Н. С. Михаленко
(ГГУ имени Ф. Скорины, Гомель)
Науч. рук. **Е. А. Ружицкая**, канд. физ.-мат. наук, доцент

ИСПОЛЬЗОВАНИЕ CSRF-ТОКЕНА ПРИ РАЗРАБОТКЕ WEB-ПРИЛОЖЕНИЙ

CSRF-токены – это случайные строки символов, которые генерируются сервером и отправляются клиенту, обычно в виде скрытых полей формы или HTTP-заголовков. Клиент затем возвращает этот токен с каждым последующим запросом к тому же серверу, а сервер проверяет, совпадает ли он с выданным ранее. Таким образом, сервер может убедиться, что запрос является подлинным и не был подделан злоумышленником. CSRF-токены также называют anti-CSRF-токенами, синхронизирующими токенами или токенами состояния.

CSRF-токены защищают web-приложения от атак CSRF. Такая атака может позволить злоумышленнику украсть деньги со счета пользователя, изменить его электронную почту или пароль, а также опубликовать вредоносный контент от его имени. CSRF-токены уникальны для каждой сессии и запроса. Данный факт делает невозможным его повторное использование. Также CSRF-токены помогают web-приложениям соответствовать стандартам безопасности и нормативным требованиям, таким как OWASP Top 10 и GDPR.

Генерация и хранение CSRF-токенов требует учета особенностей web-фреймворка и архитектуры приложения. Один из распространенных методов – использование криптографически безопасного генератора случайных чисел (RNG) для создания токена и хранения его в переменной сессии на стороне сервера. Однако этот метод может быть неэффективен для бесстатичных или распределенных приложений, так как требует управления сессиями.

Альтернативный вариант – использование HMAC (код аутентификации сообщений на основе хэша) для генерации токена и его хранения в cookie на стороне клиента. Этот метод масштабируемый и эффективный, но требует секретного ключа. Однако данный метод делает сайт уязвимым для атаки с перехватом и подделкой cookie.

Еще один подход – использование случайного одноразового значения, которое сохраняется в скрытом поле формы на стороне клиента. Этот метод гибкий и совместимый с разными архитектурами, но требует генерации нового токена для каждой формы. Он также может быть уязвим для межсайтового скриптинга, если злоумышленник внедрит вредоносный код в страницу. Выбор подходящего метода зависит от архитектуры приложения, его уровня безопасности и потребностей в масштабируемости.

Чтобы сделать токен недействительным, сервер должен удалять или перезаписывать его на стороне клиента или сервера, в зависимости от метода хранения. Аннулирование токена должно происходить при выходе пользователя из системы, изменении пароля или выполнении чувствительных действий. В случае использования токенов с ограниченным временем действия, сервер должен автоматически генерировать новый токен при истечении срока.

Если CSRF-токен истек или стал недействительным, система должна автоматически обновлять страницу или перенаправлять пользователя на страницу подтверждения. Также стоит предусмотреть возможность повторной отправки запроса без потери введенных данных. Следует применять токены только там, где они действительно необходимы: для POST, PUT, DELETE и других запросов, изменяющих состояние.

CSRF-токены не помогут, если сайт подвержен XSS-атакам. Злоумышленник может внедрить скрипт, который украдет CSRF-токен и отправит запрос от имени пользователя. Поэтому важно использовать CSRF-токены вместе с защитой от XSS, такой как Content Security Policy (CSP) и HTTPOnly cookie.

В некоторых случаях SameSite-флаг в cookie может быть альтернативой CSRF-токенам. Установка SameSite=Strict или SameSite=Lax позволяет браузеру автоматически блокировать кросс-сайтовые запросы, что в свою очередь позволяет повысить безопасность web-приложения и защитить его от атак CSRF, обеспечивая при этом удобство использования.