УДК 004:056.52

ВИДЫ АУТЕНТИФИКАЦИИ

О.И. РАЧИЦКИЙ
(Представлено: канд. техн. наук, доц. А.Ф. ОСЬКИН)

Рассматриваются разнообразные типы и виды аутентификации, представленные в сфере безопасности веб-приложений. Отмечается наличие большого количества других протоколов и видов авторизации. Однако в данной работе речь пойдет о самых значимых, которые оказали влияние на индустрию и стали стандартами.

В целом процесс аутентификации представляет собой ответ на вопрос, «является ли клиент тем, за кого себя выдает». Проверки подлинности личности клиента осуществляются по самым различным протоколам и самыми различными способами.

В зависимости от степени доверительных отношений, структуры, особенностей сети и удаленности объекта проверка может быть односторонней или взаимной. Также различают однофакторную и строгую (криптографическую) аутентификации. Из однофакторных систем, наиболее распространенными на данный момент являются парольные системы аутентификации. У пользователя есть идентификатор и пароль, то есть секретная информация, известная только пользователю (и, возможно, системе), которая используется для прохождения аутентификации. В зависимости от реализации системы, пароль может быть одноразовым или многоразовым. В целом можно выделить следующие типы аутентификации: базовая, дайджест-аутентификация, с использованием средств протокола HTTPS, по предъявлению цифрового сертификата, с использованием куки; а также группа децентрализованных типов аутентификации: OAuth 1.0, OAuth 2.0, OpenAuth, OpenID Connect.

При использовании *базовой аутентификации* имя пользователя и пароль включаются в состав запроса. Любой злоумышленник, перехвативший пакет, способен получить доступ к секретной информации. Из ключевых недостатков такой аутентификации следует отметить невысокий уровень безопасности – пароль можно подсмотреть, угадать, подобрать, сообщить посторонним лицам, получить посредством перехвата сообщений между клиентом и аутентификационным сервером.

Далее рассмотрим *дайджест-аутентификацию*, при которой пароль пользователя передается в хешированном виде. Казалось бы, что по уровню безопасности передачи паролей этот тип мало чем отличается от базовой аутентификации, так как злоумышленнику предоставляется возможность перехватить сообщение в любом его виде, и злоумышленник все равно получает доступ к конечной точке. Но это не совсем так – пароль превращается в некоторый хеш по определенному алгоритму и всегда с добавлением произвольной строки символов, которая генерируется для каждого соединения. Таким образом, при каждом соединении генерируется новый хэш пароля и перехват его ничего не даст.

Протокол *HTTPS* позволяет шифровать все данные, передаваемые между браузером и сервером, а не только имена пользователей и пароли. Протокол HTTPS следует использовать в любом случае для обеспечения безопасности передаваемых данных. Это поддерживается практически всеми современными облачными платформами, включая выбранную для развертывания системы Heroku.

Механизмы аутентификации с применением *цифровых сертификатов*, как правило, используют протокол с запросом и ответом. Сервер аутентификации отправляет пользователю последовательность символов, так называемый запрос. В качестве ответа выступает запрос сервера аутентификации, подписанный с помощью закрытого ключа пользователя.

Множество различных сервисов и старых сайтов используют в качестве средства аутентификации *куки*. Если куки удастся похитить, то, подделав эти данные, можно аутентифицироваться в качестве другого пользователя. В случае, когда вводимые данные плохо фильтруются или не фильтруются вовсе, похитить куки становится очень просто. Для защиты куки от похищения методом запуска вредоносных JavaScript скриптов в браузере клиента используется флаг HTTPonly, который поддерживают современные браузеры, однако это гарантирует защиту только внутри браузера, для защиты куки от перехвата во время передачи по незащищенным каналам используется флаг Secure, который шифрует куки средствами браузера. Тем не менее даже в этом случае куки все равно остаются уязвимостью с точки зрения безопасности аутентификации, так как браузер будет отправлять все куки с подходящим доменом при запросе на любой сайт из этого домена. На этой особенности построены CSRF (cross-request forgery) атаки.

Далее можно рассмотреть *децентрализованную аутентификацию*. Одним из главных минусов таких систем является то, что взлом дает доступ сразу ко многим сервисам. Однако взамен они предоставляют наивысший уровень безопасности, так как они изначально созданы и нацелены на осуществление

наибезопаснейшей аутентификации пользователей с созданием необходимых сессий и использованием самых последних наработак в области шифрования и безопасности.

OpenID – это открытая децентрализованная система аутентификации пользователей, она позволяет пользователю иметь одни аутентификационные данные для различных сервисов. Безопасность обеспечивается подписью всех сообщений. Возможной уязвимостью OpenID является подверженность атакам посредством перехвата запросов.

Многофакторная аутентификация используется для повышения безопасности посредством использования нескольких факторов аутентификации сразу. Но не всякая комбинация нескольких методов является многофакторной аутентификацией. Используются факторы различных типов: отличительные свойства (сканирование лица, отпечаток пальцев, радужная оболочка глаз); некоторое знание, которым обладает клиент (пароль, пин-код, последовательность действий); физическая вещь (карта доступа, пропуск, флеш-память, сим-карта с определенным телефонным номером).

Использование классических многоразовых паролей является серьезной уязвимостью при использовании открытых систем или незащищенных протоколов передачи данных. Это подтолкнуло разработчиков систем безопасности к созданию генераторов одноразовых паролей. Такие устройства или программы генерируют пароли для каждого соединения или сессии, чтобы при разрыве или перехвате сессии было невозможно подделать личность клиента. Для сохранения принципа двухфакторности аутентификации помимо сгенерированного устройством значения пользователь вводит постоянный пароль или подтверждает свой вход в систему вводом кода SMS-сообщения, которое на его личный номер высылает система.

Заключение

В результате проведенного исследования рассмотрены основные виды проверки подлинности клиентов на стороне приложений. Показано, что существует также много других видов аутентификации, более или менее популярных, каждый со своими плюсами и минусами. Очевидно, что правильное комбинирование систем аутентификации может значительно усложнить кражу данных клиентов.

ЛИТЕРАТУРА

1. Authentication Cheat Sheet [Электронный ресурс]. – Режим доступа: https://www.owasp.org/index.php/Authentication_Cheat_Sheet. – Дата обращения: 26.09.2017.
2. Understanding Authentication, Authorization, and Encryption [Электронный ресурс]. – Режим доступа: https://www.bu.edu/tech/about/security-resources/bestpractice/auth/. – Дата обращения: 26.09.2017.
3. Understanding and selecting authentication methods [Электронный ресурс]. – Режим доступа http://www.techrepublic.com/article/understanding-and-selecting-authentication-methods/. – Дата обращения: 26.09.2017.
4. Аутентификация и авторизация в микросервисных приложениях [Электронный ресурс]. – Режим доступа: https://habrahabr.ru/company/dataart/blog/311376/. – Дата обращения: 26.09.2017.