УДК 004.056.5

Р. А. Стахов, О. П. Сирота

ПРЕИМУЩЕСТВА ОТКРЫТОГО JSONWEBTOKEN ДЛЯ УПРОЩЕНИЯ ДОСТУПА В КЛИЕНТ-СЕРВЕРНЫХ ПРИЛОЖЕНИЯХ

Аннотация: в статье приведены основные механизмы работы открытого стандарта токенов доступа JsonWebToken (JWT) для получения доступа в веб приложениях. Показан общий принцип работы создания токенов, их проверки и аутентификации. Описаны преимущества данного стандарта доступа при аутентификации.

Ключевые слова: JWT, JSON, токен, авторизация, доступ, аутентификация, стандарт.

Введение

JSON Web Tokenявляетсястандартом RFC 7519, ответственный за создание так называемых токенов аутентификации, использующий в качестве основы формат JSON. Данный стандартприменяется при передачеаутентификационных данных[2] в вебпрограммах.

В веб программах в качестве аутентификации клиента на сервере самым широко используемым способом было использование идентификатора сетевой сесси. ограниченной по времени, который передавался в файлах cookiesили непосредственно в адресной строке. Такой метод работы именуется sliding expiration.

Однако с данным способамисвязано множество осложнений и уязвимостей. Использование для хранения данных по аутентификации сессии на сервере усложняет масштабирование сервисов. Куки же в свою очередь поддаются перехвату и подменепри условии использования незашифрованного соединения с сервером. Особую опасность представляютобщественные точки доступа Wi-Fi с отсутствием SSL. К тому же, с помощью куковдовольно легко провести процесс деанонимизации[1] и дешифрованиядействий пользователя на сайте. Они крайне уязвимы к мошенническим действиямтипа подмены, кражи при помощи «снифферов» и всевозможных уязвимостей браузеров клиентов. Популярные браузеры дают пользователям возможность обойтись без использования куков, но в таком случае некоторые сайты для них будут недоступны.

Сейчасразработчики все меньше используют идентификатор сетевой сессии, занимаясь поиском аналогичныхвариантов аутентификации. Наиболее оптимальным из них является JSON Web Token (JWT) — технология, содержащаявнутри себя всезашифрованныеданныедляаутентификации и авторизации[3]. Технология является самодостаточной, так что хранение данных пользователяизлишне.

Сейчас разработка переходит на одностраничные web-приложения (SPA) и использует обращения серверу при помощи API. То же самое API используетсяв мобильных приложениях. Мощный тренд использования микросервисов накладывает требования на масштабирование сервисов и заставляет отказываться от хранения данных, в том числе аутентификационных, на стороне сервиса непосредственно в сессии.

Для унифицировании процессааутентификации используютсятокены доступа. Стандарт JWTи является стандартом для токенов доступа.

Сперва происходит запрос доступа у серверной частичерез высылкуаутентификационных данных в виде токена.Сервер высылает токен доступа с имеющимсясрокомдействия.Полученный токен открывает доступ к ресурсам на сервере ресурсов.В случае окончания действия токена необходимо провести дополнительную процедуру аутентификации.

Состав и строение JWT довольно просты, что позволяет использовать данный стандарт во всех местах, где требуется аутентификация и авторизация, не требуя дополнительных затрат, времени и ресурсов. Данный стандарт имеет широкий уровень поддержки и сопровождения.

В состав JWT – токенавходят три части: заголовок,полезная нагрузка ицифровая подпись, включающая данныеалгоритма защиты токена. Первые двечасти представляют собойJSON-объекты[2]соответствующегостроения. Последняячасть вычисляется с использованием первых двух и основывается наизбранном алгоритмешифрования. Также токены можно перекодировать в болееемкийвид. Для этого к первым двум частям необходимо применитьшифрование Base64-URL. Далее происходит процесс добавления цифровой подписи и разделения всех трех элементов символами точки.

Ниже представлен пример JWT – токена:

```
{
  "alg": "HS512",
  "typ": "JWT"
}
{
  "sub": "12345",
  "name": "John Gold",
  "admin": true
}
```

Зашифрованное представление данных выглядит следующим образом (переводы строки добавлены для наглядности):

```
eyJhbGciOiJIUzUxMiIsInR5cCI6IkpXVCJ9.
eyJzdWIiOiIxMjM0NSIsIm5hbWUiOiJKb2huIEdvbGQiLCJhZG1pbiI6dHJ1ZX0.
```

LIHjWCBORSWMEibq-tnT8ue_deUqZx1K0XxCOXZRrBI

Взаголовкеуказана необходимая информация,описывающая сам токен.

Обязательный ключ в одном екземпляре. Таким образом, исключается двусмысленность трактования выходной информации и возможность подмены ключевых элементов для вскрытия и расшифровки защищенной информации третьими лицами. В теле JWT может быть указан один из нескольких параметров, отвечающий за поведение токена.

Параметр alg –вид алгоритма, используемого для подписи/шифрования (в случае неуказания в JWT используется значение "none").

Параметртур –указанный тип токена (type). Необходим, когда токены смешиваются с другими объектами, имеющими JOSE заголовки. Должен иметь значение "JWT".

Параметрсty –указанный тип содержимого (contenttype). Если внутри токенакроме служебных ключей присутствуют пользовательские, то данный ключ должен отсутствовать. В противном случае должен иметь значение "JWT".

В частиполезной нагрузкиуказывается информация о самом пользователе (имя пользователя и уровень его доступа), также используютсядополнительные служебные ключи. Все они являются необязательными:

Параметр iss обозначаетрегистрочувствительную строку или URI, уникальный идентификатор стороны, генерирующей токен (issuer).

Параметр sub представляет собой чувствительную к регистру строку или URI, представляющую уникальный идентификатор стороны, о которой содержится информация в данном токене (subject). Значения с этим ключом обязаны быть уникальными в контексте стороны, ответственной за генерацию JWT.

Параметр aud указывает списокк получателей данного токена. При приеме принимающей стороны получает JWT с данным ключом, он должен проверить наличие самого себя в получателях - иначе токен будет проигнорирован (audience).

Параметр exp обозначает время в формате Unix Time, определяющее момент, когда токен станет не валидным (expiration).

Параметр nbf обозначает в противоположность ключу exp, это время в формате Unix Time, определяющее момент, когда токен станет валидным (not before).

Параметр jti обозначает строку, определяющую уникальный идентификатор данного токена (JWT ID).

Access и refresh токены

Access-токен - это токен, который предоставляет доступ его владельцу к защищенным ресурсам сервера. Обычно он имеет короткий срок жизни и может нести в себе дополнительную информацию, такую как IP-адрес стороны, запрашивающей данный токен.

Refresh-токен - это токен, позволяющий клиентам запрашивать новые access-токены по истечению их времени жизни, при чем не требуя у пользователя вводить логин и пароль. Данные токены обычно выдаются на длительный срок.

Схема работы

Как правило, при использовании JSON токенов в клиент-серверных приложениях реализована следующая схема.

Клиент проходит аутентификацию в приложении (к примеру, с использованием логина и пароля). В случае успешной аутентификаци сервер отправляет клиенту access- и refresh-токены[3].При дальнейшем обращении к серверу, клиент использует access-токен. Сервер проверяет токен на валидность и предоставляет клиенту доступ к ресурсам. В случае, если access-токен становится не валидным, клиент отправляет refresh-токен, в ответ на который сервер предоставляет два обновленных токена.В случае, если refresh-токен становится не валидным, клиент опять должен пройти процесс аутентификации.

Преимущества

JWT имеет ряд преимуществ над другими способами аутентификации и авторизации, благодаря чему его применение будет весьма эффективным по сравнению с другими.

Снижается нагрузка на сервер –при использовании JWT от сервера теперь не требуется хранить дополнительных данные о сессии пользователя. Также, сервер может не заниматься созданием токенов, а предоставить это внешним сервисам.

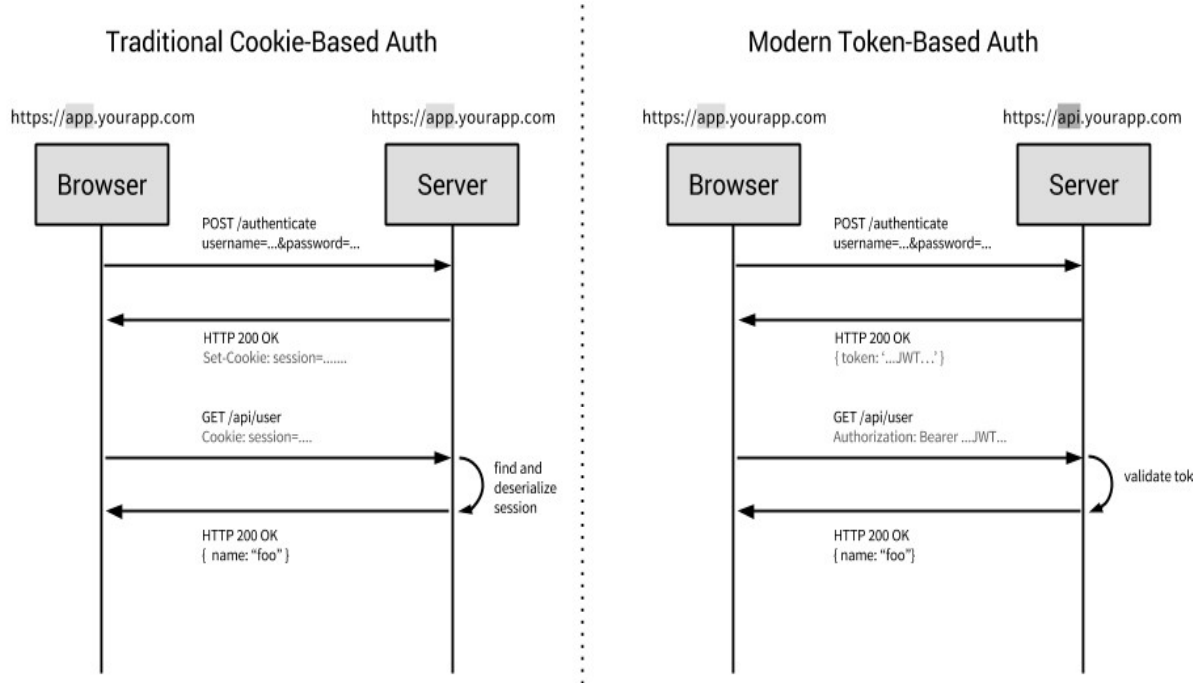

Рис. 1. – сравнение аутентификации
с помощью идентификатора сессии в Cookie и JWT

 ISSN 1560-8956

В JSON токенах можно хранить дополнительную полезную информацию о пользователях.

JWT делает возможным предоставление одновременного доступа к различным доменам и сервисам.

Ниже будет приведена сравнительная характеристика использования JWTи Cookies.

Таблица 1

Преимущества JWT в сравнении с Cookies

JWT	Идентификатор сессии в Cookies
Сервер взаимодействует с JWT-токеном при авторизации, не требуя личных данных клиента.	При входе на сервер пользователь обязан оставить свои аутентификационные данные.
Информация о сессии и клиенте не используется сервером. Сервер получает всю информацию с помощью токена.	Информация о сессии и клиенте обязана храниться на сервере.
При наличии валидного токена технология совместного использования ресурсов работает.	Не поддерживает технологию CORS (совместное использование ресурсов разными доменами).
В токене JWT возможно хранить любой тип метаданных, пока он поддерживается форматом JSON	В cookiехранится лишь индекс текущей сессии клиента
Целиком и полностью подходят для использования в «Интернете вещей»	Не подходят для «Интернета вещей», если не создан концепт работы куков для каждого устройства в сети.
Весьма короткий срок «жизни» с возможностью его регулирования, что минимизирует возможности его перехвата в долговременной перспективе.	Срок жизни весьма долгий, что вместе с предыдущими недостатками приводит к частым случаям их кражи и использования в мошеннических целях

Возможные атаки

Не будет лишним указать возможные способы атаки на стандарт JWT. Токен имеет в своем составе три части, которые могут существовать независимо друг от друга. Таким образом, появляется возможностьизъять цифровую подпись из токена и сменить заголовок, оставив токен без подписи. Если сервер не снабдили проверкой на наличие подписи у входящего токена, то мошенники имеют возможность указатьдругие значения в полезной нагрузке. Проблема решается путемотбрасывания неподписанных объектов.

CSRF (CrossSiteRequestForgery, межсайтоваяподделказапросов). Одним из наиболее часто используемых методов борьбы с CSRF является добавление специальных заголовков, содержащихзашифрованную информацию, составляющую подтверждение отправки запроса через доверенный сервер. Таким образом, если JWT используется не в качестве cookie, возможность CSRF атакиминимальна[3].

XSS (Cross-SiteScriptng). Токены могут храниться в браузере двумя способами: в DOM-хранилище или в куках. В первом случае система может быть подвержена XSS атаке, т.к. JavaScript имеет доступ к DOM-хранилищу и злоумышленник может извлечь оттуда токен для дальнейшего использования от имени пользователя. При использовании куки можно выставить HttpOnly флаг, который предотвращает доступ JavaScript к хранилищу. Таким образом, злоумышленник не сможет извлечь токен и приложение становится защищенным от XSS.

Подписанные JSON токены описываются JWS спецификацией (RFC 7515).

Поддерживаемые алгоритмы подписи

Стандарт использует современные стандарты шифрования, которые применяются во всем мире. Также возможно использование

Подпись заголовка и полезной нагрузки производится следующими алгоритмами.

Обязательный для поддержки всеми реализациями алгоритм HMAC с использованием SHA-256 (HS256).

Рекомендованные алгоритмы RSASSA PKCS1 v1.5 с использованием SHA-256 (RS256) и ECDSA с использованием P-256 and SHA-256 (ES256).

Также поддерживаются вариации рекомендованных алгоритмов с использованием SHA-384 и SHA-512 соответственно HS384, HS512,RS384, RS512,ES384, ES512.

Структура заголовка подписанного JWTтокена

Использование подписанного JWT позволяет повысить защищенность токена при его использовании. Ниже указаны дополнительные ключи.

Ключ jku:URI на набор открытых ключей в JSON-формате, используемых для подписи данного токена (JSON Web Key Set URL).

Ключ jwk: Ключ, используемый для подписи данного токена (JSON Web Key).

Ключ kid: Уникальный идентификатор используемого ключа для случая, когда указывается набор ключей (Key ID).

Ключ x5u: URI на набор сертификатов X.509. Первый сертификат в наборе должен являться тем, который использовался для подписи данного токена (X.509 URL).

Ключ x5c: Массив сертификатов X.509 в формате JSON, использованных для подписи данного токена (X.509 certifcate chain).

Ключ x5t: Цифровой отпечаток SHA1 сертификата X.509 (X.509 certi cate SHA-1 fingerprint).

Ключ crit: Массив строк с названиями ключей данного заголовка, которые должны обрабатываться парсером JWT. Если должны быть обработаны все ключи, то не используется (critical).

Реализации

Реализации JWT существуют в следующих языках программирования и фреймворках: Clojure, .NET, Go, Haskell, Python, Java, JavaScript, Lua, Perl, PHP, Ruby, Rust, Scala, Erlang, Common Lisp и Elixir.

Выводы

В статье приведены основные механизмы и принципы работы открытого стандарта создания токенов доступа JsnoWebToken (JWT) для передачи доступа в клиент-серверных приложениях.Показан общий принцип работы формирования токенов доступа, их проверки и аутентификации. Описаны составляющие и компоненты данного стандарта.Указаны преимущества стандарта JWT, его составных частей, а также гибкость применения в задачах аутентификации.

Список использованных источников

1. Johnson M: Newadwancedpersonaldataprotection/ WileyInformationTechnologies, 2016.
2. TroelsenI:JWTviewviaC# libraries/ Apress, 2016
3. AlbahariJ:JWT – a brand new protection system/ Bookjoy, 2015