24-го сентября 2020

Атака CSRF

Материал на этой странице устарел, поэтому скрыт из оглавления сайта.

Нельзя говорить про AJAX и не упомянуть про важнейшую деталь его реализации – защиту от CSRF-атак.

CSRF (Cross-Site Request Forgery, также XSRF) – опаснейшая атака, которая приводит к тому, что хакер может выполнить на неподготовленном сайте массу различных действий от имени других, зарегистрированных посетителей.

Какие это действия – отправка ли сообщений, перевод денег со счёта на счёт или смена паролей – зависят от сайта, но в любом случае эта атака входит в образовательный минимум веб-разработчика.

Злая форма

«Классический» сценарий атаки таков:

- Вася является залогиненным на сайт, допустим, `mail.com`. У него есть сессия в куках.
- Вася попал на «злую страницу», например хакер пригласил его сделать это письмом или как-то иначе.
- На злой странице находится форма такого вида:

```
<form action="http://mail.com/send" method="POST">
  <input type="hidden" name="message" value="Сообщение">
  ...
</form>
```

- При заходе на злую страницу JavaScript вызывает `form.submit`, отправляя таким образом форму на `mail.com`.
- Сайт `mail.com` проверяет куки, видит, что посетитель авторизован и обрабатывает форму. В данном примере форма предполагает посылку сообщения.

Итог атаки – Вася, зайдя на злую страницу, ненароком отправил письмо от своего имени. Содержимое письма сформировано хакером.

Защита

В примере выше атака использовала слабое звено авторизации.

Куки позволяют сайту `mail.com` проверить, что пришёл именно Вася, но ничего не говорят про данные, которые он отправляет.

Иначе говоря, куки не гарантируют, что форму создал именно Вася. Они только удостоверяют личность, но не данные.

Типичный способ защиты сайтов – это «секретный ключ» (`secret`), специальное значение, которое генерируется случайным образом при авторизации и сохраняется в сессии посетителя. Его знает только сервер, посетителю мы его даже не будем показывать.

Разумеется, для разных посетителей `secret` будет разным.

Затем на основе ключа генерируется «токен» (`token`). Токен делается так, чтобы с одной стороны он был отличен от ключа `secret`, в частности, может быть много токенов для одного ключа, с другой – чтобы было легко проверить по токену, сгенерирован ли он на основе данного ключа или нет.

Для этого для каждого токена используется дополнительное случайное значение, которое называют «соль» `salt`.

Формула вычисления токена:

```
token = salt + ":" + MD5(salt + ":" + secret)
```

Например:

1. В сессии хранится `secret="abcdef"`, это значение создаётся один раз.
2. Для нового токена сгенерируем `salt`, например пусть `salt="1234"`.
3. `token = "1234" + ":" + MD5("1234" + ":" + "abcdef") = "1234:5ad02792a3285252e524ccadeeda3401"`.

Это значение – с одной стороны, случайное, с другой – получив такой `token` от пользователя, мы можем его проверить: разбить по символу двоеточия, взять его левую часть `1234` в качестве `salt` и, зная `secret`, запустить вычисление по формуле выше. Если мы получили то же самое, то `token` правильный, это не хакер.

Обратим внимание: не зная `secret`, невозможно сгенерировать `token`, который сервер воспримет как правильный.

Далее, токен добавляется в качестве скрытого поля к каждой форме, генерируемой на сервере.

То есть, «честная» форма для отсылки сообщений, созданная на `http://mail.com`, будет выглядеть так:

```
<form action="http://mail.com/send" method="POST">
  <input type="hidden" name="csrf" value="1234:5ad02792a3285252e524ccadeeda34(
  <textarea name="message">
    ...
  </textarea>
</form>
```

При её отправке сервер проверит поле `csrf`, удостоверится в правильности токена, и лишь после этого отошлёт сообщение.

«Злая страница» при всём желании не сможет сгенерировать подобную форму, так как не владеет `secret`, и токен будет неверным.

Такой токен также называют «подписью» формы, которая удостоверяет, что форма сгенерирована именно на сервере.

Подпись с полями формы

Эта подпись говорит о том, что автор формы – сервер, но ничего не гарантирует относительно её содержания.

Есть ситуации, когда мы хотим быть уверены, что некоторые из полей формы посетитель не изменил самовольно. Тогда мы можем включить в MD5 для формулы токена эти поля, например:

```
token = salt + ":" + MD5(salt + ":" + secret + ":" + fields.money)
```

При отправке формы сервер проверит подпись, подставив в неё известный ему `secret` и присланное значение `fields.money`. При несовпадении либо `secret` не тот (хакер), либо `fields.money` изменено.

Токен и AJAX

Теперь перейдём к AJAX-запросам.

Что если посылка сообщений в нашем интерфейсе реализуется через XMLHttpRequest?

Как и в случае с формой, мы должны «подписать» запрос токеном, чтобы гарантировать, что его содержимое прислано на сервер именно интерфейсом сайта, а не «злой страницей».

Здесь возможны варианты, самый простой – это дополнительная кука.

1. При авторизации сервер устанавливает куку с именем `CSRF-TOKEN`, и пишет в неё токен.
2. Код, осуществляющий XMLHttpRequest, получает куку и ставит заголовок `X-CSRF-TOKEN` с ней:

```
var request = new XMLHttpRequest();

var csrfCookie = document.cookie.match(/CSRF-TOKEN=([\w-]+)/);
if (csrfCookie) {
```

```
5    request.setRequestHeader("X-CSRF-TOKEN", csrfCookie[1]);
6  }
```

3. Сервер проверяет, есть ли заголовок и содержит ли он правильный токен.

Защита действует потому, что прочитать куку может только JavaScript с того же домена. «Злая страница» не сможет «переложить» куку в заголовок.

Если нужно сделать не XMLHttpRequest, а, к примеру, динамически сгенерировать форму из JavaScript – она также подписывается аналогичным образом, скрытое поле или дополнительный URL-параметр генерируется по куке.

Итого

- CSRF-атака – это когда «злая страница» отправляет форму или запрос на сайт, где посетитель, предположительно, залогинен.

  Если сайт проверяет только куки, то он такую форму принимает. А делать это не следует, так как её сгенерировал злой хакер.
- Для защиты от атаки формы, которые генерирует `mail.com`, подписываются специальным токеном. Можно подписывать не все формы, а только те, которые осуществляют действия от имени посетителя, то есть могут служить объектом атаки.
- Для подписи XMLHttpRequest токен дополнительно записывается в куку. Тогда JavaScript с домена `mail.com` сможет прочитать её и добавить в заголовок, а сервер – проверить, что заголовок есть и содержит корректный токен.
- Динамически сгенерированные формы подписываются аналогично: токен из куки добавляется как URL-параметр или дополнительное поле.

Комментарии

перед тем как писать...