# МЕХАНІЗМИ БЕЗПЕКИ У МІКРОСЕРВІСНІЙ АРХІТЕКТУРІ

Виконав студент ІПЗ БП-4
Загривий Олег Сергійович

Науковий керівник
асистент кафедри, Андрощук Максим Віталійович

# Актуальність

Незупинний розвиток програмної індустрії: продовження покращення ефективності реалізацій мікросервісних архітектур завдяки новим можливостям обладнання.

Зростання використання мікросервісних архітектур в індустрії.

Збільшення кількості застосунків на ринку, в тому числі мікросервісних.

# Постановка задачі

Проблеми:

- Багато точок входу для потенційних атак у мікросервісних системах.
- Ризики пов’язані з архітектурою (комунікація, керування доступом, захист даних).

Мета:

- Дослідити різні аспекти безпеки в мікросервісних архітектурах.
- Розробити мікросервісний застосунок, забезпечений широким спектром заходів безпеки.

Завдання:

- Вивчити важливість та складність впровадження заходів безпеки у розподілених системах.
- Описати основні механізми забезпечення безпеки в мікросервісних архітектурах.
- Спланувати розробку промислового мікросервісного застосунку, включаючи вибір мов програмування та технологій.
- Розробити застосунок, починаючи з монолітної архітектури та завершуючи мікросервісною, з імплементацією заходів безпеки.
- Створити програмний продукт, забезпечений широким спектром заходів безпеки.

# Дослідження механізмів забезпечення безпеки

Захист та аутентифікація

1. Захист на різних рівнях
   - Розгляд різних шарів системи та визначення методів захисту на кожному рівні.
2. Аутентифікація та авторизація
   - Аутентифікація: Перевірка ідентичності користувача перед наданням доступу.
   - Авторизація: Керування правами доступу до ресурсів системи.
3. Логування та моніторинг безпеки
   - Запис подій та моніторинг системи для виявлення потенційних загроз.

# Дослідження механізмів забезпечення безпеки

Захист баз даних та стандарти

1. Захист баз даних
   - Застосування методів захисту даних в базі даних для запобігання несанкціонованому доступу та втраті інформації.

2. Compliance, Standards
   - Відповідність стандартам та вимогам щодо безпеки та конфіденційності даних (наприклад, GDPR, HIPAA).

# Дослідження механізмів забезпечення безпеки

Шифрування та захист даних

1. Шифрування
   - Застосування шифрування для захисту конфіденційності даних під час передачі та зберігання.
2. Аналіз інцидентів
   - Виявлення та аналіз інцидентів безпеки для вдосконалення заходів захисту.
3. Зберігання резервних копій
   - Регулярне створення та зберігання резервних копій для відновлення даних у випадку втрати чи пошкодження.
4. Фізичний захист
   - Захист апаратного забезпечення та інфраструктури від несанкціонованого доступу.

# Планування розробки застосунку

Для створення було обрано промисловий застосунок, тобто, C2 за рівнем NCSC (не військова система, власник даних відповідає за доступ).

Вибір мови програмування та технологій:

- Java: Популярна мова для розробки серверних додатків
- Spring Boot: Фреймворк, що спрощує створення та розгортання Java-додатків, забезпечуючи автоматичну конфігурацію та швидкий старт проєктів.

Вибір бази даних:

- Development – H2 Database: Вбудована БД для розробки та тестування
- Production – PostgreSQL: Високопродуктивна БД для production.

# Модель бази даних

Модель обрано для розробки з актуальним розбиттям на мікросервіси.

Модель: Книжковий магазин

Функціональність:

- Зберігання книг: Управління інвентарем та наявністю книг.
- Користувачі: контроль над користувачами та їх можливостями.
- Замовлення книг: Обробка замовлень та управління кошиком покупок.
- Оплата замовлень: Обробка платежів та управління статусом оплат.

# Розробка застосунку

Створення моноліту:

- Початкова розробка застосунку як єдиної монолітної архітектури.

Перехід від моноліту до мікросервісів:

- Розбиття на сервіси:
  - Виділення окремих сервісів з моноліту.
  - Додавання шлюзу (Spring Cloud Gateway) для контролю спілкування до сервісів та між ними.
- Забезпечення коректної роботи сервісів:
  - Тестування та інтеграція нових сервісів для забезпечення їхньої надійності та сумісності.

Контейнеризація:

- Використання Docker для контейнеризації сервісів.
- Управління та координація контейнерів здійснюється за допомогою Docker Compose для спрощення розгортання та керування контейнеризованими додатками.

# Аутентифікація та авторизація

Аутентифікація:

- Реалізовано створення та перевірка JWT токенів на сервісі auth-service.
- JWT токен генерується з клеймами та нечутливими даними користувача, ролі користувача включно.
- Ключ для підпису токена передається через docker-compose та зберігається у .env файлі для забезпечення безпеки.

Авторизація:

- Додано секретний ключ до шлюзу для перевірки JWT токенів, що дає можливість ефективної перевірки токенів без необхідності звертатися до авторизаційного сервісу.
- Впроваджено механізм виходу з акаунту на шлюзі, який зберігає токени що були вже використані, і періодично видаляє їх за датою закінчення терміну дії
- Фільтрація маршрутів у шлюзі налаштована залежно від ролей, що містяться у JWT токені, для контролю доступу до ресурсів.

Transport Layer Security:

- Впроваджено TLS termination для забезпечення безпеки комунікації між сервісами.
- Створено та налаштовано keystore для зберігання сертифікатів та приватних ключів, що використовуються для TLS/SSL.

# Захист від атак

Види атак та встановлені методи захисту:

1. DDoS/DoS:
   - Фільтрація та обмеження трафіку: На рівні шлюзу розроблено механізм, який відслідковує та аналізує кількість запитів від користувачів, і блокує подальші запити при перевищенні порогу.
2. Bruteforce:
   - Обмеження спроб входу та блокування: Розроблено механізм занесення IP адресів користувачів до чорного списку після 5 невдалих спроб введення пароля, та очищення після 24 годин.
3. SQL Injection:
   - Використання параметризованих запитів та ORM: Для захисту від SQL Injection застосовано підготовлені запити та методи Spring Data JPA, що автоматично екранують спеціальні символи, забезпечуючи захист бази даних від шкідливих SQL-запитів.

# Захист від атак

Види атак та встановлені методи захисту:

4. Session fixation:
    - Використання JWT з шифруванням, контроль дати закінчення дії (expiration date) та механізму виходу для ускладнення атаки.
5. Sensitive data exposure:
    - Заборона доступу та захист конфіденційних даних: Блокування доступу до ресурсів з конфіденційною інформацією для авторизованих/не авторизованих користувачів. Використання шару DTO для контролю доступу до чутливих даних.
6. Man-in-the-Middle:
    - Обмеження доступу та захист комунікації: Обмеження доступу лише до шлюзу та використання HTTPS та TLS для забезпечення безпеки комунікації між клієнтами та сервісами.
7. Zero-day exploits:
    - Стабільне програмне забезпечення та швидка реакція: Використання стабільного ПЗ та його залежностей, обмеження версій, і швидка реакція на виявлення вразливостей для захисту від атак Zero-day exploits.

# Висновки

**Результати**

- Аналіз мікросервісної архітектури:
  - Проаналізовано мікросервісну архітектуру та її вразливості безпеки.
  - Розроблено промисловий мікросервісний застосунок з урахуванням кращих практик та заходів безпеки.
- Аналіз вразливостей:
  - Проведено докладний аналіз вразливостей розробленого застосунку.
  - Виявлено слабкі місця та потенційні загрози для безпеки системи.
- Заходи забезпечення безпеки:
  - Розроблено та впроваджено заходи для мінімізації ризиків.
  - Захищено систему від потенційних атак з мінімізацією втрати продуктивності.

**Подальші шляхи розвитку**

- Логування та моніторинг:
  - Впровадження повної системи логування та моніторингу для ефективного виявлення помилок та критичних ситуацій.
- Аналіз коду:
  - Проведення статичного та динамічного аналізу коду з метою виявлення потенційних вразливостей та їх усунення.
- Відмовостійкість:
  - Впровадження відмовостійкості та тестування під час масштабування сервісів для забезпечення стабільної роботи системи.

ДЯКУЮ ЗА УВАГУ