Аникин Д.А. Анализ методов авторизации и аутентификации REST API // Международный журнал информационных технологий и энергоэффективности. – 2023. – Т. 8 № 5(31) ч.2. с. 120–124

Международный журнал информационных технологий и энергоэффективности

Сайт журнала:
http://www.openaccessscience.ru/index.php/ijcse/

УДК 004.422

АНАЛИЗ МЕТОДОВ АВТОРИЗАЦИИ И АУТЕНТИФИКАЦИИ REST API

Аникин Д.А.
ФГБУО ВО «МИРЭА - Российский технологический университет», Москва, Россия (119454, г. Москва, пр. Вернадского, 78), e-mail: danil-anikin-98@mail.ru

Настоящая статья посвящена рассмотрению и анализу различных методов аутентификации и авторизации, которые могут быть использованы при проектировании приложения, основанного на REST API. Современные информационные системы в большинстве случаев основаны на технологии обработки пользовательских запросов, для реализации контроля доступа используются различные технологии. В результате их анализа была дана подробная характеристика каждой из них, выявлены сильные и слабые стороны и выработаны рекомендации по выбору технологии.

Ключевые слова: Информационные технологии, REST API, программирование, авторизация, аутентификация.
.

ANALYSIS OF REST API AUTHORIZATION AND AUTHENTICATION METHODS REST API

Anikin D.A.
MIREA - Russian Technological University, Moscow, Russia (119454, Moscow, Vernadskogo Ave., 78), e-mail: danil-anikin-98@mail.ru

This article is devoted to the consideration and comparative analysis of various authentication and authorization methods that can be used when designing an application based on the REST API. Modern information systems in the vast majority of cases are based on the technology of processing user requests, various technologies are used to implement access control. Because of their analysis, a detailed description of each of them was given, strengths and weaknesses were identified and recommendations on the choice of technology were developed.

Keywords: Information technology, REST API, programming, authorization, authentication.

По мере развития информационных технологий приложения, разрабатываемые программистами, стремительно усложнялись. К новым более совершенным системам не были применимы те же подходы, методики разработки и организации кода, которые были актуальны ранее. Для решения приходящих проблем были разработаны различные решения.

В ходе разработки приложений, к которым должны были иметь доступ множество пользователей, возникли следующие проблемы [1]:

- Необходимость разделения клиентской и серверной частей приложения, так как серверу приходится обрабатывать запросы не только одного пользователя на локальной машине.

- Необходимость добиться кроссплатформенного доступа к сервису, чтобы пользователи не зависели от операционной системы, установленной на их клиентской машине.
- Необходимость создания универсального стиля взаимодействия веб-приложений, использующих общепризнанные сетевые протоколы. Обращение к сервисам не должно было зависеть от языка программирования и средства реализации серверной части приложения.
- Необходимость добиться более простой масштабируемости для всё более разрастающихся и развивающихся систем.

С целью решения вышеперечисленных проблем была разработана технология REST API. Representational State Transfer (REST) Application Programming Interface (API) – это не единственный строго определённый протокол взаимодействия компонентов программы, это архитектурный стиль разработки. Он описывает процесс проектирования интерфейса разработчиком для взаимодействия своего приложения со внешними элементами. Принципы и ограничения данного архитектурного стиля были определены Роем Филдингом в 2000 году, он является одним из создателей протокола HTTP, если интерфейс полностью соответствует принципам, то он называется RESTful [2].

Принцип работы REST API заключается в следующем: клиент отправляет запрос на сервер, сервер обрабатывает клиентский запрос, сохраняет логи обработки, подготавливает ответ и возвращает его клиенту. Однако далеко не все клиенты должны иметь доступ ко всем методам, предоставляемым сервером, так как в таком случае создается угроза безопасности и защиты данных, объектов или материалов, которые предоставляет API. Для решения этой проблемы была разработана концепция авторизации.

Авторизация – это предоставление конкретному лицу или группе лиц прав на выполнение определённых действий, а также процесс проверки, подтверждения данных прав при попытке выполнения этих действий, зачастую этот процесс подразумевает проверку подлинности логина и пароля пользователя или токена доступа [3].

Помимо авторизации, процесс обеспечения защиты подразумевает идентификацию и авторизацию пользователей. Идентификация — процедура, в результате выполнения которой для субъекта идентификации выявляется его идентификатор, однозначно определяющий этого субъекта в информационной системе. Аутентификация — процедура проверки подлинности.

Недостаточно точно организованный процесс разделения доступа к API может привести к таким последствиям, как: неограниченное количество запросов пользователей к определённым методам, что приведет к замедлению их обработки на серверной части приложения, отсутствие связки конкретного пользователя с историей его запросов, защита целостности данных информационной системы от злоумышленников, невозможность отследить, какие точки доступа используются чаще всего без использования сторонних средств [4].

Проведём анализ наиболее популярных методов авторизации REST API.

Basic Authentication: является одним из самых простых методов аутентификации, который применяется в веб-приложениях. В его основе лежит процесс передачи логина и пароля, для которых используется закодированное значение в заголовке запроса, обычно кодирование производится с помощью алгоритма Base64. При получении запроса сервером производится дешифровка сообщения и проверка заголовка, который содержит логин и

пароль, в зависимости от результата проверки будет принято решение принять или отклонить запрос [5].

Token-based Authentification (Аутентификация на основе токенов) основана на том, что токен используется для авторизации пользователя. Токеном является специальный код, который генерируется для каждого отдельного пользователя и который предоставляет различные уровни доступа. Этот токен может содержать как и информацию о пользователе, так и любые другие сведения, важные для приложения [6].

OAUTH 2.0: позволяет пользователям предоставлять доступ к своим данным другим приложениям без необходимости предоставления собственных данных для входа. В основе своей он имеет отдельный сервер аутентификации для связи с сервером API. Ключевой чертой, по которой можно выявить использование данного метода – наличие возможности авторизации с помощью учётной записи сторонних сервисов.

Данный протокол работает по следующему принципу: пользовательское приложение отправляет ключ приложения и секретные данные на страницу входа в систему на сервере аутентификации, при успешном прохождении сервер аутентификации присваивает и возвращает пользователю токен для доступа к основному сервису. Запрос с полученным токеном перенаправляется на сервер с необходимыми ресурсами. Токен доступа добавляется в заголовок в качестве параметра Bearer, сам сервер проверяет токен доступа и принимает решение об обработке запроса [7]. Диаграмма последовательности для работы протокола OAuth 2.0 представлена на Рисунке 1.

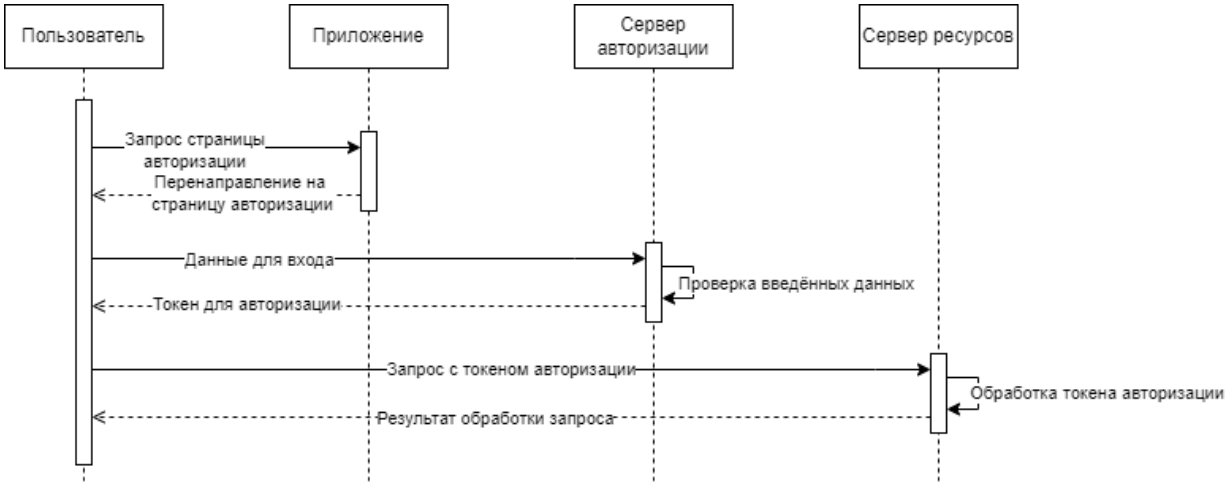

Рисунок 1 – Диаграмма последовательности авторизации OAuth 2.0

Bearer token (Токен-маркер): представляет собой маркер доступа, который используется для аутентификации, хранится он, в свою очередь, на стороне клиента и передаётся в качестве одного из параметров заголовка запроса.

API KEY: заключается в передачи ключевой пары, обычно состоящей из секретного и открытого ключа. Открытый ключ входит в состав запроса, закрытый же используется только при обмене данными между серверами [8].

AWS Signature: метод аутентификации, который используется для обеспечения безопасности доступа к API Amazon Web Services. Signature HMAC-SHA1 или Signature Version 4 — это две версии, которые используются при аутентификации AWS. Первая версия (HMAC-SHA1) используется для старых версий API Amazon [9]. Вторая версия (Version 4) для

более новых версий API. AWS Signature основывается на передаче ключевой пары, которая используется для создания контрольной суммы, обеспечивающей авторизацию пользователя.

Исходя из проведённого анализа, можно определить, что выбор метода аутентификации API зависит от планируемого использования и выдвинутых требований приложения, описанных в техническом задании. Рекомендуется выбрать метод, который наиболее точно отвечает на потребности в безопасности и простоте доступа к сервису.

Таким образом, обеспечение авторизации и аутентификации REST API является крайне важным этапом, позволяющим обеспечить безопасность передаваемых данных, а также управлять доступом к ресурсам и методам серверной части приложения. Были рассмотрены такие методы авторизации, как: Basic Authentication, Token-based Authentification, OAUTH 2.0, JSON Web Tokens, Bearer token, API KEY, AWS Signature, была дана развёрнутая характеристика каждого из них и были рассмотрены основные принципы работы. Из проведённого анализа было установлено, что выбор метода авторизации целиком и полностью основывается на том, какие требования были выдвинуты к разрабатываемому приложению по части безопасности и контроля уровней доступа, нет какого-то одного ультимативного выбора для всех ситуаций, существуют различные технологии, которые применимы в различных ситуациях.

Список литературы

1. Brenda, Jin Designing Web APIs / Jin Brenda, Sahni,&,Amir Saurabh. — First edition. — Sebastopol : O’Reilly Media, 2018. — 232 с. — Текст : непосредственный.
2. Mark, Masse REST API Design Rulebook / Masse Mark. — First edition. — Sebastopol : O’Reilly Media, 2012. — 93 с. — Текст : непосредственный.
3. Neil, Madden API Security in Action / Madden Neil. — First edition. — : y Manning Publications Co, 2020. — 43 с. — Текст : непосредственный.
4. Andrew, Hoffman Web Application Security / Hoffman Andrew. — First edition. — : O'Reilly Media, 2020. — 331 с. — Текст : непосредственный.
5. HTTP authentification. — Текст : электронный // mdn web docs : [сайт]. — URL: https://developer.mozilla.org/en-US/docs/Web/HTTP/Authentication (дата обращения: 12.05.2023).
6. Token-based Authentication. — Текст : электронный // jetbrains : [сайт]. — URL: https://www.jetbrains.com/help/youtrack/server/2fa-with-token.html (дата обращения: 12.05.2023).
7. OAuth 2.0. — Текст : электронный // oauth : [сайт]. — URL: https://oauth.net/2/ (дата обращения: 12.05.2023).
8. OAuth 2.0. — Текст : электронный // OpenAPI guide : [сайт]. — URL: https://swagger.io/docs/specification/about/ (дата обращения: 12.05.2023).
9. Signing AWS API requests. — Текст : электронный // aws : [сайт]. — URL: https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_aws-signing.html (дата обращения: 12.05.2023).

References

1. Brenda, Jin Designing Web APIs / Jin Brenda, Sahni,&,Amir Saurabh. — First edition. — Sebastopol : O'Reilly Media, 2018. — p. 232 — Text: immediate.
2. Mark, Masse REST API Design Rulebook / Masse Mark. — First edition. — Sebastopol : O'Reilly Media, 2012. — p. 93 — Text: immediate.
3. Neil, Madden API Security in Action / Madden Neil. — First edition. — y Manning Publications Co, 2020. — p. 43 — Text: immediate.
4. Andrew, Hoffman Web Application Security / Hoffman Andrew. — First edition. — O'Reilly Media, 2020. — p. 331 — Text: immediate.
5. HTTP authentification. — Text: electronic // mdn web docs: [site]. Available at: https://developer.mozilla.org/en-US/docs/Web/HTTP/Authentication (accessed: 12.05.2023).
6. Token-based Authentication. — Text: electronic // jetbrains: [site]. Available at: https://www.jetbrains.com/help/youtrack/server/2fa-with-token.html (accessed: 12.05.2023).
7. OAuth 2.0. — Text: electronic // oauth: [site]. Available at: https://oauth.net/2/ (accessed: 12.05.2023).
8. OAuth 2.0. — Text: electronic // OpenAPI guide: [site]. Available at: https://swagger.io/docs/specification/about/ (accessed: 12.05.2023).
10. Signing AWS API requests. — Текст : электронный // aws : [сайт]. — URL: https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_aws-signing.html (дата обращения: 12.05.2023).