Лабораторна робота №3
«Дослідження безпечної реалізації та експлуатації децентралізованих додатків»
Абдуллаєва Есміра, Товстенко Артем, Пашинський Максим ФБ-31мп

Мета роботи: отримання навичок роботи із децентралізованими додатками та оцінка безпеки інформації при їх функціонуванні

Завдання:

дослідження вимог OWASP (безпека web-додатків) та складання аналогічних вимог для обраної системи децентралізованих додатків.

Хід роботи

OWASP (Open Web Application Security Project) - це некомерційна організація, яка займається покращенням безпеки програмного забезпечення. OWASP Top 10 - це список десяти найкритичніших ризиків безпеки для веб-додатків. Цей список є цінним ресурсом для розробників та власників веб-додатків, які прагнуть покращити безпеку своїх систем.

1. Дослідження вимог OWASP 10 2021

A01:2021 – Broken Access Control

Неправильний контроль доступу - це ризик, який виникає, коли користувачі отримують доступ до ресурсів, до яких вони не мають права. Це може призвести до того, що зловмисники зможуть красти дані, змінювати інформацію або навіть повністю захопити веб-сайт.

Як виникає:

- Відсутність належних обмежень на рівні API або інтерфейсу користувача.
- Використання дефолтних конфігурацій безпеки.
- Неправильне управління правами користувачів.

Приклади:

- Користувач без адміністративних прав може отримати доступ до адміністративного інтерфейсу.
- Виконання заборонених дій через маніпуляцію URL-адресою або параметрами запиту.

Запобігання:

- Використовувати моделі контролю доступу на основі ролей або атрибутів.
- Перевіряти та обмежувати доступ до ресурсів на сервері.
- Проведення тестування на проникнення та ревізії коду для виявлення вразливостей.

A02:2021 – Cryptographic Failures

Криптографічні помилки - це ризик, який виникає, коли криптографічні алгоритми або протоколи використовуються неправильно. Це може призвести до того, що зловмисники зможуть перехопити дані, розшифрувати паролі або підробити цифрові підписи.

Як виникає:

- Використання застарілих або небезпечних криптографічних алгоритмів.
- Недостатня захищеність ключів шифрування.
- Відсутність належного керування ключами.

Приклади:

- Використання MD5 або SHA1 для хешування паролів.
- Відправка конфіденційних даних через незашифровані канали зв’язку (HTTP замість HTTPS).

Запобігання:

- Використовувати сучасні та перевірені криптографічні алгоритми.
- Впровадити надійне керування ключами.
- Зашифровувати всі конфіденційні дані, що передаються через мережу.

A03:2021 – Injection

Введення даних - це ризик, який виникає, коли зловмисні дані вводяться в веб-додаток. Це може призвести до того, що зловмисники зможуть виконати довільний код, отримати доступ до конфіденційних даних або навіть повністю захопити веб-сайт.

Як виникає:

- Відсутність належного фільтрування та валідації вводу користувачів.
- Динамічне складання запитів з вводу користувачів без використання параметризованих запитів.

Приклади:

- SQL-ін’єкції, що дозволяють зловмиснику виконувати довільні SQL-команди.
- Ін’єкції команд ОС, що дозволяють виконувати системні команди.

Запобігання:

- Використовувати параметризовані запити або підготовлені вирази.
- Валідувати та фільтрувати ввід користувачів.
- Використовувати ORM (Object-Relational Mapping) для взаємодії з базами даних.

A04:2021 – Insecure Design

Небезпечний дизайн - це ризик, який виникає, коли веб-додаток розроблений таким чином, що робить його вразливим до атак. Це може призвести до того, що зловмисникам буде легко знайти та експлуатувати вразливості.

Як виникає:

- Недостатній аналіз загроз на етапі проектування.
- Відсутність безпечних паттернів та практик розробки.

Приклади:

- Відсутність багатофакторної аутентифікації для критичних операцій.
- Недостатній захист даних, що передаються або зберігаються.

Запобігання:

- Впроваджувати методології Secure Development Lifecycle (SDLC).
- Використовувати безпечні паттерни проектування.
- Проводити регулярні ревізії дизайну та архітектури з урахуванням безпеки.

A05:2021 – Security Misconfiguration

Неправильна конфігурація безпеки - це ризик, який виникає, коли веб-додаток не налаштований належним чином для забезпечення безпеки. Це може призвести до того, що зловмисникам буде легко знайти та експлуатувати вразливості.

Як виникає:

- Використання дефолтних налаштувань безпеки.
- Неправильна конфігурація серверів, баз даних або інших компонентів.

Приклади:

- Використання дефолтних облікових записів та паролів.
- Відкриті порти або сервіси, що не потрібні для роботи додатку.

Запобігання:

- Виконувати регулярний аудит конфігурацій.

- Використовувати інструменти автоматизованої конфігурації та управління.
- Вимикати непотрібні сервіси та порти.

A06:2021 – Vulnerable and Outdated Components

Вразливі та застарілі компоненти - це ризик, який виникає, коли у веб-додатку використовуються компоненти, які мають відомі вразливості. Це може призвести до того, що зловмисникам буде легко знайти та експлуатувати вразливості.

Як виникає:

- Використання застарілих бібліотек та фреймворків.
- Відсутність моніторингу та оновлення компонентів.

Приклади:

- Використання застарілих версій бібліотек з відомими вразливостями.
- Використання плагінів або модулів з невідомих джерел.

Запобігання:

- Використовувати інструменти для моніторингу вразливостей в компонентах.
- Регулярно оновлювати компоненти та бібліотеки.
- Використовувати лише перевірені джерела для завантаження компонентів.

A07:2021 – Identification and Authentication Failures

Помилки ідентифікації та автентифікації - це ризик, який виникає, коли веб-додаток не може правильно ідентифікувати або автентифікувати користувачів. Це може призвести до того, що зловмисники зможуть отримати доступ до несанкціонованих ресурсів або видати себе за законних користувачів.

Як виникає:

- Використання слабких або легко відгадуваних паролів.
- Відсутність належного управління сесіями.

Приклади:

- Використання слабких паролів без вимог до складності.
- Недостатній захист токенів сесій.

Запобігання:

- Впроваджувати багатофакторну аутентифікацію.
- Вимагати складні паролі та регулярно їх оновлювати.
- Використовувати безпечні методи управління сесіями.

A08:2021 – Software and Data Integrity Failures

Помилки цілісності програмного забезпечення та даних - це ризик, який виникає, коли цілісність програмного забезпечення або даних не може бути гарантована. Це може призвести до того, що зловмисники зможуть змінити код або дані, що призведе до непередбачуваних наслідків.

Як виникає:

- Відсутність перевірки цілісності при завантаженні або оновленні програмного забезпечення.
- Відсутність механізмів верифікації даних.

Приклади:

- Використання програмного забезпечення з недостовірних джерел.
- Відсутність перевірки цифрових підписів.

Запобігання:

- Використовувати цифрові підписи для перевірки цілісності програмного забезпечення.
- Впроваджувати механізми верифікації даних.
- Забезпечувати контроль доступу до даних та програмного забезпечення.

A09:2021 – Security Logging and Monitoring Failures

Помилки ведення журналів та моніторингу - це ризик, який виникає, коли події безпеки не реєструються або не моніторяться належним чином. Це може ускладнити або унеможливити виявлення атак, а також розслідування їх причин.

Як виникає:

- Відсутність належного логування подій безпеки.
- Відсутність інструментів для моніторингу та аналізу логів.

Приклади:

- Відсутність логування невдалих спроб аутентифікації.
- Недостатній моніторинг критичних подій.

Запобігання:

- Впроваджувати належне логування всіх критичних подій.
- Використовувати інструменти моніторингу та аналізу логів.
- Регулярно перевіряти та аналізувати логи на предмет виявлення інцидентів безпеки.

A10:2021 – Server-Side Request Forgery (SSRF)

Підробка запитів з серверного боку - це ризик, який виникає, коли веб-додаток завантажує віддалений ресурс без належного валідування URL-адреси, що запитується. Це може дозволити зловмисникам сканувати внутрішню мережу, отримувати доступ до локальних файлів на сервері або виконувати інші шкідливі дії.

Як виникає:

- Відсутність належної перевірки та фільтрації вводу користувачів, що використовується для створення запитів.

Приклади:

- Зловмисник може змусити сервер зробити запит до внутрішньої мережі або зовнішнього сервера, розкриваючи конфіденційні дані.

Запобігання:

- Обмежити можливість серверу робити запити до небезпечних ресурсів.
- Використовувати належну перевірку та фільтрацію вводу користувачів.
- Впроваджувати списки дозволених та заборонених ресурсів для запитів.

2. Вимоги до безпеки для децентралізованих додатків

Децентралізовані додатки (DApps) - це новий тип веб-додатків, які працюють на блокчейні. DApps мають ряд переваг перед традиційними веб-додатками, включаючи підвищену безпеку та прозорість. Однак DApps також мають свої унікальні ризики безпеки.

OWASP Top 10 можна використовувати як відправну точку для дослідження вимог безпеки DApps. Кожен з десяти ризиків OWASP Top 10 має бути оцінений з точки зору його релевантності для DApps. Для ризиків, які вважаються релевантними, мають бути розроблені конкретні вимоги безпеки.

Приклад децентралізованого додатку (DApp) - **децентралізована біржа (DEX)**. Децентралізована біржа дозволяє користувачам обмінювати криптовалюти без посередників, використовуючи смарт-контракти для автоматизації торгівлі.

1) **Контроль доступу до ресурсів (Access Control)**

Опис: Гарантування, що тільки авторизовані користувачі можуть виконувати певні операції на біржі.

Вимоги:

Смарт-контракти: Використовувати смарт-контракти для контролю доступу до функцій торгівлі.

Моделі доступу: Впровадити моделі контролю доступу на основі ролей (адміністратори, трейдери).

Перевірка прав: Забезпечити перевірку прав доступу на кожному етапі взаємодії з DEX.

2) **Безпека криптографії (Cryptographic Security)**

Опис: Захист даних користувачів шляхом правильної реалізації криптографічних алгоритмів.

Вимоги:

Сучасні алгоритми: Використовувати сучасні та перевірені криптографічні алгоритми для шифрування даних.

Захист ключів: Забезпечити належний захист криптографічних ключів та механізмів їх генерації.

Шифрування: Переконатися, що всі конфіденційні дані зашифровані як при зберіганні, так і при передачі.

3) **Запобігання ін’єкціям (Injection Prevention)**

Опис: Захист від ін’єкцій, які можуть призвести до виконання небажаних команд або доступу до даних.

Вимоги:

Валідація вводу: Виконувати ретельну валідацію та фільтрацію вводу користувачів.

Параметризовані запити: Використовувати параметризовані запити або підготовлені вирази.

Безпечні бібліотеки: Використовувати перевірені бібліотеки для обробки вводу.

4) **Безпека дизайну (Secure Design)**

Опис: Забезпечення безпеки на етапі проектування та архітектури DEX.

Вимоги:

SDLC: Впроваджувати методології Secure Development Lifecycle (SDLC).

Безпечні паттерни: Використовувати безпечні паттерни проектування.

Аналіз загроз: Проводити регулярний аналіз загроз та ревізії дизайну.

5) **Конфігурація безпеки (Security Configuration)**

Опис: Гарантування правильної конфігурації всіх компонентів DEX.

Вимоги:

Аудит конфігурацій: Виконувати регулярний аудит конфігурацій всіх компонентів DEX.

Автоматизація: Використовувати інструменти автоматизованої конфігурації та управління.

Смарт-контракти: Забезпечити правильну конфігурацію смарт-контрактів.

6) **Актуальність компонентів (Up-to-date Components)**

Опис: Використання актуальних версій компонентів для запобігання вразливостям.

Вимоги:

Оновлення: Регулярно оновлювати компоненти та бібліотеки DEX.

Моніторинг вразливостей: Використовувати інструменти для моніторингу вразливостей в компонентах.

Перевірені джерела: Завантажувати компоненти лише з перевірених джерел.

7) **Аутентифікація та управління сесіями (Authentication and Session Management)**

Опис: Забезпечення надійної аутентифікації користувачів та управління сесіями.

Вимоги:

Багатофакторна аутентифікація: Впроваджувати багатофакторну аутентифікацію для критичних дій.

Безпечні методи: Використовувати безпечні методи управління сесіями, такі як токени.

Складні паролі: Вимагати складні паролі та регулярне їх оновлення.

8) Цілісність даних та ПЗ (Data and Software Integrity)

Опис: Забезпечення цілісності даних та програмного забезпечення DEX.

Вимоги:

Цифрові підписи: Використовувати цифрові підписи для перевірки цілісності програмного забезпечення.

Верифікація даних: Впроваджувати механізми верифікації даних.

Контроль доступу: Забезпечити контроль доступу до даних та програмного забезпечення.

9) Логування та моніторинг (Logging and Monitoring)

Опис: Впровадження належного логування та моніторингу подій безпеки.

Вимоги:

Логування подій: Впроваджувати логування всіх критичних подій.

Інструменти моніторингу: Використовувати інструменти моніторингу та аналізу логів.

Аналіз логів: Регулярно перевіряти та аналізувати логи для виявлення інцидентів безпеки.

10) Захист від SSRF (SSRF Protection)

Опис: Запобігання можливості направлення небажаних запитів до внутрішніх ресурсів через DEX.

Вимоги:

Обмеження запитів: Обмежити можливість серверу робити запити до небезпечних ресурсів.

Валідація та фільтрація: Використовувати належну перевірку та фільтрацію вводу користувачів.

Білий список: Впроваджувати списки дозволених та заборонених ресурсів для запитів.

Висновок

В процесі дослідження було розглянуто вимоги OWASP Top Ten, які є критично важливими для забезпечення безпеки веб-додатків. Ці вимоги включають захист від різних типів атак та вразливостей, таких як зламаний контроль доступу, криптографічні помилки, ін’єкції, небезпечний дизайн, помилки конфігурації безпеки, використання застарілих компонентів, помилки ідентифікації та аутентифікації, порушення цілісності даних та програмного забезпечення, недоліки логування та моніторингу, а також підробка запитів з серверного боку (SSRF).

На основі цих вимог були розроблені аналогічні вимоги для децентралізованих додатків (DApps), зокрема для децентралізованої біржі (DEX). Запропоновані вимоги забезпечують високий рівень безпеки для децентралізованої біржі, використовуючи підхід, аналогічний OWASP Top Ten, але з урахуванням специфічних особливостей децентралізованих додатків. Це дозволяє знизити ризики та підвищити надійність і захищеність децентралізованих платформ у сучасному цифровому середовищі.