«МИЦЕЛИЙ»

Описание программного продукта

СОДЕРЖАНИЕ

1 Глоссарий ... 3
2 Назначение ... 4
3 Функциональные возможности ... 5
Сервис Grants ... 5
Сервис SecurityProxy ... 5
4 Архитектура ... 6
Используемый стек технологий ... 6
5 Функциональные требования ... 8
6 Нефункциональные требования ... 9
Требования к надежности ... 9
Время отклика ... 9
Требование к нагрузке ... 9
Масштабируемость ... 9
Безопасность ... 9
7 Эксплуатационные характеристики ... 10
Требования к программному обеспечению ... 10
Требования к аппаратному обеспечению ... 10
Обеспечение отказоустойчивости и высокой доступности ... 11

1 Глоссарий

Термин/сокращение	Определение
Контракт сервиса	Формальная, точная и верифицируемая спецификация публичных API сервиса
Онтология	Подробная и всеобъемлющая формализация некоторой области знаний с помощью концептуальной схемы
ABAC	Attribute-based access control, авторизация на основе атрибутов
Consumer-Driven Contracts	Контракты, диктуемые потребителем. Концепция при которой сервис описывает свой контракт максимально широко и универсально для всех клиентов, а каждый конкретный потребитель конкретизирует, по какому контракту конкретно он будет работать с данным сервисом
GraphQL API	Язык запросов и обработки данных с открытым исходным кодом для API; также предоставляет механизм выполнения запросов, при котором клиенты определяют структуру данных, которые должны быть возвращены сервером
JWT	JSON Web Token, это JSON объект, который определен в открытом стандарте RFC 7519. Он считается одним из безопасных способов передачи информации между двумя участниками
RBAC	Role-based access control, авторизация на основе ролей

2 Назначение

«Мицелий» – программный продукт для микросервисных платформ, предназначенный для разграничения и контроля доступа к данным, предоставляемым API протоколов GraphQL или REST.

3 Функциональные возможности

«Мицелий» состоит из сервисов:

- Grants: выполняет функции управления моделью контроля доступа на основе RBAC/ABAC подходов;
- SecurityProxy: осуществляет контроль доступа к данным, предоставляемым GraphQL API сервисов-источников данных.
- AdminConsole: консоль администрирования БД Grants

«Мицелий» позволяет настраивать права доступа:

- Для GraphQL к отдельным полям схемы
- Для REST к каждому ресурсу (endpoint)

Права доступа, роли, пользователи настраиваются с помощью консоли администрирования.

Примечание: *Сервис SecurityProxy разворачивается перед каждым бизнес сервисом, данные которого необходимо защитить, и проксирует входящие вызовы, осуществляя контроль доступа к данным.*

Политики доступа к данным указываются опционально либо разработчиками в метаданных GQL\REST-контрактов сервисов-источников либо в консоле администрирования (динамические права).

Сервис Grants

Сервис обеспечивает следующую функциональность:

- предоставление списка ролей пользователя, прошедшего аутентификацию;
- предоставление открытых данных сертификатов для проверки подписи JWT-токена;
- формирование JWT-токена для авторизации в сервисе источнике данных для указанной роли пользователя, прошедшего аутентификацию;
- хранение данных пользователя, необходимых для выпуска JWT-токена;
- хранение ролей, доступных пользователю для авторизации;
- хранение настроек контроля доступа для сервисов-источников.
- Хранение и авториазцация по API KEY – доступ для сервисов.

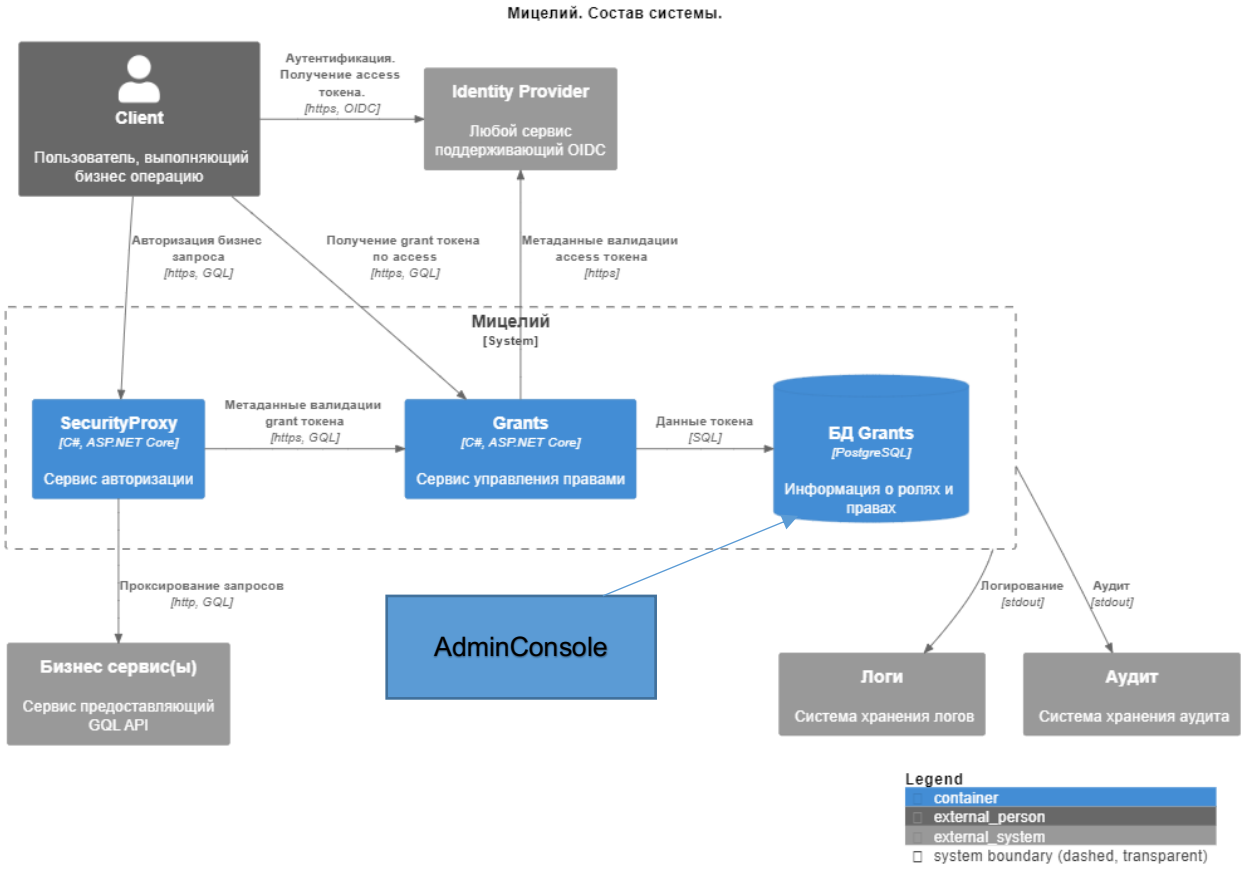
Сервис SecurityProxy

Сервис обеспечивает следующие функциональные возможности:

- контроль доступа к данным, предоставляемым GraphQL\REST API сервисом-источником данных, на основе правил авторизации;
- проверка корректности разметки GraphQL API сервисов-источника данных;
- предоставление контрактов защищаемого сервиса внешним потребителям без ограничения доступа к контрактам для неавторизованных запросов;
- проксирование запросов, прошедших авторизацию к сервису-источнику данных.

4 Архитектура

На схеме представлена функциональная архитектура продукта.



Используемый стек технологий

При разработке и эксплуатации продукта используются языки программирования, компоненты с открытым исходным кодом, стандарты:

Название	Описание	Лицензия	Источник
C# .NET	Модульная платформа для разработки программного обеспечения с открытым исходным кодом	MIT License	https://learn.microsoft.com/ru-ru/dotnet/csharp/
Semver	Методология семантического версионирования		https://semver.org/lang/ru/
PostgreSQL	Свободная объектно-реляционная система управления базами данных (СУБД).	PostgreSQL License	www.postgresql.org
GraphQL API	Язык запросов и обработки данных с открытым исходным кодом для API. Предоставляет	MIT License	https://graphql.org/

	также механизм выполнения запросов, при котором клиенты определяют структуру данных, которые должны быть возвращены сервером		
REST	Архитектурный стиль взаимодействия компонентов распределённого приложения в сети		https://ru.wikipedia.org/wiki/REST
OpenAPI	Описание схемы REST		https://www.openapis.org/
Docker	Открытая платформа для разработки, доставки и запуска приложений	Apache License 2.0	https://www.docker.com/

5 Функциональные требования

Технология авторизации в «Мицелий» использует следующие принципы:

- дискретное управление доступом к данным модели, т.е. возможность предоставить различный уровень доступа к отдельным атрибутам модели или всей модели целиком;
- комбинирование управление доступом на основе ролей, контекста и атрибутов;
- децентрализованная авторизация по принципу App-to-App Single sign-on, поддерживаемая основными системами аутентификации, представленными на рынке;
- обеспечение омниканальности – возможность авторизоваться внутренним и внешним сотрудникам и системам.

Также продукт должен отвечать следующим функциональным условиям:

- должен быть реализован механизм контроля доступа к данным, включая данные, составляющие служебную и банковскую тайну в омниканальной среде с большим количеством контекстов исполнения кода;
- правила авторизации запросов к защищаемым данным ПО «Мицелий» должен считывать из метаданных контрактов сервисов (правила авторизации определяют уровень доступа к Модели, Методу, Атрибуту любого уровня иерархической вложенности, Фильтру, Контексту исполнения, Аргументу вызова);
- продукт должен валидировать схему защищаемого сервиса на корректность с точки зрения целостности модели и дополнительно валидировать метаданные схемы с точки зрения полноты описания и защищённости API, закрываемого авторизацией сервиса. Основное правило – все API должны быть покрыты правилами авторизации явно. Любой пропуск в правилах, должен приводить к соответствующей ошибке при попытке обратиться к данному API. Опционально все подобные ошибки логируются при запуске ПО «Мицелий» в момент валидации контрактов, но блокируют дальнейшую работу ПО «Мицелий»;
- продукт должен поставлять контракты защищаемого сервиса внешним потребителям, не ограничивая доступ к контрактам для неавторизованных запросов;
- для системы должна быть реализована гибкая настройка источника контрактов, в качестве которого может выступать как само защищаемое API, так и сторонний источник, поставляющий только контракты;
- продукт должен обеспечивать хранение настроек разграничения доступа для пользователей системы в базе данных;
- продукт должен обеспечивать журналирование и аудит работы комплекса.

6 Нефункциональные требования

Требования к надежности

Должна быть обеспечена доступность компонентов «Мицелий» на уровне 99.9%.

Время отклика

Продукт должен обрабатывать GraphQL\REST запрос не более чем за 300ms на 95 перцентиле.

Требование к нагрузке

«Мицелий» SecurityProxy должен обрабатывать 1000 запросов в минуту, обеспечивая выполнения условия по Времени отклика.

Сервис Grants должен обрабатывать 300 запросов в минуту, обеспечивая выполнения условия по времени отклика.

Масштабируемость

Продукт должен обеспечивать возможность горизонтального масштабирования.

Безопасность

Продукт должен аутентифицировать каждый запрос от пользователя или внешней системы.

Продукт должен записывать аудит по каждой операции.

7 Эксплуатационные характеристики

Пример:

Требования к программному обеспечению

ПРИМЕР: Требования к программному обеспечению:

- Операционная система: Astra Linux 2.12 или Ubuntu 20.04+ LTS или Red Hat Linux 7/8 + или CentOS 8.1+
- Система Контейнеризации: kubernetes или OKD 4 или Red Hat Open Shift 4 или компонент Service Mesh - ISTIO, текущая версия 1.4.6.

Требования к аппаратному обеспечению

Канальные приложения

Название	CPU	RAM, GB	Storage (HDD, SSD)
Сервер интерфейсов	128	512	880
Сервер БД PostgreSQL кластерный	128	512	3400
Сервер MQ	128	256	1040
Перекладчик в S-terra	64	256	1040
Сервер хранения и обработки системы журналирования	32	256	10000

Сервисы устойчивых бизнес операций

Название	CPU	RAM, GB	Storage (HDD, SSD)
Сервер баз данных	3528	10584	358092
Кворумный сервер	1890	5418	31878
Nginx	1890	5418	31878
WEB	1890	5418	31878
Кеш микросервисов	1890	5418	31878
Redis	1890	5418	31878

Общие продуктовые сервисы

Название	CPU	RAM, GB	Storage (HDD, SSD)
PostgreSQL	36	1536	6000
etcd	6	12	135

Обеспечение отказоустойчивости и высокой доступности

Для обеспечения высокой доступности и отказоустойчивости, система развертывается с использованием ресурсов нескольких ЦОД по схеме Active-Active.

Используется схема, позволяющая продолжить операции ввода-вывода после потери одного из ЦОД. Для реализации требования отказоустойчивости используется технология растянутого кластера, с механизмом виртуализации. Механизм виртуализации позволяет управлять запуском виртуальных машин и перезапускать их на доступных физических машинах, в случае сбоя основных.