

Молдавский Государственный Университет Молдовы  
Факультет Математики и Информатики  
Департамент Информатики

Лабораторная работа №5  
по предмету “Безопасность Информационных Систем”  
тема:” Инструменты сканирования уязвимостей OWASP-ZAP”

Преподаватель: Dr Conf. Unif. Новак Л.  
Выполнила: Павлышина Александра I2302

Кишинев, 2024

## Цель работы

1. Использовать инструменты OWASP для сканирования уязвимостей в веб-приложениях (2-3 веб-приложения).
2. Определить, какие уязвимости встречаются, и описать их.
3. Каковы методы решения тех проблем, которые вызваны определенными уязвимостями?
4. Определить другие приложения для сканирования уязвимостей для веб-приложений.

## Ход работы

OWASP ZAP (Zed Attack Proxy) — это бесплатный инструмент с открытым исходным кодом для тестирования безопасности веб-приложений, разработанный OWASP (Open Web Application Security Project). Он служит как прокси-сервер, перехватывая трафик между пользователем и тестируемым веб-приложением, что позволяет анализировать, изменять и сканировать запросы и ответы.

Я устанавливаю OWASP ZAP с официального сайта, захожу в приложение и настраиваю прокси сервер в браузере и настройках. Проверяю подключение, перейдя на сайт <http://example.com>, все правильно работает.

## Изменить прокси-сервер

Использовать прокси-сервер

☒ Вкл.

IP-адрес прокси-сервера: 127.0.0.1      Порт: 8080

Не использовать прокси-сервер для адресов, которые начинаются с указанных ниже записей. Для разделения записей используйте точку с запятой (;).

☐ Не использовать прокси-сервер для локальных (внутрисетевых) адресов

Сохранить      Отмена

Сессия без названия - 20241028-180435 - ZAP 2.15.0

Файл   Плавка   Вид   Анализ   Отчет   Инструменты   Импортить   Экспорт   Онлайн   Справка

Стандартный режим

Сайты   Быстрый старт   Запрос   Ответ   Запрос

Контексты  
Контекст по умолчанию  
Сайты  
http://example.com

Welco... ZAP by Checkmarx

ZAP это простой в использовании инструмент для поиска уязвимостей в веб-приложениях.

Если вы новичок в ZAP, то лучше всего начать с одного из вариантов ниже.

Автоматическое ска...    Руководство Иссле...    Учить больше

История   Поиск   Оповещения   Вывод


Фильтр: Выкл   Экспорт

Иденти...	Исто...	Req. Отметк...	M...	URL-адрес	Прич...	Наивысшее пред...	Приме...	Теги	Размер Re...
1	...	28.10.2024, 18...	GET	http://example.com/	3...	Not ...	4...	Низкий	0 байт

Оповещения   0   0   1   1   Основной прокси: localhost:80   Текущие сканирования   0   0   0   0   0   0   0

Первое веб-приложение, которое я просканирую, будет Cineplex ([CineplexMD](https://cineplex.md/)). Во вкладке “Быстрый старт” в поле URL для атаки я ввожу адрес веб-сайта и нажимаю на кнопку “Атака”.

⚡ Быстрый старт ➡ Запрос ⬅ Ответ ⏏ Прерывание 📄 Консоль запуска сценариев 🖨 Запрос



## Автоматическое сканирование

Этот экран позволяет запустить автоматическое сканирование приложения - просто введите его URL-адрес ниже и нажмите «Атака».

Пожалуйста имейте в виду, что вы должны атаковать только те приложения, к которым в получили доступ специально для тестирования.

URL для атаки:

Используйте традиционного паука:

☒

Использование ajax spider

If Modern

 с 

Chrome

⚡ Атака

⏏ Остановить

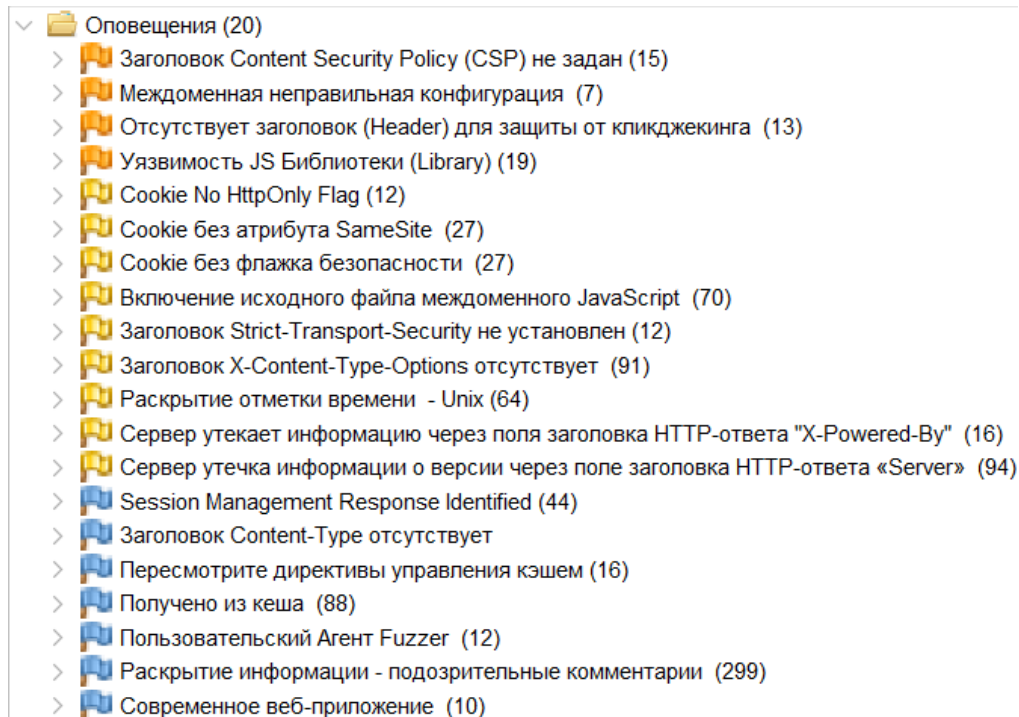
Текущее состояние:

Атака завершена - смотрите вкладку "Оповещения" с информацией об обнаруженных проблемах

Начинается сканирование, подробности которого можно проследить в нижней панели в “Паук”.

История		Поиск		Прерывания		Оповещения		Активное Сканирование		Подбор и обнаружение ресурсов		Параметры		Автоматизация		Фаззер		Паук	
Новое сканирование		Текущее состояние:		0: https://cineplex.md/				100 %		Текущие сканирования: 0		Найденные URI: 60		Добавлены узлы					
URL-адреса		Добавленные узлы		Сообщения															
Обработано		Метод		URI															
		GET		https://cineplex.md/assets/img/youtube_lg.svg															
		GET		https://cineplex.md/assets/img/tiktok.svg															
		GET		https://cineplex.md/assets/img/visa.png															
		GET		https://cineplex.md/assets/img/mastercard.png															
		GET		https://cineplex.md/assets/img/ios_download.svg															
		GET		https://play.google.com/intl/en_us/badges/static/images/badges/ro_badge_web_generic.png		Вне рамки													
		GET		http://www.w3.org/2000/svg		Вне рамки													
		GET		https://fontawesome.com/		Вне рамки													
		GET		https://fontawesome.com/license/free		Вне рамки													

После завершения сканирования мы переходим в этой же панели во вкладку “Оповещения”, где можно наблюдать все уязвимости данной веб-страницы.



CSP (Content Security Policy) — стандарт защиты сайтов от атак с внедрением контента

- Отсутствие заголовка CSP означает, что браузер принимает весь загружаемый контент без ограничений, что увеличивает вероятность атак типа XSS и внедрения вредоносного кода.

Методы устранения:

- Добавьте заголовок Content-Security-Policy с настройками, соответствующими политике безопасности вашего приложения. Например:

```
Content-Security-Policy: default-src 'self'; script-src 'self' trusted.com; img-src 'self';|
```

- Неправильная междоменная конфигурация возникает, когда веб-приложение разрешает доступ всем доменам (Access-Control-Allow-Origin: \*), что делает его уязвимым для атак через междоменные запросы.

Методы устранения:

- Настройте заголовок Access-Control-Allow-Origin так, чтобы только доверенные домены имели доступ. Например:

```
Access-Control-Allow-Origin: https://trusted-domain.com|
```

- Если заголовок защиты от клидджекинга (X-Frame-Options) отсутствует, злоумышленники могут внедрить ваш сайт в iFrame на другой странице.

Методы устранения:

- Добавьте заголовок X-Frame-Options с параметрами DENY или SAMEORIGIN для предотвращения внедрения страницы в iFrame.

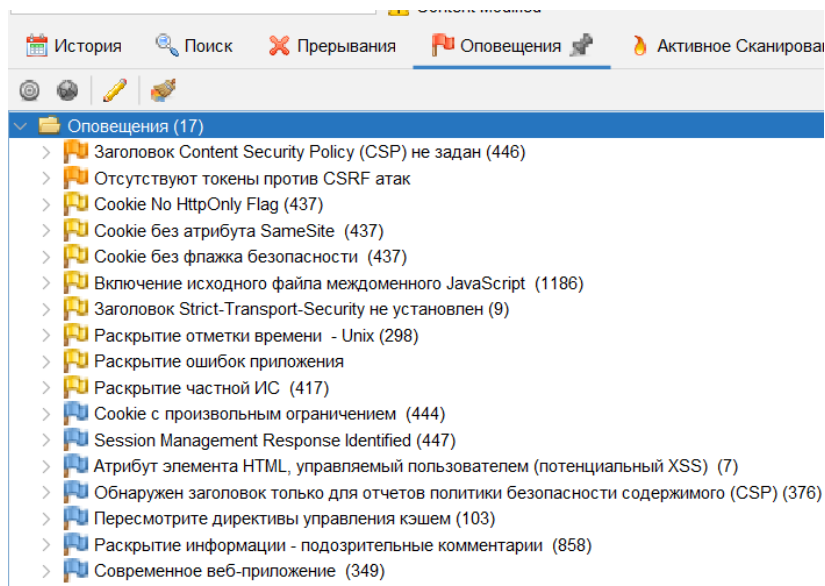
```
X-Frame-Options: DENY|
```

- Устаревшие или уязвимые JavaScript-библиотеки могут содержать уязвимости, такие как XSS, уязвимости обхода CORS, уязвимости подделки запросов и другие.

Методы устранения:

- Используйте всегда последнюю стабильную версию каждой JavaScript-библиотеки.
- Периодически проводите проверку на наличие уязвимостей в зависимостях с помощью инструментов, таких как Snyk, Dependabot или npm audit.

Далее я проделываю те же шаги, но уже с сайтом **Netflix** ([Netflix](https://www.netflix.com/)). У него можно заметить такие уязвимости:



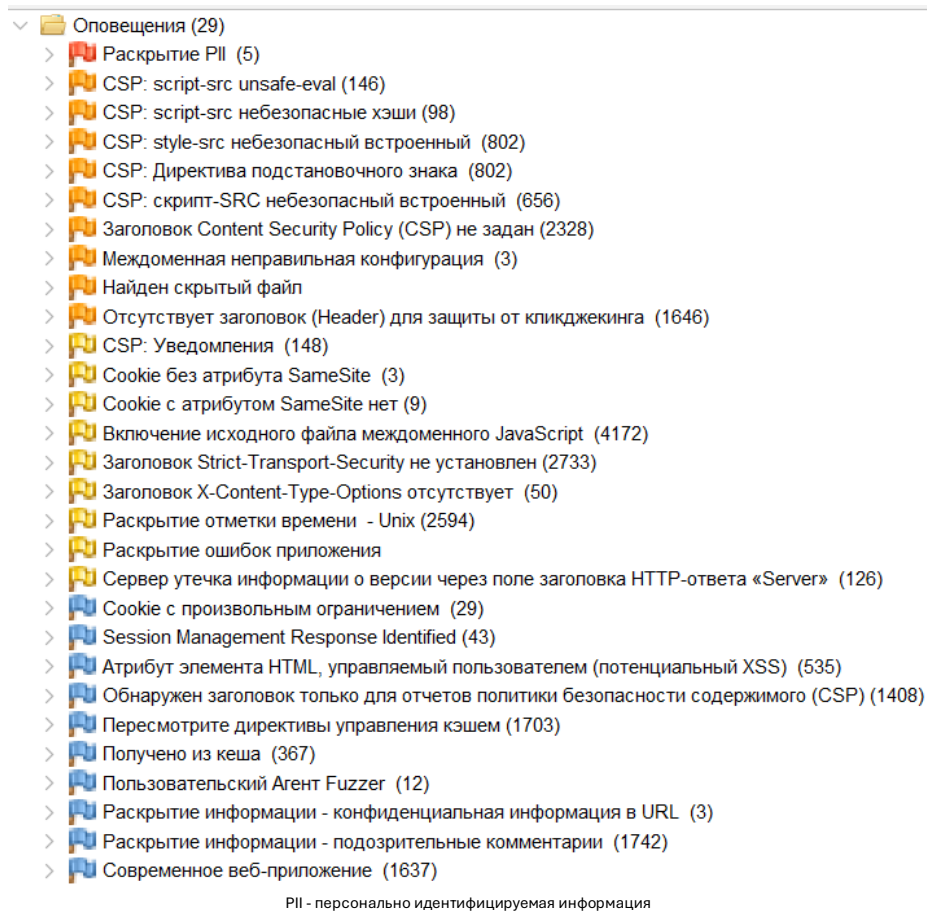
- Отсутствие CSRF-токенов означает, что приложение не использует специальный уникальный маркер (токен), который генерируется для каждой сессии или действия. Этот токен должен быть включен в запросы, отправляемые на сервер, чтобы подтвердить, что запрос инициирован из доверенного источника (веб-приложения) и не является подделанным.

Методы устранения:

1. Внедрение CSRF-токенов:
  - Генерируйте уникальный CSRF-токен для каждой пользовательской сессии или действия. Этот токен должен быть включен в каждый запрос, отправляемый с клиента на сервер.
2. Проверка токенов на сервере:
  - На стороне сервера при каждом запросе проверяйте, что переданный CSRF-токен совпадает с токеном, сохраненным в сессии пользователя.

- Если токен отсутствует или не совпадает, сервер должен отклонить запрос и вернуть ошибку.
3. Использование фреймворков с защитой от CSRF:
- Многие современные фреймворки, такие как Django, Spring, Ruby on Rails, имеют встроенную защиту от CSRF. Включите эту опцию, если она поддерживается, и убедитесь, что все формы и запросы защищены CSRF-токенами.

И третье веб-приложение - это **YouTube** ([YouTube](https://www.youtube.com/)). У данного сайта можно найти следующие уязвимости:



- Раскрытие PII может произойти, если такие данные случайно отображаются в URL, содержимом страниц или логах, где злоумышленники могут их перехватить или использовать.

Методы устранения:

- Скрывайте PII из URL-адресов и исключайте их из файлов журналов.
- Используйте шифрование для передачи PII.

- Ограничьте доступ к PII только для авторизованных пользователей и защитите данные в базе данных.
- Директива `unsafe-eval` позволяет выполнять JavaScript-код через `eval()` и аналогичные функции. Это ослабляет политику безопасности и увеличивает вероятность внедрения вредоносного кода.

Методы устранения:

- Удалите `unsafe-eval` из `script-src` в заголовке CSP и избегайте использования `eval()` в коде.
  - Замените `eval()` на безопасные альтернативы, например, прямое выполнение кода через функции.
- Небезопасные хэши в `script-src` допускают выполнение встроенного JavaScript-кода, определенного хэшем. Если хэш небезопасен или недостаточно уникален, злоумышленник может создать код с тем же хэшем.

Методы устранения:

- Используйте только доверенные хэши и избегайте добавления встроенного JavaScript-кода.
  - Перенесите JavaScript в отдельные файлы и указывайте `nonce` или `sha-256` в CSP, чтобы управлять загрузкой только доверенного контента.
- Включение встроенного JavaScript в `script-src` делает сайт уязвимым для XSS-атак, так как злоумышленники могут легко добавить вредоносный код прямо в HTML.

Методы устранения:

- Вынесите весь JavaScript-код в отдельные файлы и исключите возможность использования встроенного JavaScript.
  - Настройте CSP с использованием `nonce` для каждой страницы или используйте безопасные хэши.
- Использование подстановочных знаков, например, `*` в CSP, позволяет загружать контент с любого источника, что снижает эффективность политики CSP.

Методы устранения:

- Указывайте только конкретные и доверенные источники в каждой директиве CSP. Например, замените `script-src *` на `script-src 'self' https://trusted.com`.
  - Избегайте использования `*` для чувствительных директив, таких как `script-src` и `style-src`.
- Если CORS настроен неправильно, это позволяет злоумышленникам получать доступ к данным вашего приложения с другого домена, а также выявлять и скачивать скрытые файлы.

Методы устранения:

- Настройте CORS, ограничив доступ доверенными доменами. Например:



```
Access-Control-Allow-Origin: https://trusted-domain.com
```

- Скрывайте конфиденциальные и внутренние файлы и проверяйте, что они недоступны через прямой URL.

### Альтернативные приложения для сканирования уязвимостей

**Burp Suite** — широко используемый инструмент для проверки безопасности веб-приложений, поддерживающий продвинутые функции.

**Acunetix** — коммерческое решение, специализирующееся на сканировании веб-приложений.

**Nikto** — бесплатный инструмент для сканирования веб-серверов на наличие потенциальных проблем безопасности.

**Netsparker** — еще один популярный инструмент с функциями автоматического сканирования.

## Вывод

В ходе лабораторной работы проведено сканирование уязвимостей в веб-приложениях с использованием инструментов OWASP, выявлены типичные уязвимости (отсутствие CSP, неправильная конфигурация CORS, CSRF) и предложены методы их устранения. Также рассмотрены дополнительные инструменты для выявления и предотвращения угроз в веб-приложениях.

## Библиография

1. [Обзор OWASP ZAP. Сканер для поиска уязвимостей в веб-приложениях / Хабр](#)
2. [ZAP](#)
3. [OWASP ZAP: Погружение в Мир Ручного Тестирования – Exploit.By](#)
4. [What is OWASP Zed Attack Proxy \(ZAP\)?](#)
5. [Топ-23 сервиса быстрой проверки безопасности сайта – 2023](#)
6. [Веб безопасность: SOP, CORS, CSRF | Mad Devs—блог об IT](#)