




Уязвимость открытого перенаправления

15 АВГУСТА 2018 Г.

 Время чтения ~ 7 минут

TL; DR

В этом посте говорится об открытых / непроверенных перенаправлениях и переадресации. Вы узнаете, *что это такое* , как их найти , как использовать и как исправить .

На заметку

- Прочтите об от [Заявление отказе ответственности](#) перед чтением этого сообщения.
- Требуются базовые знания о HTTP.
- Если у вас есть некоторые знания в области программирования, это тоже поможет, потому что мы рассмотрим код в этом посте.

Вступление

Что такое перенаправления?

Перенаправление означает разрешение веб-сайту перенаправлять запрос ресурсов на другой URL-адрес / конечную точку. Предположим, вы делаете запрос к `apple.com` и `apple.com` может перенаправить вас на другой веб-сайт (`new-apple.com`), так что вы окажетесь на `new-apple.com` хотя исходный запрос был сделан для `apple.com` . Это называется «**перенаправление**» . В HTTP есть разные типы перенаправления, ознакомьтесь с ними ниже.

Код состояния перенаправления - 3xx

- [300 вариантов выбора](#)
- [301 перемещен навсегда](#)
- [302 Найдено](#)
- [303 См. Другое](#)
- [304 Не изменено](#)
- [305 Использовать прокси](#)
- [307 Временное перенаправление](#)
- [308 постоянное перенаправление](#)

Перенаправление может происходить на стороне сервера или на стороне клиента.

На стороне сервера : запрос на перенаправление отправляется на сервер, затем сервер уведомляет браузер о перенаправлении на URL-адрес, указанный в ответе.

На стороне клиента : браузер получает уведомление о перенаправлении на указанный URL-адрес без вмешательства сервера.

Что такое открытые перенаправления?

Open редирект основан на том, что название говорит, Open LY позволяет **перенаправления** на любой сайт.

Почему это проблема?

Ну, это плохо сразу, подумай на минутку, а что, если `apple.com`, **НАДЕЖНЫЙ** веб-сайт позволяет выполнять перенаправление на любой другой веб-сайт. Тогда злоумышленник может просто перенаправить `apple.com` к `attacker.com`, и люди все время на это попадают, веря, что им доверяют, но на самом деле это не так. Поэтому разрешать перенаправления на любой веб-сайт без остановки посередине или без надлежащего уведомления для пользователя - это **плохо**.

Объяснение

Допустим, есть «хорошо известный» веб-сайт - `https://example.com/`.

Предположим, есть ссылка вроде

```
https://example.com/signup?redirectUrl=https://example.com/login
```

Эта ссылка ведет на страницу подписки. После регистрации вы будете перенаправлены на `https://example.com/login` который указан в параметре HTTP GET `redirectUrl`.

Что будет, если мы изменим `example.com/login` К `attacker.com`?

```
https://example.com/signup?redirectUrl=https://attacker.com/
```

Посетив этот URL-адрес, мы будем перенаправлены на `attacker.com` после регистрации это означает, что у нас есть уязвимость открытого перенаправления. Это классический открытый редирект, который часто используется для фишинга (дополнительную информацию см. В разделе «Эксплуатация»).

Почему это происходит?

Это происходит из-за недостаточных проверок перенаправления в серверной части, что означает, что сервер неправильно проверяет, находится ли URL-адрес перенаправления в их белом списке или нет. Вот несколько примеров уязвимого кода

PHP (на стороне сервера)

```
<?php
$url_to_redirect = $_GET['redirect_url'];
header('Location: ' . $url_to_redirect);
die();
```

Здесь php-код слепо захватывает URL-адрес из `redirect_url` параметр и перенаправляет на этот URL-адрес с помощью `Location` Заголовок HTTP.

Java (на стороне сервера)

```
response.sendRedirect(request.getParameter("u"));
```

Здесь jsp-страница берет URL-адрес из параметра `u` и слепо перенаправляет его на указанный URL.

Javascript (на стороне клиента)

```
window.location.href = "https://attacker.com";
```

Мы можем присвоить строку URL `location.href` из `window` объект. Это вызовет перенаправление. Если на месте нет проверок, то это ошибка.

HTML (на стороне клиента)

```
<meta http-equiv="refresh" content="0;URL='http://attacker.com/'" />
```

HTML-метатеги могут обновлять сайт с указанным URL-адресом, поскольку он `content` а также вы можете указать время задержки обновления.

Как их найти?

- Посетите каждую конечную точку цели, чтобы найти эти параметры «перенаправления».
- Просмотрите историю своих прокси, возможно, вы что-нибудь найдете. Обязательно используйте фильтры.
- Брутфорс тоже помогает.
- Вы можете обнаружить множество конечных точек, прочитав код javascript.
- Google - ваш друг, пример запроса: `inurl:redirectUrl=http site:target.com`
- Поймите и проанализируйте, где требуется перенаправление в целевом приложении, например *перенаправление на панель управления постами*

входа в систему или что-то в этом роде.

Некоторые фокусы

- Проверьте базовую модификацию URL-адреса, например `target.com/?redirect_url=https://attacker.com` .
- Попробуйте использовать двойную косую черту `target.com//attacker.com` .
- Попробуйте `target.com/@attacker.com` . В этом случае интерпретация будет такой: `target.com` это имя пользователя и `attacker.com` будет домен.
- Тест на протокол javascript `javascript:confirm(1)` .
- Попробуйте `target.com/?image_url=attacker.com/.jpg` если загружается ресурс изображения.
- Попробуйте использовать IP-адрес вместо имени домена.
- Вы можете пойти дальше с точки зрения представления IP в десятичном, шестнадцатеричном или восьмеричном формате.
- Вы также можете попробовать `target.com/?redirect_url=target.com.attacker.com` чтобы обойти слабые реализации регулярных выражений.
- Китайский разделитель. как точка - `https://attacker%E3%80%82com` .
- Проверка для юникода реверсора (**строкового** «\ u202e») `target.com@%E2%80%AE@attacker.com` .
- Без косой черты `https:attacker.com` .
- Обратные косые черты `http://\\attacker.com` или же `https://\\attacker.com` .
- Другой домен `redirect_url=.jp` в результате перенаправления `target.com.jp` что не то же самое, что `target.com` .
- Попробуй юникод (включая смайлики) безумие `tArget.com` или же `Attacker.com` ('А' означает «\ uD835 \ uDC00»).
- Их гораздо больше, пожалуйста, обратитесь к **Интересные находки и чит-листы** разделу « » этой статьи. Если вы хотите больше трюков, ОТКРОЙТЕ ДЛЯ СЕБЯ ЕМ!

Эксплуатации

Фишинг

Предположим, что цель `example.com`. У него есть страница восстановления пароля по адресу `example.com/forgot-password`. Вы вводите адрес электронной почты и нажимаете кнопку «**Забыли пароль**», и он отправляет вам электронное письмо со ссылкой для сброса пароля, и эта ссылка может выглядеть так

```
https://example.com/reset-password/some-random-token?redirect=https://example.com/login
```

Если мы вмешаемся в `redirect` параметр и изменим его на

```
https://example.com/reset-password/some-random-token?redirect=https://attacker.com/login
```

Это перенаправляет пользователя на *злую страницу входа в систему*, страница если исходная и пользователь могут быть подвергнуты фишингу.

Связь с SSRF

Если вы не знаете о `SSRF`, возможно, вы захотите погуглить, я опущу еще один пост, посвященный SSRF, так что следите за обновлениями. В любом случае, допустим, что у нас есть цель - `example.com` и вы обнаружили ошибку SSRF на `https://example.com/?get-image=https://images.example.com/cat.jpg`. По сути, мы хотим заставить сервер делать запрос от нашего имени. Мы можем изменить `get-image` значение параметра, и сервер делает запрос к этой конечной точке. Но в этом случае он ограничивает запросы своим собственным (под) доменом (ами), например `images.example.com`. Это означает, что вы можете отправить запрос `*.example.com` (Любой поддомен) в качестве сервера и не может получить доступ ни к чему за пределами этой области. Допустим, вы также обнаружили Open Redirect ошибку на `https://test.example.com/redirect_url=https://www.example.com/`, теперь вы можете связать их вместе, чтобы SSRF работал для любого домена, например

Сначала мы Open Redirect исправляем ошибку, изменяя `redirect_url` параметр

```
https://test.example.com/?redirect_url=https://www.attacker.com/
```

Затем мы можем использовать это с SSRF , добавив открытый Open Redirect URL-адрес сверху в качестве `get-image` параметр.

```
https://example.com/?get-image=https://test.example.com/?redirect_url=https://www.attacker.c
```

Теперь это заставит сервер сделать запрос к одному из его собственных поддоменов (`test.example.com`), а затем перенаправляется на `attacker.com` заставить работать ошибку SSRF.

```
-----
⇒ | get-image | ⇒ | test.example.com | ⇒ { REDIRECTS } ⇒ | attacker.com |
-----
```

Подобно тому, как эти открытые перенаправления можно использовать по-разному, я не могу обсуждать их все, это просто зависит от цели.

"Вы ограничены только вашим воображением."

Смягчение

- Используйте перенаправления только в том случае, если они вам действительно нужны.
- Если вы хотите их использовать, убедитесь, что вы правильно проверили домены из белого списка и разрешили совпадающие.
- Вы также можете использовать `hmac` если хотите, но с этим довольно легко справиться (атаки на расширение длины), так что не усложняйте ситуацию, используйте правильный белый список.

Крутые находки

Это список всех уникальных отчетов Open Redirect Reports на HackerOne, которые я смог найти на первых 20 страницах результатов Google.

- [\[Report-226408\]](#) Открыть перенаправление на Shopify

- [\[Report-211213\] Открыть перенаправление на Nextcloud](#)
- [\[Report-246897\] Открытое перенаправление в Twitter](#)
- [\[Report-103772\] Открыть перенаправление на Shopify](#)
- [\[Report-309058\] Открыть перенаправление в Wordpress](#)
- [\[Report-260744\] Открытое перенаправление и XSS в Twitter](#)
- [\[Report-320376\] Открыть перенаправление на HackerOne](#)
- [\[Report-111968\] Обход межстраничного перенаправления / открытое перенаправление в сеансе HackerOne Zendesk](#)
- [\[Report-244721\] Открыть редирект на Mail.Ru](#)
- [\[Report-236599\] Открытое перенаправление на ExpressionEngine](#)
- [\[Report-299403\] Открыть перенаправление на HackerOne](#)
- [\[Report-239503\] Открытое перенаправление и раскрытие информации на HackerOne](#)
- [\[Report-210875\] Открытое перенаправление через заголовок хоста](#)
- [\[Report-119236\] Открыть переадресацию на Uber](#)
- [\[Report-126203\] Открыть переадресацию на Uber](#)
- [\[Report-28865\] Обход открытого фильтра перенаправления на HackerOne](#)
- [\[Report-144525\] Обход открытого перенаправления на New Relic](#)
- [\[Report-104087\] Обход открытого перенаправления с использованием svg в Slack](#)
- [\[Report-179568\] Открыть перенаправление через window.opener на Open-Xchange](#)
- [Открыть Redirect to RCE в приложении Google Hangouts Electron и в твите RCE](#)

Cheat Sheets

Этот список может быть не уникальным, загрузите их все, удалите дубликаты и отсортируйте их.

- [Шпаргалка по открытому перенаправлению EdOverflow](#)
- [cujanovic Open Redirect Payloads](#)

- [Шаблоны URL-адресов для перенаправления fuzzdb](#)
- [ak1t4 Открытые полезные нагрузки перенаправления](#)
- [случайные полезные нагрузки открытого перенаправления robbie](#)
- [Памятка по непроверенным перенаправлениям и переадресации OWASP](#)

Ресурсы

- [Открыть редирект от zseano](#)
- [Открытое перенаправление с помощью SI9INT](#)
- [Использование Burp для проверки открытых перенаправлений](#)
- [Обнаружение непроверенных перенаправлений и переадресации](#)
- [Открытые редиректы, которые имеют значение](#)
- [Перенаправление URL-адресов на стороне клиента](#)
- [Развлечения с редиректами 2010](#)

На этом пока все, ребята. Спасибо за чтение. Хорошего дня :)

- s0cket7

WEBSEC ОТКРЫТОЕ ПЕРЕНАПРАВЛЕНИЕ

ОБНОВЛЕНО 15 АВГУСТА 2018 Г. BY S0CKET7

 ДОЛЯ

 ТВИТНУТЬ

 +1

Читать далее

Запись о веб-эксплуатации Pico CTF 2018

Обзор всех 18 веб-вызовов от PicoCTF. Продолжить чтение

IDOR **приводит к захвату аккаунта**

Опубликовано 16 августа 2018 г.

Прохождение Linux CTF **для начинающих**

Опубликовано 11 августа 2018 г.

© 2018 s0cket7. Питаться от Джекил