

Тестирование на XSS и другие уязвимости с помощью OWASP ZAP

Life-Hack [Жизнь-Взлом]/Хакинг • September 02, 2019

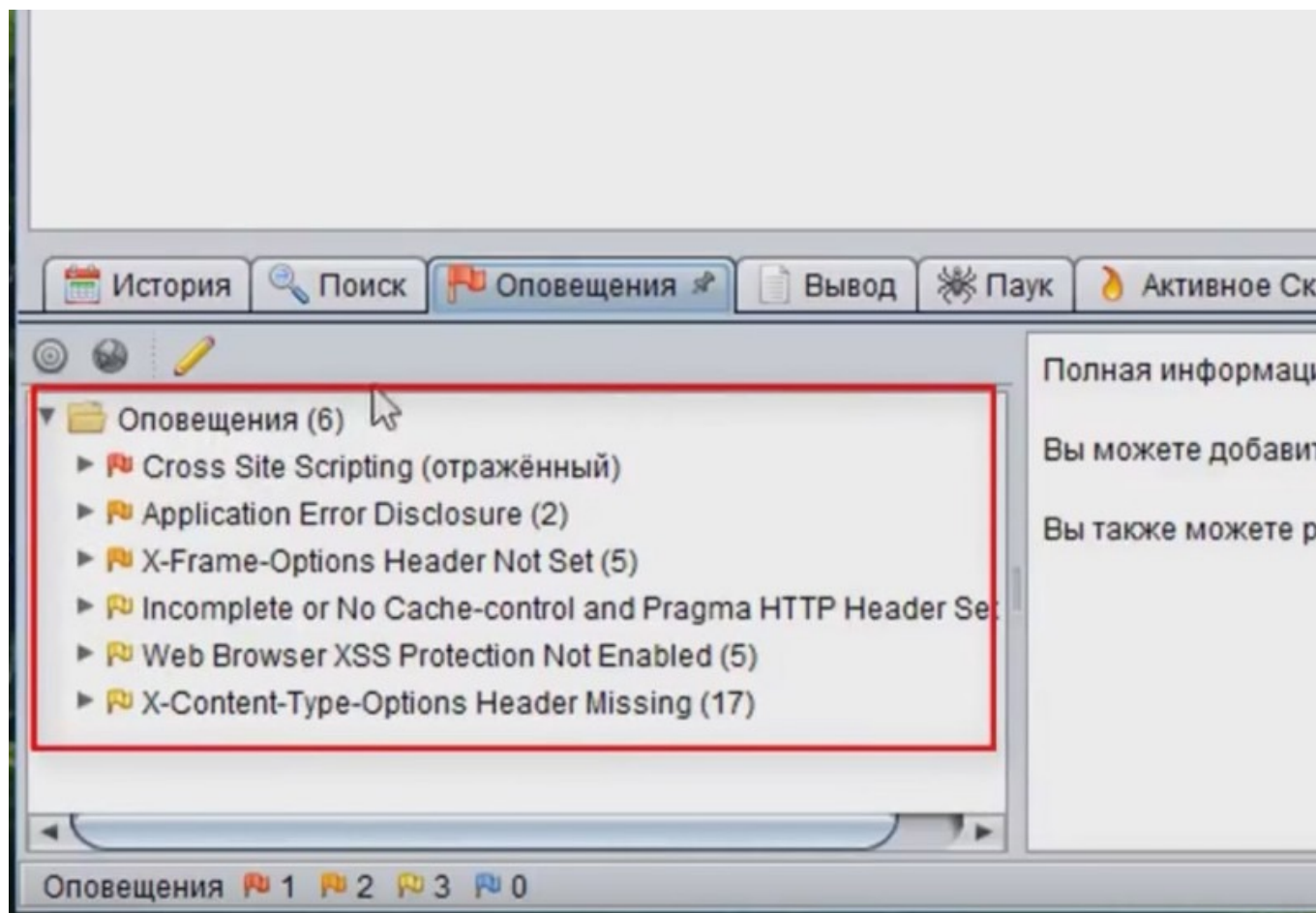
#Обучение

Начнем с того, что такое **OWASP ZAP** — это в первую очередь инструмент, который достаточно прост в использовании для тестирования на проникновение в ваше приложение, а также для поиска уязвимостей в web-приложениях. Программа предназначена как для пользователей имеющих опыт работы в сфере информационной безопасности таких как разработчики и мануальные QA.

Режимы OWASP ZAP

- 1) **Безопасный режим** — с этим режимом нельзя совершить вам что-либо потенциально опасное для вашего приложения.
- 2) **Защищенный режим** — с помощью этого режима пользователь (наш бот) может выполнять только вредоносные действия по URL-адресам, указанным в области браузера.
- 3) **Стандартный режим** — В этом режиме наш пользователь (Бот), может делать все, что имеет значение для нашего приложения.
- 4) **Режим АТТАСК** — при нахождении новых узлов в области действия шпиона, активно сканируются, как только они обнаруживаются.

Оповещение



После того как запустите сканер, все найденные ошибки **OWASP ZAP** будут отсортированные под разные серьезности уязвимостей и будут находится во вкладке “Оповещение”. Под красный флажок попадут самые серьезные, такие как **XSS**, **SQLinjection**. Под оранжевый менее серьезные, типа **CSRF** и тд. Ну и остальные малозначимые на, которые мало кто обращает внимание, хотя стоило бы.

Чтоб посмотреть более подробно найденные уязвимости, кликнув несколько по какой-то из уязвимостей.

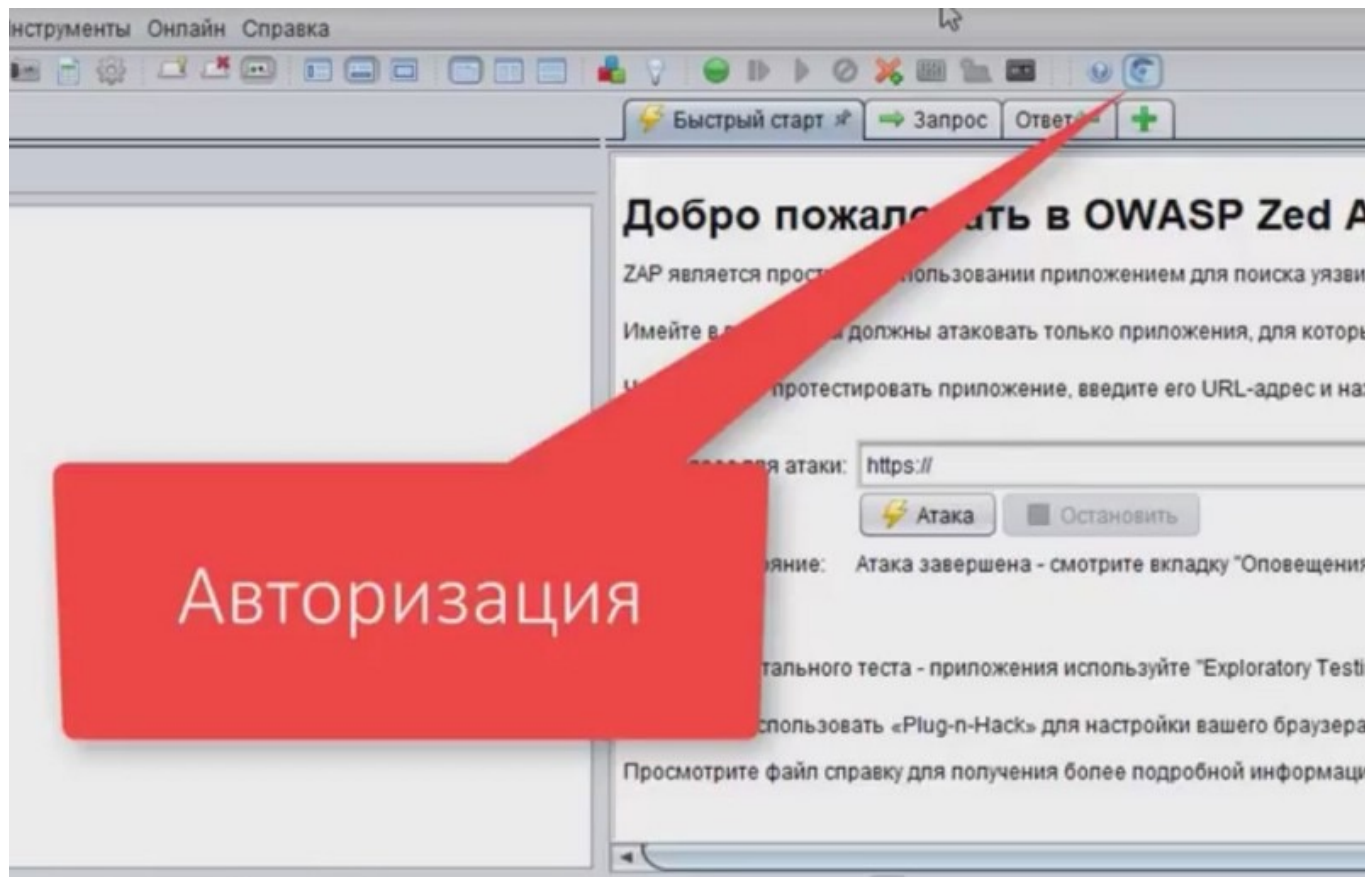
При этом откроется окно в котором будет расписано:

The screenshot shows a window titled "Изменить оповещение" (Edit Notification) with a close button. The main content area is for a "Cross Site Scripting (отражённый)" (Cross Site Scripting (reflected)) vulnerability. The fields are as follows:

- URL-адрес: `https://xss-game.appspot.com/level1/frame?query=%3C%2Fb%3E%3Cscript%3Ealert%281%29%3B%3C%2Fscript%3E%3C%3E`
- Риск: High
- Confidence: Medium
- Параметр: query
- Атака: `<script>alert(1);</script>`
- Evidence: `<script>alert(1);</script>`
- CWE ID: 79
- WASC ID: 8
- Описание: Cross-site Scripting (XSS) is an attack technique that involves echoing attacker-supplied code into a user's browser instance. A browser instance can be a standard web browser client, or a browser object embedded in a software product such as the browser within WinAmp, an RSS reader, or an email client. The code itself is usually written in HTML/JavaScript, but may also extend to VBScript, ActiveX, Java, Flash, or any other browser-supported technology.
- Дополнительно: (Empty text area)
- Решение: Phase: Architecture and Design
Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid. Examples of libraries and frameworks that make it easier to generate properly encoded output include Microsoft's Anti-XSS library, the OWASP ESAPI
- Ссылка: (Empty text area)

- что это за уязвимость и на, что она влияет
- ее критичность
- где ее можно воспроизвести
- литература как ее можно починить

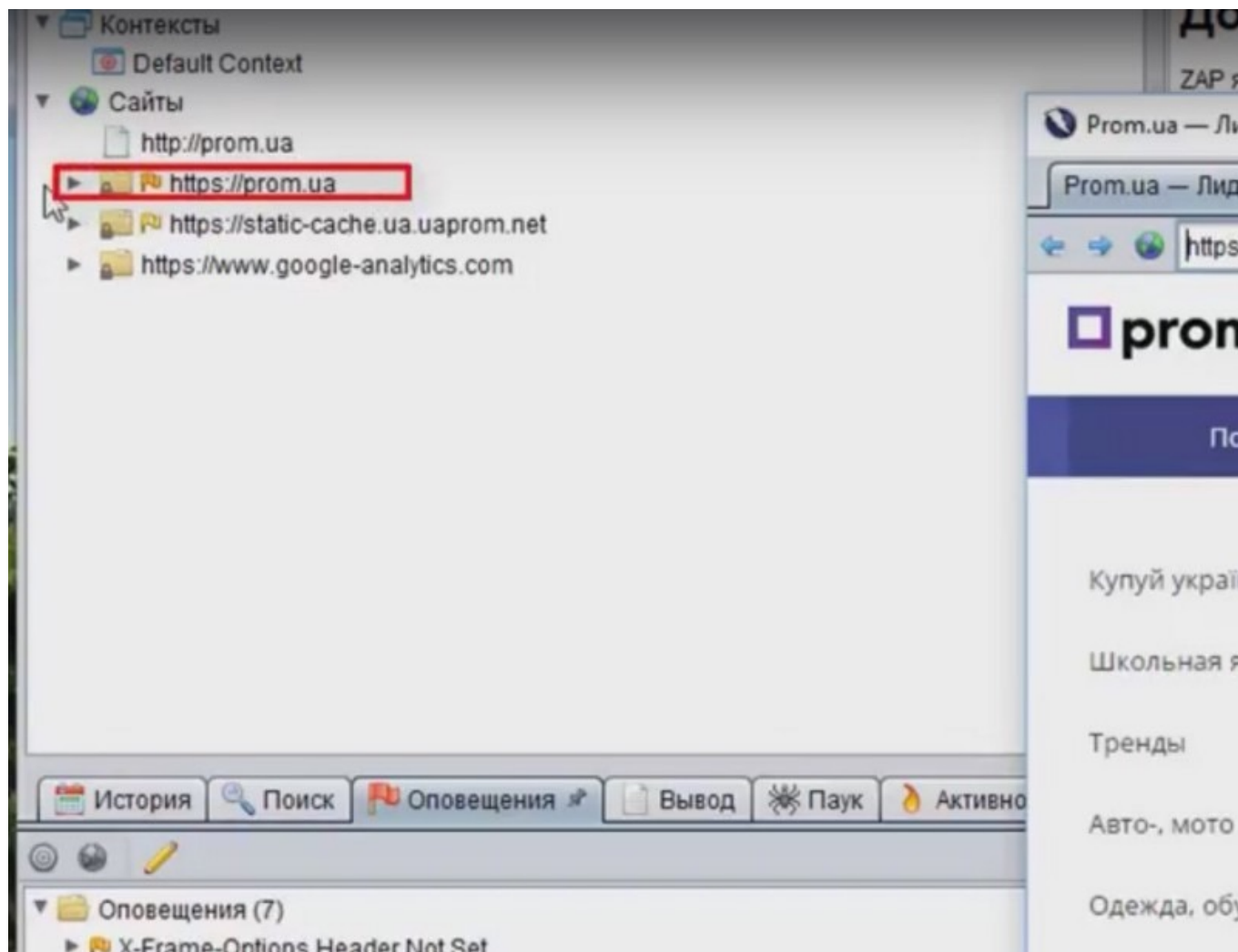
Авторизация



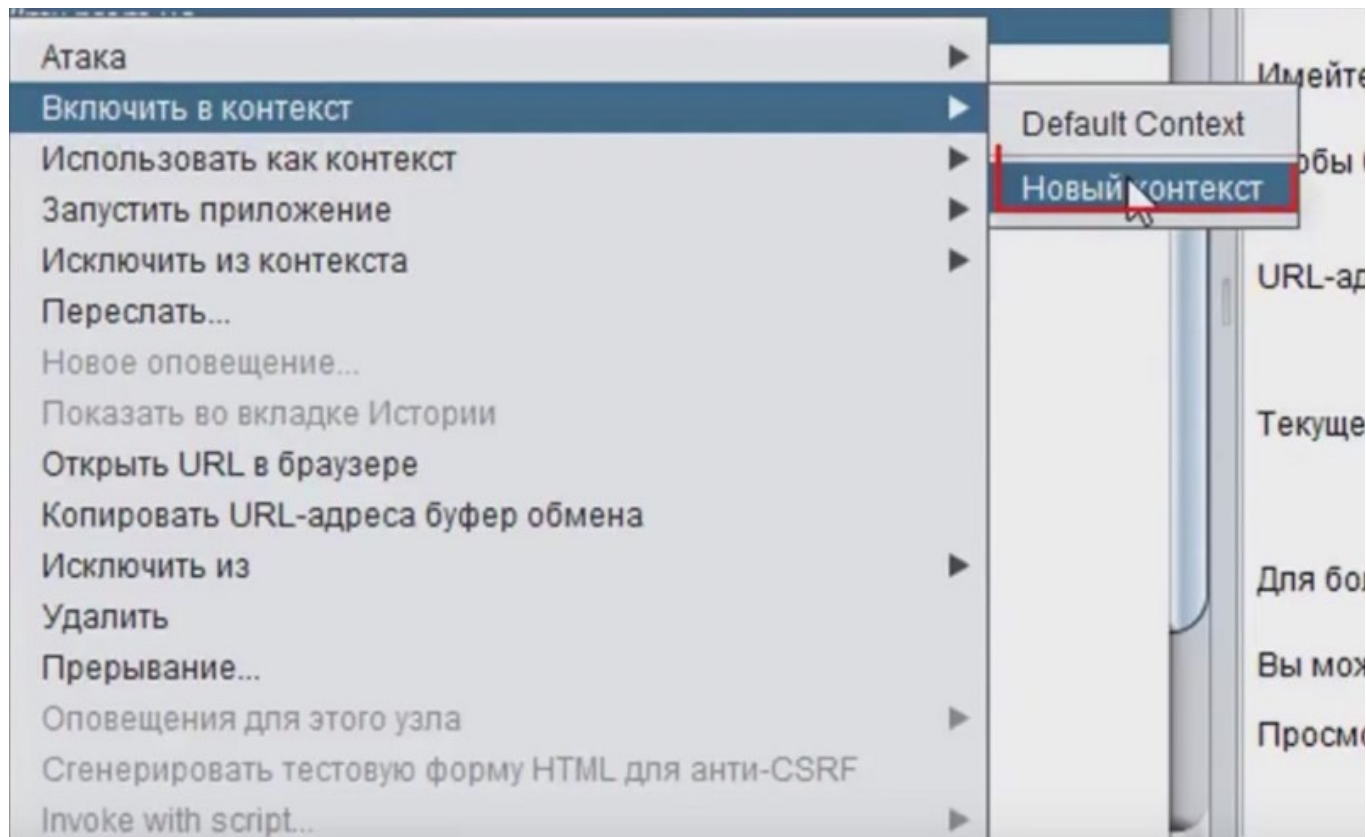
Если в вашем проекте присутствует авторизация, то те ссылки, которые доступны для авторизованного пользователя, не будут доступны для **OWASP ZAP** без настроенного юзера, под которым сможет зайти наше приложение для сканирования ссылок внутри.

Для того, чтоб быстро настроить этого юзера, кликаем по иконке браузера. При этом откроется сам браузер, в котором введены настройки прокси для приема данных приложением **OWASP**.

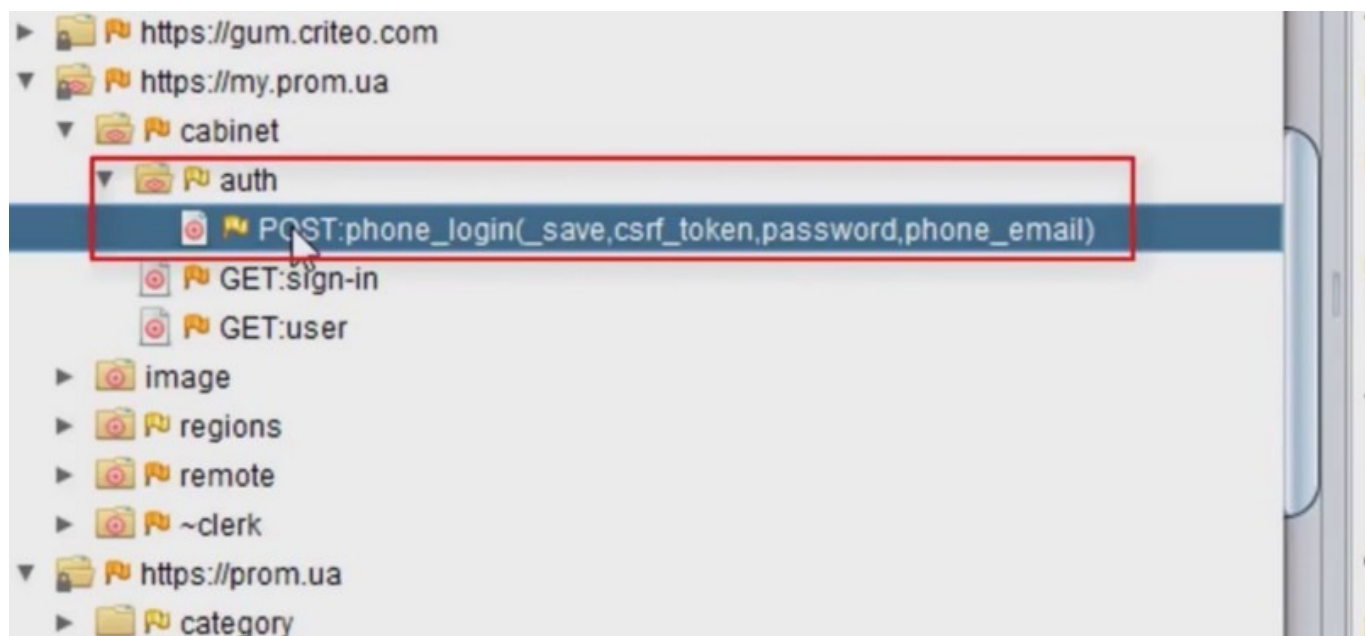
Выполните авторизацию на сайт через этот браузер. Все данные, которые вы ввели будут перехвачены нашим приложением **OWASP ZAP**.

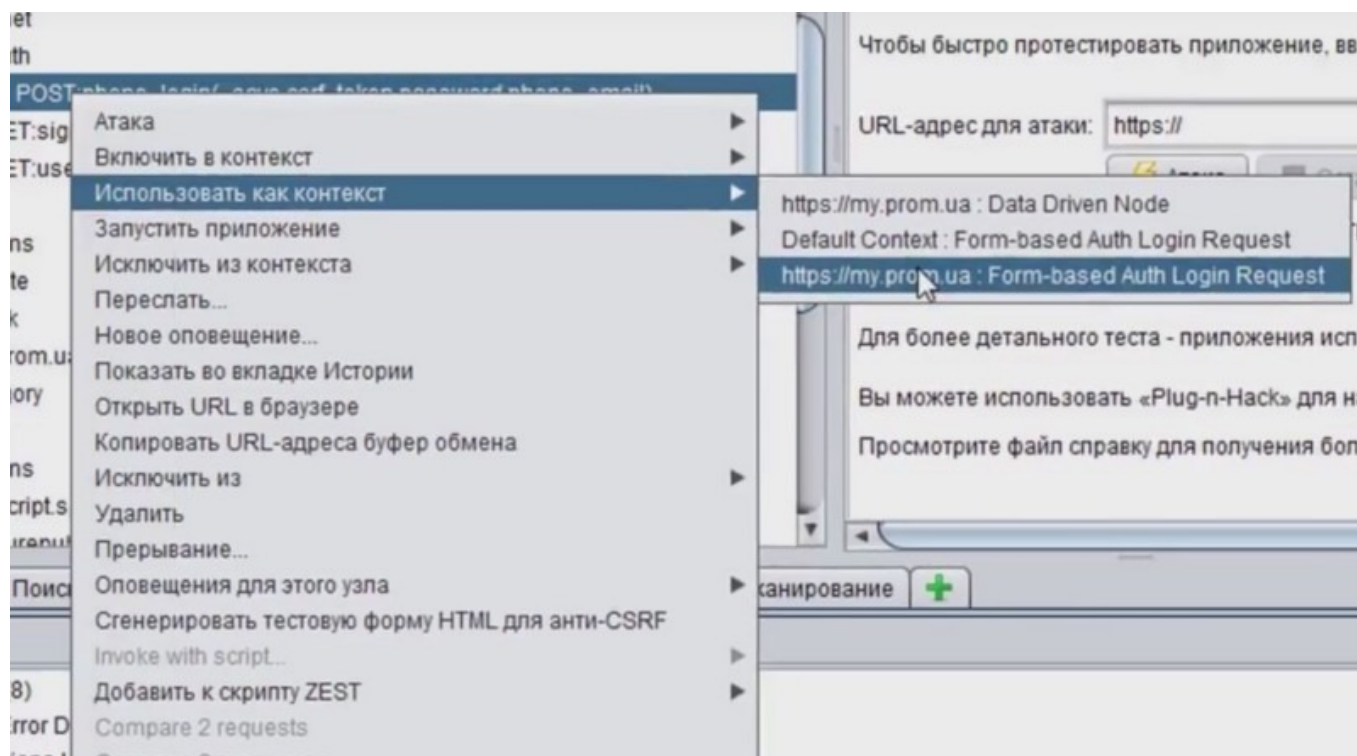


Затем, чтоб наше приложение на сканировало все что в него попадет (Лишние ресурсы), не касающегося нашего проекта, нужно задать “контекст” только для одной папки, которую мы хотим проверить. Для этого выбираем нужную папку и включаем ее в контекст.

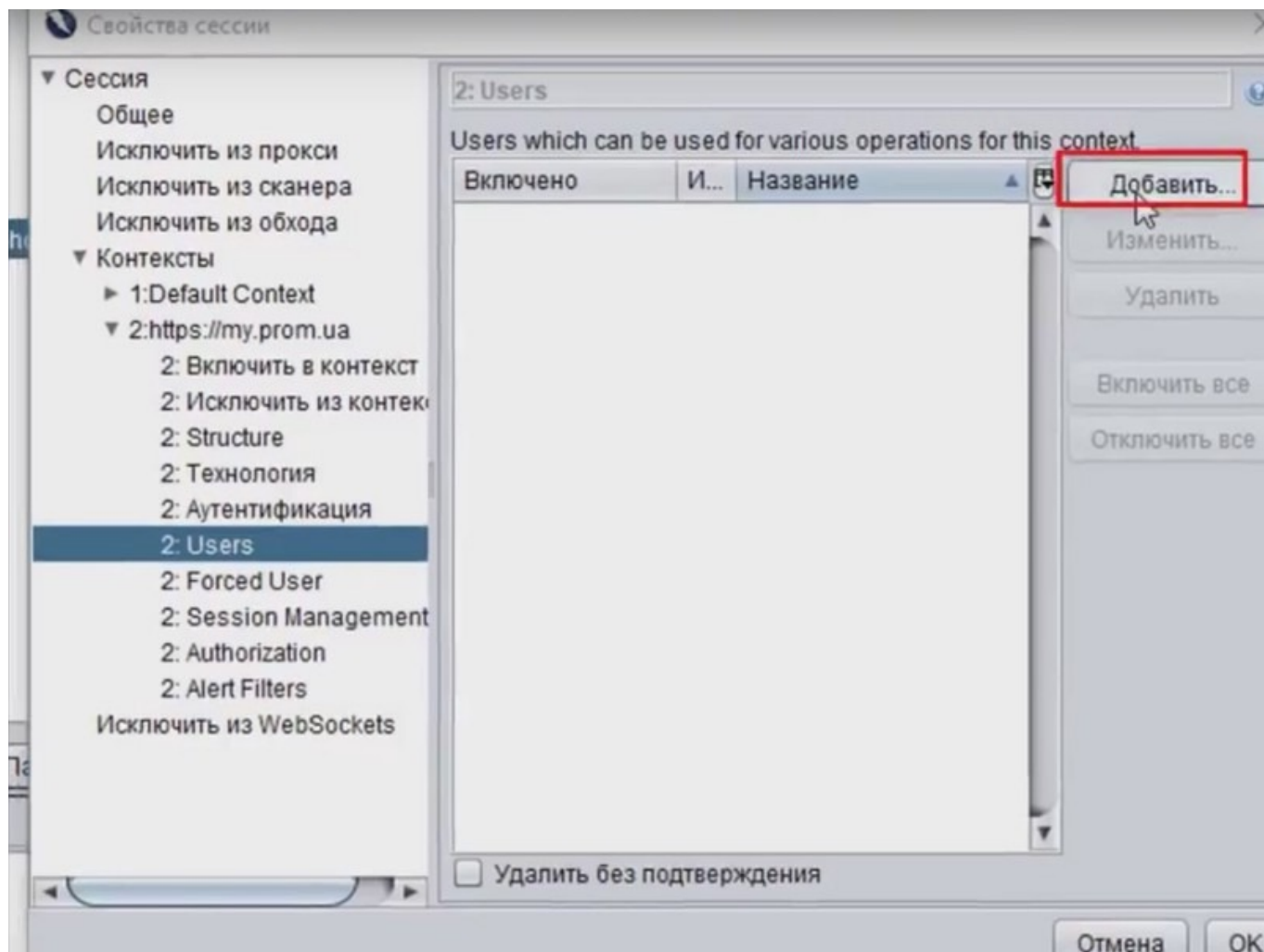


После того как мы задали папке контекст, мы должны найти в ней запрос который выполнялся на авторизацию. А затем создать юзера для этого метода.



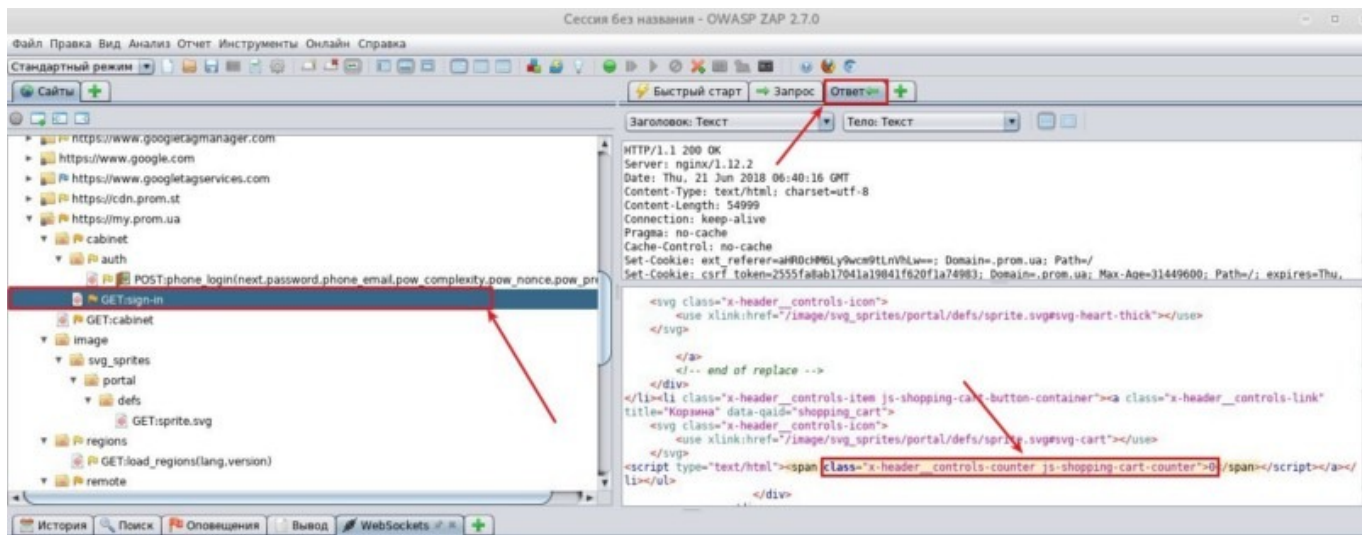


После выполненных операций создаем юзера с мылом и паролем, чтоб он мог авторизоваться, при сканировании нашего проекта. Для этого заходим в раздел юзеров. И в этом разделе добавляем нашего пользователя, который сможет авторизоваться.

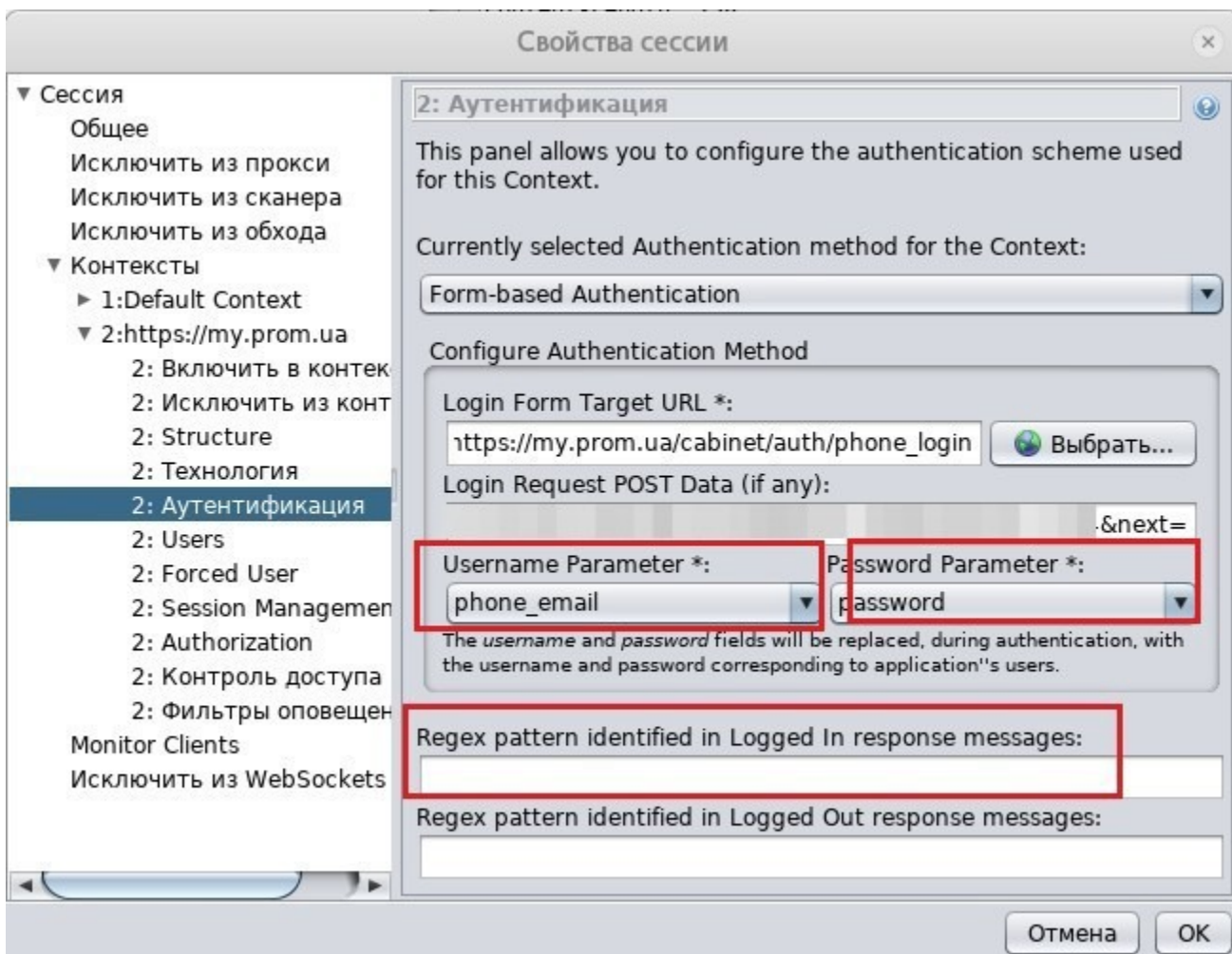


В открывшемся попапе задаем этому юзеру имя(не имеет значение какое, это делается для вашего понимания, что это за юзер), пароль и логин(email), который подходит для авторизации в нашу систему.

Теперь нам надо показать **OWASP ZAP** какой-то локатор (Xpath) на странице авторизованного юзера, чтоб она понимала, что авторизация прошла успешно. Для этого нам надо зайти на запрос, который мы получили от сервера, с html разметкой, которую получает авторизованные юзер. Затем находим какой-то локатор к которому привяжем наш сканер. Этот локатор **OWASP** будет искать после выполнения авторизации и понимать он успешно ли прошел авторизацию на сайте.

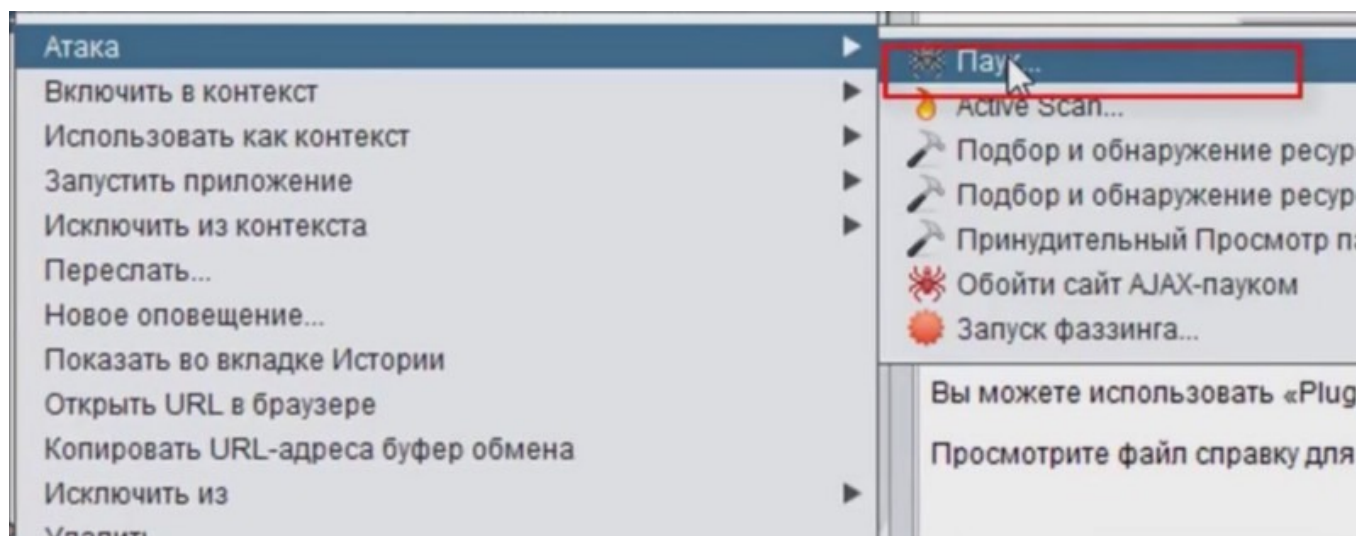


Этот локатор вставляем в раздел аутентификации, в котором мы задавали параметры входа. В инпут **Regex pattern indentified in Logged**

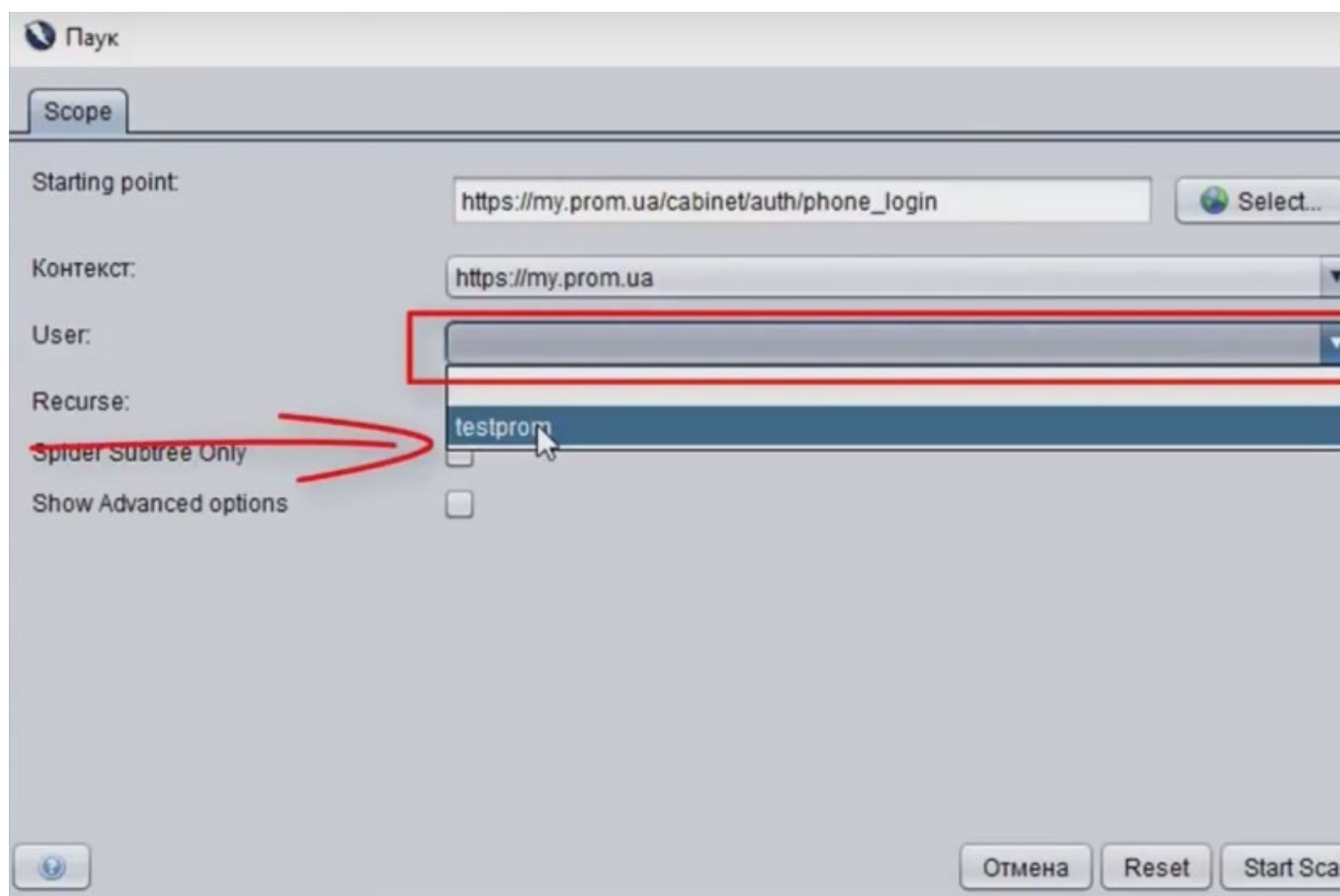


Как только мы выполнили все, что написано выше, можем начинать атаку и проверку на, что способно наше приложение. Для этого кликаем по нашей папке, которой

задали контекст и жмем скан либо атака.



После этого в выпадающем списке находим нашего пользователя, под которым зайдет сканер



И воуля!!!

Пошло сканирование вашего проекта на всевозможные уязвимости.

Но помните никакая из программ не может гарантировать, что будут найдены все уязвимости.

И помните все показанное выше, сделано в целях обучения!!!

Можно применять только на своих проектах, после разрешения.

После этого в выпадающем списке находим нашего пользователя, под которым зайдет сканер

Как только мы выполнили все, что написано выше, можем начинать атаку и проверку на, что способно наше приложение. Для этого кликаем по нашей папке, которой задали контекст и жмем скан либо атака.

Этот локатор вставляем в раздел аутентификации, в котором мы задавали параметры входа. В инпут **Regex pattern indentified in Logged**

В открывшемся попапе задаем этому юзеру имя(не имеет значение какое, это делается для вашего понимания, что это за юзер), пароль и логин(email), который подходит для авторизации в нашу систему.

Теперь нам надо показать **OWASP ZAP** какой-то локатор (Xpath) на странице авторизованного юзера, чтоб она понимала, что авторизация прошла успешно. Для этого нам надо зайти на запрос, который мы получили от сервера, с html разметкой, которую получает авторизованные юзер. Затем находим какой-то локатор к которому привяжем наш сканер. Этот локатор **OWASP** будет искать после выполнения авторизации и понимать он успешно ли прошел авторизацию на сайте.

После выполненных операций создаем юзера с мылом и паролем, чтоб он мог

авторезироваться, при сканировании нашего проекта. Для этого заходим в раздел юзеров. И в этом разделе добавляем нашего пользователя, который сможет авторезироваться.

После того как мы задали папке контекст, мы должны найти в ней запрос который выполнялся на авторизацию. А затем создать юзера для этого метода.

Затем, чтоб наше приложение на сканировало все что в него попадет (Лишние ресурсы), не касающегося нашего проекта, нужно задать “контекст” только для одной папки, которую мы хотим проверить. Для этого выбираем нужную папку и включаем ее в контекст.

Если в вашем проекте присутствует авторизация, то те ссылки, которые доступны для авторизованного пользователя, не будут доступны для **OWASP ZAP** без настроенного юзера, под которым сможет зайти наше приложение для сканирования ссылок внутри.

Для того, чтоб быстро настроить этого юзера, кликаем по иконке браузера. При этом откроется сам браузер, в котором введенны настройки прокси для приема данных приложением **OWASP**.

Выполните авторизацию на сайт через этот браузер. Все данные, которые вы ввели будут перехвачены нашим приложением **OWASP ZAP**.

- что это за уязвимость и на, что она влияет
- ее критичность
- где ее можно воспроизвести
- литература как ее можно починить

Авторизация

После того как запустите сканер, все найденные ошибки **OWASP ZAP** будут отсортированные под разные серьезности уязвимостей и будут находится во вкладке “Оповещение”. Под красный флажок попадут самые серьезные, такие как **XSS**, **SQLinjection**. Под оранжевый менее серьезные, типа **CSRF** и тд. Ну и

остальные малозначимые на, которые мало кто обращает внимание, хотя стоило бы.

Чтоб посмотреть более подробно найденные уязвимости, кликнув несколько по какой-то из уязвимостей.

Программа тут

Источник