAuth надає численні переваги, він не позбавлений проблем. Однією з основних проблем OAuth є викрадення токенів. Токени, такі як токени доступу або токени оновлення, можуть бути вразливими до перехоплення зловмисниками, що

дозволяє їм отримати несанкціонований доступ до ресурсів користувача. Щоб вирішити цю проблему, для захисту токенів можна використовувати передові методи шифрування. Шифрування токенів за допомогою стійких криптографічних алгоритмів значно знижує ризик викрадення токенів [39]. Крім того, можна реалізувати прив’язку токенів, яка гарантує, що токени будуть прив’язані до певних клієнтських пристроїв або каналів, що зробить їх непридатними для використання в разі перехоплення.

Ще однією проблемою OAuth є питання надто широкої довіри. Коли користувачі надають дозвіл на доступ до своїх ресурсів, обсяг доступу, який запитують програми, може бути занадто широким, що потенційно може призвести до надмірного витоку даних. Щоб вирішити цю проблему, постачальники OAuth повинні пропонувати більш деталізовані варіанти згоди. Це дозволить користувачам визначати рівень доступу, що надається кожному додатку, гарантуючи, що буде надано доступ лише до необхідних даних. Забезпечуючи точний контроль над рівнями доступу до даних, користувачі можуть з більшою впевненістю надавати дозволи додаткам. Крім того, недостатні механізми відкликання токенів становлять значний ризик безпеки в системах OAuth. Якщо токен скомпрометовано, необхідно негайно відкликати його, щоб запобігти несанкціонованому доступу до ресурсів користувача. Щоб ефективно вирішити цю проблему, реалізації OAuth повинні реалізовувати надійні механізми відкликання токенів. Ці механізми повинні дозволяти негайно і безпечно відкликати скомпрометовані токени, роблячи їх непридатними для використання [40]. Впровадження списків відкликання токенів в режимі реального часу або інтеграція зі службами відкликання постачальників ідентифікаційних даних може забезпечити швидке та ефективне відкликання. Використання передових методів шифрування та прив'язки токенів може захистити від викрадення токенів. Надання більш деталізованих варіантів згоди може запобігти надмірно широким дозволам на доступ. Впровадження ефективних і негайних механізмів відкликання токенів може вирішити проблему скомпрометованих токенів. Впроваджуючи ці

рішення, впровадження OAuth може забезпечити безпечну і зручну роботу як для розробників додатків, так і для кінцевих користувачів.

Метод OpenID Connect являє собою рівень ідентифікації, побудований на основі протоколу OAuth 2.0, що забезпечує стандартизований спосіб для додатків автентифікувати користувачів і отримувати доступ до даних їхніх профілів. Хоча OpenID Connect пропонує значні переваги, він також створює певні проблеми. Однією з проблем OpenID Connect є складний процес інтеграції. Впровадження OpenID Connect може бути складним завданням для розробників через відсутність чіткої та вичерпної документації та стандартизованих бібліотек. Щоб вирішити цю проблему, необхідно спростити процес інтеграції, надавши кращу документацію, включаючи покрокові інструкції, приклади коду та ресурси для усунення несправностей [41]. Крім того, розробка стандартизованих бібліотек і SDK для популярних мов програмування може значно полегшити роботу розробників, сприяючи ширшому впровадженню та скороченню часу на реалізацію. Також важливою проблемою є обмежена підтримка міждоменної автентифікації. Міждоменна автентифікація дозволяє користувачам безперешкодно проходити автентифікацію в різних додатках або на різних веб-сайтах без необхідності повторного введення облікових даних. Такий спрощений досвід підвищує задоволеність користувачів і зменшує тертя. Щоб вирішити цю проблему, постачальники OpenID Connect повинні розширити підтримку міждоменної автентифікації. Цього можна досягти шляхом впровадження галузевих стандартів, таких як протоколи єдиного входу (SSO), або шляхом розробки рішень, які полегшують безпечний і довірчий зв’язок між різними доменами.

Питання конфіденційності також є важливим фактором при впровадженні OpenID Connect. Протокол покладається на атрибути користувача, якими обмінюються постачальники ідентифікаційних даних та постачальники послуг, що викликає занепокоєння з приводу збору і використання особистої інформації. Щоб зменшити ці занепокоєння, слід запровадити сильніші засоби контролю конфіденційності. Провайдери OpenID Connect повинні пропонувати надійні

механізми управління згодою користувачів, що дозволять користувачам контролювати, які атрибути передаються кожному провайдеру і з якою метою. Також слід розробити чіткі та прозорі політики управління даними, які гарантуватимуть, що дані користувачів обробляються безпечно та відповідно до правил конфіденційності. OpenID Connect забезпечує стандартизовану основу для автентифікації особистості, проте важливо вирішити складнощі інтеграції, забезпечити безперебійну міждоменну автентифікацію і надати пріоритет конфіденційності користувачів.

Метод SAML це фреймворк на основі XML, який використовується для обміну даними автентифікації та авторизації між постачальниками ідентифікаційних даних та постачальниками послуг. Хоча SAML пропонує значні переваги, існує ряд проблем, які необхідно вирішити для забезпечення оптимальної продуктивності, сумісності з мобільними пристроями та простоти усунення помилок. Однією з проблем SAML є проблеми з продуктивністю, особливо при обробці великих корисних навантажень. Повідомлення SAML можуть містити значну кількість метаданих та інформації про атрибути, що призводить до уповільнення часу обробки [42]. Щоб вирішити цю проблему, можна застосувати методи оптимізації для ефективнішої обробки більшого корисного навантаження. Це може включати оптимізацію структури XML, стиснення даних або впровадження механізмів кешування для зменшення накладних витрат на обробку. Покращуючи продуктивність, SAML може забезпечити більш плавну та швидку автентифікацію.

Обмежена підтримка мобільних пристроїв є ще однією проблемою для SAML. Мобільні пристрої мають різні розміри екранів, можливості та обмеження підключення в порівнянні з традиційними системними середовищами. Щоб покращити сумісність з мобільними пристроями, необхідно зосередитися на адаптивному дизайні [43]. Це передбачає розробку інтерфейсів SAML, які плавно адаптуються до різних розмірів і орієнтації екрану. Крім того, зменшення використання даних за допомогою оптимізованих протоколів зв’язку або вибіркового обміну атрибутами може покращити користувацький досвід на мобільних пристроях

з обмеженою пропускною здатністю або з переривчастим з'єднанням. Складність у вирішенні помилок також є значною проблемою для SAML. Коли під час процесу автентифікації виникають помилки, користувачі можуть зіткнутися із зашифрованими повідомленнями про помилки або намагатися усунути проблему. Для вирішення цієї проблеми можна розробити чіткіші повідомлення про помилки, які надаватимуть змістовні пояснення та практичні кроки для їх усунення. Крім того, можна надати вичерпні інструкції з усунення несправностей, щоб допомогти користувачам діагностувати типові проблеми і знаходити відповідні рішення. Спрощуючи вирішення помилок, впровадження SAML може зменшити розчарування користувачів і спростити процес автентифікації. Оптимізація обробки SAML для ефективної обробки більшого корисного навантаження може підвищити продуктивність. Покращення сумісності з мобільними пристроями завдяки адаптивному дизайну та зменшенню використання даних може покращити користувацький досвід на мобільних пристроях. Розробка більш зрозумілих повідомлень про помилки і надання вичерпних інструкцій з усунення проблем може сприяти швидшому їх вирішенню.

JSON Web Token, популярний метод безпечної передачі інформації між сторонами у вигляді JSON-об'єктів. Хоча JWT мають ряд переваг, існують проблеми, які необхідно вирішити для забезпечення ефективного управління ключами, відкликання токенів та захисту від атак. Однією з проблем JWT є труднощі з управлінням ключами. Оскільки JWT покладається на криптографічні ключі для підписання та перевірки токенів, безпечне управління цими ключами може бути складним завданням, особливо в розподілених системах [44]. Щоб вирішити цю проблему, дуже важливо стандартизувати практики управління ключами в організаціях і на різних платформах. Це включає в себе впровадження безпечних механізмів зберігання ключів, встановлення політик ротації ключів та автоматизацію процесів генерації ключів. Стандартизувавши практики управління ключами, можна мінімізувати ризик витоку ключів або несанкціонованого доступу до них. Відкликання токенів є ще однією проблемою в системах на основі JWT. Після випуску

JWT досить складно відкликати його дійсність до закінчення терміну дії. Це стає проблематичним, коли йдеться про скомпрометовані токени або необхідність негайного відкликання. Щоб вирішити цю проблему, слід створити надійні процеси відкликання токенів. Це може включати в себе впровадження централізованих служб управління токенами, які підтримують список відкликаних токенів, або інтеграцію з механізмами відкликання токенів постачальників ідентифікаційних даних. Наявність ефективних процесів відкликання токенів дозволяє пом'якшити вплив скомпрометованих токенів.

JWT можуть бути вразливими до атак, якщо вони не реалізовані безпечно. Наприклад, якщо зловмисник отримує доступ до приватного ключа, який використовується для підписання JWT, він може підробити дійсні токени і видавати себе за законних користувачів. Щоб підвищити безпеку, слід впроваджувати передові заходи. Одним з таких заходів є використання асиметричного підписання, коли для підписання та перевірки використовуються різні ключі [45]. Крім того, слід проводити регулярні аудити безпеки для виявлення та усунення будь-яких вразливостей у реалізації JWT. Впроваджуючи ці заходи, можна значно знизити ризик несанкціонованого доступу та імітації.

HTTP-аутентифікація механізм, який використовується для автентифікації користувачів і надання доступу до захищених ресурсів через протокол HTTP. Хоча HTTP-аутентифікація забезпечує простий механізм автентифікації, існують проблеми, які необхідно вирішити для забезпечення безпечної передачі даних, захисту від перехоплення сеансів та надання більшої кількості варіантів автентифікації. Ключовою з проблем HTTP-автентифікації є слабке шифрування під час передачі даних. За замовчуванням HTTP не забезпечує шифрування, а це означає, що конфіденційні облікові дані та дані користувача можуть бути перехоплені зловмисниками [46]. Щоб вирішити цю проблему, дуже важливо використовувати більш надійні методи шифрування, такі як захист на транспортному рівні (TLS) або рівень захищених сокетів (SSL). Впровадження TLS гарантує, що всі дані, які

передаються між клієнтом і сервером, будуть зашифровані, захищаючи конфіденційну інформацію від несанкціонованого доступу. Перехоплення сеансу є ще однією проблемою в HTTP-аутентифікації. Зловмисники можуть перехоплювати маркери сеансу або файли cookie і використовувати їх для того, щоб видавати себе за легітимних користувачів, отримуючи несанкціонований доступ до захищених ресурсів. Щоб зменшити цей ризик, слід вдосконалити методи управління сеансами. Безпечні файли cookie можна використовувати для зберігання інформації про сеанси з відповідними прапорами безпеки, такими як secure і httpOnly, щоб запобігти міжсайтовим скриптовим атакам і гарантувати, що файли cookie передаються тільки через безпечні з'єднання [47]. Крім того, впровадження методів завершення сеансу, таких як встановлення короткої тривалості сеансу або використання ковзного завершення сеансу, може допомогти обмежити часові рамки для перехоплення сеансу. Хоча HTTP-аутентифікація пропонує простий механізм автентифікації, важливо вирішити проблему слабкого шифрування під час передачі даних, захиститися від перехоплення сеансу. Використання більш надійних методів шифрування, таких як TLS, забезпечує безпечну передачу даних. Покращення управління сеансами за допомогою захищених файлів cookie та методів закінчення сеансу знижує ризик перехоплення сеансу.

3.2 Порівняльна характеристика ефективності та безпеки різних методів аутентифікації та авторизації

Порівняльний аналіз різних методів автентифікації та авторизації, таких як багатофакторна автентифікація (MFA), OAuth, OpenID Connect, Security Assertion Markup Language (SAML), JSON Web Token (JWT) та HTTP-автентифікація, дає уявлення про їхню ефективність та безпеку. Щоб всебічно оцінити ці методи, важливо враховувати широкий спектр критеріїв. Для порівняння методів було обрано наступні критерії:

1. Надійність безпеки: Цей критерій оцінює рівень захисту, який пропонує метод автентифікації та авторизації від несанкціонованого доступу, спроб злому та порушень. Метод повинен використовувати надійні алгоритми шифрування для захисту конфіденційних даних під час передачі та зберігання. Підтримувати надійні механізми автентифікації, такі як надійні паролі, біометричні дані або апаратні токени, щоб забезпечити доступ до ресурсів лише авторизованим користувачам. Мати вбудовані функції або механізми для запобігання поширеним атакам, таким як атаки грубої сили, фішинг або перехоплення сеансу;

2. Зручність для користувача: Цей критерій оцінює простоту використання та зручність процесу автентифікації для кінцевих користувачів. Він враховує фактори, які покращують користувацький досвід. Метод повинен забезпечувати безперебійну та інтуїтивно зрозумілу автентифікацію, зводячи до мінімуму потребу в складних кроках або технічних знаннях. Процес автентифікації повинен бути швидким і не викликати значних затримок або незручностей для кінцевих користувачів. Метод має бути доступним на різних пристроях і платформах, дозволяючи користувачам легко проходити автентифікацію незалежно від їхнього місцезнаходження чи пристрою;

3. Масштабованість: Цей критерій вимірює здатність методу автентифікації та авторизації обробляти зростаючу кількість користувачів і транзакцій без шкоди для продуктивності та безпеки. Метод повинен зберігати свою ефективність і час відгуку навіть при збільшенні кількості користувачів Підтримувати механізми балансування навантаження для розподілу навантаження на автентифікацію між декількома серверами або вузлами, щоб уникнути вузьких місць. Забезпечувати цілісність та ефективність для забезпечення безперервної доступності в ситуаціях підвищеного попиту або системних збоїв;

4. Можливість інтеграції: Цей критерій оцінює, наскільки добре метод автентифікації та авторизації інтегрується з існуючими системами та програмним забезпеченням в організації. Метод повинен надавати добре задокументовані API та бібліотеки, які сприяють безперешкодній інтеграції з різними платформами,

фреймворками та мовами програмування. Підтримувати інтеграцію з популярними постачальниками ідентифікаційних даних, такими як Active Directory, LDAP або хмарні сервіси ідентифікації, що дозволяє організаціям використовувати існуючі каталоги користувачів. Забезпечувати легку інтеграцію з рішеннями SSO, дозволяючи користувачам автентифікуватися один раз і безперешкодно отримувати доступ до декількох додатків або сервісів;

5. Економічна ефективність: Цей критерій оцінює загальні витрати, пов'язані з впровадженням і підтримкою методу автентифікації та авторизації. Чи вимагає метод будь-яких ліцензійних платежів або підписок, аналіз довгострокові фінансові наслідки для організації. Вимоги до інфраструктури, включаючи обладнання, програмне забезпечення та мережеві компоненти, необхідні для підтримки методу. Ресурси, необхідні для регулярних оновлень, виправлення помилок і технічної підтримки, щоб підтримувати ефективність і безпеку методу;

6. Адаптивність: Цей критерій вимірює гнучкість методу автентифікації та авторизації в адаптації до мінливих середовищ безпеки та технологій. Чи може метод легко інтегруватися з новими технологіями безпеки, такими як біометрія, апаратні токени або нові стандарти автентифікації. Метод відповідає галузевим стандартам і протоколам, забезпечуючи сумісність з майбутніми технологіями та фреймворками. Чи може метод масштабуватися і адаптуватися до мінливих потреб організації, наприклад, до нових груп користувачів, політик доступу або сценаріїв автентифікації;

7. Відповідність: Цей критерій оцінює, наскільки метод автентифікації та авторизації відповідає законодавчим і нормативним вимогам, таким як Загальний регламент про захист даних (GDPR) або іншим галузевим стандартам. Метод повинен підтримувати механізми для безпечної обробки та захисту даних користувачів, забезпечуючи дотримання законів та нормативних актів про конфіденційність даних. Підтримка функції управління згодою, що дозволяють користувачам надавати явну згоду на збір та обробку даних. Дозвіл забезпечувати дотримання прав користувачів,

таких як доступ до даних, їх модифікація чи видалення, відповідно до чинних нормативно-правових актів;

8. Конфіденційність даних: Цей критерій оцінює, наскільки добре метод автентифікації та авторизації захищає дані користувачів та їхню конфіденційність. Надійність механізмів шифрування, що використовуються для захисту даних користувача під час передачі та зберігання. Мінімізація збирання та зберігання персональних даних для зменшення ризиків для конфіденційності. Забезпечення механізмів для збереження даних користувачів лише на той час, на який це необхідно, дозвіл видаляти дані, коли це необхідно;

9. Аудит та моніторинг: Цей критерій оцінює здатність методу автентифікації та авторизації відстежувати, перевіряти та контролювати події доступу та автентифікації з метою забезпечення відповідності та безпеки. Генерація журналів та записів подій автентифікації, включно з позначками часу, ідентифікаторами користувачів та спробами доступу. Надання можливості звітування для аналізу шаблонів автентифікації, виявлення аномалій та створення аудиторських слідів. Інтеграція з системами SIEM для централізованого управління журналами та моніторингу подій безпеки;

10. Крос-платформна підтримка: Крос-платформна підтримка означає сумісність методу автентифікації та авторизації з різними платформами та пристроями, включаючи мобільні та десктопні системи. Підтримка основних операційних систем, таких як Windows, macOS, Linux, iOS та Android. Сумісність з популярними веб-браузерами, такими як Chrome, Firefox,Opera, Safari та Edge. Підтримка специфічних для мобільних пристроїв методів автентифікації, такі як розпізнавання відбитків пальців або ідентифікація за обличчям;

11. Механізми відновлення: Механізми відновлення стосуються ефективності варіантів відновлення облікового запису у випадках, коли користувачі втратили або забули свої облікові дані. Чи пропонує метод безпечний і простий процес скидання паролів за допомогою електронної пошти, SMS або секретних запитань для

користувачів. Чи надає метод альтернативні методи перевірки для відновлення облікового запису, коли фактори MFA недоступні, наприклад, резервні коди або токени для відновлення облікового запису. Включення механізмів запобігання несанкціонованому доступу під час процесу відновлення, дозволяючи при цьому законним користувачам відновити доступ до своїх облікових записів;

12. Управління токенами: Управління токенами фокусується на тому, наскільки добре токенами в системах, заснованих на токенах, керують протягом усього процесу автентифікації та авторизації. Як токени генеруються та розповсюджуються серед користувачів, щоб переконатися, що токени є унікальними, захищеними від несанкціонованого доступу та безпечно доставлені. Чи підтримує метод механізми відкликання, щоб зробити токени недійсними або відкликати їх у разі необхідності, наприклад, у випадку скомпрометованого пристрою або облікового запису користувача. Чи мають токени термін придатності, щоб обмежити їхню дійсність і зменшити ризик несанкціонованого доступу;

13. Керування сеансами: Керування сеансами відноситься до ефективності методу автентифікації та авторизації в управлінні сеансами користувачів і запобіганні перехопленню сеансів. Використання методом безпечних протоколів (наприклад, HTTPS) для створення сеансу для захисту від підслуховування та атак. Використання механізмів завершення сеансу для автоматичного виходу з системи неактивних користувачів, що зменшує ризик несанкціонованого доступу, якщо сеанс залишився без нагляду. Використання технологій, як маркери сеансу або файли cookie, щоб гарантувати, що кожен запит асоціюється з дійсним та автентифікованим сеансом.

14. Багатодоменна підтримка: Підтримка декількох доменів оцінює здатність методу автентифікації та авторизації безперешкодно працювати в різних доменах або середовищах. Аутентифікація користувачів один раз і отримати доступ до декількох додатків або служб в різних доменах без необхідності повторної автентифікації. Підтримка федеративних протоколів ідентичності, таких як SAML або OAuth, що дозволяють користувачам отримувати доступ до ресурсів у різних доменах або

середовищах, використовуючи бажаного постачальника ідентичності. Підтримка механізмів надання та управління дозволами на доступ у різних доменах або середовищах, забезпечуючи узгоджений і контрольований доступ до ресурсів;

15. Настроюваність: ступінь, до якого метод автентифікації та авторизації може бути налаштований для задоволення конкретних потреб організації або користувача. Налаштування елементів користувацького інтерфейсу, таких як екрани входу в систему або елементи брендингу, щоб відобразити свою унікальну ідентичність бренду. Можливості кастомізації для визначення та впровадження політик доступу, правил автентифікації та механізмів авторизації, пристосованих до конкретних вимог організації.

Враховуючи вище описані критерії та методи авторизації та аутентифікації було створено порівняльну таблицю цих методів (табл. 3.1).

Таблиця 3.1.

Порівняльна таблиця методів

Критерії	**MFA**	**OAuth**	**OpenID Connect**	**SAML**	**JWT**	**HTTP аутентифікація**
Надійність безпеки	8	7	7	8	9	5
Зручність для користувача	7	8	8	6	7	6
Масштабованість	6	7	7	7	8	5
Можливість інтеграції	6	8	8	7	7	5
Вартісна ефективність	7	6	6	5	8	7
Адаптивність	7	7	7	6	8	5
Відповідність нормам	8	7	7	8	9	6
Конфіденційність даних	7	7	7	7	8	6
Аудит та моніторинг	7	8	8	7	8	6
Підтримка крос-платформ	6	8	8	6	7	5

Кінець таблиці 3.1.

Механізми відновлення	8	6	6	6	5	7
Управління токенами	5	7	7	8	9	4
Управління сесіями	6	7	7	7	8	5
Підтримка мультидоменів	5	8	8	7	6	4
Можливість налаштування	6	7	7	6	7	5

Таблиця 3.1 надає оцінки шістьом популярним методам аутентифікації та авторизації: багатофакторної аутентифікації (MFA), OAuth, OpenID Connect, SAML, JSON Web Token (JWT) та HTTP-аутентифікації. Оцінки були надані на основі 15 критеріїв, які охоплюють аспекти безпеки, зручності, ефективності, масштабованості та інтеграційних можливостей.

Оцінки від 1 до 10 відображають наступне:

1-3: Низька ефективність/безпека; значні недоліки в цій області;

4-6: Середня ефективність; деякі обмеження, але прийнятний рівень функціональності;

7-8: Висока ефективність; добре виконує свої функції, з незначними недоліками;

9-10: Виняткова ефективність та безпека; відповідає або перевищує всі очікування;

На основі отриманих результатів можна зробити наступні висновки:

JWT виокремлюється високими оцінками в багатьох категоріях, особливо в надійності безпеки та управлінні токенами. Це вказує на його сильні сторони в захисті даних та гнучкості управління сесіями;

OAuth та OpenID Connect методи отримали високі оцінки за зручність для користувачів, масштабованість, можливість інтеграції та підтримку крос-платформ. Це свідчить про їхню ефективність у великих масштабних середовищах та гнучкість у використанні;

MFA показує високі результати у категоріях надійності безпеки, адаптивності та механізмів відновлення. Це підтверджує його важливість як основного засобу захисту облікових записів;

SAML демонструє міцні позиції у відповідності нормам, управлінні токенами та надійності безпеки, вказуючи на його напрямленість для корпоративних систем із суворими вимогами до безпеки;

HTTP-аутентифікація має нижчі оцінки у більшості категорій, що відображає її обмежені можливості порівняно з іншими, більш сучасними методами;

Цей аналіз підкреслює, що кожен метод має свої сильні та слабкі сторони, і вибір відповідного методу залежить від специфічних потреб та контексту його застосування. Для організацій важливо оцінювати ці методи на основі своїх унікальних вимог до безпеки, масштабованості, зручності та витрат, щоб вибрати найбільш підходящий. Такий підхід дозволяє не лише забезпечити оптимальний рівень захисту, але й гарантує, що процеси аутентифікації та авторизації будуть максимально ефективними та користувацьки орієнтованими. Врахування мінливих тенденцій у кібербезпеці та технологічних нововведень є ключовим для підтримання актуальності та ефективності систем безпеки.

висновки до розділу 3

У третьому розділі роботи були розглянуті ключові аспекти ефективності та безпеки сучасних технологій аутентифікації та авторизації, що мають вирішальне значення для забезпечення безпеки бізнесу.

Обговорено та проаналізовано питання, як-от зручність для користувачів, безпека даних, масштабованість систем та інтеграційна здатність. Особлива увага приділяється впровадженню адаптивної багатофакторної аутентифікації, поліпшенню механізмів управління токенами та розширенню можливостей аудиту та моніторингу.

Наступним створено таблицю порівняння, яка висвітлює різні аспекти методів аутентифікації, таких як MFA, OAuth, OpenID Connect, SAML, JWT та HTTP-аутентифікація. Оцінки за критеріями, від надійності безпеки до налаштування, цей аналіз надає зрозуміле уявлення про сильні та слабкі сторони кожного методу. Ця аналітика сприяє глибшому розумінню того, як різні технології можуть слугувати конкретним потребам організацій у контексті захисту бізнесу.

ВИСНОВКИ

У даній роботі були ретельно розроблені та детально описані ключові методи аутентифікації та авторизації, які відіграють вирішальну роль у забезпеченні безпеки сучасного бізнесу.

В першому розділі було розкрито основні поняття та визначення, пов’язані з аутентифікацією та авторизацією. Зокрема, було класифіковано та описано різні методи аутентифікації, а також висвітлено основні принципи їхньої роботи. Також у цьому розділі були окреслені загальні методи авторизації в веб-додатках, що дало чітке розуміння основних механізмів контролю доступу.

У другому розділі було детально проаналізовано сучасні технології, включаючи багатофакторну аутентифікацію (MFA), OAuth, OpenID Connect, SAML, JSON Web Token (JWT), та HTTP-аутентифікацію. Цей аналіз дозволив глибше зрозуміти як особливості кожного методу, так і їхнє місце у загальній картині захисту інформаційних систем.

Третій розділ зосереджувався на оцінці ефективності впровадження згаданих технологій з метою забезпечення бізнес-безпеки. Були висвітлені шляхи вирішення актуальних проблем, які стосуються аутентифікації та авторизації, а також проведено порівняльний аналіз різних методів за кількома ключовими параметрами.

Ця робота, включаючи теоретичні виклади, детальний аналіз та комплексні оцінки, надає цінне розуміння важливості та складності аутентифікаційних та авторизаційних технологій у сучасному цифровому світі. Розробка, вивчення та оцінка цих методів є критичною задачею для забезпечення надійного захисту бізнесу від кіберзагроз.

ПЕРЕЛІК ПОСИЛАНЬ

1. QualityAssuranceGroup. Аутентифікація і авторизація: що це і в чому відмінність? URL: qagroup.com.ua/publications/autentyfikatciia-i-avtoryzatciia/.

2. GetPCI. PCI DSS Certification URL: https://getpci.com/.

3. Legal IT Group. GDPR Україна URL: https://legalitgroup.com/category/gdpr-ta-personalni-dani/gdpr-ukrayina/.

4. MiniOrange. What is Authentication? Different Types of Authentication URL: https://www.miniorange.com/blog/different-types-of-authentication-methods-for-security/.

5. MicroSoft. Kerberos Authentication Overview URL: https://learn.microsoft.com/en-us/windows-server/security/kerberos/kerberos-authentication-overview.

6. LoginRadius. LDAP Authentication: Meaning and How it Works? URL: loginradius.com/blog/identity/what-is-ldap-authentication-and-how-it-works/.

7. optbank. Технологія безпечних розрахунків в інтернеті URL: https://www.otpbank.com.ua/privateclients/pay-cards/3dsecure/.

8. Fortinet. Authentication Token URL: https://www.fortinet.com/resources/cyberglossary/authentication-token.

9. descope. What is Password-Based Authentication? URL: https://www.descope.com/learn/post/password-authentication.

10. ondato. A Complete Guide to Biometric Authentication Methods URL: https://ondato.com/blog/benefits-of-biometric-authentication/.

11. Гайворонський М.В., Новіков О.М. Безпека інформаційно-комунікаційних систем. - К.: Видавнича група BHV,2009. – 608 с., іл.

12. Дронь М.М., Малайчук В.П., Петренко О.М. Основи теорії захисту

інформації: Навч. посібник. – Д.: Вид-во Дніпропетр. ун-ту, 2001. – 312 с.

13. ChannelforIt. Методи аутентифікації без пароля URL: https://channel4it.com/publications/metodi-autentifkac-bez-parolya.html.

14. Mohammad A. Alia, Abdelfatah Aref Tamimi, and Omaima N. A. AL-Allaf. Cryptography Based Authentication Methods. Proceedings of the World Congress on Engineering and Computer Science 2014 Vol I. Режим доступу: https://www.iaeng.org/publication/WCECS2014/WCECS2014_pp199-204.pdf

15. LinkedIn. A Comprehensive Overview of Access Control Models [Електроннийресурс] – Режим доступу: linkedin.com/pulse/comprehensive-overview-access-control-models-rbac-abac-jay-.

16. Connect2id. OpenID Connect explained URL: https://connect2id.com/learn/openid-connect.

17. frontegg. OAuth vs. JWT: What Is the Difference? URL: https://frontegg.com/blog/oauth-vs-jwt.

18. What is Multi-Factor Authentication (MFA)?? URL: https://aws.amazon.com/what-is/mfa/.

19. Time Based One Time Password (TOTP, OTP) URL: https://www.hypr.com/security-encyclopedia/time-based-time-password-totp-otp.

20. Engage your customers globally with SMS APIs URL: https://www.twilio.com/docs/usage/api.

21. The Best Authenticator Apps for 2023 URL: https://www.pcmag.com/picks/the-best-authenticator-apps.

22. speakeasy URL: npmjs.com/package/speakeasy.

23. base32 URL: npmjs.com/package/base32.

24. twilio URL: npmjs.com/package/twilio.

25. What is OAuth? How the open authorization framework works URL: https://www.csoonline.com/article/562635/what-is-oauth-how-the-open-authorization-framework-works.html.

26. oauth URL: npmjs.com/package/oauth.

27. OAuth 2.0 URL: https://oauth.net/2/.

28. What is OpenID Connect URL: https://openid.net/developers/how-connect-works/.

29. What is Security Assertion Markup Language (SAML)? URL: https://www.oracle.com/security/cloud-security/what-is-saml/.

30. What is SAML and How Does it Work? URL: https://www.varonis.com/blog/what-is-saml.

31. Passport-SAML URL: npmjs.com/package/passport-saml.

32. a SAML 2.0 authentication provider for Passport, the Node.js authentication library. URL:.passportjs.org/packages/passport-saml/.

33. Introduction to JSON Web Tokens. URL:. https://jwt.io/introduction.

34. What is JSON Web Token (JWT) | Implementation of JWT in Node JS. URL:. https://www.cronj.com/blog/what-is-json-web-token-jwt/.

35. An implementation of JSON Web Tokens. URL:. npmjs.com/package/jsonwebtoken.

36. How Does HTTP Authentication Work. URL:. https://hackernoon.com/how-does-http-authentication-work-ys1c3yph.

37. Adaptive Multi-Factor Authentication (MFA). URL:. https://www.cyberark.com/products/adaptive-multi-factor-authentication/.

38. S. Das, B. Wang, and L. Jean Camp.MFA is a Waste of Time! Understanding Negative Connotation Towards MFA Applications via User Generated Content.

39. Secure OAuth 2.0: How To Keep OAuth Secure?. URL:. https://www.securing.pl/en/secure-oauth-2-0-how-to-keep-oauth-secure/.

40. OAuth 2.0 Threat Model and Security Considerations. URL:. https://datatracker.ietf.org/doc/html/rfc6819.

41. How can you use OpenID Connect for web service authentication?. URL:. https://www.linkedin.com/advice/0/how-can-you-use-openid-connect-web-service-authentication.

42. How do you design and optimize SAML performance and scalability?. URL:. https://www.linkedin.com/advice/0/how-do-you-design-optimize-saml-performance-scalability.

43. Why You Wouldn’t Use SAML in a SPA and Mobile App. URL:. https://medium.com/@sundas.choudry/why-you-wouldnt-use-saml-in-a-spa-and-mobile-app-3890b5dd81e1.

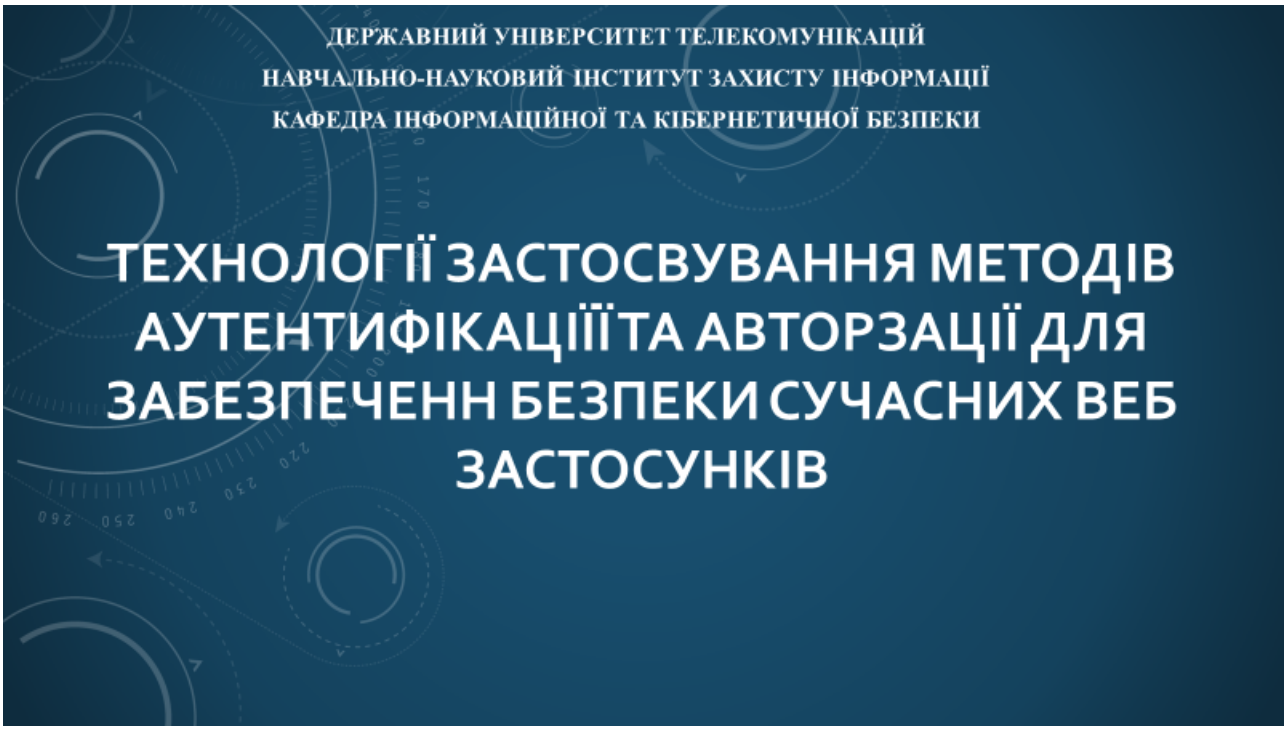
44. Introduction to JSON Web Tokens. URL:. hhttps://jwt.io/introduction/.

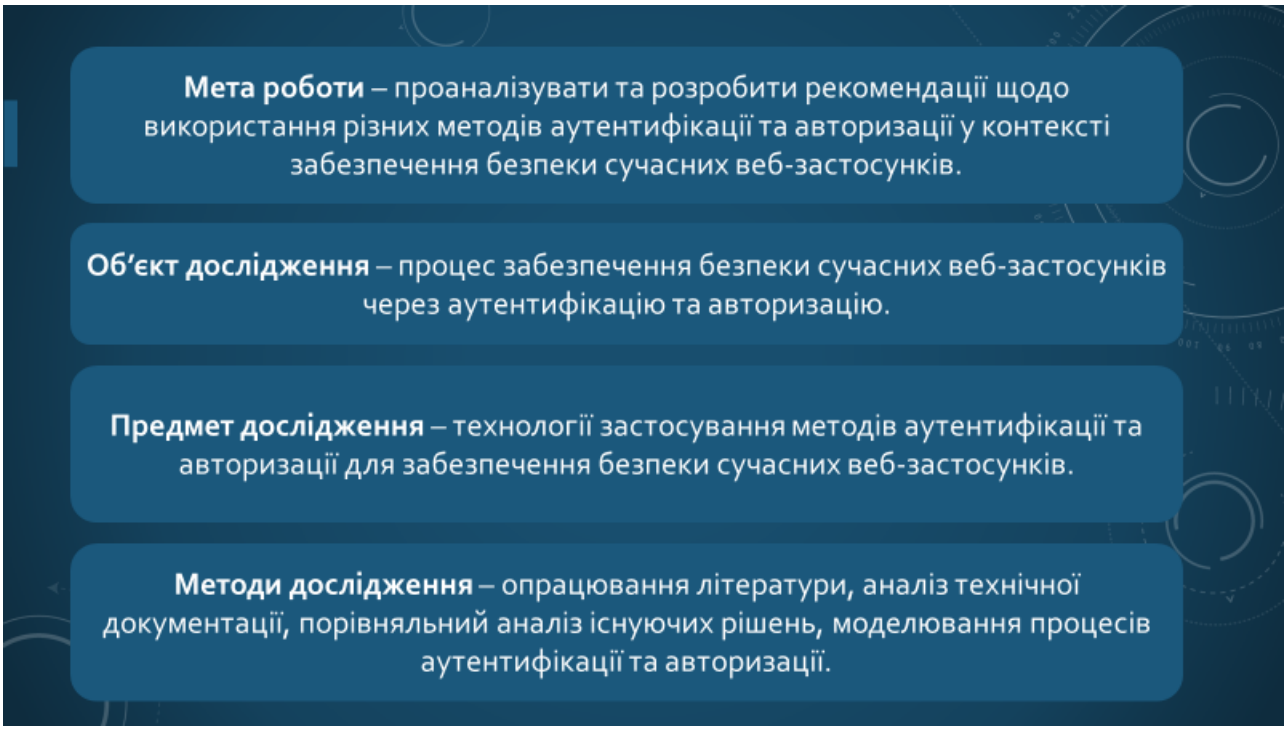
45. JWT Security Best Practices. URL:. https://curity.io/resources/learn/jwt-best-practices/.

46. HTTP authentication. URL:. https://developer.mozilla.org/en-US/docs/Web/HTTP/Authentication.

47. Session Management Cheat Sheet. URL:. https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html.

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ (Презентація)





ЗАВДАННЯ РОБОТИ

Огляд алгоритму роботи та ключових тверджень у процесах авторизації та аутентифікації;

Аналіз існуючих технологій авторизації та аутентифікації;

Огляд та варіанти вирішення актуальних проблем авторизації та аутентифікації;

Розробка порівняльної характеристики методів;

АВТОРИЗАЦІЯ

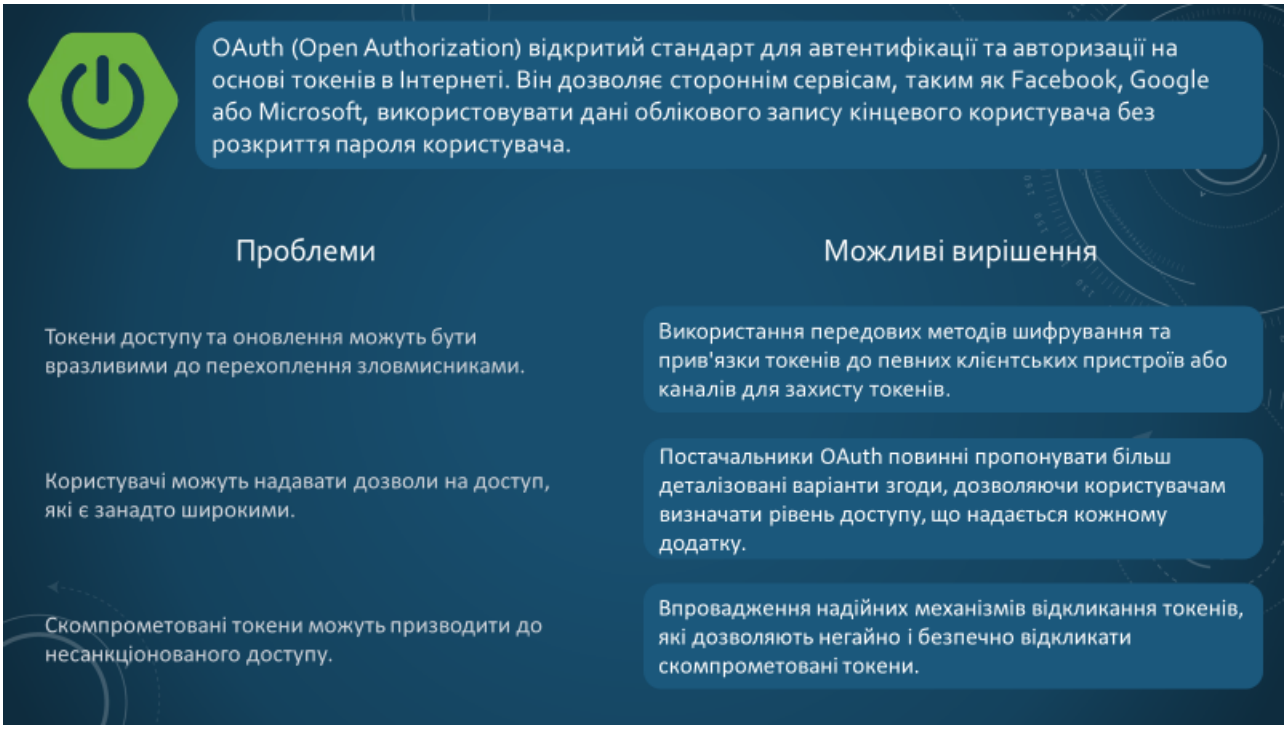
ХТО ВИ?

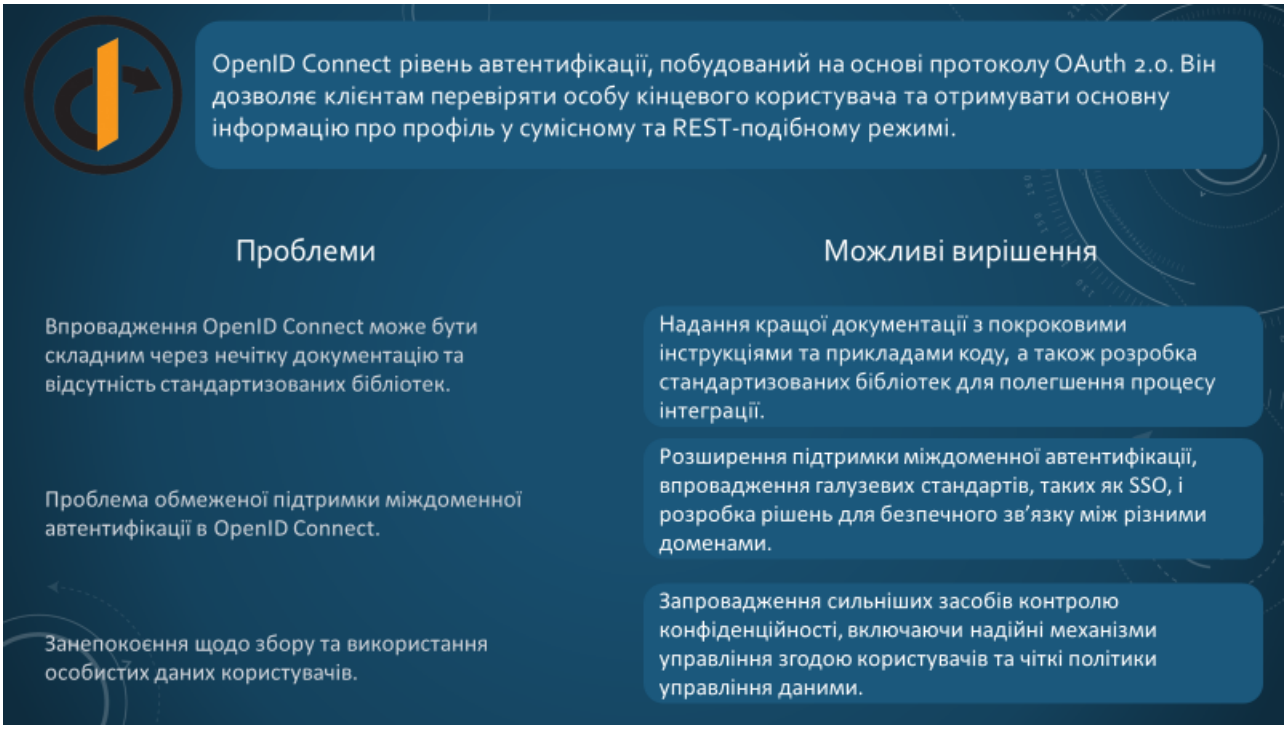
АУТЕНТИФІКАЦІЯ

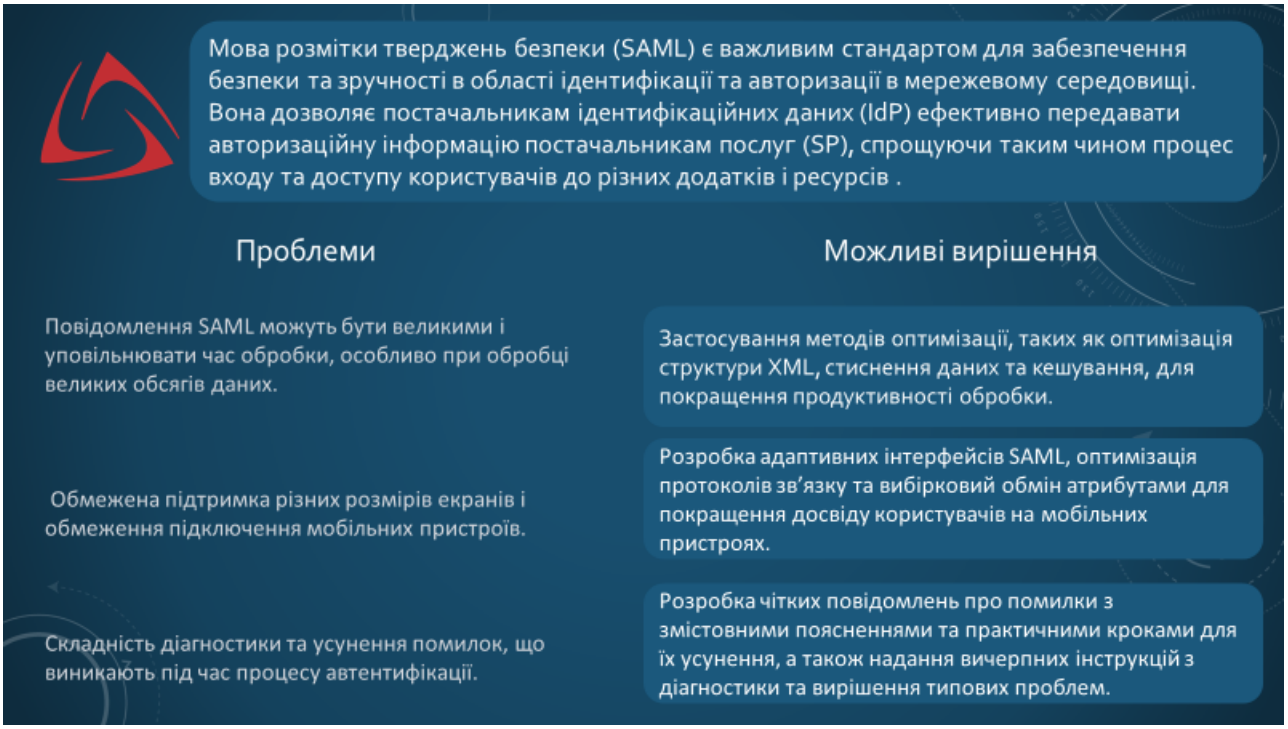
ЩО ВИ МОЖЕТЕ РОБИТИ?

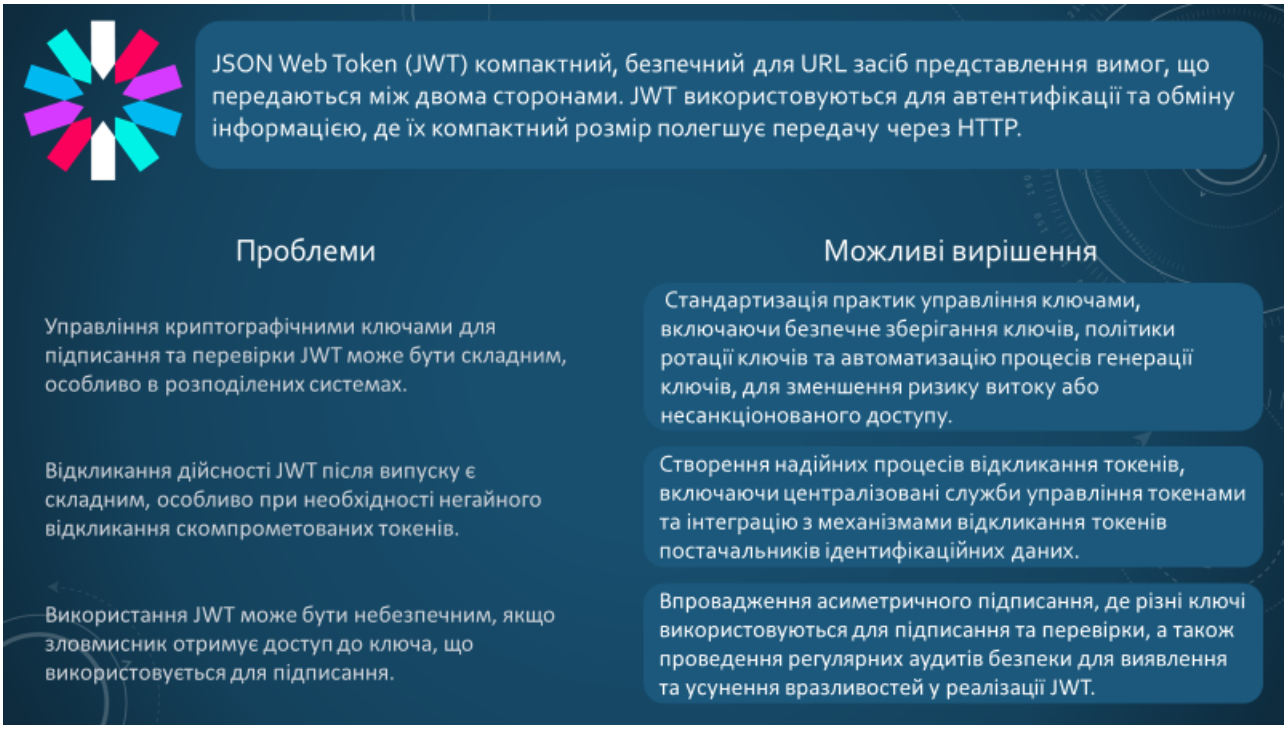
Багатофакторна автентифікація (MFA) це механізм безпеки, який вимагає від користувачів надання двох або більше факторів перевірки для отримання доступу до ресурсу

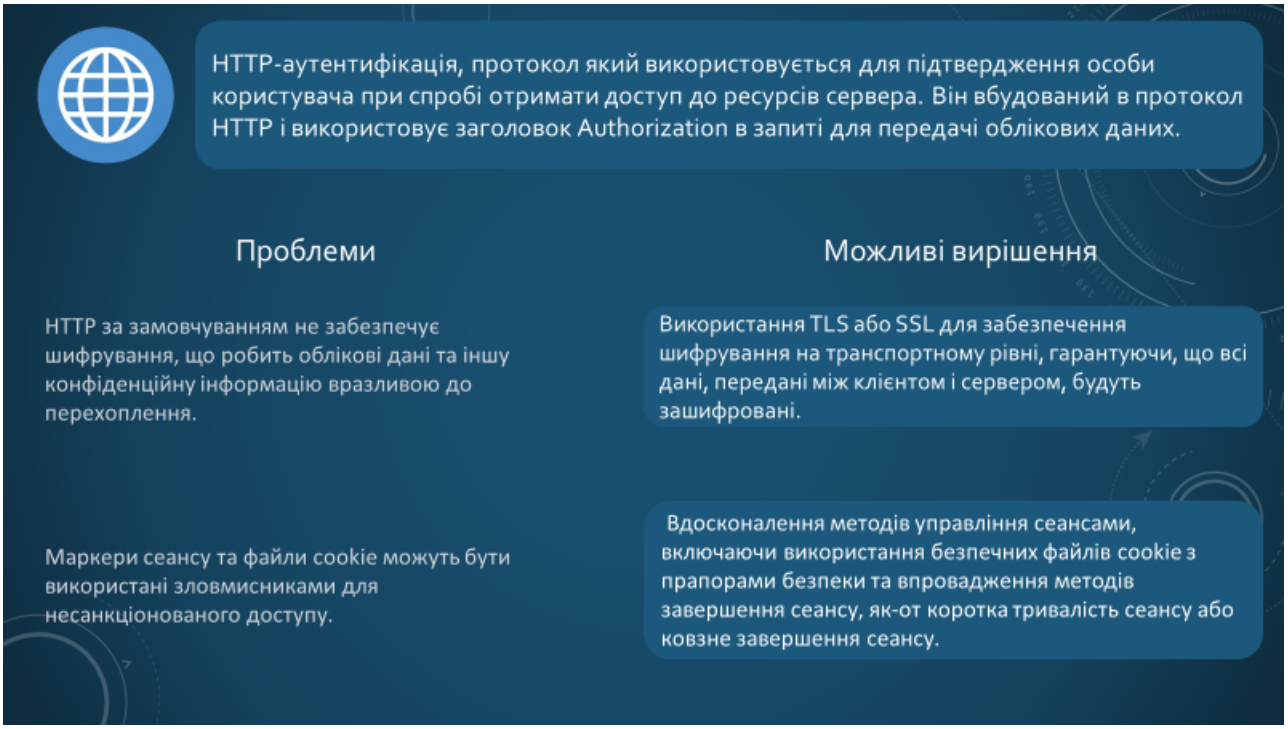
Проблеми	Можливі вирішення
Традиційні методи MFA можуть бути незручними для користувачів, оскільки вони вимагають запам'ятовування та введення декількох факторів.	Впровадження адаптивної MFA, яка використовує алгоритми машинного навчання для оптимізації процесу автентифікації в залежності від контексту.
Обмежені можливості резервного копіювання при втраті основних методів автентифікації, як-от фізичних токенів.	Розробка альтернативних методів, таких як одноразові коди через електронну пошту або SMS, або використання резервних ключів.
Залежність від фізичних пристроїв у MFA може призвести до їх втрати або пошкодження.	Використання віртуальних варіантів MFA, як-от мобільні додатки з біометричною автентифікацією або одноразовими паролями.


OAuth (Open Authorization) відкритий стандарт для автентифікації та авторизації на основі токенів в Інтернеті. Він дозволяє стороннім сервісам, таким як Facebook, Google або Microsoft, використовувати дані облікового запису кінцевого користувача без розкриття пароля користувача.
Проблеми
Можливі вирішення
Токени доступу та оновлення можуть бути вразливими до перехоплення зловмисниками.
Використання передових методів шифрування та прив'язки токенів до певних клієнтських пристроїв або каналів для захисту токенів.
Користувачі можуть надавати дозволи на доступ, які є занадто широкими.
Постачальники OAuth повинні пропонувати більш деталізовані варіанти згоди, дозволяючи користувачам визначати рівень доступу, що надається кожному додатку.
Скомпрометовані токени можуть призводити до несанкціонованого доступу.
Впровадження надійних механізмів відкликання токенів, які дозволяють негайно і безпечно відкликати скомпрометовані токени.


OpenID Connect рівень автентифікації, побудований на основі протоколу OAuth 2.0. Він дозволяє клієнтам перевіряти особу кінцевого користувача та отримувати основну інформацію про профіль у сумісному та REST-подібному режимі.
Проблеми
Можливі вирішення
Впровадження OpenID Connect може бути складним через нечітку документацію та відсутність стандартизованих бібліотек.
Надання кращої документації з покроковими інструкціями та прикладами коду, а також розробка стандартизованих бібліотек для полегшення процесу інтеграції.
Проблема обмеженої підтримки міждоменної автентифікації в OpenID Connect.
Розширення підтримки міждоменної автентифікації, впровадження галузевих стандартів, таких як SSO, і розробка рішень для безпечного зв'язку між різними доменами.
Занепокоєння щодо збору та використання особистих даних користувачів.
Запровадження сильніших засобів контролю конфіденційності, включаючи надійні механізми управління згодою користувачів та чіткі політики управління даними.


Мова розмітки тверджень безпеки (SAML) є важливим стандартом для забезпечення безпеки та зручності в області ідентифікації та авторизації в мережевому середовищі. Вона дозволяє постачальникам ідентифікаційних даних (IdP) ефективно передавати авторизаційну інформацію постачальникам послуг (SP), спрощуючи таким чином процес входу та доступу користувачів до різних додатків і ресурсів .
Проблеми
Можливі вирішення
Повідомлення SAML можуть бути великими і уповільнювати час обробки, особливо при обробці великих обсягів даних.
Застосування методів оптимізації, таких як оптимізація структури XML, стиснення даних та кешування, для покращення продуктивності обробки.
Обмежена підтримка різних розмірів екранів і обмеження підключення мобільних пристроїв.
Розробка адаптивних інтерфейсів SAML, оптимізація протоколів зв'язку та вибірковий обмін атрибутами для покращення досвіду користувачів на мобільних пристроях.
Складність діагностики та усунення помилок, що виникають під час процесу автентифікації.
Розробка чітких повідомлень про помилки з змістовними поясненнями та практичними кроками для їх усунення, а також надання вичерпних інструкцій з діагностики та вирішення типових проблем.


JSON Web Token (JWT) компактний, безпечний для URL засіб представлення вимог, що передаються між двома сторонами. JWT використовуються для автентифікації та обміну інформацією, де їх компактний розмір полегшує передачу через HTTP.
Проблеми
Можливі вирішення
Управління криптографічними ключами для підписання та перевірки JWT може бути складним, особливо в розподілених системах.
Стандартизація практик управління ключами, включаючи безпечне зберігання ключів, політики ротації ключів та автоматизацію процесів генерації ключів, для зменшення ризику витоку або несанкціонованого доступу.
Відкликання дійсності JWT після випуску є складним, особливо при необхідності негайного відкликання скомпрометованих токенів.
Створення надійних процесів відкликання токенів, включаючи централізовані служби управління токенами та інтеграцію з механізмами відкликання токенів постачальників ідентифікаційних даних.
Використання JWT може бути небезпечним, якщо зловмисник отримує доступ до ключа, що використовується для підписання.
Впровадження асиметричного підписання, де різні ключі використовуються для підписання та перевірки, а також проведення регулярних аудитів безпеки для виявлення та усунення вразливостей у реалізації JWT.



ПОРІВНЯННЯ МЕТОДІВ

Критерії	MFA	OAuth	OIDC	SAML	JWT	HTTP
Надійність безпеки	8	7	7	8	9	5
Зручність для користувача	7	8	8	6	7	6
Масштабованість	6	7	7	7	8	5
Можливість інтеграції	6	8	8	7	7	5
Вартісна ефективність	7	6	6	5	8	7
Адаптивність	7	7	7	6	8	5
Відповідність нормам	8	7	7	8	9	6
Конфіденційність даних	7	7	7	7	8	6
Аудит та моніторинг	7	8	8	7	8	6
Підтримка крос-платформ	6	8	8	6	7	5
Механізми відновлення	8	6	6	6	5	7
Управління токенами	5	7	7	8	9	4
Управління сесіями	6	7	7	7	8	5
Підтримка мультидоменів	5	8	8	7	6	4
Можливість налаштування	6	7	7	6	7	5