

# Введення до аутентифікації та авторизації.



**GENIUS**  
space

# Розрізнення між аутентифікацією та авторизацією.

**Аутентифікація та авторизація** - це два важливі поняття в інформаційній безпеці та управлінні доступом, і вони використовуються для забезпечення безпеки систем та ресурсів.

**Аутентифікація** - це процес перевірки ідентифікації користувача, тобто встановлення його особистості. Ваша ідентифікація - це спосіб, яким система перевіряє, хто ви є. Це може включати в себе такі речі, як логін, пароль, відбиток пальця, карта доступу, обличчя та інші методи. Коли ви вводите свій логін і пароль на веб-сайті або у програмі, система проводить аутентифікацію, щоб переконатися, що ви дійсно той, за кого себе видаєте.

**Авторизація** - це процес надання користувачеві доступу до певних ресурсів чи функцій після успішної аутентифікації. Якщо ви успішно пройшли аутентифікацію (довели свою ідентифікацію), це ще не означає, що ви можете мати доступ до всього в системі. Авторизація визначає, які ресурси або дії ви можете виконувати після входу. Наприклад, після входу на свій електронний поштовий акаунт, ви авторизовані для перегляду та відправлення електронних листів, але ви не можете редагувати налаштування інших користувачів.

**Аутентифікація** – це підтвердження того, ким є користувач **на вході**. Це проходження перевірки автентичності.

**Авторизація** – це те, що користувачу дозволяється робити **після входу**. Це надання і перевірка прав на вчинення будь-яких дій в системі.

## Як проходить процедура аутентифікації?

Наприклад, користувач хоче прочитати свої свіжі листи, що прийшли на email.

Зайдовши на сайт пошти, він поки що бачить лише рекламу і новини. Але свої листи він зможе прочитати лише після введення та відправки свого логіну та паролю. До цього моменту – ні-ні. Система ж не знає, хто він. Ось це і є процес **аутентифікації**.

Що може перевіряти система:

чи існує користувач з таким ім'ям,

чи збігається введений пароль з його обліковим записом.

може запитувати ще наявність сертифіката, IP-адресу або додатковий код верифікації.

Якщо користувач пройшов аутентифікацію – він дійсно той, ким здається.

Після цього починається процес **авторизації**. І тут з поштою все просто: всі можуть переглядати листи і документи, редагувати їх і створювати нові.

А ось в соцмережах, сайтах або на форумах, часто відвідувачі належать до певної групи.

Тут авторизація допомагає системі визначити, що дозволено тим чи іншим користувачам:

право писати повідомлення користувачу,

право додавати в повідомлення певні матеріали.

право переглядати фотографії інших користувачів.

право на редагування або копіювання документів.

Авторизація перевіряє наявність прав на конкретні дії. Це відбувається не тільки під час входу в систему, але і при будь-якій спробі вчинити будь-які маніпуляції з даними.

У цьому полягає відмінність аутентифікації від авторизації: перша – процедура одноразова для поточної сесії, другу користувач проходить постійно перед запуском будь-якого процесу.

Процес аутентифікації запускається користувачем при вході в систему: він надає ідентифікаційні дані: логін / пароль, відбиток пальця, встановлений сертифікат, карта і її PIN-код. При цьому можливі помилки з боку клієнта. Авторизація запускається сервером автоматично, якщо аутентифікація завершена успішно, і дії користувача на даний процес не впливають

Щоб запам'ятати краще, в чому різниця між аутентифікацією і авторизацією – уявіть закриту тематичну вечірку, де вхід лише за запрошенням. При вході гість пред'являє запрошення (в якому його ПІБ та інші дані), а охорона перевіряє, чи можна цю людину впустити. Якщо запрошення справжнє і прізвище є в списку – вхід дозволений, він пройшов **аутентифікацію**, вечір розпочався.

Та щоб вечірка вдалася, слід пройти **авторизацію**: це певні дозволи, дрескоди чи заборони і табу, встановлені організаторами вечірки, яких чітко треба дотримуватись.

# Токени аутентифікації та їх роль у веб-програмах.

Токени аутентифікації грають важливу роль у веб-програмах для забезпечення безпеки та контролю доступу.

**Токен аутентифікації** - це компактний об'єкт, який містить інформацію про користувача та його права, і який виданий системою після успішної аутентифікації користувача. Токени часто представляють собою унікальний рядок або числове значення, яке надсилається користувачу після входу в систему та використовується для подальшого звернення до ресурсів або послуг системи.

Роль токенів аутентифікації в веб-програмах:

1. Забезпечення безпеки: Токени допомагають зберегти паролі та інші конфіденційні дані від користувачів. Замість передачі пароля кожного разу, коли потрібно звернутися до ресурсів, використовується токен, що робить процес більш безпечним.
2. Управління сесією: Токени можуть мати обмежений строк дії, що дозволяє системі контролювати тривалість сеансу користувача. Після закінчення строку дії токена користувач повинен повторно аутентифікуватися.
3. Складний доступ: Токени можуть містити інформацію про ролі та дозволи користувача. Це дозволяє системі точно керувати тим, до яких ресурсів або функцій має доступ користувач.

4. Один вхід (Single Sign-On, SSO): Токени можуть бути використані для реалізації SSO, де користувач може увійти в одну систему і автоматично отримувати доступ до інших систем без повторної аутентифікації.
5. Децентралізований доступ: Токени дозволяють користувачам отримувати доступ до ресурсів через API інших сервісів. Це особливо важливо в сучасних веб-додатах, де ресурси можуть бути розподілені на різних серверах або службах.
6. Легкість інтеграції: Токени дозволяють інтегрувати авторизацію та контроль доступу в існуючі додатки і послуги, не потребуючи повноцінного обміну користувальницькими інформацією.

Існує кілька типів токенів аутентифікації, які використовуються в різних контекстах та сценаріях:

**Токени на основі сесії (Session-based tokens):** Цей тип токенів використовується для аутентифікації на веб-сайтах і веб-додатках. Після успішної аутентифікації, сервер створює унікальний ідентифікатор сесії, який зберігається в токені аутентифікації (зазвичай у вигляді куکі або заголовка запиту). Цей токен надсилається на сервер з кожним наступним питанням, щоб підтвердити ідентичність користувача. Сервер перевіряє та зберігає інформацію про активну сесію.

**Токени на основі токенів доступу (Access tokens):** Цей тип токенів широко використовується в API і служить для аутентифікації та авторизації запитів. Після успішної аутентифікації, сервер видає токен доступу, який містить інформацію про ідентичність користувача або програми, а також список дозволених дій. Цей токен надсилається разом з кожним API-запитом для підтвердження ідентичності та отримання доступу до ресурсів.

**JSON Web Tokens (JWT):** JWT є відкритим стандартом для створення токенів аутентифікації та передачі інформації між сторонами у безпечному форматі. JWT складається з трьох частин: заголовка, заяви та підпису. В заявлі можна включити додаткові властивості, такі як ідентифікатор користувача, роль або термін дії токена. JWT часто використовується для аутентифікації та авторизації веб-додатків, мікро сервісів та систем однієї і тієї ж довіри.

**Токени на основі одноразових паролів (One-Time Password tokens):** Цей тип токенів використовується для забезпечення додаткового рівня безпеки під час аутентифікації. Одноразовий пароль генерується на сервері або на пристрії користувача та надсилається йому для введення під час аутентифікації. Цей токен діє лише один раз та застосовується для запобігання повторного використання паролів.

Ось як відбувається аутентифікація на основі токенів:

- Користувач надає свої облікові дані (наприклад, ім'я та пароль).
- Сервер видає токен доступу, який містить інформацію про ідентичність користувача та дозволи на доступ до ресурсів. Цей токен надсилається користувачу для подальшого використання.
- Користувач передає токен доступу на сервер з кожним запитом до захищених ресурсів або послуг. Це може відбуватись через заголовок запиту, параметр URL або в тілі запиту, залежно від використаного протоколу та API.
- Сервер перевіряє та декодує токен доступу для перевірки його валідності, цілісності та авторизаційних даних. Якщо токен є дійсним та містить необхідні дозволи, сервер надає доступ користувачу до ресурсу або послуги, до якої генерувався запит.
- Токен доступу може мати обмежений термін дії. Після закінчення терміну дії токена користувач повинен повторно пройти аутентифікацію для отримання нового токена доступу.

Крім цих основних типів токенів, є також **фізичні токени авторизації** – це пристрої, які зазвичай мають форму картки, ключа або пластикового пристроя, та можуть бути активовані кнопкою або спеціальними сенсорами.

Переваги використання токенів авторизації:

**Безпека**: токени авторизації можуть бути шифрованими та підписаними, що забезпечує безпеку при передачі даних. Це дозволяє запобігти підробці токенів та несанкціонованому доступу до ресурсів.

**Безпарольна аутентифікація**: токени дозволяють уникнути потреби у введенні пароля кожного разу під час аутентифікації. Це зручно для користувачів, оскільки вони можуть отримати токен доступу після першої аутентифікації та використовувати його для подальшого доступу без повторного введення пароля.

**Масштабованість**: токени дозволяють розподілити навантаження на сервер, оскільки перевірка токенів може здійснюватися безпосередньо на ресурсі або сервісі, що надає доступ. Це особливо важливо в розподілених системах або мікросервісній архітектурі.

**Гнучкість**: токени надають можливість контролювати рівні доступу та дозволи користувачів до ресурсів. Вони можуть містити додаткові дані, такі як ідентифікатор користувача, роль, термін дії токену тощо, що дозволяє гнучкіше налаштовувати систему авторизації.

Недоліки та основні вразливості:

**Потенційна крадіжка токенів**: якщо токен доступу потрапить в несанкціоновані руки, зловмисники можуть мати доступ до ресурсів та інформації. Тому важливо добре захищати токени та використовувати надійні методи передачі токенів.

**Управління токенами**: при використанні токенів необхідно розробляти механізми управління їхнім життєвим циклом. Це включає виділення, оновлення та скасування токенів, а також перевірку їхньої валідності та контроль доступу.

**Потенційне збільшення обсягу даних**: токени, особливо JWT, можуть містити багато додаткової інформації, що призводить до збільшення обсягу переданих даних. Це може вплинути на продуктивність мережі та швидкість передачі даних.

**Залежність від сервера для перевірки токенів**: для перевірки токена доступу серверу необхідно звертатися до відповідного сервісу або бази даних. Це може створити додаткове навантаження на сервер та збільшити час відповіді.

**XSS-атаки**: вразливості типу Cross-Site Scripting (XSS) можуть дозволити зловмисникам отримати доступ до токенів, які зберігаються на стороні клієнта, та використовувати їх для отримання доступу до ресурсів або здійснення несанкціонованих дій під ім'ям користувача.

**CSRF-атаки**: вразливості типу Cross-Site Request Forgery (CSRF) можуть використовуватися для передачі недостовірних запитів зловмисником, який має доступ до токенів. Це може привести до виконання несанкціонованих дій під ім'ям аутентифікованого користувача.