Евгений Бондаренко
Positive Technologies

Безопасная разработка во фронтенде

Евгений Бондаренко

Тимлид фронтенда PT NAD,
Positive Technologies

t.me/fyzlog

Обеспечиваем практическую кибербезопасность

20 лет

опыта исследований и разработок

1,5+ тыс.

сотрудников: инженеров по ИБ, разработчиков, аналитиков и других специалистов

250+

экспертов в нашем исследовательском центре безопасности

200+

обнаруженных уязвимостей нулевого дня в год

250+

аудитов безопасности корпоративных систем делаем ежегодно

50%

всех уязвимостей в промышленности и телекомах обнаружили наши эксперты

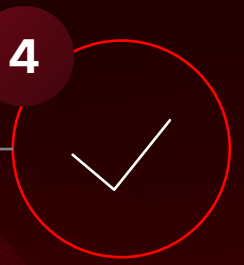
- Создаем продукты и решения
- Проводим аудиты безопасности
- Расследуем инциденты
- Исследуем угрозы

Что будет

1. 
Зачем нам нужна безопасная разработка

2. 
Как работает безопасная разработка

3. 
Безопасная разработка во фронтенде

4. 
Заключение

1

Зачем нам нужна безопасная разработка

Анализ защищенности РТ

86%

Компаний, участвовавших в пентестах

имеют уязвимости в коде веб-приложений

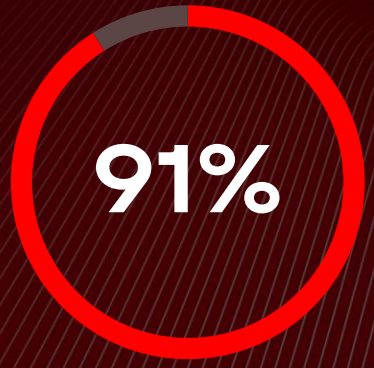
63%

Веб-приложений с уязвимостями

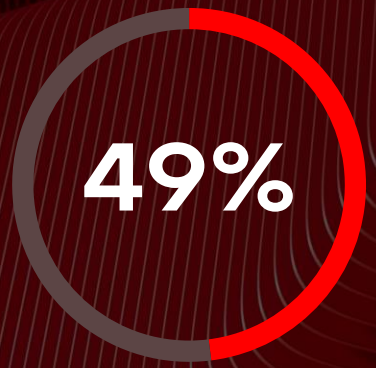
используются для проникновения во внутреннюю сеть компании

Уровень безопасности веб-приложений

Веб-приложений могут быть атакованы



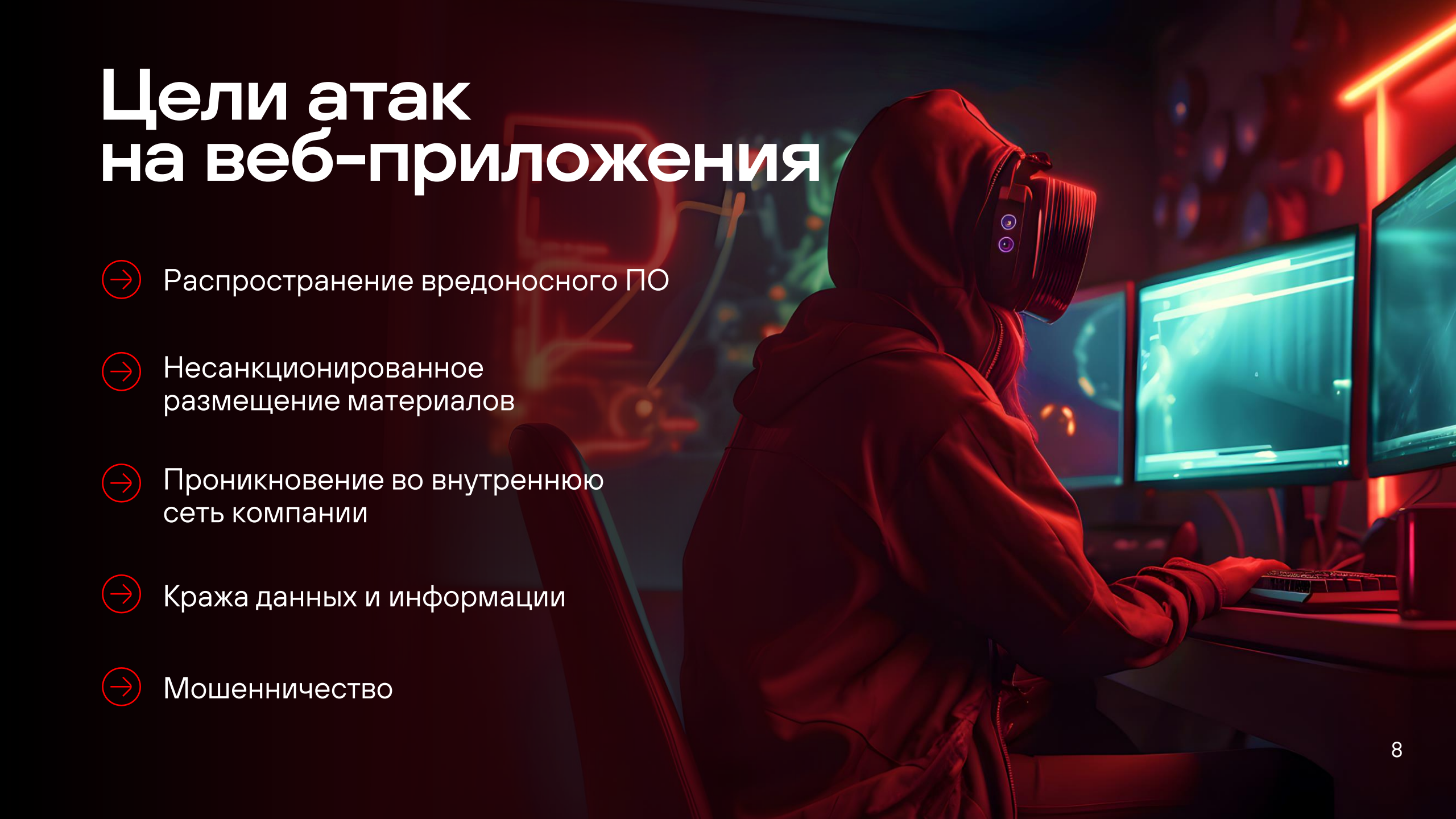
Нарушение конфиденциальности данных



Низкий и экстремально низкий уровни безопасности

Цели атак на веб-приложения

- Распространение вредоносного ПО
- Несанкционированное размещение материалов
- Проникновение во внутреннюю сеть компании
- Кража данных и информации
- Мошенничество



Зачем безопасная разработка во фронтенде

- Уменьшить риски для бизнеса
- Снизить затраты на устранение уязвимостей и последствий атак
- Повысить устойчивость против атак
- Управление инцидентами безопасности и реагированием на них
- Внедрение принципов и инструментов повышения надежности во все этапы разработки

в 30-60 раз выше

стоимость исправления уязвимости в продакшене, чем на этапе проектирования

© NIST

2

Как работает безопасная разработка

Безопасная разработка

Методика разработки приложений

- Устойчивость к атакам
- Обеспечение конфиденциальности, целостности и доступности данных
- Уменьшение уязвимостей через устранение багов и логических ошибок в коде

Принципы

- Отсутствие гарантий безопасности
- Глубинная защита
- Отказобезопасность
- Минимизация полномочий
- Разделение ответственности
- Упрощение механизмов безопасности
- Полное посредничество
- Публичная информация об архитектуре
- Минимизация общих процессов
- Психологическая приемлемость
- Самое слабое место
- Использование существующих компонентов

Software Development Lifecycle (SDLC)

Secure SDLC

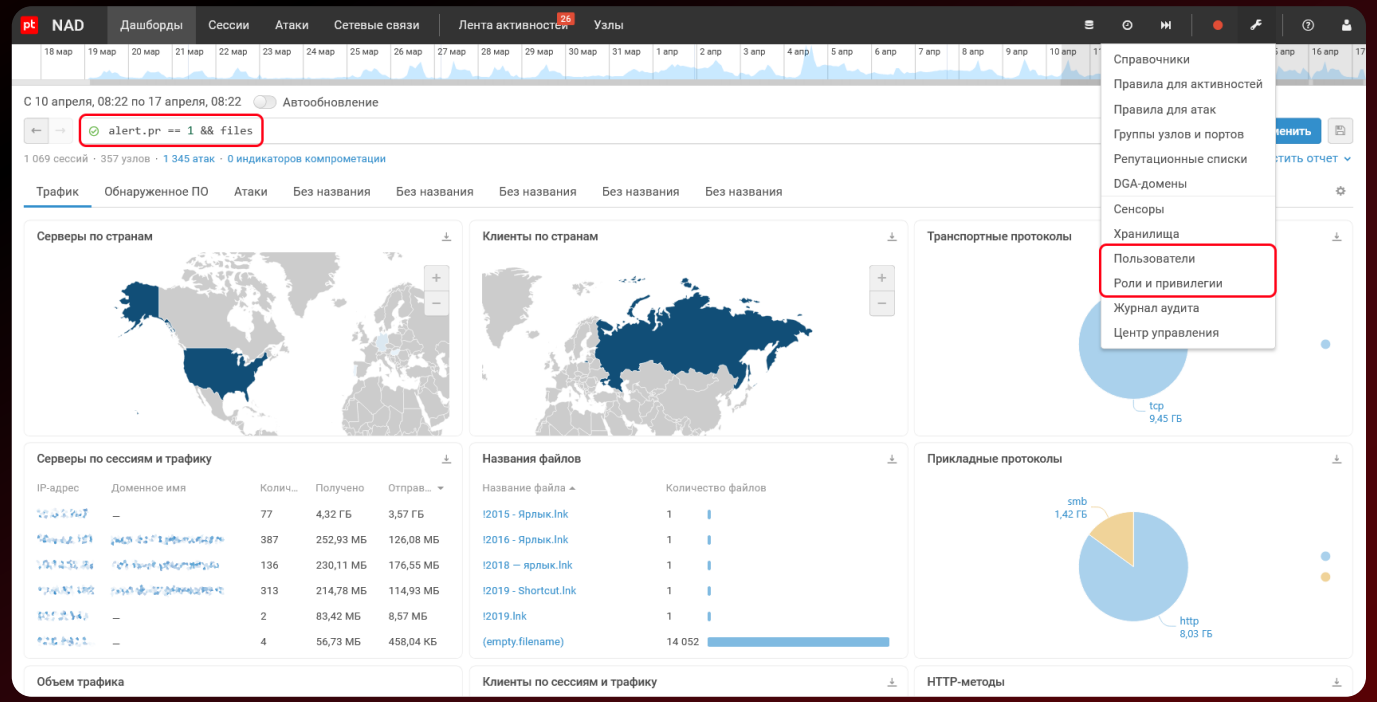
Требования 1 | Проекти-рование | Разработка | Тестиро-вание | Эксплуа-тация

Требования безопасности

- → Какие данные мы храним?
- → Как мы передаем данные?
- → Есть ли ограничения доступа?

Анализ рисков и угроз

- → Небезопасный ввод в фильтр
- → Операции с ограниченным доступом



Secure SDLC

Требования | Проектирование (2) | Разработка | Тестирование | Эксплуатация

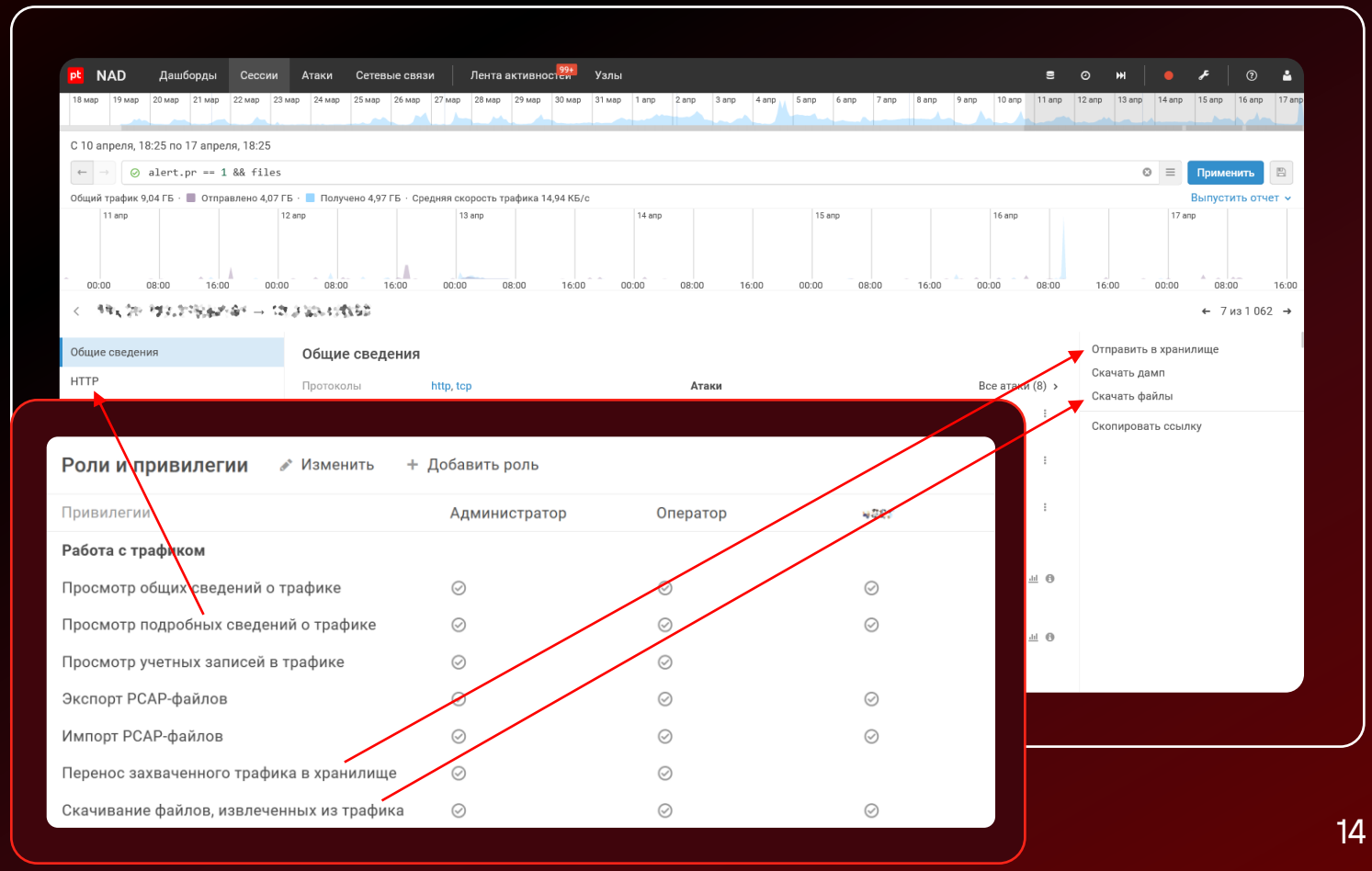
Учет требований безопасности

- Безопасное хранение и передача чувствительных данных
- Разграничение доступа к функциям и данным

Модели рисков и угроз

- Применение фильтра со зловредной нагрузкой
- Получение доступа к недоступным данным и разделам

Дизайн и ревью архитектуры



Secure SDLC

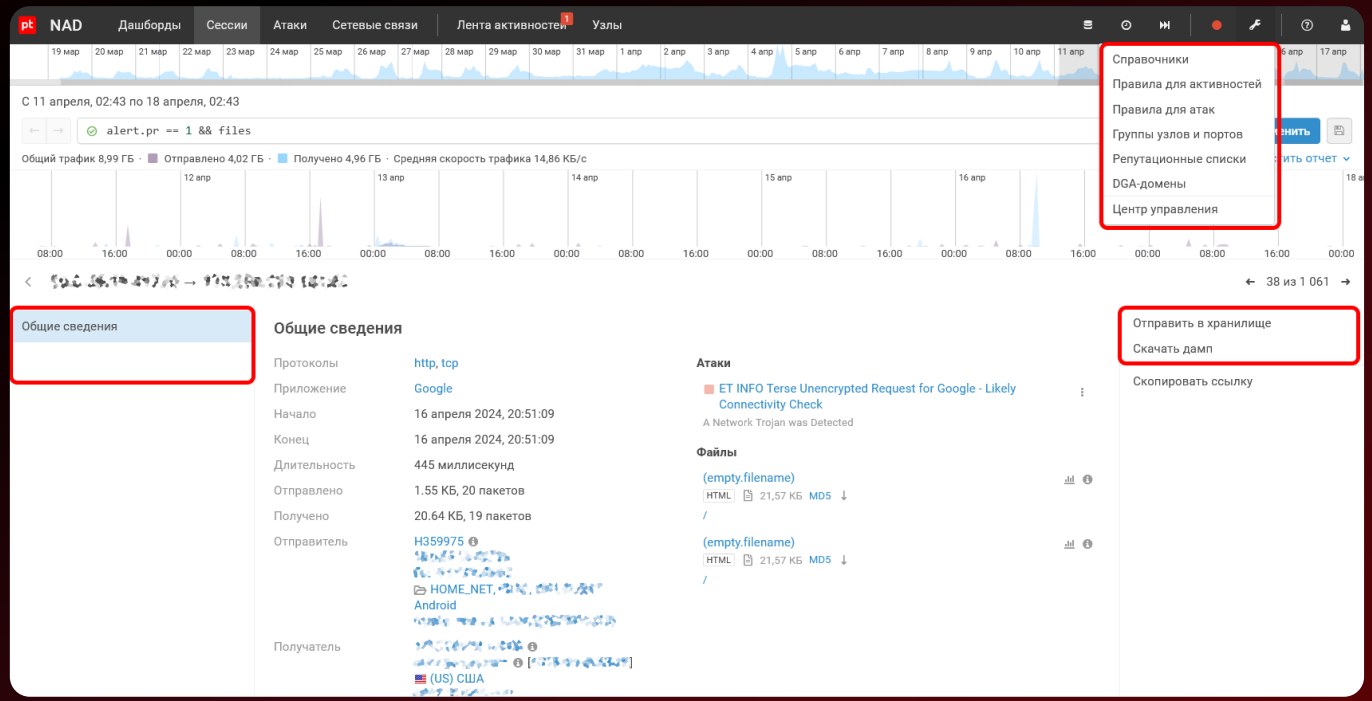
Требования | Проектирование | **Разработка** 3 | Тестирование | Эксплуатация

Реализация

- Экранирование и очистка пользовательского ввода
- Проверка прав доступа к разделам, данным и операциям

Инструменты

- Соглашение о безопасном кодировании
- Ревью кода и решения
- Статический анализ кода
- Анализ зависимостей



Secure SDLC

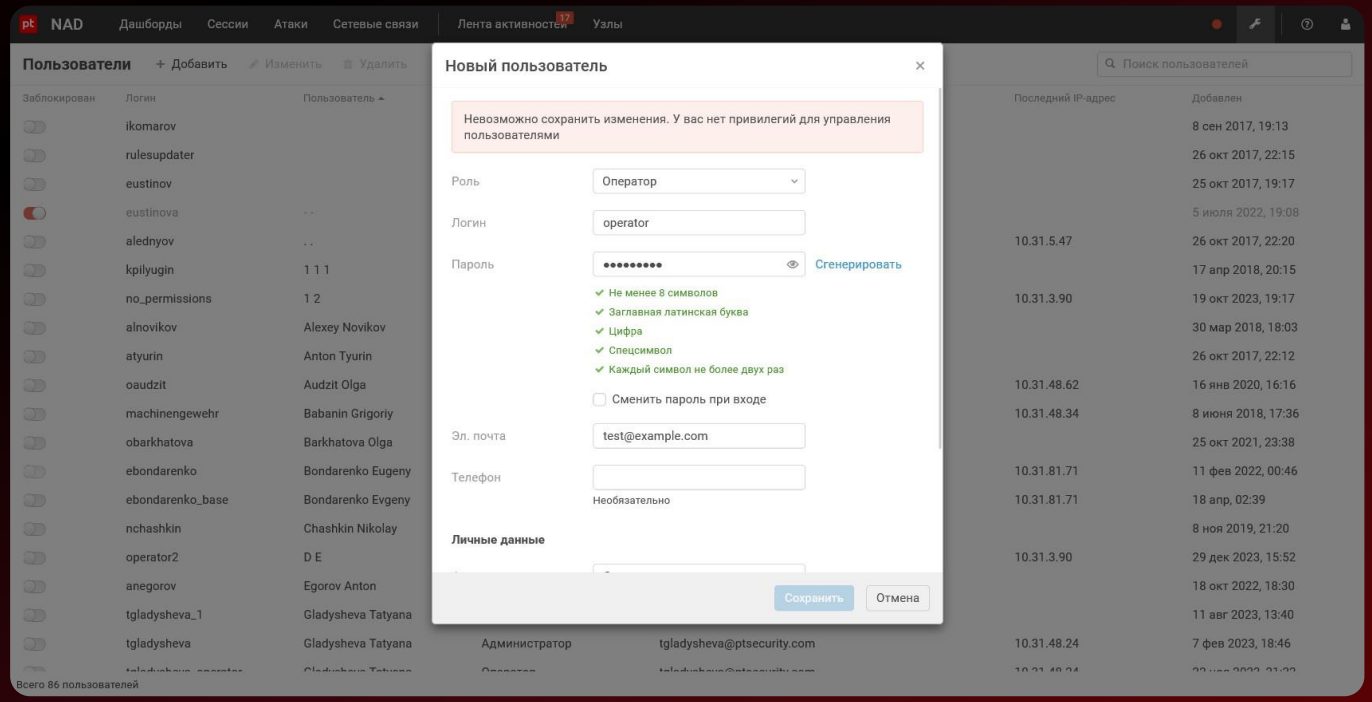
Требования | Проекти-рование | Разработка | Тестиро-вание (4) | Эксплуа-тация

Тестирование угроз

- Ввод зловредной нагрузки в фильтр
- Получение неавторизованного доступа к данным и операциям

Инструменты

- Динамический анализ кода
- Привлечение пентестеров
- Bug bounty

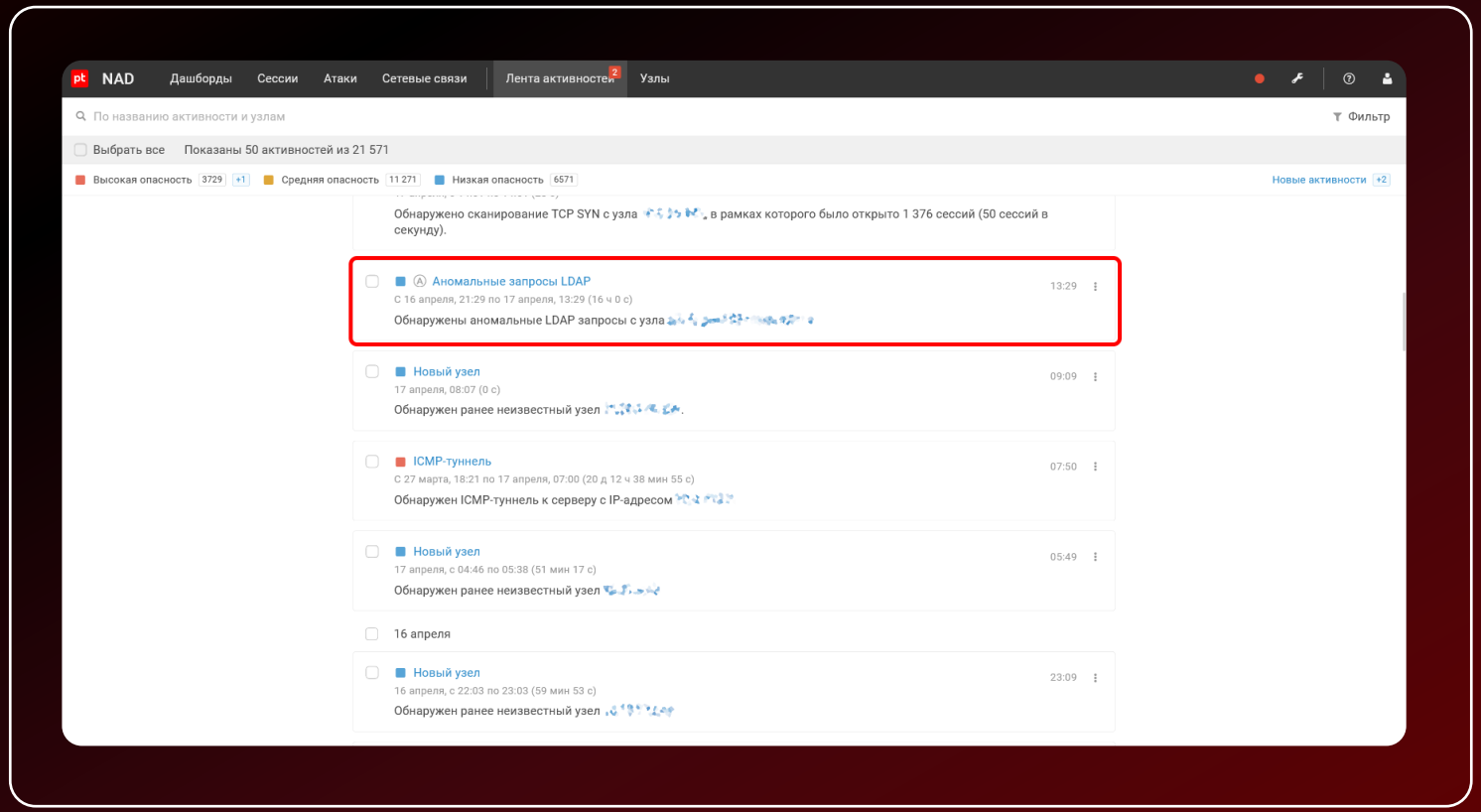


Secure SDLC

Требования | Проектирование | Разработка | Тестирование | Эксплуатация

5

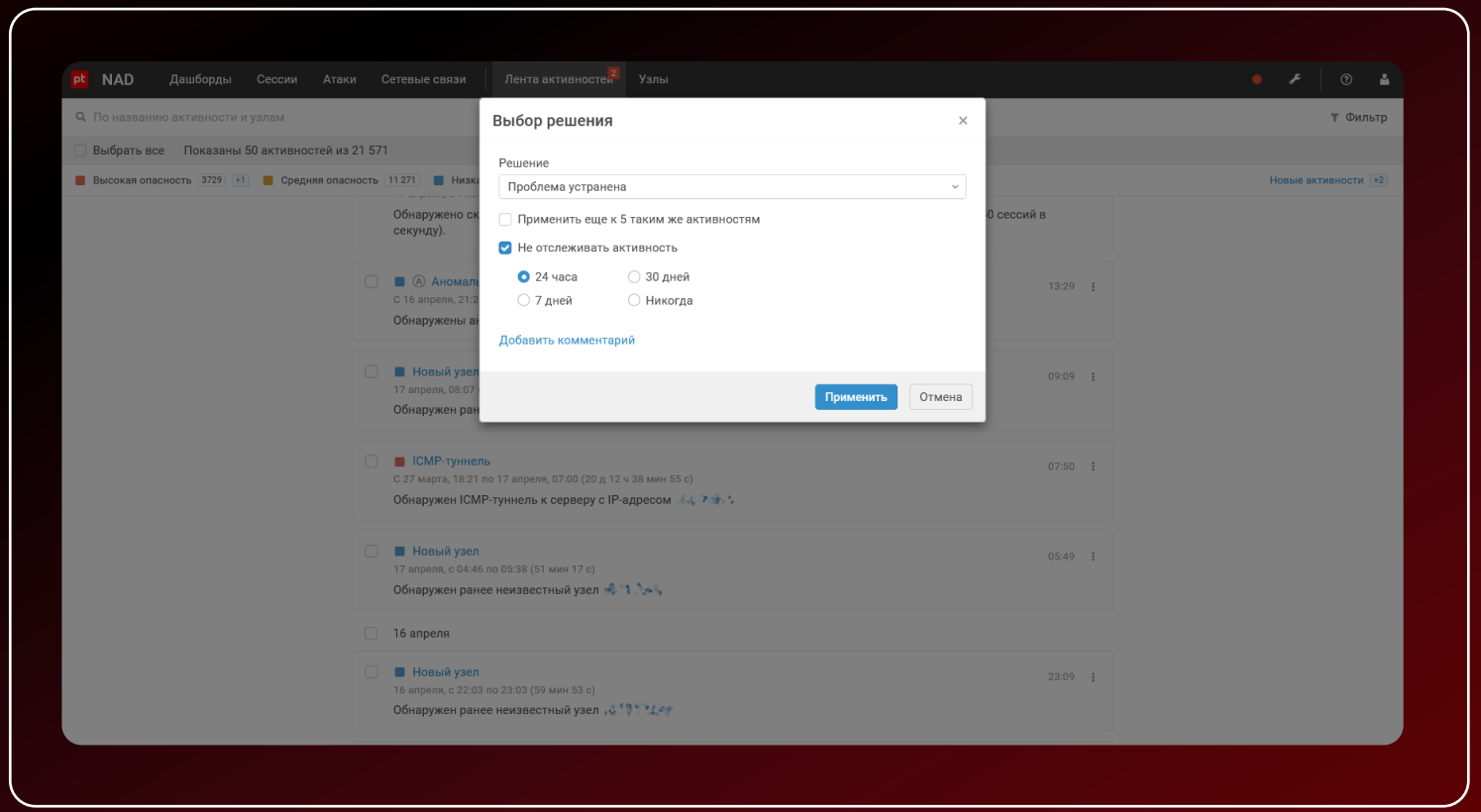
- Журналирование, аудит, мониторинг событий безопасности
- Классификация инцидентов
- Реакция на инциденты
- Обработка ошибок и исключений



Secure SDLC

Требования | Проектирование | Разработка | Тестирование | Эксплуатация 5

- Журналирование, аудит, мониторинг событий безопасности
- Классификация инцидентов
- Реакция на инциденты
- Обработка ошибок и исключений

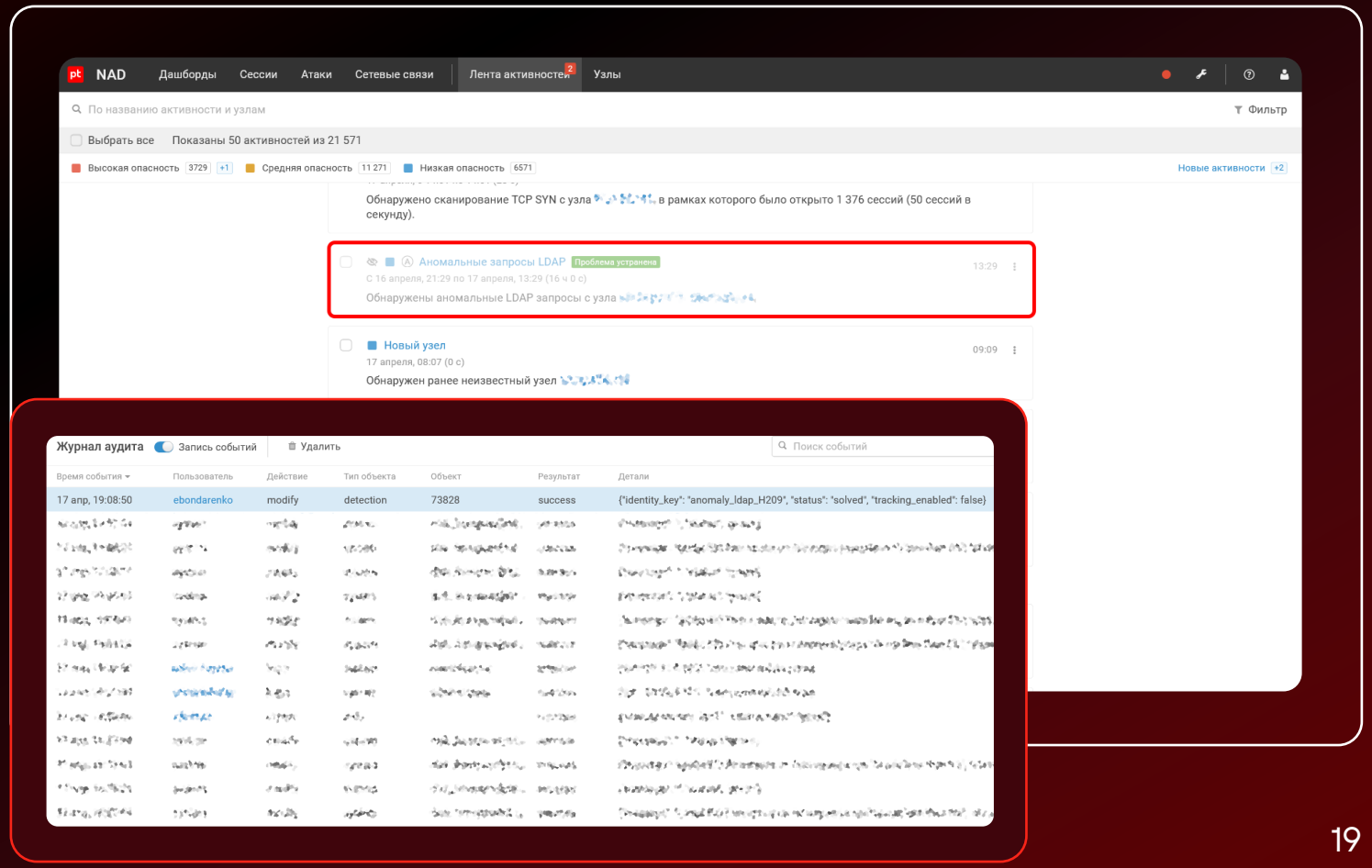


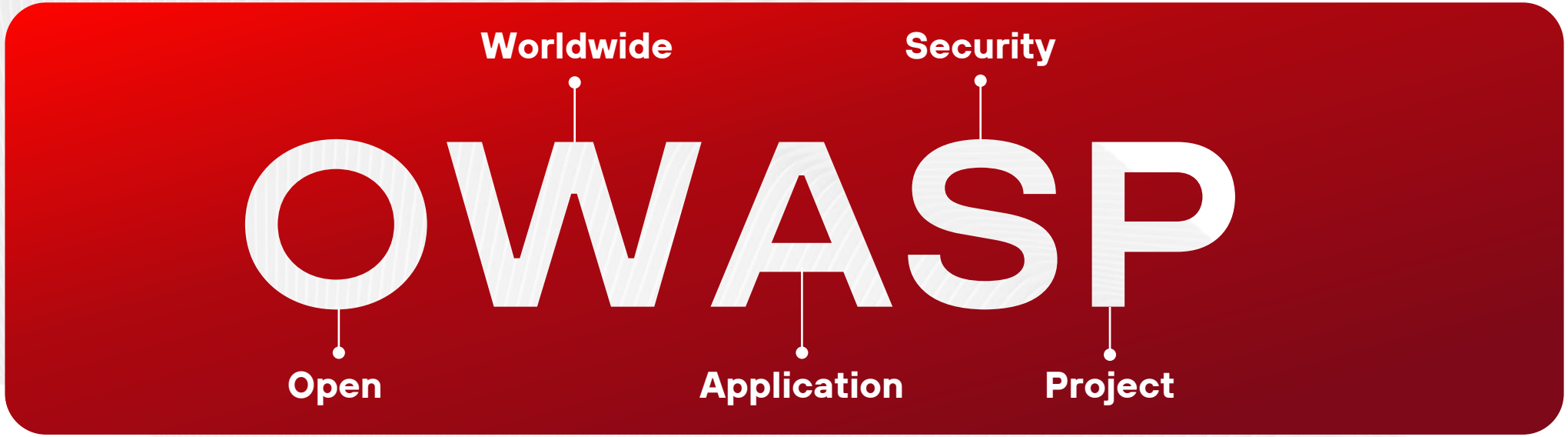
Secure SDLC

Требования | Проектирование | Разработка | Тестирование | Эксплуатация

5

- Журналирование, аудит, мониторинг событий безопасности
- Классификация инцидентов
- Реакция на инциденты
- Обработка ошибок и исключений





Открытый проект по обеспечению безопасности веб-приложений

Создают инструменты, руководства, документацию по безопасности веб-приложений

Направления
- Защита
- Обнаружение
- SSDLC

Разрабатывают стандарты

https://owasp.org/

OWASP Projects

- OWASP Top Ten
- OWASP Application Security Verification Standard
- OWASP Cheat Sheet Series
- OWASP Web Security Testing Guide
- OWASP Juice Shop

OWASP Top-10

https://owasp.org/www-project-top-ten/

1. A01:2021-Broken Access Control
2. A02:2021-Cryptographic Failures
3. A03:2021-Injection
4. A04:2021-Insecure Design
5. A05:2021-Security Misconfiguration
6. A06:2021-Vulnerable and Outdated Components
7. A07:2021-Identification and Authentication Failures
8. A08:2021-Software and Data Integrity Failures
9. A09:2021-Security Logging and Monitoring Failures
10. A10:2021-Server-Side Request Forgery

OWASP Top-10

Client-Side

https://owasp.org/www-project-top-10-client-side-security-risks/

1. Broken Client-side Access Control
2. DOM-based XSS
3. Sensitive Data Leakage
4. Vulnerable and Outdated Components
5. Lack of Third-party Origin Control
6. JavaScript Drift
7. Sensitive Data Stored Client-Side
8. Client-side Security Logging and Monitoring Failures
9. Not Using Standard Browser Security Controls
10. Including Proprietary Information on the Client-Side

OWASP Top-10

Client-Side

https://owasp.org/www-project-top-10-client-side-security-risks/

1. Broken Client-side Access Control
2. DOM-based XSS
3. Sensitive Data Leakage
4. Vulnerable and Outdated Components
5. Lack of Third-party Origin Control
6. JavaScript Drift
7. Sensitive Data Stored Client-Side
8. Client-side Security Logging and Monitoring Failures
9. Not Using Standard Browser Security Controls
10. Including Proprietary Information on the Client-Side

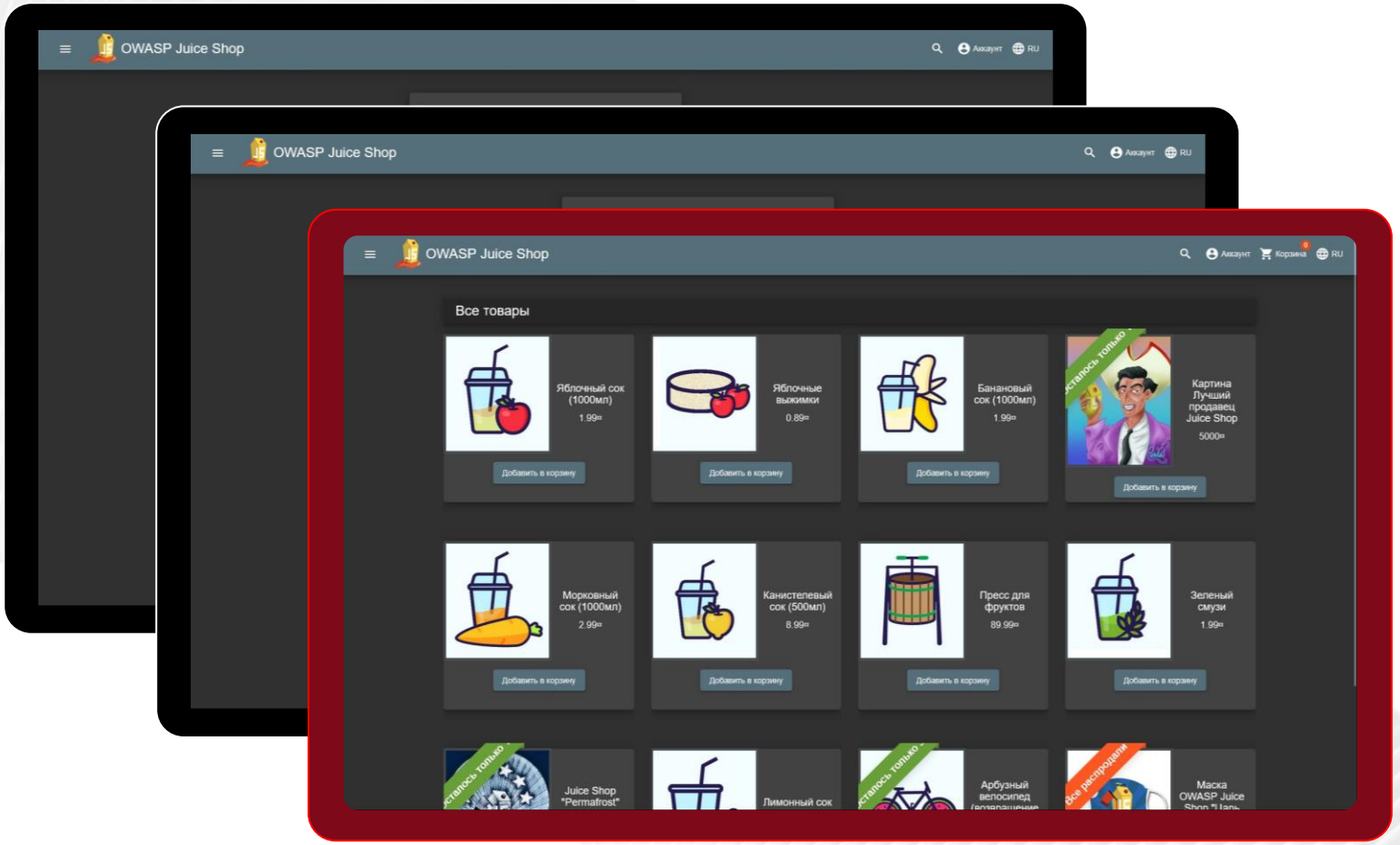
3

Безопасная разработка во фронтенде

OWASP Juice Shop

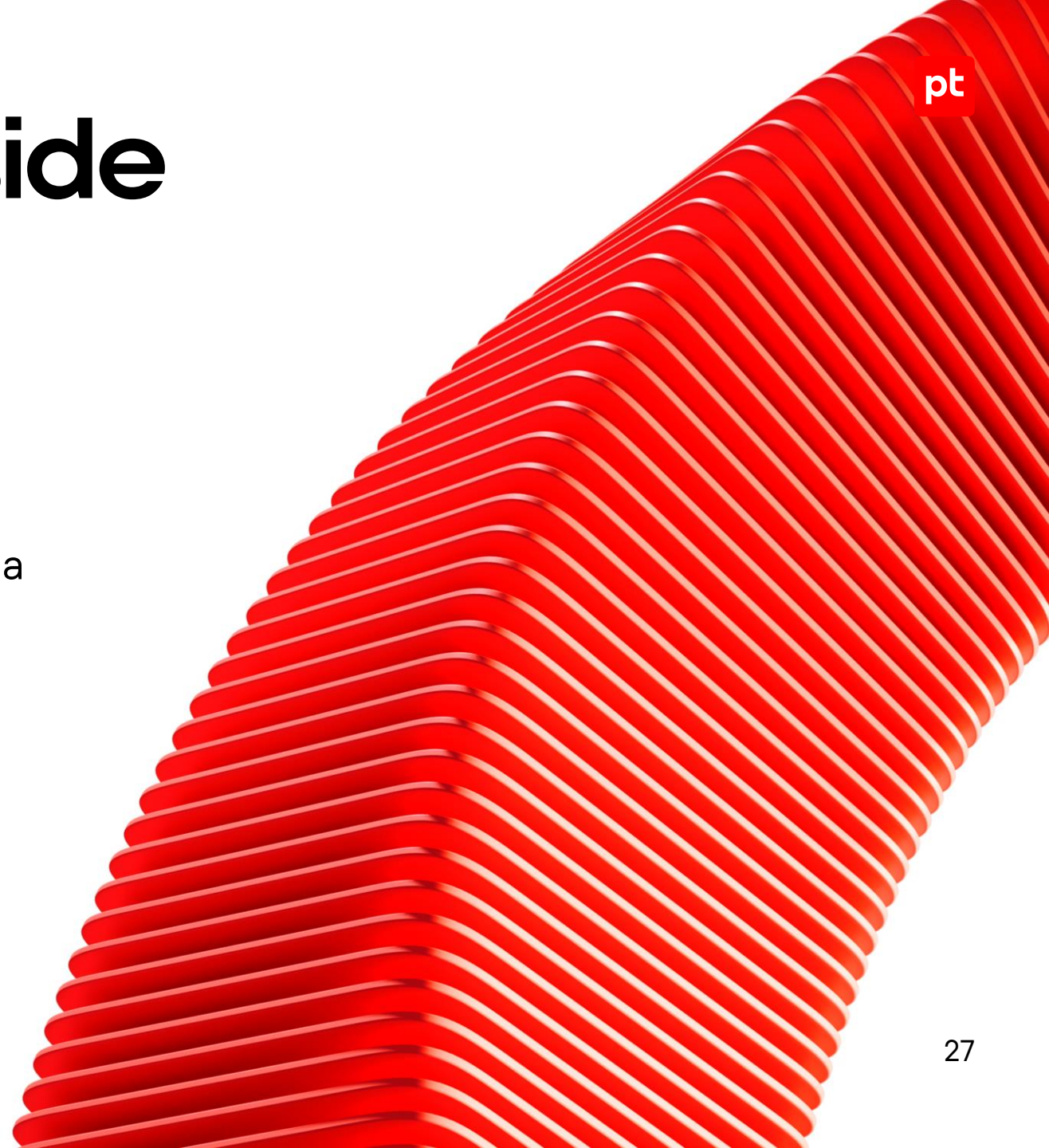
Пример небезопасного веб-приложения

- Angular v15
- Express 4



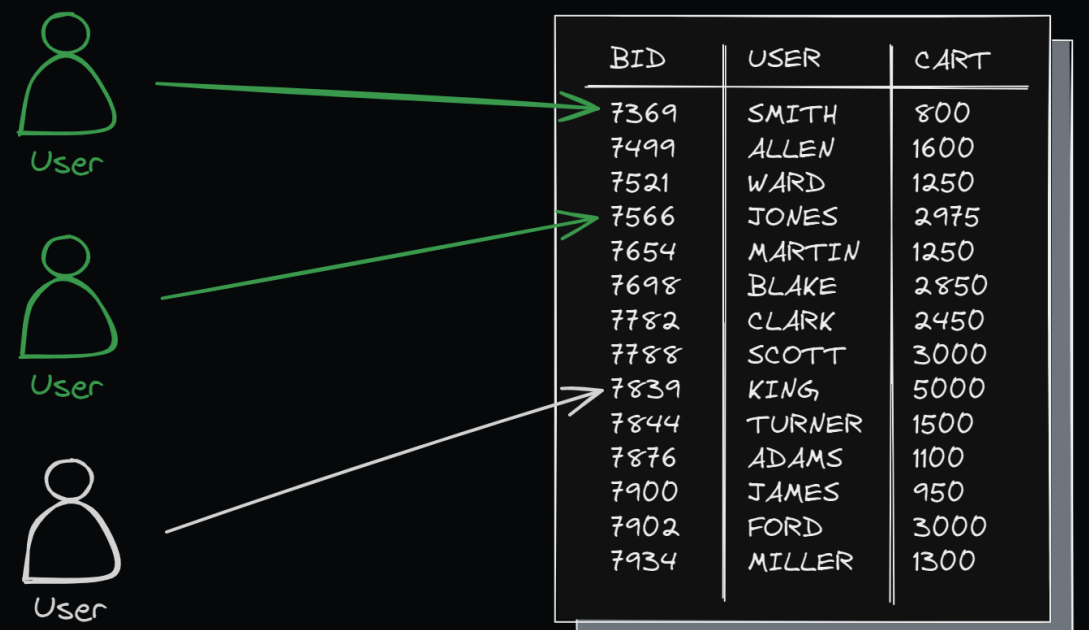
Broken Client-side Access Control

- Нарушение доступа к данным
- Нарушение доступа к операциям
- Нарушение конфиденциальности данных
- Нарушение целостности данных

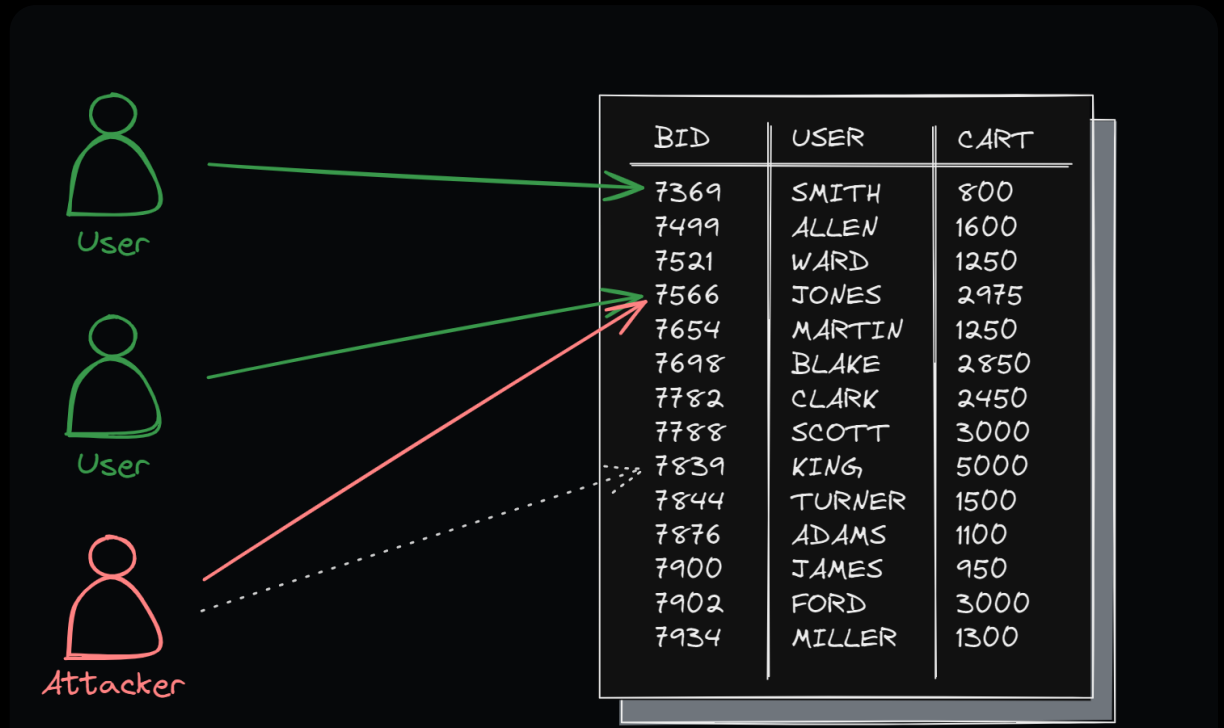


Broken Client-side Access Control

Broken Client-side Access Control

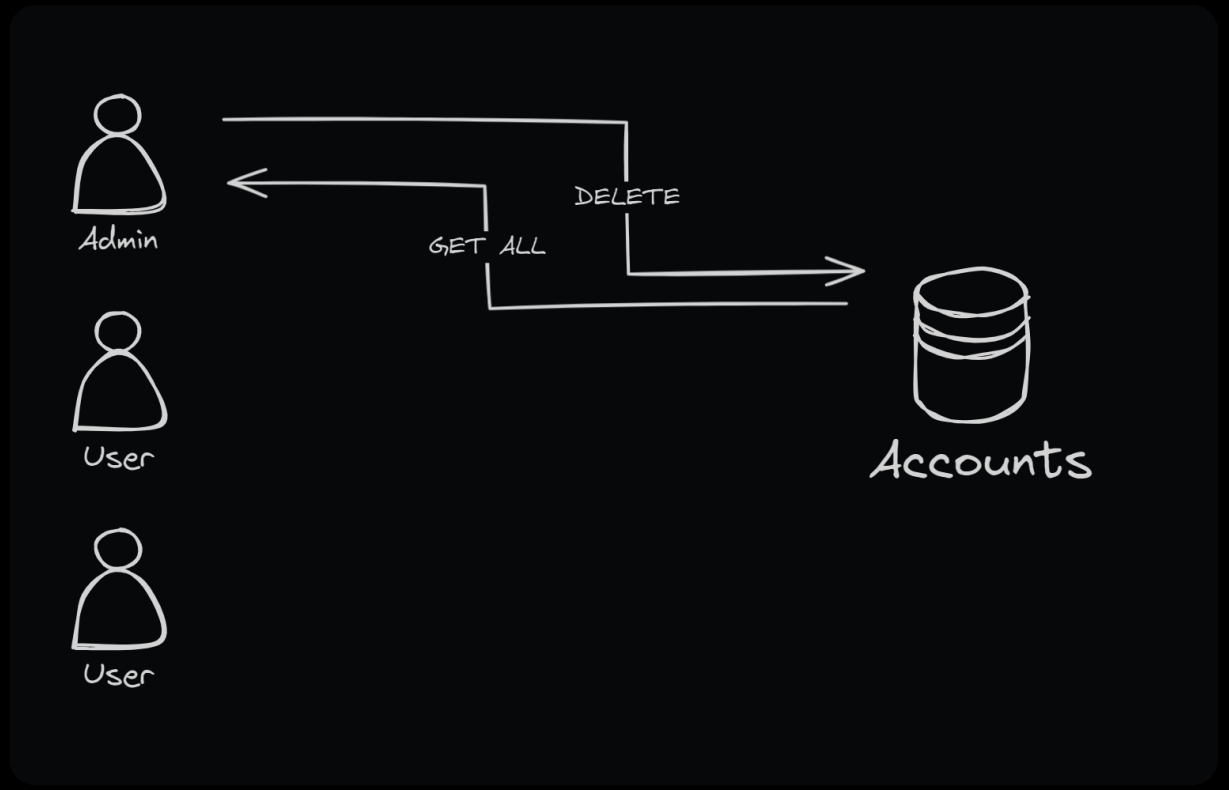


Broken Client-side Access Control

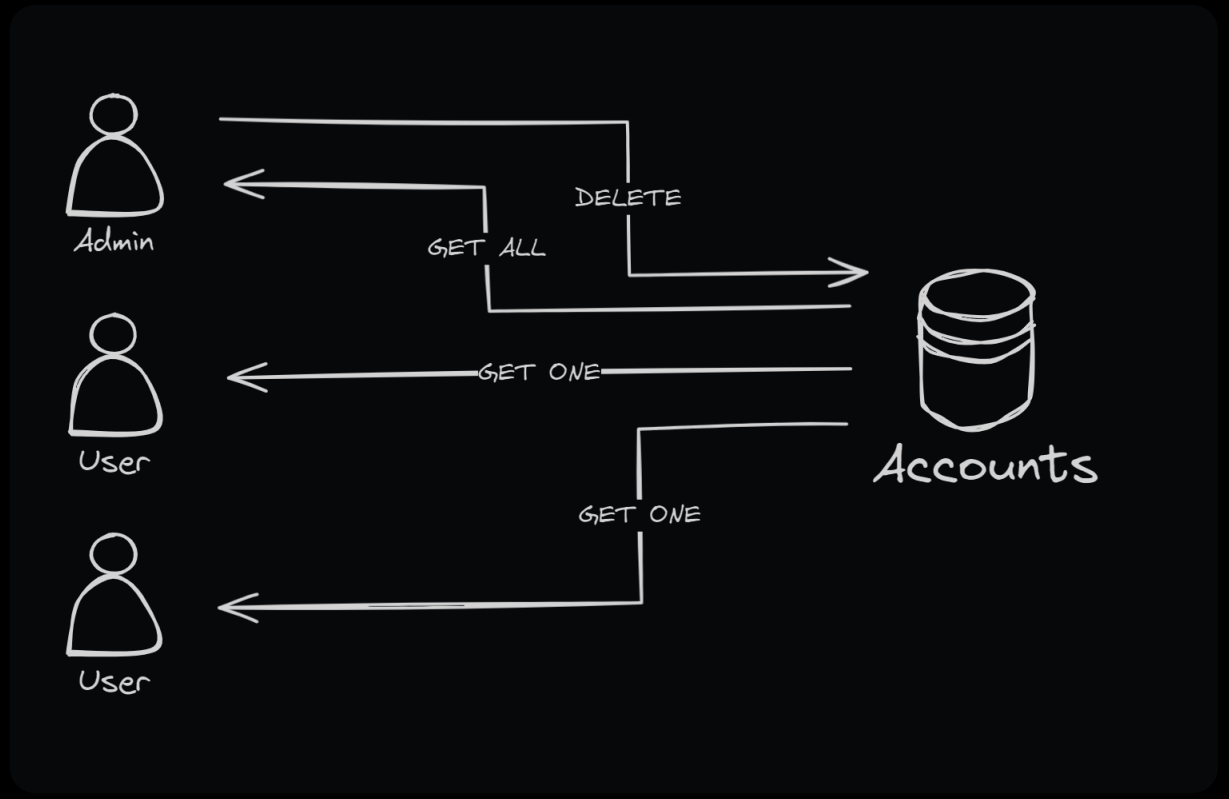


Broken Client-side Access Control

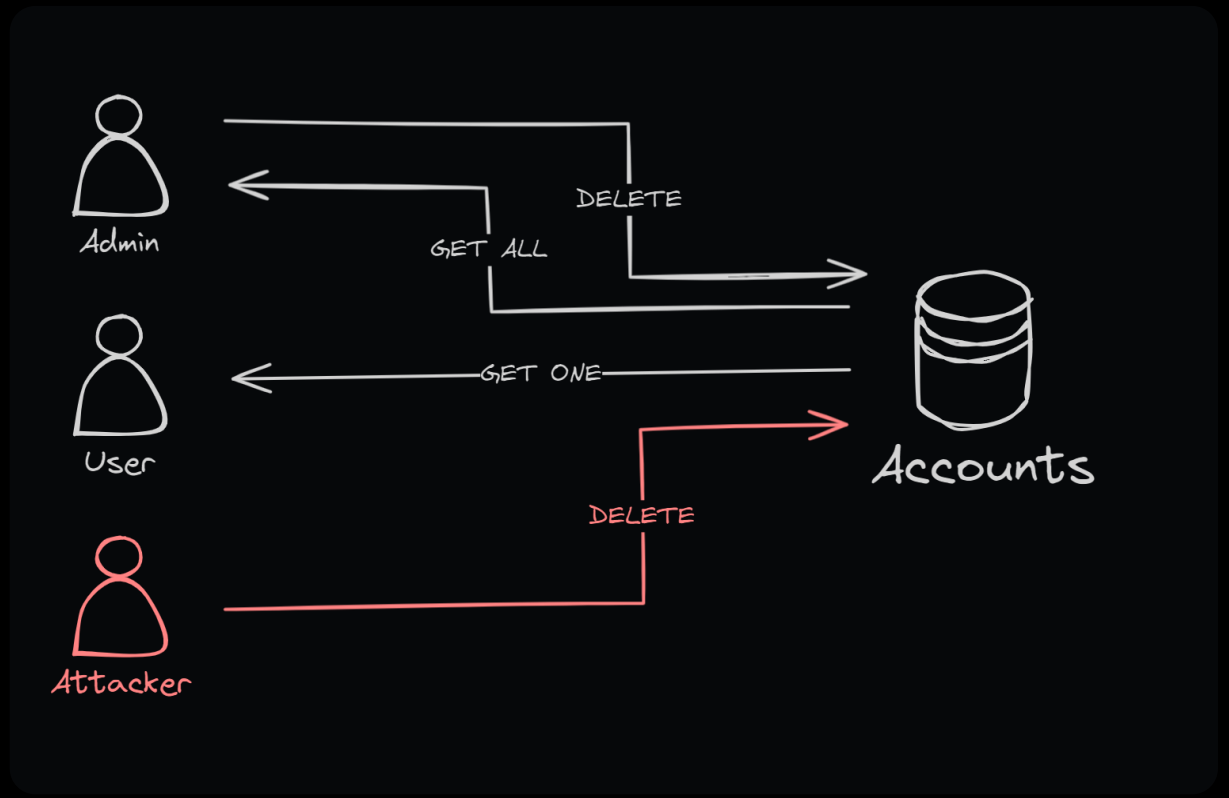
Broken Client-side Access Control



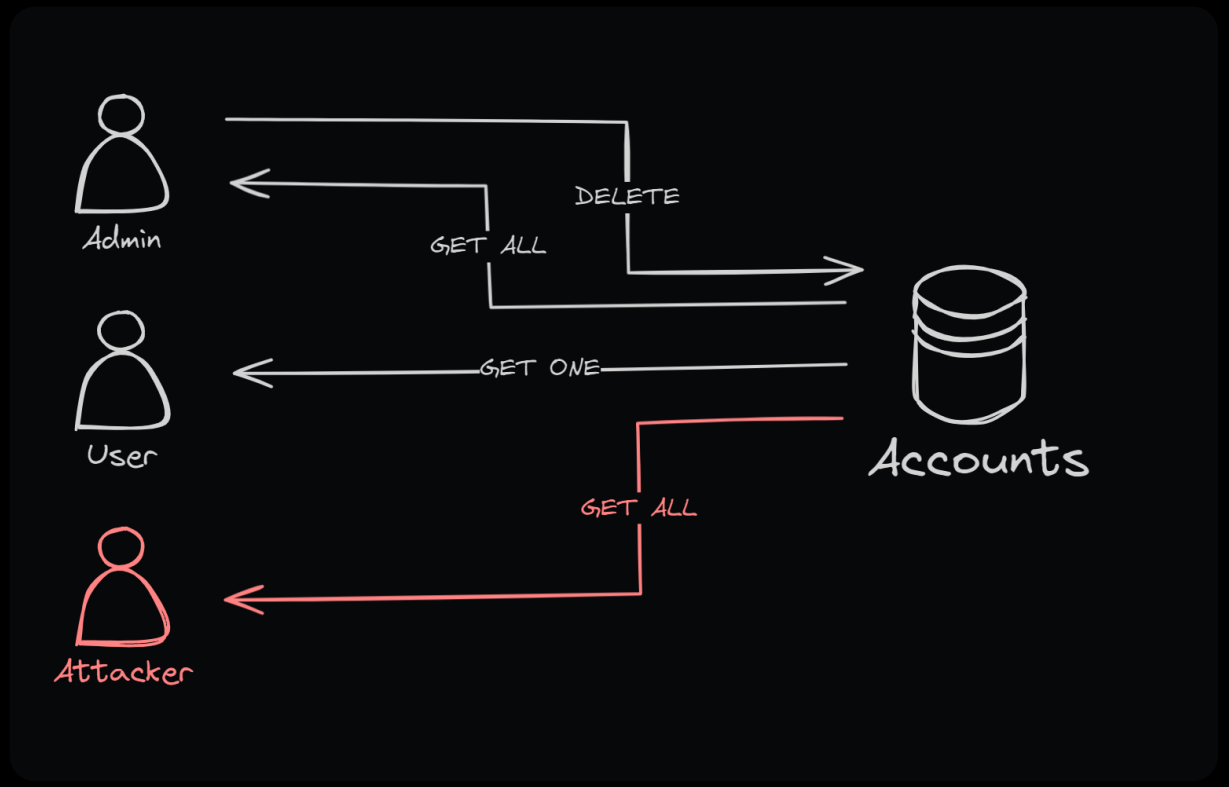
Broken Client-side Access Control



Broken Client-side Access Control



Broken Client-side Access Control

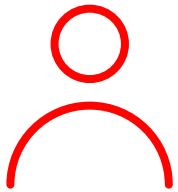


Broken Client-side Access Control

Требования
SDLC
1

Проекти-
рование
SDLC

Разработка
SDLC



Пользователь может

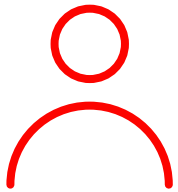
+ Добавлять товары в корзину
+ Посмотреть товары в корзине
+ Изменить товары в корзине

Broken Client-side Access Control

Требования
SDLC
1

Проекти-
рование
SDLC

Разработка
SDLC



Пользователь может

+ Добавлять товары в корзину
+ Посмотреть товары в корзине
+ Изменить товары в корзине

Корзина пользователя

+ Создается при создании новой сессии
+ «Живет» в течение пользовательской сессии
+ Доступна во всех вкладках браузера

Broken Client-side Access Control

Требования
SDLC

Проекти-
рование
SDLC
2

Разработка
SDLC

>

Корзина пользователя доступна по адресу `/rest/basket/:id`

Broken Client-side Access Control

Требования
SDLC

Проекти-
рование
SDLC
2

Разработка
SDLC

>

Корзина пользователя доступна по адресу `/rest/basket/:id`

>

Операции GET, ...

Broken Client-side Access Control

Требования
SDLC

Проекти-
рование
SDLC
2

Разработка
SDLC

> Корзина пользователя доступна по адресу `/rest/basket/:id`

> Операции GET, ...

> Идентификатор `bid` создается при открытии новой сессии

Broken Client-side Access Control

Требования
SDLC

Проекти-
рование
SDLC
2

Разработка
SDLC

- Корзина пользователя доступна по адресу `/rest/basket/:id`
- Операции GET, ...
- Идентификатор `bid` создается при открытии новой сессии
- Идентификатор `bid` доступен в сущности `authentication` при выполнении операции `login`

Broken Client-side Access Control

Требования
SDLC

Проекти-
рование
SDLC

Разработка
SDLC
3

>

Операции к API
реализуем в BasketService
через HttpClient

Broken Client-side Access Control

Требования
SDLC

Проекти-
рование
SDLC

Разработка
SDLC
3

>

Операции к API
реализуем в BasketService
через HttpClient

>

Идентификатор `bid`
будет браться
из SessionStorage

Broken Client-side Access Control

Требования
SDLC

Проекти-
рование
SDLC

Разработка
SDLC
3

>

Операции к API
реализуем в BasketService
через HttpClient

>

Идентификатор `bid`
будет браться
из SessionStorage

>

Идентификатор `bid`
будем получать при
авторизации и сохранять
в SessionStorage

Разработка (SDLC)

Broken Client-side Access Control

frontend\src\app\login\login.component.ts

```
this.userService.login(this.user).subscribe((authentication: any) => {
    ...
    sessionStorage.setItem('bid', authentication.bid)
    ...
}, ...)
```

Разработка (SDLC)

Broken Client-side Access Control

frontend\src\app\purchase-basket\purchase-basket.component.ts

```
this.basketService
    .find(parseInt(sessionStorage.getItem('bid'), 10))
    .subscribe((basket) ⇒ {
        ...
    })
```

Разработка (SDLC)

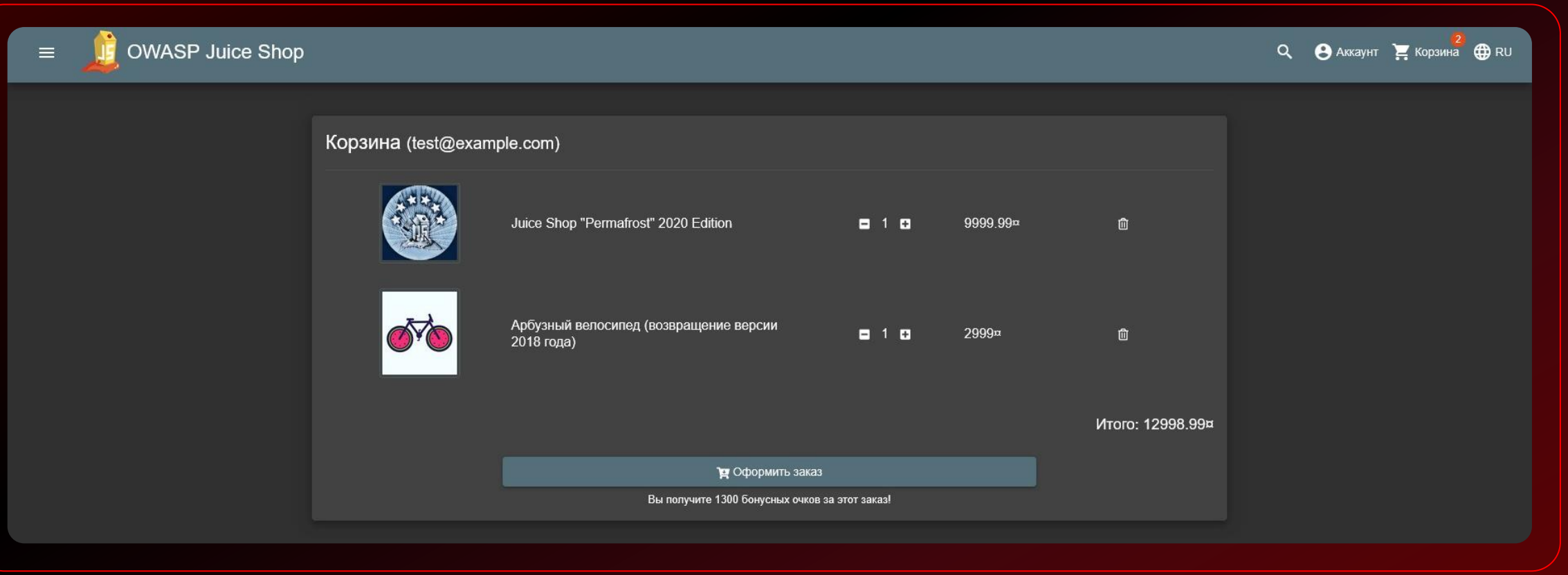
Broken Client-side Access Control

frontend\src\app\Services\basket.service.ts

```
export class BasketService {
    ...
    find (id?: number) {
        return this.http.get(`${this.hostServer}/rest/basket/${id}`).pipe(
            map((response: any) => response.data),
            catchError((error) => { throw error }))
    }
    ...
}
```

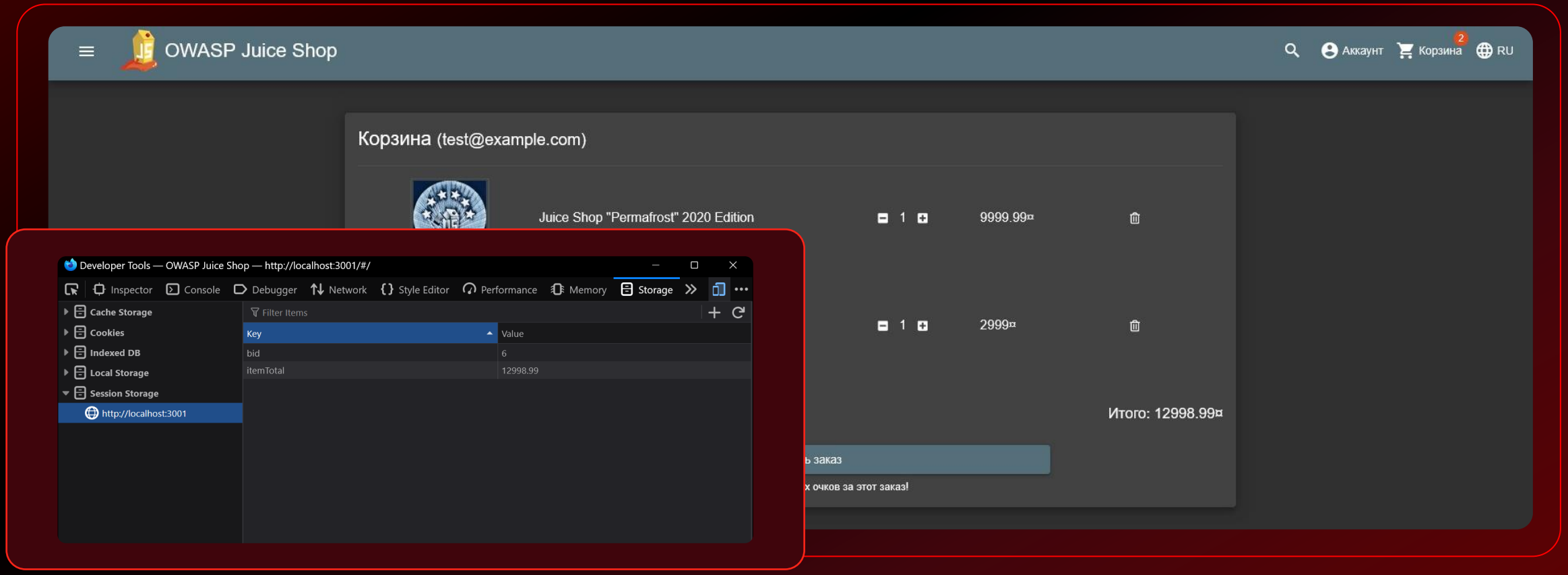
Juice Shop: View Basket

Broken Client-side Access Control



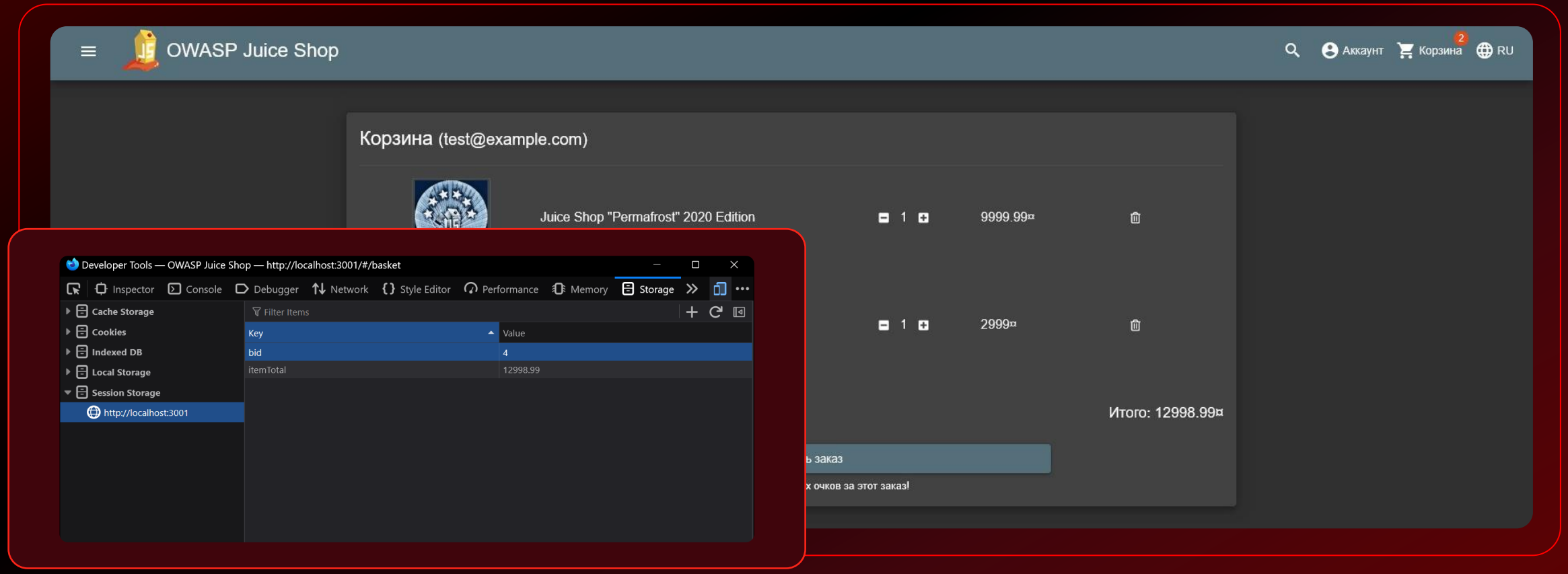
Juice Shop: View Basket

Broken Client-side Access Control



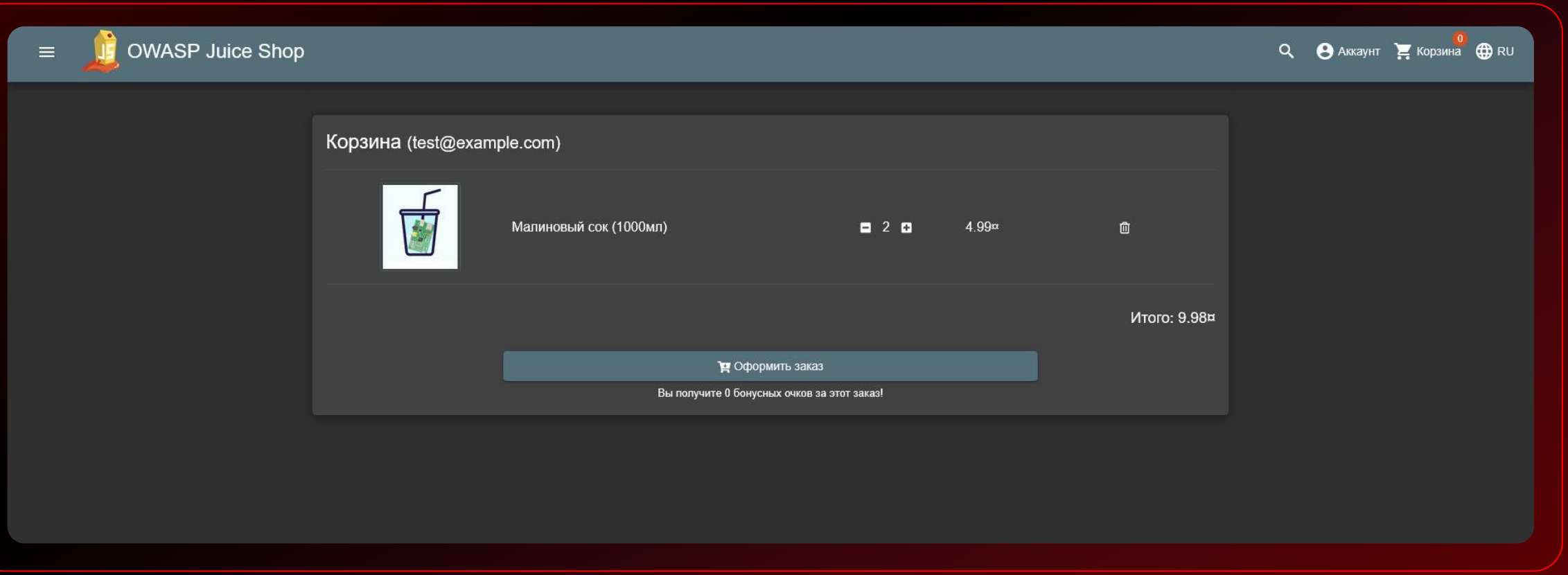
Juice Shop: View Basket

Broken Client-side Access Control



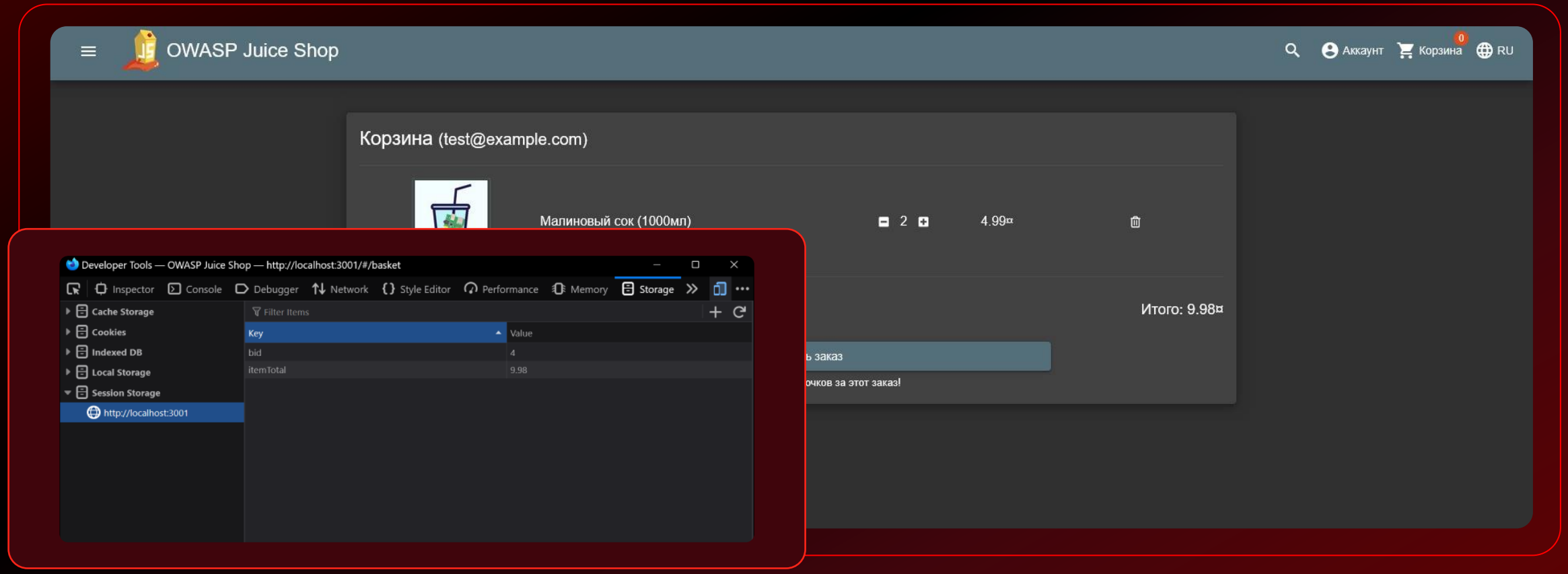
Juice Shop: View Basket

Broken Client-side Access Control



Juice Shop: View Basket

Broken Client-side Access Control



Cheat Sheet

Broken Client-side Access Control

https://cheatsheetseries.owasp.org/cheatsheets/Authorization_Cheat_Sheet.html

- Запрещайте все по умолчанию
- Определите границы и права доступов к данным и операциям
- Создайте тесты, проверяющие границы и права доступов
- Старайтесь не раскрывать уникальные идентификаторы критических данных и объектов
- При реализации учитывайте, как работают те или иные механизмы, проверяйте что их использование не нарушает границы и права доступа

Broken Client-side Access Control

Требования SSDLC 1

Проекти-рование SSDLC

Разработка SSDLC

Тестирование SSDLC

Пользователь может

+ Добавлять товары в корзину
+ Посмотреть товары в корзине
+ Изменить товары в корзине

Broken Client-side Access Control

Требования SSDLC 1 | Проекти-рование SSDLC | Разработка SSDLC | Тестирование SSDLC

Пользователь может

- Добавлять товары в корзину
- Посмотреть товары в корзине
- Изменить товары в корзине

Корзина пользователя

- Создается при создании новой сессии
- «Живет» в течение пользовательской сессии
- Доступна во всех вкладках браузера

Broken Client-side Access Control

Требования SSDLC 1 | Проекти-рование SSDLC | Разработка SSDLC | Тестирование SSDLC

Пользователь может

- Добавлять товары в корзину
- Посмотреть товары в корзине
- Изменить товары в корзине

Корзина пользователя

- Создается при создании новой сессии
- «Живет» в течение пользовательской сессии
- Доступна во всех вкладках браузера

Риски

- Злоумышленник может получить доступ к чужой корзине
- Злоумышленник может изменить чужую корзину

Broken Client-side Access Control

Требования SSDLC 1 | Проекти-рование SSDLC | Разработка SSDLC | Тестирование SSDLC

Пользователь может

- Добавлять товары в корзину
- Посмотреть товары в корзине
- Изменить товары в корзине

Корзина пользователя

- Создается при создании новой сессии
- «Живет» в течение пользовательской сессии
- Доступна во всех вкладках браузера

Риски

- Злоумышленник может получить доступ к чужой корзине
- Злоумышленник может изменить чужую корзину

Устранение рисков

- Пользователь может иметь доступ только к своей корзине
- Пользователь может изменить только свою корзину

Broken Client-side Access Control

Требования SSDLC

Проекти-рование SSDLC 2

Разработка SSDLC

Тестирование SSDLC

>

Идентификатор `bid` создается при открытии новой сессии

Broken Client-side Access Control

Требования SSDLC

Проекти-рование SSDLC 2

Разработка SSDLC

Тестирование SSDLC

>

Идентификатор `bid` создается при открытии новой сессии

>

Операции GET, ...

Broken Client-side Access Control

Требования SSDLC | Проекти-рование SSDLC 2 | Разработка SSDLC | Тестирование SSDLC

>

Идентификатор `bid` создается при открытии новой сессии

>

Операции GET, ...

Корзина пользователя доступна по адресу `/rest/basket/:id`

>

Корзина пользователя доступна по адресу `/rest/basket`

Broken Client-side Access Control

Требования SSDLC | Проекти-рование SSDLC 2 | Разработка SSDLC | Тестирование SSDLC

Идентификатор `bid` создается при открытии новой сессии

Корзина пользователя доступна по адресу `/rest/basket/:id`

Корзина пользователя доступна по адресу `/rest/basket`

Операции GET, ...

Идентификатор `bid` доступен в сущности `authentication` при выполнении операции `login`

Идентификатор `bid` извлекается из серверной пользовательской сессии при каждой операции в корзине

Broken Client-side Access Control

Требования SSDLC | Проекти-рование SSDLC 2 | Разработка SSDLC | Тестирование SSDLC

Идентификатор `bid` создается при открытии новой сессии

Корзина пользователя доступна по адресу `/rest/basket/:id`

Корзина пользователя доступна по адресу `/rest/basket`

Идентификатор `bid` хранится только в серверной пользовательской сессии

Операции GET, ...

Идентификатор `bid` доступен в сущности `authentication` при выполнении операции `login`

Идентификатор `bid` извлекается из серверной пользовательской сессии при каждой операции в корзине

Broken Client-side Access Control

Требования SSDLC

Проекти-рование SSDLC

Разработка SSDLC 3

Тестирование SSDLC

>

Операции к API
реализуем в BasketService
через HttpClient

Broken Client-side Access Control

Требования SSDLC

Проекти-рование SSDLC

Разработка SSDLC 3

Тестирование SSDLC

> Операции к API реализуем в BasketService через HttpClient

✕ Идентификатор `bid` будет браться из SessionStorage

Broken Client-side Access Control

Требования SSDLC

Проекти-рование SSDLC

Разработка SSDLC

3

Тестирование SSDLC

>

Операции к API реализуем в BasketService через HttpClient

Идентификатор `bid` будет браться из SessionStorage

Идентификатор `bid` будем получать при авторизации и сохранять в SessionStorage

Разработка (SSDLC)

Broken Client-side Access Control

frontend\src\app\login\login.component.ts

```
this.userService.login(this.user).subscribe((authentication: any) => {
    ...
-   sessionStorage.setItem('bid', authentication.bid)
    ...
}, ...)
```

Разработка (SSDLC)

Broken Client-side Access Control

frontend\src\app\purchase-basket\purchase-basket.component.ts

```
 this.basketService
-    .find(parseInt(sessionStorage.getItem('bid'), 10))
+    .find()
     .subscribe((basket) => {
     ...
 })
```

Разработка (SSDLC)

Broken Client-side Access Control

frontend\src\app\Services\basket.service.ts

```
 export class BasketService {
      ...
-     find (id?: number) {
+     find () {
-       return this.http.get(`${this.hostServer}/rest/basket/${id}`).pipe(...)
+       return this.http.get(`${this.hostServer}/rest/basket`).pipe(...)
      }
      ...
 }
```

Broken Client-side Access Control

Требования SSDLC | Проекти-рование SSDLC | Разработка SSDLC 3 | Тестирование SSDLC

Ревью MR командами разработки и AppSec

>

Линтинг

- eslint

Broken Client-side Access Control

Требования SSDLC

Проекти-рование SSDLC

Разработка SSDLC

3

Тестирование SSDLC

Ревью MR командами разработки и AppSec

> **Линтинг**

- eslint

> **Статический анализ кода**

- SonarQube
- PT Application Inspector

Broken Client-side Access Control

Требования SSDLC | Проекти-рование SSDLC | **Разработка SSDLC** | Тестирование SSDLC

3

Ревью MR командами разработки и AppSec

> **Линтинг**

- eslint

> **Статический анализ кода**

- SonarQube
- PT Application Inspector

> **Проверка зависимостей**

- npm audit

Broken Client-side Access Control

Требования SSDLC

Проекти-рование SSDLC

Разработка SSDLC

Тестирование SSDLC

4

>

Тесты для проверки получения корзины другого пользователя через `/rest/basket`

Broken Client-side Access Control

Требования SSDLC

Проекти-рование SSDLC

Разработка SSDLC

Тестирование SSDLC

4

> Тесты для проверки получения корзины другого пользователя через `/rest/basket`

> Тесты для проверки возможности получить корзину другого пользователя другими методами

Broken Client-side Access Control

Требования SSDLC

Проекти-рование SSDLC

Разработка SSDLC

Тестирование SSDLC

4

>

Тесты для проверки получения корзины другого пользователя через `/rest/basket`

>

Тесты для проверки возможности получить корзину другого пользователя другими методами

>

Динамический анализ кода:

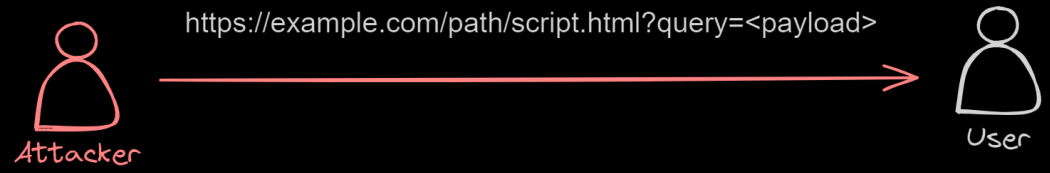
- OWASP ZAP
- PT BlackBox

DOM-based XSS

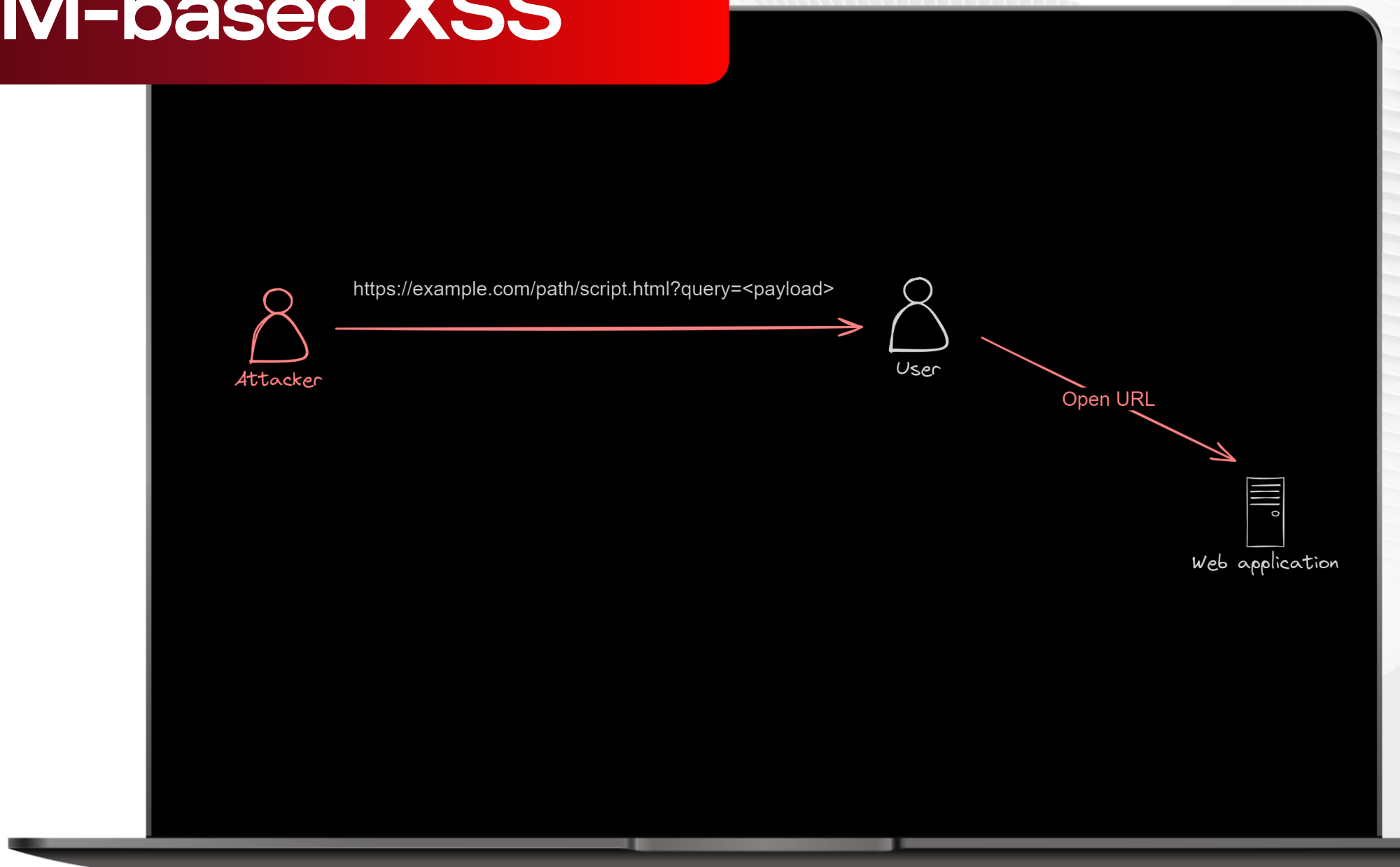
Type-0 XSS

- Зловредный код выполняется только на клиенте
- Есть доступ к сессии, данным на странице и т. п.
- Зловредная нагрузка не всегда доходит до сервера

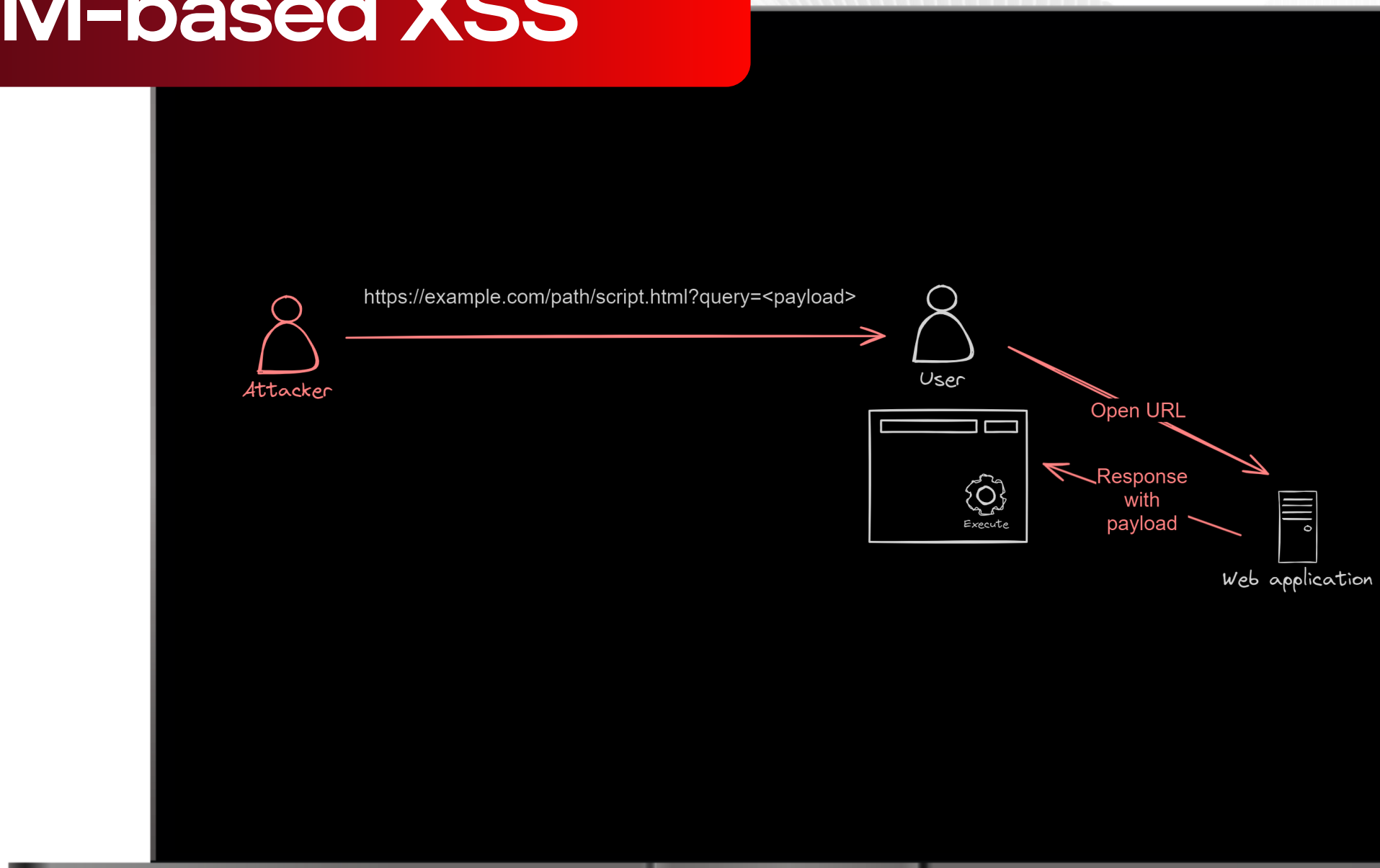
DOM-based XSS



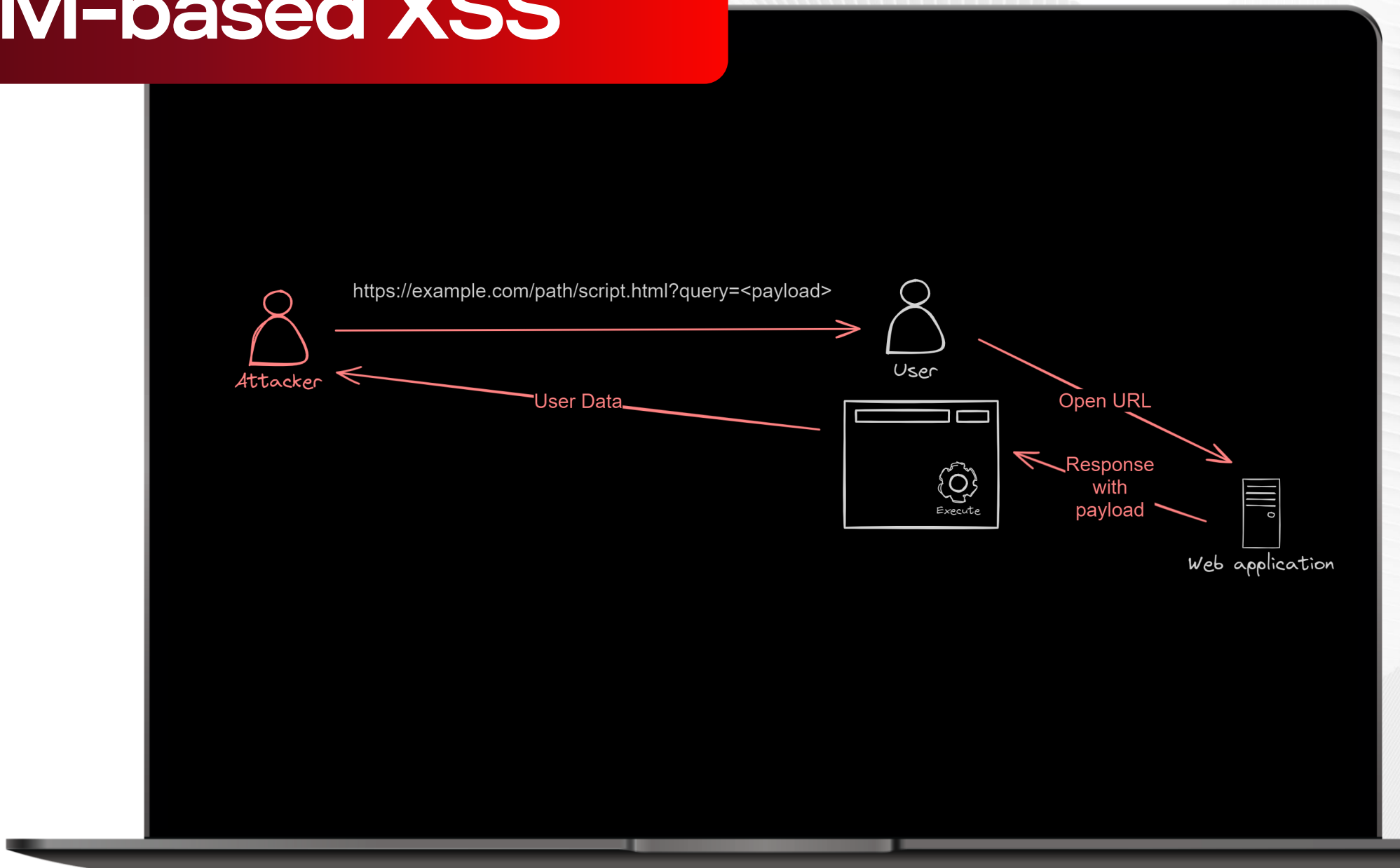
DOM-based XSS



DOM-based XSS



DOM-based XSS



DOM-based XSS

Требования SDLC 1

Проекти-рование SDLC

Разработка SDLC

>

Пользователь может ввести поисковый запрос в строку ввода и посмотреть результаты поиска на отдельной странице

DOM-based XSS

Требования
SDLC
1

Проекти-
рование
SDLC

Разработка
SDLC

>

Пользователь может ввести поисковый запрос в строку ввода и посмотреть результаты поиска на отдельной странице

>

Пользователь должен видеть свой поисковый запрос на странице результатов поиска

DOM-based XSS

Требования
SDLC
1

Проекти-рование
SDLC

Разработка
SDLC

>

Пользователь может ввести поисковый запрос в строку ввода и посмотреть результаты поиска на отдельной странице

>

Пользователь должен видеть свой поисковый запрос на странице результатов поиска

>

Поисковый запрос также должен быть добавлен в URL, чтобы пользователь мог сохранить результаты поиска или поделиться ими

DOM-based XSS

Требования SDLC

Проекти-рование SDLC

2

Разработка SDLC

>

Получаем поисковый запрос
из поля поиска, применяем
к загруженным данным из BE
и отображаем результаты

DOM-based XSS

Требования
SDLC

Проекти-
рование
SDLC
2

Разработка
SDLC

>

Получаем поисковый запрос из поля поиска, применяем к загруженным данным из BE и отображаем результаты

>

На странице результатов отображаем поисковый запрос

DOM-based XSS

Требования
SDLC

Проекти-
рование
SDLC

Разработка
SDLC

3

>

Фильтруем и отображаем товары

DOM-based XSS

Требования
SDLC

Проекти-
рование
SDLC

Разработка
SDLC
3

> Фильтруем и отображаем товары

> Отображаем поисковый запрос

DOM-based XSS

Требования
SDLC

Проекти-
рование
SDLC

Разработка
SDLC
3

> Фильтруем и отображаем товары

> Отображаем поисковый запрос

> Подставляем его в URL в параметр `q`

Разработка (SDLC)

DOM-based XSS

frontend\src\app\search-result\search-result.component.ts

```
let queryParam: string = this.route.snapshot.queryParams.q
if (queryParam) {
    queryParam = queryParam.trim()
    this.searchValue = this.sanitizer.bypassSecurityTrustHtml(queryParam)
    ...
}
```

Разработка (SDLC)

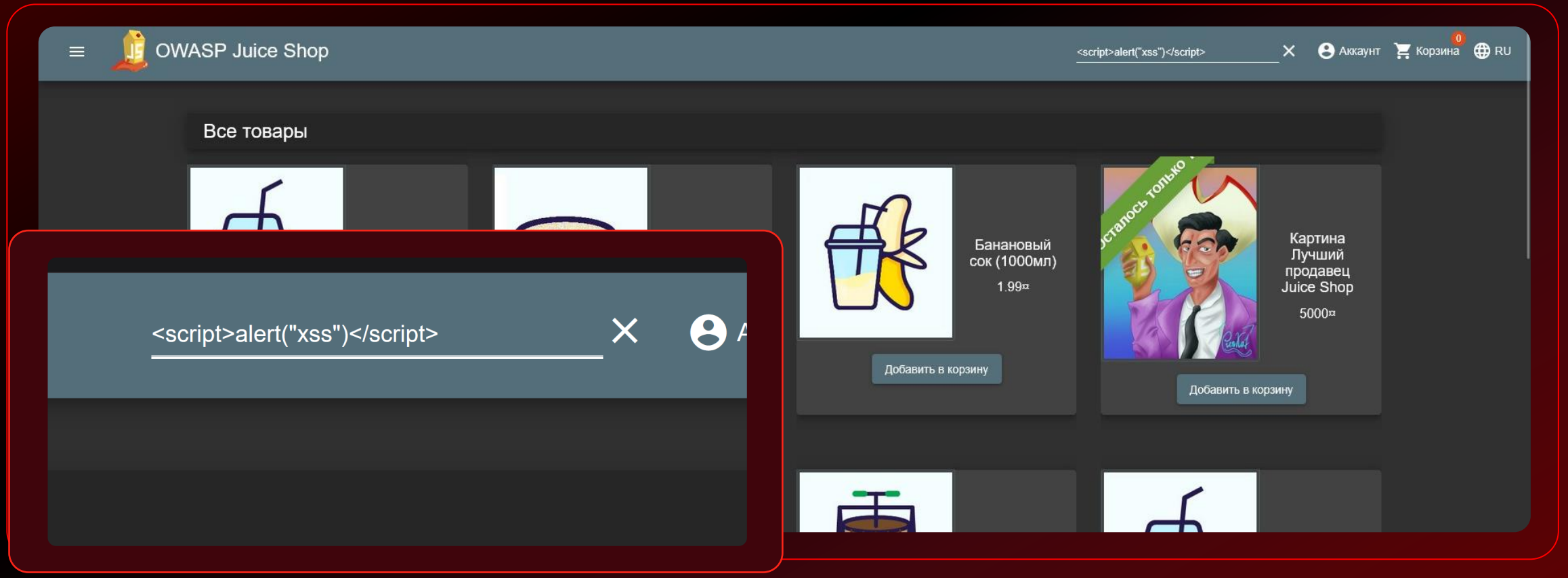
DOM-based XSS

frontend\src\app\search-result\search-result.component.html

```
<div *ngIf="searchValue">
    <span>{{"TITLE_SEARCH_RESULTS" | translate}} - </span>
    <span id="searchValue" [innerHTML]="searchValue"></span>
</div>
```

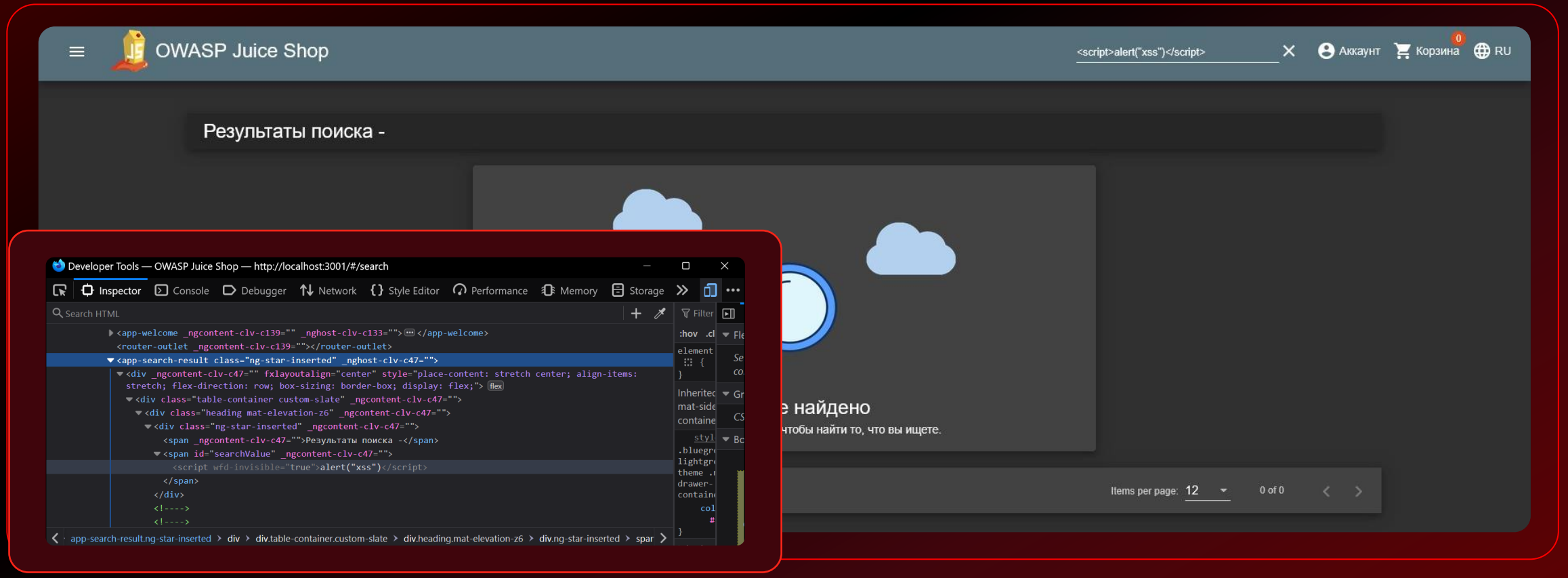
Juice Shop: Search Results

DOM-based XSS



Juice Shop: Search Results

DOM-based XSS



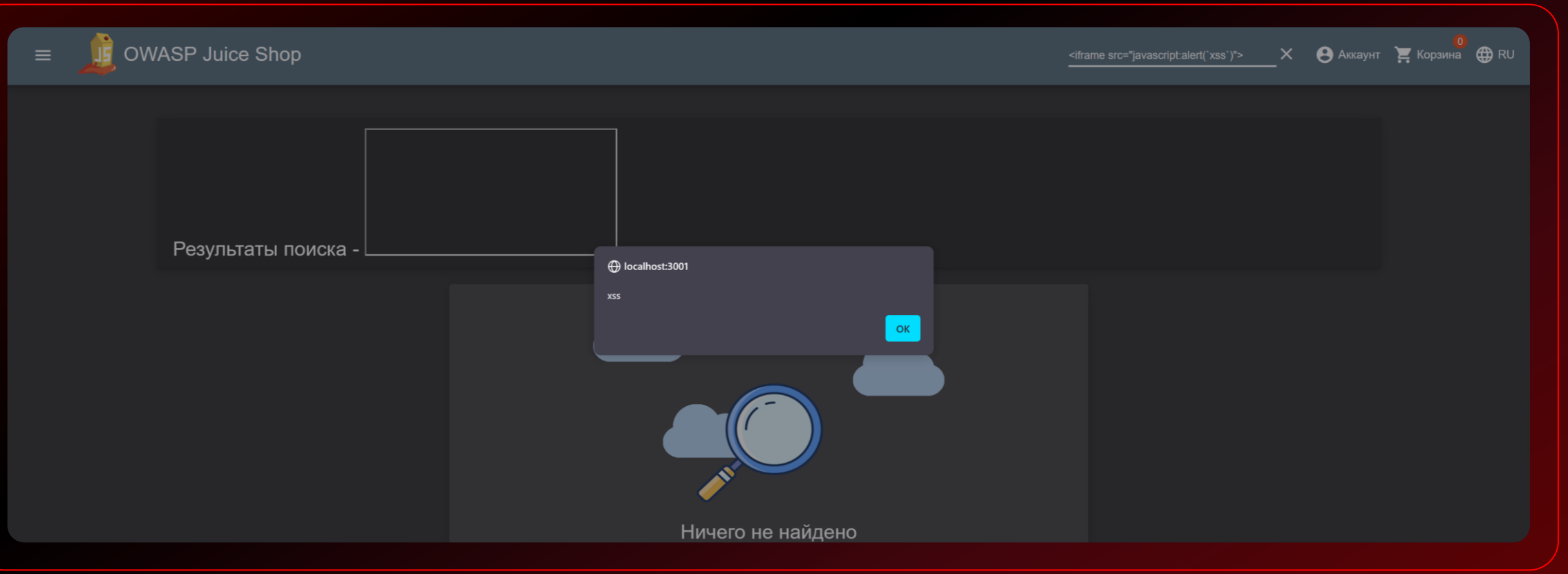
Juice Shop: Search Results

DOM-based XSS

```
<iframe src="javascript:alert(`xss`)">

http://localhost:3001/#/search?
q=%3Ciframe%20src%3D%22javascript:alert(%60xss%60)%22%3E

<img src="" onerror="javascript:alert(`xss`)">

http://localhost:3001/#/search?
q=%3Cimg%20src%3D%22%22%20onerror%3D%22javascript:alert(%60xss%60)%22%3E
```

Juice Shop: Search Results

DOM-based XSS



Cheat Sheet

DOM-based XSS

https://cheatsheetseries.owasp.org/cheatsheets/DOM_based_XSS_Prevention_Cheat_Sheet.html

- Ненадежные данные — текст
- Очищай и экранируй
- Не evalируй
- Строй DOM сам
- Не верь небезопасным источникам

DOM-based XSS

Требования SSDLC 1

Проекти-рование SSDLC

Разработка SSDLC

Тестирование SSDLC

>

Пользователь может ввести поисковый запрос в строку ввода и посмотреть результаты поиска на отдельной странице

DOM-based XSS

Требования SSDLC 1

Проекти-рование SSDLC

Разработка SSDLC

Тестирование SSDLC

>

Пользователь может ввести поисковый запрос в строку ввода и посмотреть результаты поиска на отдельной странице

>

Риск внедрения зловредной нагрузки через поле поиска и параметр в URL

DOM-based XSS

Требования SSDLC 1 | Проекти-рование SSDLC | Разработка SSDLC | Тестирование SSDLC

> Пользователь может ввести поисковый запрос в строку ввода и посмотреть результаты поиска на отдельной странице

> Пользователь будет видеть свой поисковый запрос на странице результатов поиска

> Риск внедрения зловредной нагрузки через поле поиска и параметр в URL

DOM-based XSS

Требования SSDLC 1 | Проекти-рование SSDLC | Разработка SSDLC | Тестирование SSDLC

> Пользователь может ввести поисковый запрос в строку ввода и посмотреть результаты поиска на отдельной странице

> Пользователь будет видеть свой поисковый запрос на странице результатов поиска

> Риск внедрения зловредной нагрузки через поле поиска и параметр в URL

> Необходимо экранировать и очищать небезопасные данные из поля ввода и параметра URL

DOM-based XSS

Требования SSDLC 1 | Проекти-рование SSDLC | Разработка SSDLC | Тестирование SSDLC

>

Пользователь может ввести поисковый запрос в строку ввода и посмотреть результаты поиска на отдельной странице

>

Пользователь будет видеть свой поисковый запрос на странице результатов поиска

>

Поисковый запрос также должен быть добавлен в URL, чтобы пользователь мог сохранить результаты поиска или поделиться ими

>

Риск внедрения зловредной нагрузки через поле поиска и параметр в URL

>

Необходимо экранировать и очищать небезопасные данные из поля ввода и параметра URL

DOM-based XSS

Требования SSDLC

Проекти-рование SSDLC

2

Разработка SSDLC

Тестирование SSDLC

>

Получаем поисковый запрос из поля поиска, применяем к загруженным данным из BE и отображаем результаты

DOM-based XSS

Требования SSDLC

Проекти-рование SSDLC 2

Разработка SSDLC

Тестирование SSDLC

>

Получаем поисковый запрос из поля поиска, применяем к загруженным данным из BE и отображаем результаты

>

Очищать небезопасные данные

DOM-based XSS

Требования SSDLC

Проекти-рование SSDLC 2

Разработка SSDLC

Тестирование SSDLC

>

Получаем поисковый запрос из поля поиска, применяем к загруженным данным из BE и отображаем результаты

>

На странице результатов отображаем поисковый запрос

>

Очищать небезопасные данные

DOM-based XSS

Требования SSDLC | Проекти-рование SSDLC | Разработка SSDLC | Тестирование SSDLC

2

- Получаем поисковый запрос из поля поиска, применяем к загруженным данным из BE и отображаем результаты
- На странице результатов отображаем поисковый запрос
- Очищать небезопасные данные
- Отображать поисковый запрос в виде строки

DOM-based XSS

Требования SSDLC

Проекти-рование SSDLC

Разработка SSDLC

3

Тестирование SSDLC

>

Фильтруем и отображаем товары

DOM-based XSS

Требования SSDLC

Проекти-рование SSDLC

Разработка SSDLC

3

Тестирование SSDLC

- Фильтруем и отображаем товары

- Очистка данных из поля ввода и URL через DomSanitizer.sanitize()

DOM-based XSS

Требования SSDLC

Проекти-рование SSDLC

Разработка SSDLC

3

Тестирование SSDLC

- Фильтруем и отображаем товары
- Отображаем поисковый запрос
- Очистка данных из поля ввода и URL через DomSanitizer.sanitize()

DOM-based XSS

Требования SSDLC

Проекти-рование SSDLC

Разработка SSDLC

3

Тестирование SSDLC

> Фильтруем и отображаем товары

> Отображаем поисковый запрос

> Очистка данных из поля ввода и URL через DomSanitizer.sanitize()

> Не использовать `[innerHTML]` и `Element.innerHTML` для вставки контента

DOM-based XSS

Требования SSDLC

Проекти-рование SSDLC

Разработка SSDLC

3

Тестирование SSDLC

>

Фильтруем и отображаем товары

>

Отображаем поисковый запрос

>

Подставляем его в URL в параметр `q`

>

Очистка данных из поля ввода и URL через DomSanitizer.sanitize()

>

Не использовать `[innerHTML]` и `Element.innerHTML` для вставки контента

Разработка (SSDLC)

DOM-based XSS

frontend\src\app\navbar\navbar.component.ts

```
search (value: string) {
    if (value) {
-       const queryParams = { queryParams: { q: value } }
+       const queryParams = { queryParams: { q:
this.sanitizer.sanitize(SecurityContext.HTML, value) } }
        this.ngZone.run(async () => await this.router.navigate(['/search'],
queryParams))
    }
}
```

Разработка (SSDLC)

DOM-based XSS

frontend\src\app\search-result\search-result.component.ts

```
let queryParam: string = this.route.snapshot.queryParams.q
if (queryParam) {
    queryParam = queryParam.trim()
-   this.searchValue = this.sanitizer.bypassSecurityTrustHtml(queryParam)
+   this.searchValue = this.sanitizer.sanitize(SecurityContext.HTML, queryParam)
  ...
}
```

Разработка (SSDLC)

DOM-based XSS

frontend\src\app\search-result\search-result.component.html

```
<div *ngIf="searchValue">
    <span>{{"TITLE_SEARCH_RESULTS" | translate}} - </span>
-   <span id="searchValue" [innerHTML]="searchValue"></span>
+   <span id="searchValue">{{searchValue}}</span>
</div>
```

Разработка (SSDLC)

DOM-based XSS

```
-<iframe src="javascript:alert(`xss`)">
+&lt;/body&gt;

-<img src="" onerror="javascript:alert(`xss`)">
+<img src="">
```

DOM-based XSS

Требования
SSDLC

Проекти-
рование
SSDLC

Разработка
SSDLC

Тестирование
SSDLC
4

>

Проверять поля ввода
на возможность передать
зловредную нагрузку

DOM-based XSS

Требования SSDLC

Проекти-рование SSDLC

Разработка SSDLC

Тестирование SSDLC

4

> Проверять поля ввода на возможность передать зловредную нагрузку

> Проверять параметр URL на возможность передачи зловредной нагрузки

DOM-based XSS

Требования SSDLC

Проекти-рование SSDLC

Разработка SSDLC

Тестирование SSDLC

4

>

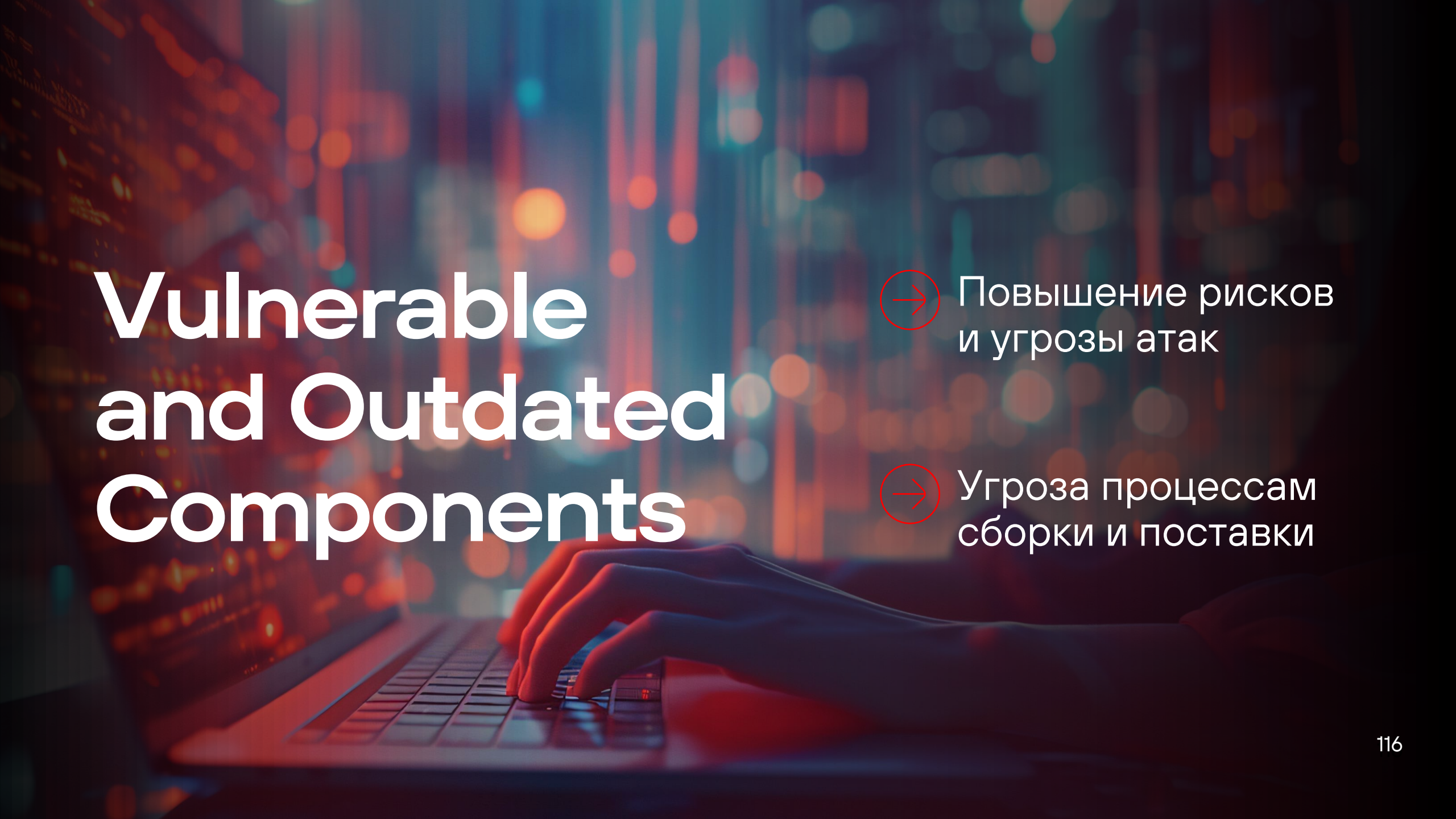
Проверять поля ввода на возможность передать зловредную нагрузку

>

Проверять параметр URL на возможность передачи зловредной нагрузки

>

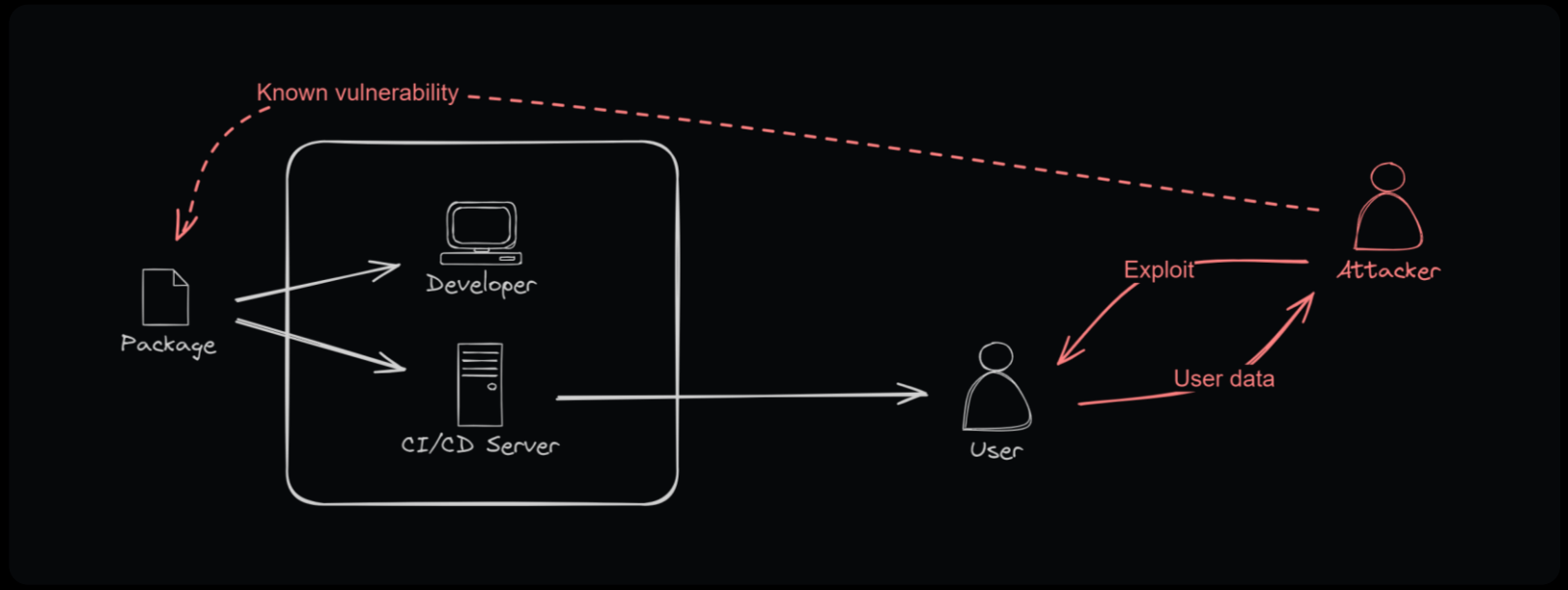
s0md3v/XSStrike



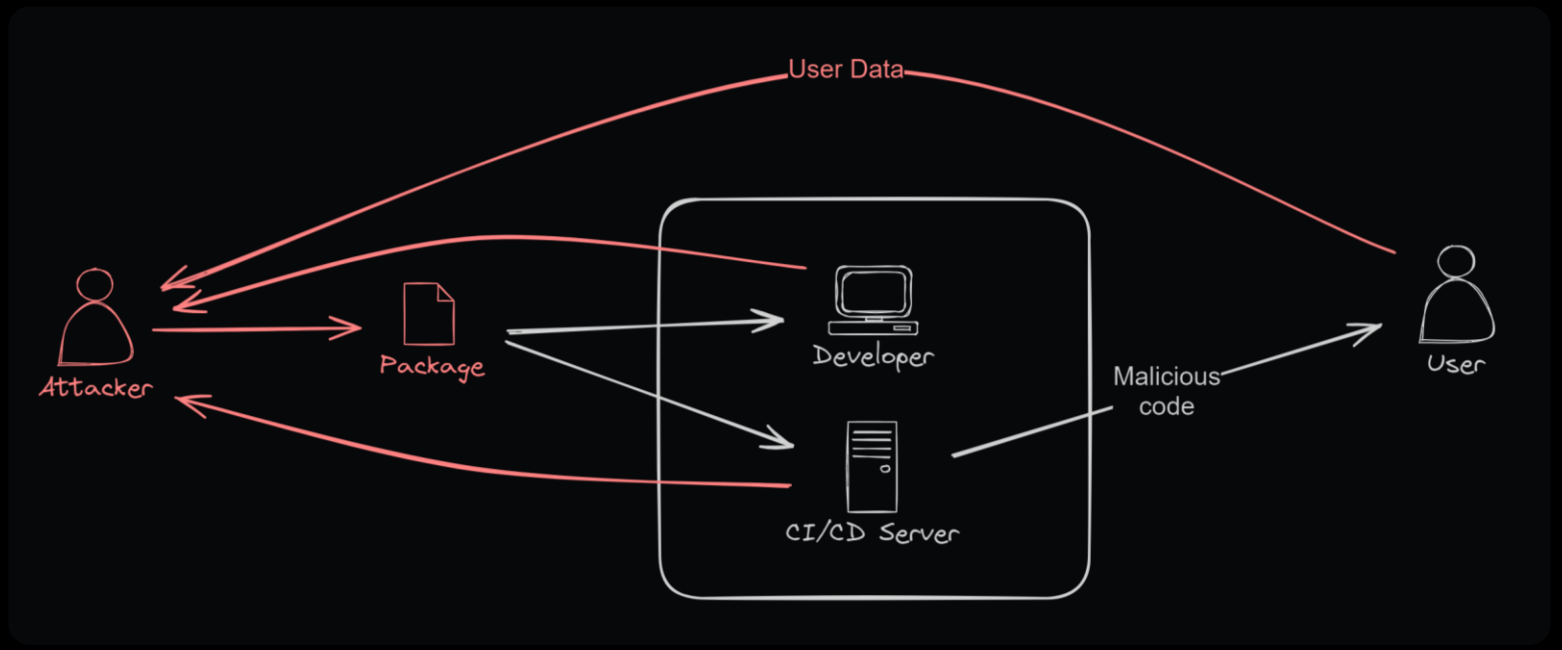
Vulnerable and Outdated Components

- Повышение рисков и угрозы атак
- Угроза процессам сборки и поставки

Vulnerable and Outdated Components



Vulnerable and Outdated Components



Cheat Sheet

Vulnerable and Outdated Components

https://cheatsheetseries.owasp.org/cheatsheets/NPM_Security_Cheat_Sheet.html

- Удаляйте неиспользуемые зависимости
- Проверяйте зависимости на уязвимости
- Используйте доверенный источник зависимостей
- Не храните секреты в репозитории
- Не запускайте скрипты хуков пакетного менеджера
- Проверяйте зависимости на наличие обновлений

Vulnerable and Outdated Components

Требования SSDLC 1

Проекти-рование SSDLC

Разработка SSDLC

Тестирование SSDLC

>

Периодичность проверки уязвимых компонентов

Vulnerable and Outdated Components

Требования SSDLC 1 | Проекти-рование SSDLC | Разработка SSDLC | Тестирование SSDLC

- Периодичность проверки уязвимых компонентов
- Периодичность обновления устаревших компонентов

Vulnerable and Outdated Components

Требования SSDLC 1 | Проекти-рование SSDLC | Разработка SSDLC | Тестирование SSDLC

- Периодичность проверки уязвимых компонентов
- Периодичность обновления устаревших компонентов
- Использование достоверного источника компонентов

Vulnerable and Outdated Components

Требования SSDLC 1 | Проекти-рование SSDLC | Разработка SSDLC | Тестирование SSDLC

- Периодичность проверки уязвимых компонентов
- Периодичность обновления устаревших компонентов
- Использование достоверного источника компонентов
- План обновления мажорных версий компонентов

Vulnerable and Outdated Components

Требования SSDLC 1 | Проекти-рование SSDLC | Разработка SSDLC | Тестирование SSDLC

- Периодичность проверки уязвимых компонентов
- Периодичность обновления устаревших компонентов
- Использование достоверного источника компонентов
- План обновления мажорных версий компонентов
- Риск эксплуатации известных уязвимостей

Vulnerable and Outdated Components

Требования SSDLC

Проекти-рование SSDLC

2

Разработка SSDLC

Тестирование SSDLC

>

Уязвимые компоненты —
в начале спринта.
Обновление с уязвимостями
medium и выше —
в течение спринта

Vulnerable and Outdated Components

Требования SSDLC | Проекти-рование SSDLC | Разработка SSDLC | Тестирование SSDLC

2

>

Уязвимые компоненты — в начале спринта. Обновление с уязвимостями medium и выше — в течение спринта

>

Неиспользуемые компоненты — в начале спринта. Удаление — в течение спринта

Vulnerable and Outdated Components

Требования SSDLC | Проекти-рование SSDLC 2 | Разработка SSDLC | Тестирование SSDLC

>

Уязвимые компоненты — в начале спринта. Обновление с уязвимостями medium и выше — в течение спринта

>

Неиспользуемые компоненты — в начале спринта. Удаление — в течение спринта

>

Мажорные версии — раз в полгода

Vulnerable and Outdated Components

Требования SSDLC | Проекти-рование SSDLC | Разработка SSDLC | Тестирование SSDLC

2

- Уязвимые компоненты — в начале спринта. Обновление с уязвимостями medium и выше — в течение спринта
- Неиспользуемые компоненты — в начале спринта. Удаление — в течение спринта
- Мажорные версии — раз в полгода
- Минорные версии — раз в квартал

Vulnerable and Outdated Components

Требования SSDLC

Проекти-рование SSDLC

Разработка SSDLC

3

Тестирование SSDLC

>

Перейти
на внутренний
артефакторий
компонентов

Vulnerable and Outdated Components

Требования SSDLC

Проекти-рование SSDLC

Разработка SSDLC

3

Тестирование SSDLC

> Перейти на внутренний артефакторий компонентов

> Закрепить зависимости

Vulnerable and Outdated Components

Требования SSDLC

Проекти-рование SSDLC

Разработка SSDLC

3

Тестирование SSDLC

> Перейти на внутренний артефакторий компонентов

> Закрепить зависимости

> Автоматическая проверка в CI/CD (npm audit, sonarqube)

Vulnerable and Outdated Components

Требования SSDLC | Проекти-рование SSDLC | Разработка SSDLC | Тестирование SSDLC

4

>

Проверка **уязвимых** компонентов

Vulnerable and Outdated Components

Требования SSDLC

Проекти-рование SSDLC

Разработка SSDLC

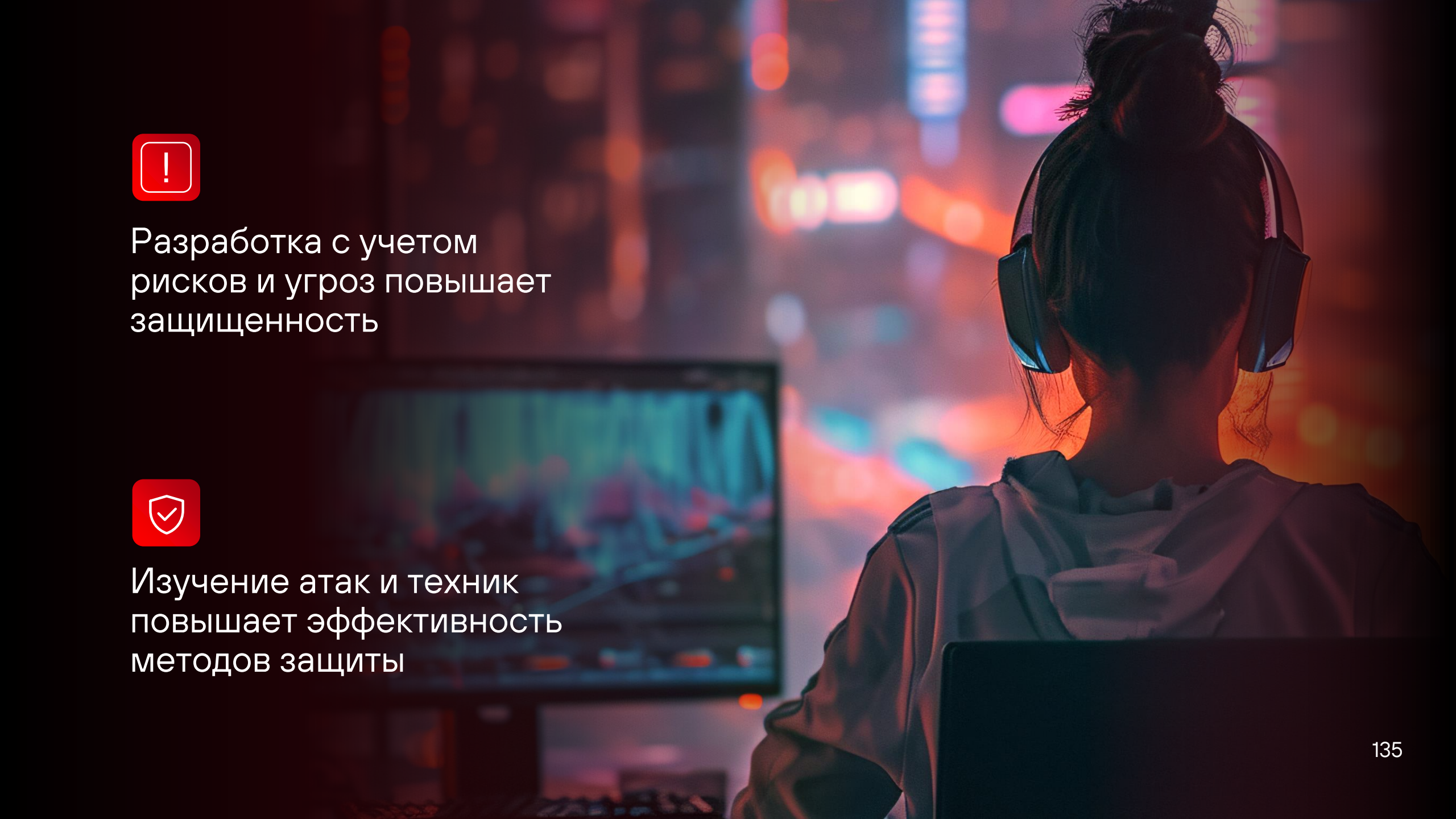
Тестирование SSDLC

4

- Проверка **уязвимых** компонентов
- Проверка **устаревших** компонентов

4

Заключение



Разработка с учетом рисков и угроз повышает защищенность

Изучение атак и техник повышает эффективность методов защиты


Спасибо!

Q&A

t.me/fyzlog

t.me/Positive_
Frontend

Ссылки на ресурсы

https://www.ptsecurity.com/ru-ru/research/analytics/results-of-pentests-2021-2022/

https://www.ptsecurity.com/ru-ru/research/analytics/kiberbezopasnost-v-2023-2024-gg-trendy-i-prognozy-chast-tretya/?sphrase_id=301187

https://www.ptsecurity.com/ru-ru/research/analytics/web-vulnerabilities-2020-2021/

https://www.ptsecurity.com/ru-ru/research/analytics/

Ссылки на ресурсы

- https://owasp.org/
- https://owasp.org/www-project-top-ten/
- https://owasp.org/www-project-top-10-client-side-security-risks/
- https://owasp.org/www-project-application-security-verification-standard/

Ссылки на ресурсы

- https://owasp.org/www-project-cheat-sheets/
- https://owasp.org/www-project-web-security-testing-guide/
- https://owasp.org/www-project-juice-shop/
- https://owasp.org/www-community/Broken_Access_Control

Ссылки на ресурсы

- https://cheatsheetseries.owasp.org/cheatsheets/Authorization_Cheat_Sheet.html
- https://github.com/OWASP/www-project-proactive-controls/blob/master/v3/en/c7-enforce-access-controls.md
- https://github.com/OWASP/ASVS/blob/master/4.0/en/0x12-V4-Access-Control.md
- https://owasp.org/www-project-web-security-testing-guide/v42/4-Web_Application_Security_Testing/05-Authorization_Testing/

Ссылки на ресурсы

- https://owasp.org/www-community/attacks/DOM_Based_XSS
- https://cheatsheetseries.owasp.org/cheatsheets/DOM_based_XSS_Prevention_Cheat_Sheet.html
- https://owasp.org/www-project-web-security-testing-guide/v42/4-Web_Application_Security_Testing/11-Client-side_Testing/01-Testing_for_DOM-based_Cross_Site_Scripting
- https://github.com/wisec/domxsswiki/wiki/

Ссылки на ресурсы

- https://github.com/payloadbox/xss-payload-list
- https://cheatsheetseries.owasp.org/cheatsheets/XSS_Filter_Evasion_Cheat_Sheet.html
- https://github.com/Edr4/XSS-Bypass-Filters
- https://gist.github.com/rvrsh3ll/09a8b933291f9f98e8ec

Ссылки на ресурсы

- https://xss-game.appspot.com/
- http://www.xssgame.com/
- https://owasp.org/Top10/A06_2021-Vulnerable_and_Outdated_Components/
- https://owasp.org/www-project-top-10-insider-threats/docs/2023/INT01_2023-Outdated_Software