Микросервис Catalog.

Руководство пользователя.

Оглавление

[Аннотация 5](#_Toc56754768)

[1. Сервис отправки sms, push сообщений, сообщений в viber, одноразовых паролей. 6](#_Toc56754769)

[1.1 Общие сведения 6](#_Toc56754770)

[1.1.1 Создание проекта Firebase с поддержкой Firebase Cloud Messaging 6](#_Toc56754771)

[1.1.2 Авторизация 7](#_Toc56754772)

[1.2 Информационная архитектура 7](#_Toc56754773)

[1.3 Компонентная архитектура 9](#_Toc56754774)

[1.4 Асинхронная отправка сообщений через RabbitMQ 11](#_Toc56754775)

[1.5 Мониторинг статусов сообщений через RabbitMQ 12](#_Toc56754776)

[1.6 Описание API 14](#_Toc56754777)

[1.6.1 Описание API микросервиса "Отправка сообщений" 14](#_Toc56754778)

[1.6.2 Описание API микросервиса "Прием статусов PUSH-сообщений" 15](#_Toc56754779)

[2 Сервис добавления штампа в pdf 16](#_Toc56754780)

[2.1 Общие сведения 16](#_Toc56754781)

[2.2 Безопасность 16](#_Toc56754782)

[2.3 Авторизация 16](#_Toc56754783)

[2.4 Описание API 16](#_Toc56754784)

[3 Сервис коротких ссылок (Shorter) 17](#_Toc56754785)

[3.1 Описание сервиса 17](#_Toc56754786)

[3.2 Авторизация 17](#_Toc56754787)

[3.3 Информационная архитектура 17](#_Toc56754788)

[3.4 Компонентная архитектура 18](#_Toc56754789)

[3.5 Описание API сервиса коротких ссылок 19](#_Toc56754790)

[4 Сервис нотификации telephony-publisher 20](#_Toc56754791)

[4.1 Общие сведения 20](#_Toc56754792)

[4.2 Схема информационных потоков 20](#_Toc56754793)

[4.3 Процесс подключения нового потребителя 20](#_Toc56754794)

[4.4 Процесс получения данных о звонках от CUCM 21](#_Toc56754795)

[4.5 Формат сообщений 21](#_Toc56754796)

[4.6 Дополнительные настройки 22](#_Toc56754797)

[4.7 Бизнес архитектура 23](#_Toc56754798)

[4.8 Информационная архитектура 23](#_Toc56754799)

[4.9 Информационная безопасность 25](#_Toc56754800)

[4.10 Обработка звонка, поступившего от клиента 25](#_Toc56754801)

[4.11 Сбор и фильтрация событий телефонии на уровне telephony-publisher 26](#_Toc56754802)

[5 Сервис печати 28](#_Toc56754803)

[5.1 Описание сервиса 28](#_Toc56754804)

[5.2 Схема взаимодействия сервиса 28](#_Toc56754805)

[5.3 Архитектурное решение 28](#_Toc56754806)

[5.4 Описание API 29](#_Toc56754807)

[5.5 Примеры заполнения шаблонов и вызова на печать документов. 31](#_Toc56754808)

[5.5.1 Опитсание плейфсхолдеров 31](#_Toc56754809)

[5.5.2 Плейсхолдеры в формате даты 31](#_Toc56754810)

[5.5.3 Условия 32](#_Toc56754811)

[5.5.4 Массив 32](#_Toc56754812)

[5.5.5 Склонения по падежам 32](#_Toc56754813)

[5.5.6 Суммы прописью и склонение валютных единиц (целой и дробной частей) 32](#_Toc56754814)

[5.5.7 Числа прописью 32](#_Toc56754815)

[5.5.8 Валюта прописью и знаком 33](#_Toc56754816)

[5.5.9 Передача картинок 33](#_Toc56754817)

[5.6 Дополнительные функции при печати документа 33](#_Toc56754818)

[5.7 Безопасность 33](#_Toc56754819)

[6 Сервис получения справочника БИКов 34](#_Toc56754820)

[6.1 Общие сведения 34](#_Toc56754821)

[6.2 Маппинг значений справочника БИК 34](#_Toc56754822)

[6.3 Применение интеграционной шины в интеграции сервиса 34](#_Toc56754823)

[6.3.1 Ограничения передаваемых данных через интеграционную шину 35](#_Toc56754824)

[6.4 Rest запросы 35](#_Toc56754825)

[6.5 Маппинг данных 35](#_Toc56754826)

[6.5.1 «Справочник БИК – Справочник действующих банков» 35](#_Toc56754827)

[6.5.2 «Справочник БИК – Справочник ликвидированных банков» 37](#_Toc56754828)

[Таблица 7.2. Маппинг данных «Справочник БИК – Справочник ликвидированных банков» . 37](#_Toc56754829)

[6.6 Структура базы данных справочника БИК 39](#_Toc56754830)

[6.7 Описание API сервиса хранения БИК-ов 39](#_Toc56754831)

[7 Сервис управления IP-телефонией 40](#_Toc56754832)

[7.1 Общие сведения 40](#_Toc56754833)

[7.2 Безопасность 40](#_Toc56754834)

[7.3 Ограничения 40](#_Toc56754835)

[7.4 Авторизация 40](#_Toc56754836)

[7.5 Интеграционная схема 40](#_Toc56754837)

[7.6 Аутентификация пользователей 42](#_Toc56754838)

[7.7 Формирование звонка 42](#_Toc56754839)

[7.8 Операции 42](#_Toc56754840)

[7.9 Описание API сервиса 44](#_Toc56754841)

[8 Сервис хранения файлов 45](#_Toc56754842)

[8.1 Общие сведения 45](#_Toc56754843)

[8.2 Авторизация 45](#_Toc56754844)

[8.3 Информационная архитектура 45](#_Toc56754845)

[8.4 Компонентная архитектура 46](#_Toc56754846)

[8.5 Описание API 47](#_Toc56754847)

[9 Сервис ЭЦП 48](#_Toc56754848)

[9.1 Общие сведения 48](#_Toc56754849)

[9.2 Ограничения 48](#_Toc56754850)

[9.3 Безопасность 48](#_Toc56754851)

[9.4 Получение хэш-а последней невидимой подписи 49](#_Toc56754852)

[9.5 Проверка подлинности документа 49](#_Toc56754853)

[9.6 Интеграционная схема сервиса ЭЦП 50](#_Toc56754854)

[9.7 Описание API сервиса подписания файлов 51](#_Toc56754855)

# Аннотация

Система «Микросервис Catalog» предназначена для решения типовых задач и представлена в виде функциональных модулей. Каждый функциональный модуль отвечает за решение и выполнение определенного рода задач.

# Сервис отправки sms, push сообщений, сообщений в viber, одноразовых паролей.

## Общие сведения

Сервис разработан для отправки сообщений по СМС, PUSH и viber, а так же отправки и проверки ОТP кодов по СМС или PUSH.

СМС и Viber сообщения отправляются через интеграционный шлюз MFMS.

PUSH сообщения отправляются через Firebase Cloud Messaging.

### Создание проекта Firebase с поддержкой Firebase Cloud Messaging

Firebase Cloud Messaging (FCM)- это кроссплатформенное решение для обмена облачными сообщениями и уведомлениями от Google, позволяющее разработчикам отправлять push-уведомления своим конечным пользователям приложений через Firebase Notification Composer или через набор API, предоставляемый Firebase.

Для настройки проекта необходимо в браузере открыть рабочее пространство по ссылке <https://console.firebase.google.com/> и создать новый проект.

Для генерации push-токенов в мобильное риложение необходимо встроить **google-services.json (Ошибка! Источник ссылки не найден.) .**

Для отправки PUSH сообщений необходимо разместить в «Менеджере Секретов», например, Vault - json, содержащий закрытый ключ проекта (Рисунок 1.2). Название в Vault должно совпадать с название проекта в запросе (атрибут **appName**).

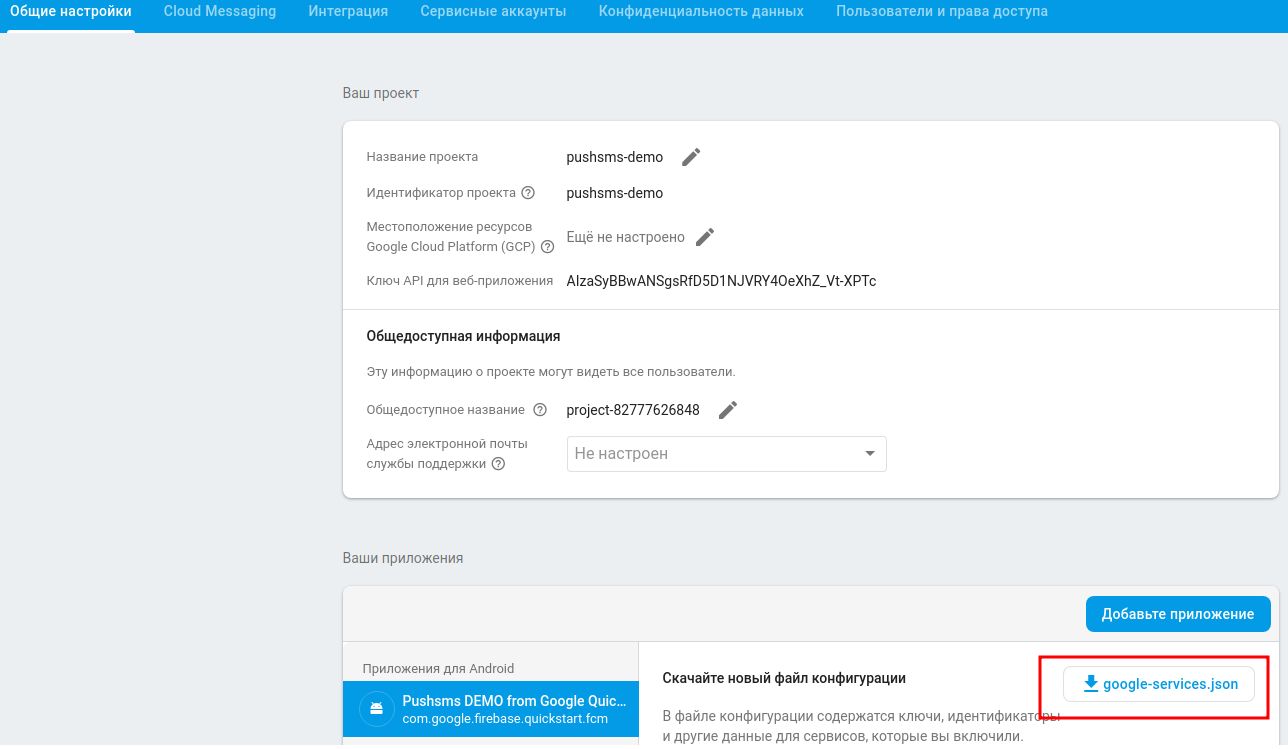


Рисунок ..

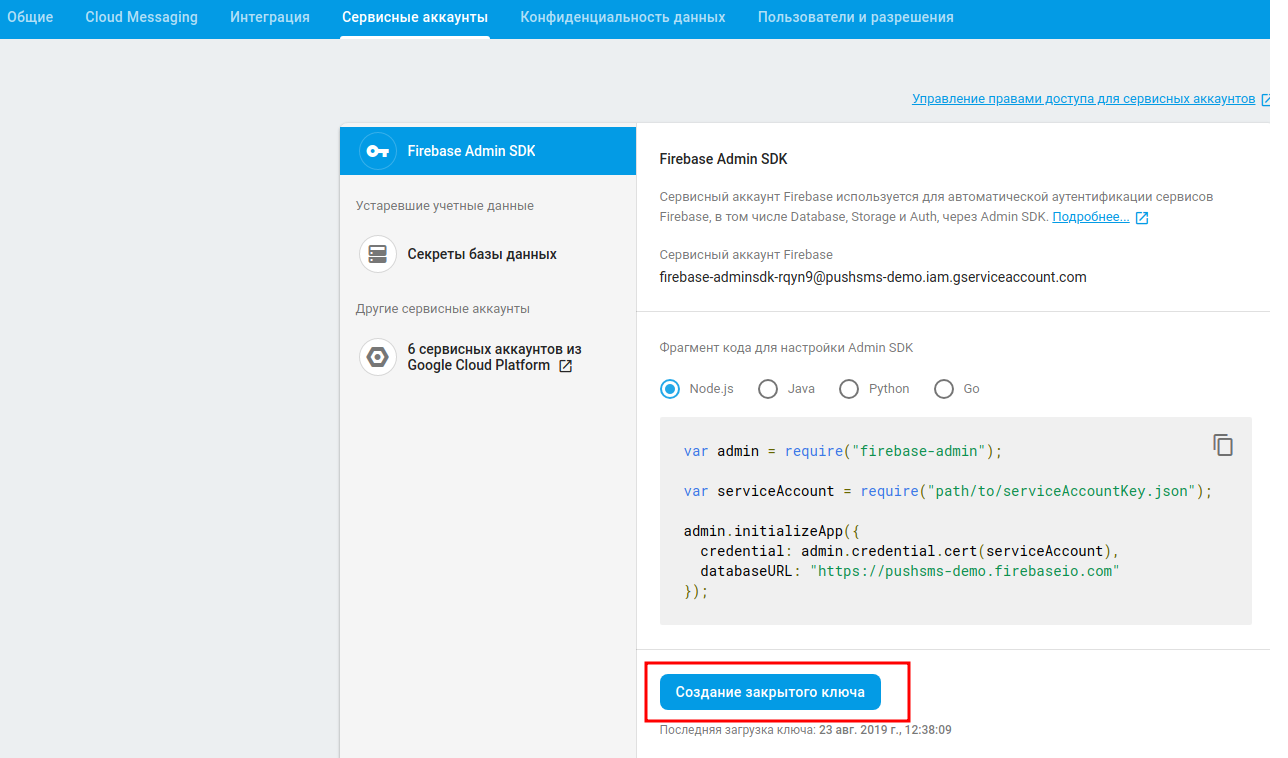


Рисунок .

### Авторизация

Запросы защищены аутентификацией. Аутентификация осуществляется по стандарту Oauth2 c JWT. Токен передается в заголовке Autorization.

## Информационная архитектура

На Рисунок 1.3 представлена схема интеграции сервиса отправки sms, push сообщений, сообщений в viber, одноразовых паролей с приложениями. В Таблица 1.1 перечислены запросы, осуществляемые к микросервисам, и дано их описание.

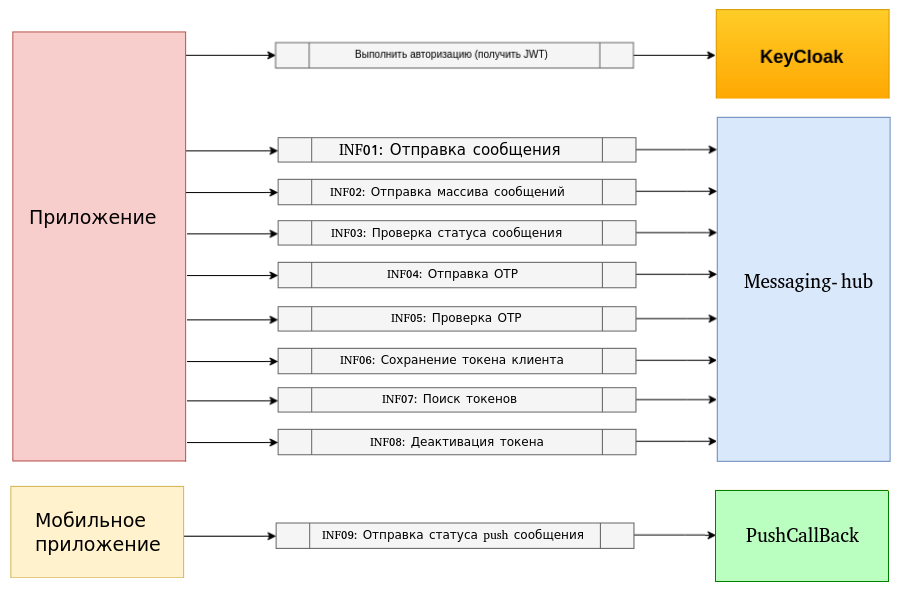


Рисунок .. Информационная архитектура сервиса.

Таблица 1.1 Описание запросов.

|  | **Код** | **Название** | **Отправитель** | **Получатель** | **Описание** |
| --- | --- | --- | --- | --- | --- |
| 1 | INF01 | Отправка сообщения | Приложение | messaging-hub | Post запрос  для отправки push, sms, push/sms, viber сообщения. |
| 2 | INF02 | Отправка массива сообщений | Приложение | messaging-hub | Post запрос  для отправки массива сообщений. |
| 3 | INF03 | Проверка статуса сообщения | Приложение | messaging-hub | Post запрос  для проверки статуса сообщения. |
| 4 | INF04 | Отправка ОТР | Приложение | messaging-hub | Post запрос для генерации и отправки ОТР кода по sms или push. |
| 5 | INF05 | Проверка ОТР | Приложение | messaging-hub | Post запрос для проверки отправленного OTP кода |
| 6 | INF06 | Сохранение токена клиента | Приложение | messaging-hub | Post запрос для сохранения сопоставления push-токена клиента и id клиента из EVA. Используется для дальнейшей отправки по clientId , когда телефон по id клиента определяется из EVA, а токен из сохраненного в системе pushsms |
| 7 | INF07 | Поиск токенов | Приложение | messaging-hub | Get запрос для поиска push-токенов Клиента в системе pushsms. |
| 8 | INF08 | Деактивация токена | Приложение | messaging-hub | Post запрос для изменения статуса push-токена в системе pushsms. |
| 9 | INF09 | Отправка статуса push сообщения | Мобильное приложение | PushCallBack | Post запрос для отправки статуса push сообщения с мобильного устройства клиента. Поскольку firebase не позволяет отслеживать статусы отправленных сообщений мобильные приложения должны отправлять статус о получении и прочтении сообщения клиентом (DELIVERED, READ, READ\_ERROR). |

## Компонентная архитектура

Компонентная архитектура сервиса представлена на Рисунок 1.4. Описание компонентов сервиса приведено в Таблица 1.2.

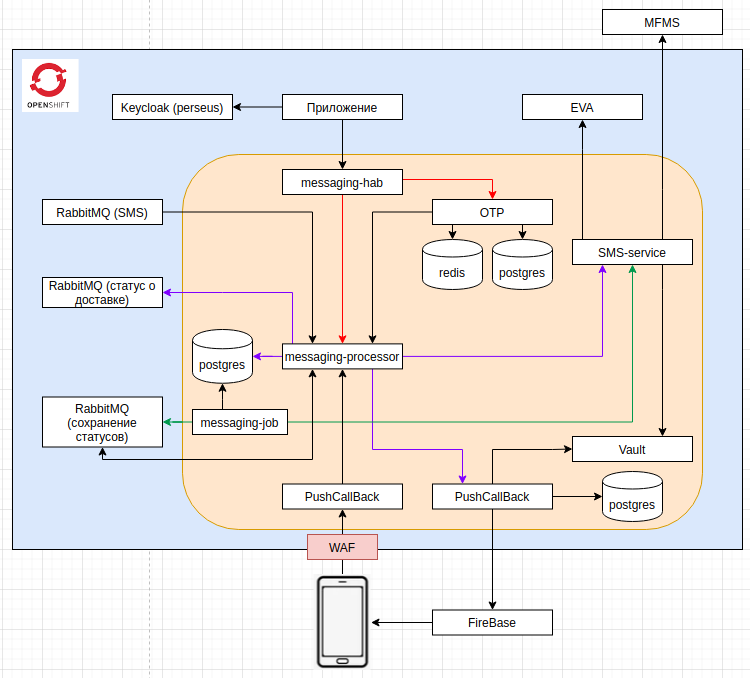


Рисунок . Компонентная архитектура сервиса.

Таблица 1.2. Описание компонентов сервиса.

| **№** | **Компонент** | **Описание компонента** |
| --- | --- | --- |
| 1 | messaging-hub | Единая точка входа для запросов на отправку всех типов сообщений. |
| 2 | messaging-processor | Сервис контроля отправки гарантированных сообщений (push+смс), хранение и обработка статусов push сообщений. |
| 3 | messaging-job | Сервис запуска job:   * мониторинг статусов по СМС для RabbitMQ ; * отложенная отправка; * отправка смс если не пришел статус по пуш сообщению. |
| 4 | PushCallBack | Сервис приёма статуса PUSH сообщения от мобильных приложений о доставке и прочтении. |
| 5 | OTP | Сервис отправки и проверки OTP кодов. |
| 6 | SMS-servise | Сервис отправки СМС. |
| 7 | push-servise | Сервис отправки push. |
| 8 | vault | Система хранения паролей и json для отправки push сообщений. |
| 9 | sms-rabbit-integration | Обработка и отправка в Rabbit статусов по доставке СМС. |
| 10 | MFMS | Интеграционный шлюз по отправке СМС. |
| 11 | EVA | Мастер система по данным клиента, для определения телефона по id клиента. |
| 12 | Приложение | Инициатор отправки сообщения. |
| 13 | FireBase | Интеграционный шлюз по отправке push. |
| 14 | RabbitMQ | Шлюз обмена сообщениями. |
| 15 | Postgres | База данных для хранения статусов по push сообщениям. |
| 16 | Postgres | База данных для хранения сопоставления master id клиента и push-токена. |
| 17 | Redis | База для отправки статус sms сообщений в rabbit. |

## [Асинхронная отправка сообщений через RabbitMQ](https://jira.bcs.ru:4464/confluence/pages/viewpage.action?pageId=192817272)

**Данные RabbitMQ:**

* Exchange;
* routingKey;
* тип сообщения:**application/json.**

Формат и описание сообщений:

<dependency>

<groupId>messaging-api</groupId>

<artifactId>messaging-api-external</artifactId>

<version>1.0.0-RELEASE</version>

</dependency>

Для интеграцией с messaging-сервисом необходимо отправить сообщение типа json.

**Пример запроса на отправку письма**

{

"appId": "string",

"type": "sms",

"appName": "string",

"subject": "string",

"text": "Текст сообщения",

"recipient": {

"clientId": "string",

"phone": "string",

"token": "string"

},

"sendStatus": "NONE",

"pushOptions": {

"title": "string",

"timeToLive": 0,

"priority": "HIGH",

"sound": "string",

"clickAction": "string",

"data": {

"additionalProp1": "string",

"additionalProp2": "string",

"additionalProp3": "string"

}

},

"timeSend": {

"from": "string",

"to": "string"

}

}

**Пример запроса при успехе**

{

"type": "sms",

"appName": "pushsms",

"subject": "BCS",

"text": "Подключение ЕБС ФОРТС отклонено.",

"recipient": {

"phone": "89991234567"

}

}

## [Мониторинг статусов сообщений через RabbitMQ](https://jira.bcs.ru:4464/confluence/pages/viewpage.action?pageId=139146021)

Если необходимо отследить статусы смс, которые были отправлены при помощи **сервиса отправки смс** необходимо:

* При отправке смс при помощи **sendMessage** указать соответствующее значение параметра (варианты значений параметра приведены в Таблица 1.3);

Таблица 1.3. Значения параметра sendMessage

|  |  |  |
| --- | --- | --- |
| **Параметр** | **Значение** | **Описание** |
| sendStatus | ALL\_STATUS\_IN\_RABBIT | Отслеживать все статусы |
| FINITE\_STATUS\_IN\_RABBIT | Отслеживать только конечные статусы |
| NONE | Не отслеживать статусы (по умолчанию) |

* Подписаться на получение сообщений через RabbitMQ в соответствии с параметрами (приведены в Таблица 1.4).

Таблица 1.4. Описание параметров сообщений через RabbitMQ.

|  |  |
| --- | --- |
| **Параметр** | **Значение** |
| routing-key | application-name из запроса **sendMessage** |
| exchange | push-sms.message-status.out |

Микросервис сохраняет в базе идентификаторы смс, требующих отслеживания, и хранит их до получения конечного статуса.

С интервалом  push-sms.rabbit-integration.service.timeout (по - умолчанию 1с) запрашивается статус сообщений из базы в шлюзе отправки смс и направляется сообщение в RabbitMQ.

Идентификаторы смс, достигшие конечного статуса удаляются из базы.

Сообщения, направляемые в RabbitMQ сериализуются в JSON и содержат следующие поля:

* Идентификатор сообщения (id - тип UUID);
* Полученный статус (messageStatus - следующие строки: "enqueued", "declined", "rejected", "failed", "sent", "cancelled", "delayed", "delivered", "undelivered", "read", "unknown" - статусы, которые получены от шлюза).

В Таблица 1.5 приведены статусы сообщений из шлюза и их состояния. Конечное состояние означает, что сообщение не может перейти из этого состояния в другое. Промежуточное – сообщение может перейти из одного состояния в другое.

Таблица 1.5. Состояния статусов сообщений из шлюза.

|  |  |
| --- | --- |
| **Статус** | **Состояние** |
| Enqueued | Промежуточное |
| Declined | Конечное |
| Rejected | Конечное |
| Failed | Конечное |
| Sent | Промежуточное |
| Canceled | Промежуточное |
| Delayed | Промежуточное |
| Delivered | Конечное |
| Undelivered | Конечное |
| Read | Промежуточное |
| Unknown | Промежуточное |

## Описание API

### Описание API микросервиса "Отправка сообщений"

**Описание запросов для диспетчера сообщений**

|  |  |
| --- | --- |
| ***Наименование запроса*** | ***Описание*** |
| POST/ v2 / sendMessage | Метод для отправки сообщения |
| POST / v2 / sendMessages | Метод для отправки нескольких сообщений |

**Описание запросов для отп-контроллера**

|  |  |
| --- | --- |
| ***Наименование запроса*** | ***Описание*** |
| POST / v2 / otp / [check](https://messaging-service.okd.t-global.bcs/swagger-ui.html#/operations/otp-controller/checkUsingPOST) | Проверять |
| POST / v2 / otp / send | Отправлять |

**Описание запросов для контроллера push-устройств**

|  |  |
| --- | --- |
| ***Наименование запроса*** | ***Описание*** |
| POST / v2 / [client-devices](https://messaging-service.okd.t-global.bcs/swagger-ui.html#/operations/push-devices-controller/saveClientDeviceUsingPOST) | Сохранение сущности |
| DELETE/ v2 / client-devices / {токен} | Удаление по токену |
| GET/ v2 / client-devices / searchByClientId / {clientId} | Найти приложения клиента |

**Описание запроса для контроллера статусов**

|  |  |
| --- | --- |
| ***Наименование запроса*** | ***Описание*** |
| POST / v2 / getStatuses | getStatuses |

### Описание API микросервиса "Прием статусов PUSH-сообщений"

**Описание запроса для контроллера приема статусов push-сообщений**

|  |  |
| --- | --- |
| ***Наименование запроса*** | ***Описание*** |
| POST / v1 / status | Прием статусов push-сообщений |

# 

# Сервис добавления штампа в pdf

## Общие сведения

Сервис предназначен для добавления в документы формата pdf штампов. Более подробную информацию о штампах вы можете найти в документации Adobe по ссылке <https://helpx.adobe.com/ru/acrobat/using/adding-stamp-pdf.html>.

Для добавления штампа сервис использует платную библиотеку [ASPOSE](https://docs.aspose.com/display/pdfnet/Add+Text+and+Image+Stamp).

## Безопасность

Все запросы защищены аутентификацией. Аутентификация осуществляется по стандарту Oauth2 c JWT. Токен должен передаваться в заголовке Autorization.

## Авторизация

Авторизация осуществляется под учетной записью в домене организации.

## Описание API

|  |  |
| --- | --- |
| ***Наименование запроса*** | ***Описание*** |
| POST[/v1/stamper](https://pdfstamp.okd.t-global.bcs/swagger-ui.html#/operations/pdf-stamp-controller/addStampUsingPOST) | Наложение штампа |

# Сервис коротких ссылок (Shorter)

## Описание сервиса

Сервис предназначен для сокращения URL. Сокращенный вариант URL создается на тот же ресурс, что и длинная ссылка. Отличие состоит только во внешнем виде. Короткие ссылки используются, например, для передачи в СМС или email.

## Авторизация

Запросы защищены аутентификацией. Аутентификация осуществляется по стандарту Oauth2 c JWT. Токен передается в заголовке Autorization.

Пример получения токена:

curl -X POST \

https://<URL>/auth/realms/<REALM>/protocol/openid-connect/token \

-H 'Cache-Control: no-cache' \

-H 'Content-Type: application/x-www-form-urlencoded' \

-d 'client\_id=bcsurlshortener-gate&grant\_type=client\_credentials&client\_secret=<secret>'

## Информационная архитектура

На Рисунок 3.1 представлена схема интеграции сервиса коротких ссылок с другими системами. В Таблица 3.1 перечислены запросы, которые осуществляются к сервису коротких ссылок, и дано их описание.



Рисунок .. Интеграционная схема сервиса коротких ссылок.

Таблица 3.1. Описание запросов.

|  | **Код** | **Название** | **Отправитель** | **Получатель** | **Тип** | **Описание** |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | INF01 | Получение короткой ссылки | Приложение | МС Shorter  (shorter-generator) | request-driven | POST запрос, содержащий ссылку, которую требуется сократить.  В ответ приходит сгенерированная ссылка формата https://realm.ru/\* и  генерируемая часть от 2 до 6 знаков (только латиница, маленькие и большие). |
| 2 | INF02 | Поиск длинной ссылки по короткой | Приложение | МС Shorter  (shorter-generator) | request-driven | GET запрос, содержащий короткую ссылку, по которой необходимо найти длинную. |
| 3 | INF03 | Redirect с короткой ссылки на длинную | Потребитель (клиент, получивший короткую ссылку) | МС Shorter  (shorter-api) | request-driven | GET запрос осуществляется при переходе по короткой ссылке. Если ссылка не найдена, происходит перенаправление по длинной ссылке. |

## Компонентная архитектура

На Рисунок 3.2 представлены компоненты сервиса, их взаимодействие между собой. В Таблица 3.2 дано описание компонентов.

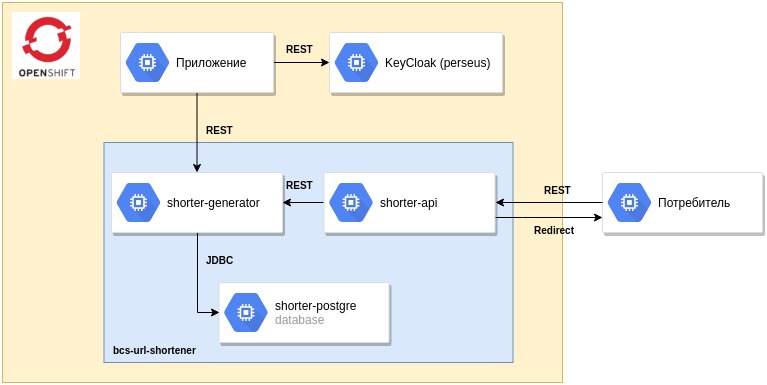


Рисунок .. Компоненты сервиса.

Таблица 3.2. Описание компонентов сервиса.

|  |  |  |
| --- | --- | --- |
| **№** | **Компонент** | **Описание компонента** |
| 1 | shorter-api | Предоставляет API для получения короткой ссылки и из короткой ссылки – длинную. |
| 2 | shorter-generator | Генерирует короткую ссылку. |
| 3 | shorter-postgres | База данных. Хранение данных о соответствии длинной и короткой ссылки. |

## Описание API сервиса коротких ссылок

|  |  |
| --- | --- |
| ***Наименование запроса*** | ***Описание*** |
| GET[/ v1 / longUrl](https://shorter-generator.okd.t-global.bcs/swagger-ui.html#/operations/bcs-url-shortener-controller/getLongUrlUsingGET) | Получения длинной ссылки по короткой |
| POST[/ v1 / url](https://shorter-generator.okd.t-global.bcs/swagger-ui.html#/operations/bcs-url-shortener-controller/generateShortUrlUsingPOST) | Генерации короткой ссылки |

# Сервис нотификации telephony-publisher

## Общие сведения

Сервис предназначен для отслеживания событий, связанных со звонками на номера сотрудников компании. Сервис осуществляет отслеживание событий по входящим и по исходящим звонкам. Сервис отслеживает информацию о звонках по номеру телефонов сотрудника только по требованию потребителя. Потребитель "подписывается" на получение событий по заданному номеру телефона.

Сервис использует JTAPI, предоставляемое Cisco Unified Communications Manager (CUCM).

При получении отслеживаемого события из CUCM сервис направляет сообщение о нем в брокер сообщений RabbitMQ.

Сервис осуществляет отслеживание следующих событий:

1. Начало звонка;
2. Начало разговора;
3. Завершение разговора.

## Схема информационных потоков

На Рисунок 4.1 представлена схема передачи данных в процессе работы сервиса telephony - publisher.

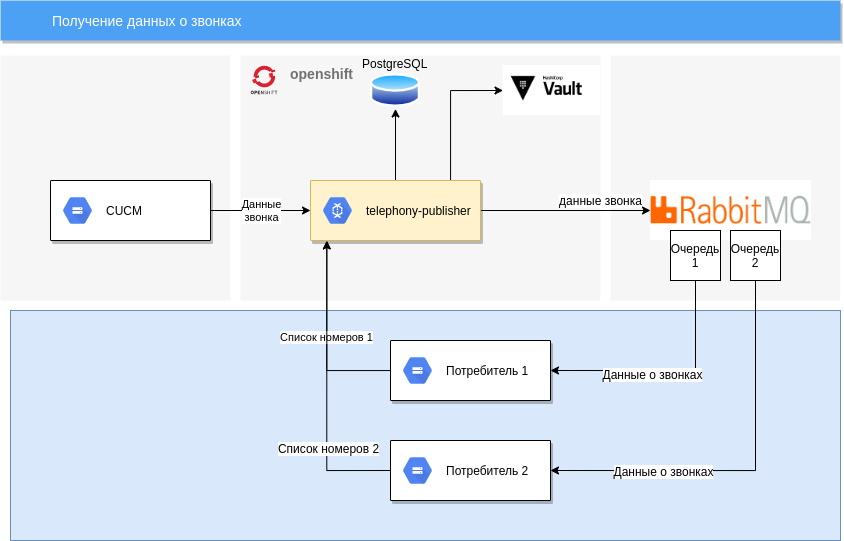


Рисунок .. Схема информационных потоков в сервисе.

## Процесс подключения нового потребителя

Процесс подключения нового потребителя происходит следующим образом:

1. Потребитель предоставляет **список номеров,** информацию о звонках, которых он хочет получать;
2. Потребитель предоставляет информацию о **дополнительных настройках;**
3. Данные, предоставленные потребителем, вносятся в конфигурацию микросервиса;
4. В RabbitMq создается очередь, в которую будут поступать сообщения;
5. Потребитель получает права на чтение очереди.

## Процесс получения данных о звонках от CUCM

Процесс получения данных о звонках в микросервис telephony - publisher от CUCM происходит согласно этапам:

1. Микросервис подписывается на получение событий о звонках с номеров телефонов, которые содержатся в списках потребителей;
2. Микросервис получает из CUCM информацию о событиях по отслеживаемым номерам телефонов сотрудников организации;
3. Микросервис формирует сообщение о событии и направляет его в соответствующую очередь.

## Формат сообщений

1. Действие «Начало звонка» содержит данные, перечисленные в Таблица 4.1.

Таблица 4.1. Данные, содержащиеся в действии «Начало звонка».

|  |  |
| --- | --- |
| callId | Идентификатор звонка. |
| phoneObsrv | Номер телефона сотрудника, по которому отправлено сообщение. |
| event | **outgoing\_start -** для исходящего звонка.  **incoming\_start** - для входящего звонка. |
| eventTime | Дата и время поступления звонка. |
| phoneFrom | Номер, с которого совершается звонок. |
| phoneTo | Номер, на который совершается звонок. |

2. Действие «Начало разговора» содержит данные, перечисленные в Таблица 4.2.

Таблица 4.2. Данные, содержащиеся в действии «Начало разговора».

|  |  |
| --- | --- |
| callId | Идентификатор звонка. |
| phoneObsrv | Номер телефона сотрудника, по которому отправлено сообщение. |
| event | **outgoing\_talk** - для исходящего звонка.  **incoming\_talk** - для входящего звонка. |
| eventTime | Дата и время поступления звонка. |
| phoneFrom | Номер, с которого совершается звонок. |
| phoneTo | Номер, на который совершается звонок. |

3. Действие «Завершение разговора» содержит данные, перечисленные в Таблица 4.3.

Таблица 4.3. Данные, содержащиеся в действии «Завершение разговора».

|  |  |
| --- | --- |
| callId | Идентификатор звонка. |
| phoneObsrv | Номер телефона сотрудника, по которому отправлено сообщение. |
| event | **outgoing\_end** - для исходящего звонка.  **incoming\_end** - для входящего звонка. |
| eventTime | Дата и время завершения разговора. |
| phoneFrom | Номер, с которого совершается звонок. |
| phoneTo | Номер, на который совершается звонок. |

## Дополнительные настройки

1. Входящие звонки типа клиент - сотрудник осуществляются через единый клиентский номер, потому что необходимо получать информацию о клиентских звонках, если клиент не напрямую звонит сотруднику, а звонит на общий клиентский номер.

2. Необходимо добавить в сообщение о звонке индивидуальный идентификатор сотрудника, потому что идентификация сотрудника осуществляется не только по номеру телефона. В сообщения о звонках будет добавлено поле **individual\_id** сотрудника, звонки которого отслеживаются. **individual\_id** является основным идентификатором сотрудника.

3. Получать данные необходимо только о входящих звонках.

4. Необходимо получать данные о звонках по номерам телефонов, соответствующих шаблону. Для этого надо добавить на отслеживаемые звонки фильтр по номеру телефона. Фильтр можно добавить на номер вызывающего абонента и на номер вызываемого.

Данная функция может быть использована, например, для удаления "технических" звонков.

Пример:

phone-number-categories:

employee: "^.{5}$"

exclude:

-

from-categorie: employee

to-categorie: employee

## Бизнес архитектура

**На** Рисунок 4.2 **представлена бизнес архитектура микросервиса** telephony – publisher. Для ее выполнения должно быть выполнено условие: была осуществлена подписка на телефоны сотрудников, по которым необходимо получать информационные сообщения в Rabbit.

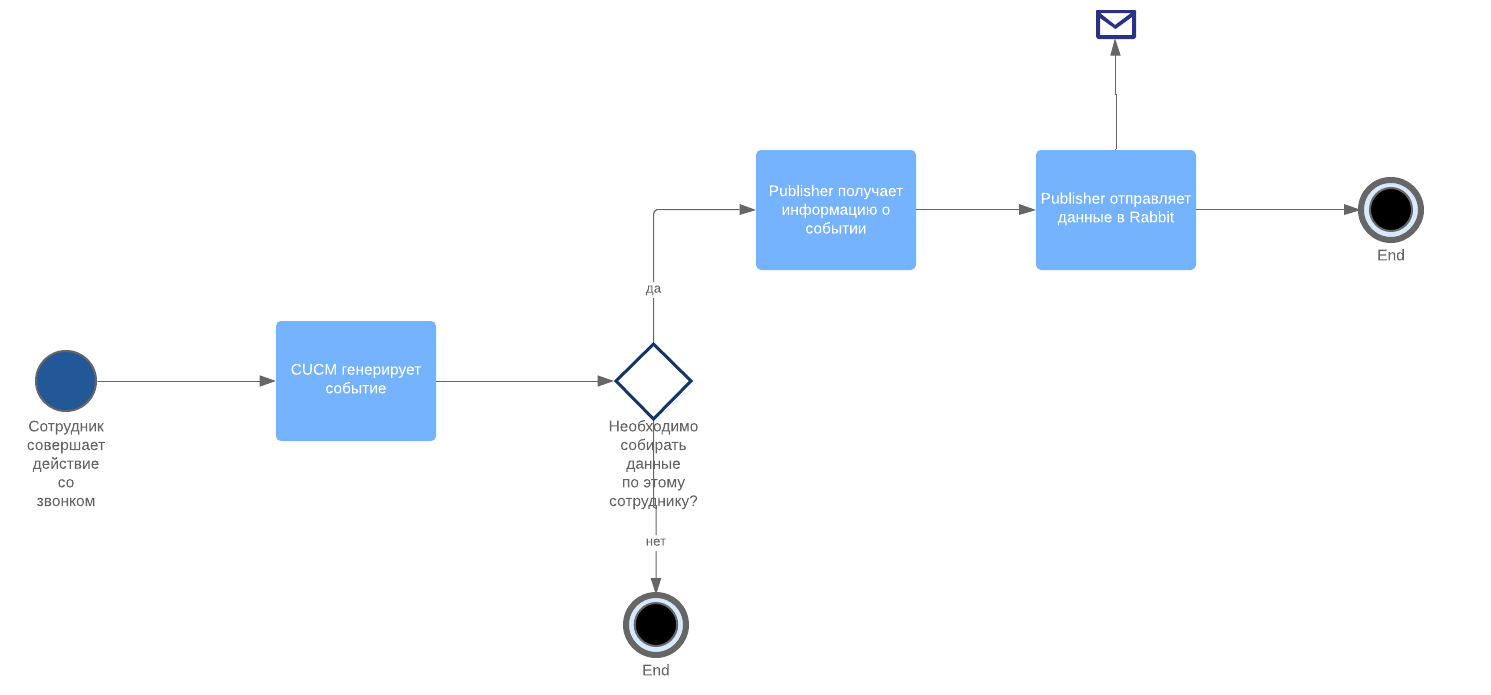


Рисунок .. Бизнес архитектура микросервиса telephony – publisher.

## Информационная архитектура

Информационная архитектура **микросервиса** telephony – publisher представлена на Рисунок 4.3. Описание информационных потоков дано в Таблица 4.4. Описание назначения информационных систем, с которыми взаимодействует **микросервис** telephony–publisher, приведено в Таблица 4.5.

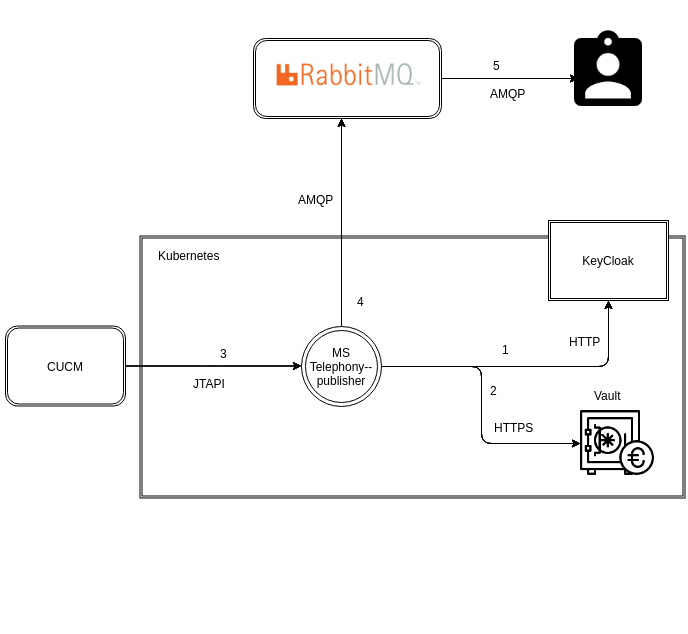


Рисунок .. Информационная архитектора микросервиса.

Таблица 4.4. Описание информационных потоков.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№ потока** | **Источник** | **Получатель** | **Тип** | **Тип взаимодействия** | **Описание** |
| 1 | telephony-publisher | keycloak | HTTP | request-driven | Микросервис **telephony -publisher** получает из сервиса аутентификации Keycloak открытый ключ для проверки JWT токенов. |
| 2 | telephony-publisher | Vault | HTTPS | request-driven | Микросервис получает данные для подключения к CUCM из хранилища секретов Vault. |
| 3 | CUCM | telephony-publisher | JTAPI | event driven | Микросервис получает из CUCM информацию о событиях по отслеживаемым номерам телефонов сотрудников организации. |
| 4 | telephony-publisher | RabbitMQ | AMQP | event driven | Микосервис направляет сообщения о событиях в RabbitMQ. |

Таблица 4.5. Назначение информационных систем.

|  | **Система** | **Назначение** |
| --- | --- | --- |
| 1 | telephony-publisher | Корпоративный микросервис, предназначенный для направления сообщений с информацией о звонках в RabbitMQ, для дальнейшего использования потребителями. |
| 2 | keycloak | Корпоративный микросервис предназначенный для аутентификации сотрудников организации, действующих клиентов организации и технических пользователей. |
| 3 | Vault | Система, предназначенная для хранения и управления секретами (пароли, ключи API и т.д.). |
| 4 | RabbitMQ | Брокер сообщений, который обеспечивает управление очередями, для обмена сообщениями между системами. |
| 5 | CUCM | Cisco Unified Communications Manager(CUCM) - система управления звонками. |

## Информационная безопасность

Набор телефонных номеров сотрудников компании, о которых микросервис получает информацию, ограничен правами доступа пользователя CUCM.

## Обработка звонка, поступившего от клиента

Процесс поступления звонка от клиента на номер оператора организации представлен на Рисунок 4.4.

Все звонки, поступающие на телефоны операторов кредитной биржи, фильтруются по источнику поступления.

Если звонок был распределен с номера триггера кредитной биржи, то события по нему будут отправлены в RabbitMQ.

Если звонок поступил из другого источника, то события будут игнорироваться.

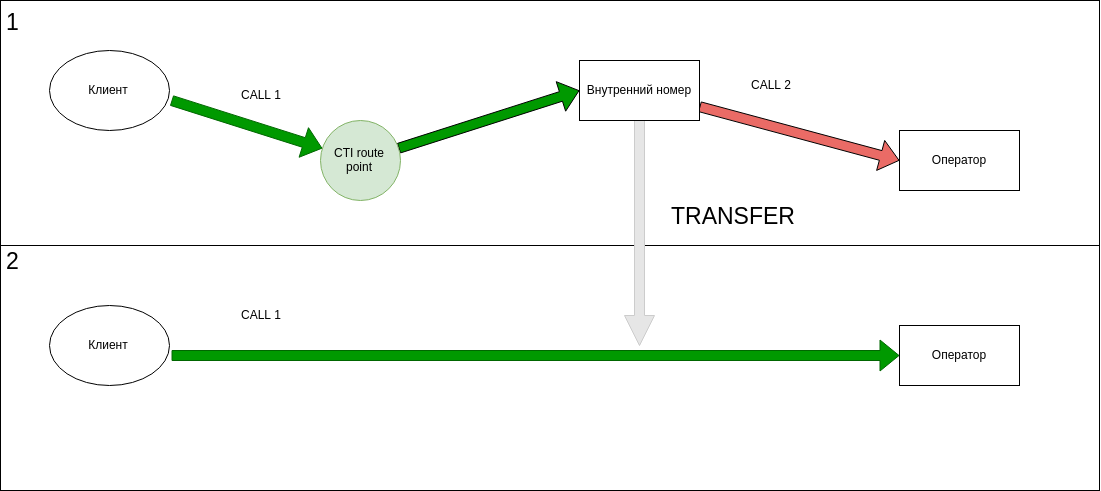


Рисунок .. Процесс поступления звонка от клиента на номер оператора

Этапы процесса поступления звонка от клиента на номер оператора:

1. клиент звонит на выделенный номер отдела в организации;
2. звонок попадает на определенный cti route point;
3. звонок попадает на некоторый внутренний номер;
4. внутренний номер инициирует звонок на номер оператора;
5. если оператор принимает звонок, то происходит трансфер;
6. клиент и оператор разговаривают.

## Сбор и фильтрация событий телефонии на уровне telephony-publisher

Сбор и фильтрация событий телефонии в микросервисе осуществляется следующим образом (Рисунок 4.5):

* микросервис telephony-publisher подписывается на получение событий по:
* Устройствам, привязанным к телефонным номерам операторов;
* route point кредитной биржи;
* при поступлении звонка на route point микросервис записывает информацию о поступившем звонке в базу данных;
* Если звонок не был принят и завершился, то в RabbitMQ направляется сообщение;
* Если микросервис получает сообщения о событиях, которые произошли на телефонных аппаратах операторов, то происходит проверка по callId;
* Если звонок с таким callId ранее поступал на route point (т. е. этот звонок изначально поступил на номер кредитной биржи), то сообщение о событии отправляется в Rabbit;
* Если нет, то событие игнорируется;

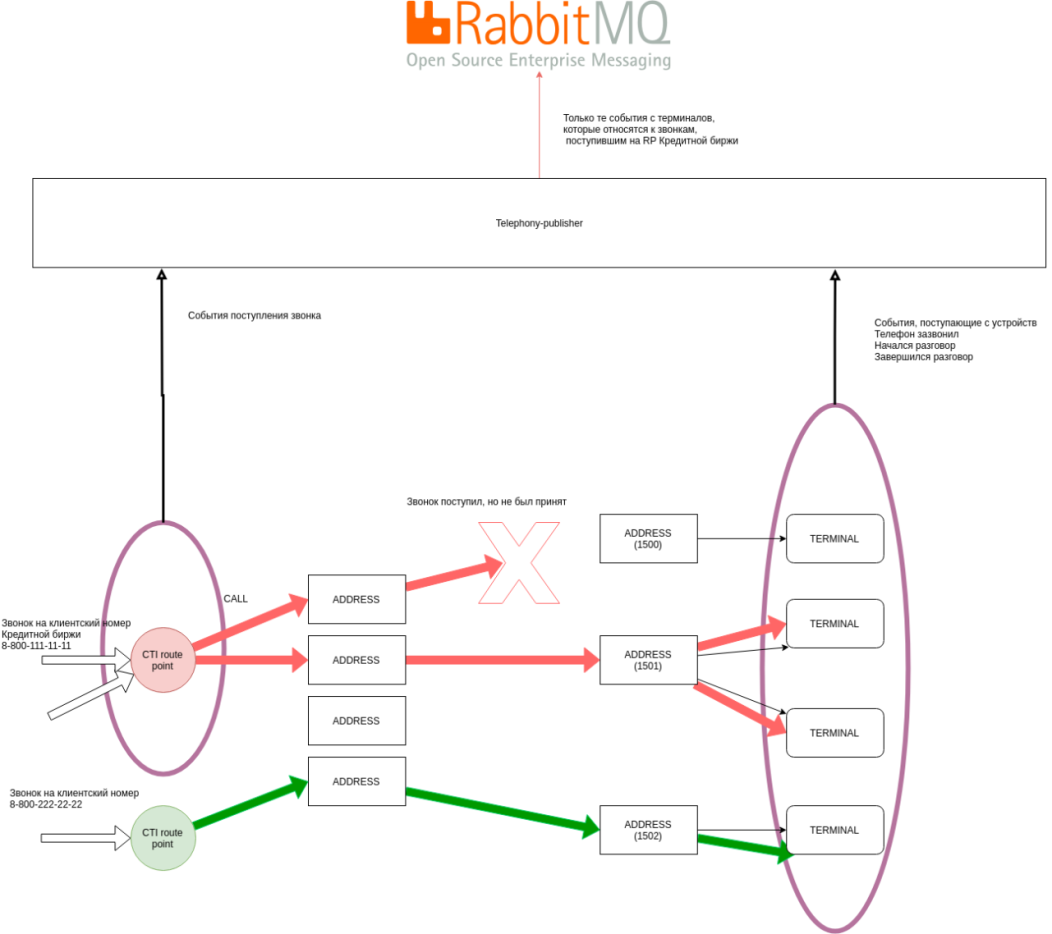


Рисунок .. Сбор и фильтрация событий в микросервисе.

Сообщения о событиях, направляемые в RabbitMQ, содержат информацию, представленную в Таблица 4.6.

Таблица 4.6. Информация, содержащаяся в сообщениях о событиях.

|  |  |
| --- | --- |
| **Наименование** | **Комментарий** |
| callId | идентификатор звонка |
| phoneObsrv | данные, при отслеживании какого номера было получено событие |
| event | incoming\_start - начало звонка incoming\_talk - начало разговора incoming\_end - окончание разговора |
| phoneFrom | номер клиента, с которого поступил звонок |
| phoneTo | номер оператора, на который поступил звонок |
| eventTime | время поступления события |

# Сервис печати

## Описание сервиса

Сервис позволяет получать печатные формы документов на основании заданных шаблонов.

Сервис конвертирует шаблоны с html/docx в pdf в приемлемом качестве. Сервис формирует печатную форму [на основе шаблона](/pages/viewpage.action?pageId=134586243) и запроса на печать с данными для заполнения плейсхолдеров шаблона.

Для передачи шаблонов в сервис печати используются два формата: base64 и MultipartFile. Файлы в формате base64 передаются в теле запроса. Файлы в формате MultipartFile передаются составным запросом, и рекомендуется для шаблонов больших размеров (до 30 Мб).

Для получения результирующего файла есть две разные операции. Одна передаёт результирующие файлы в формате base64. Другая, передает результат в формате Resources (файлы до 30 Мб). Подробное описание представлено в 5.4 Описание API.

## Схема взаимодействия сервиса

Разработчик (аналитик) создаёт шаблон, на основании которого будет генерироваться результирующий документ (Рисунок 5.1).

Шаблон сохраняется в PrintService с указанием необходимых данных с помощью операции Save. В результате выполнения операции возвращается версия шаблона.

Из системы делается вызов операции report-single или report, если требуется печать нескольких документов, с указанием шаблона и данными для плейсхолдеров. В результате приходит сгенерированный документ.

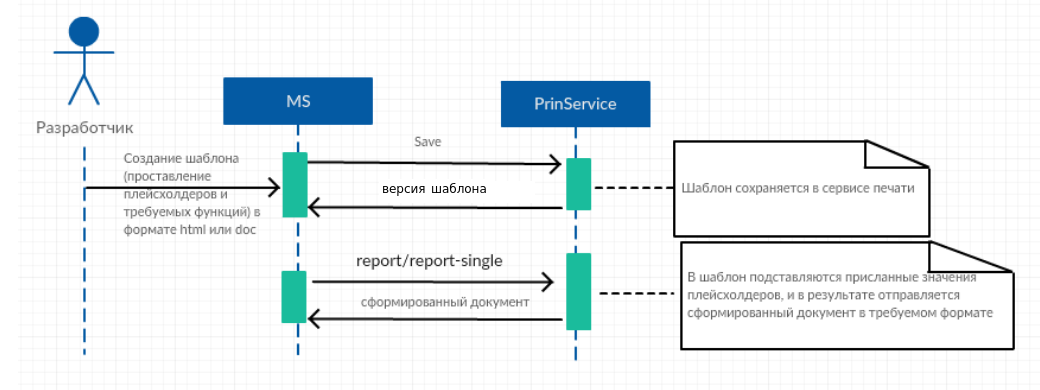


Рисунок .. Схема генерации документа.

## Архитектурное решение

Сервис предполагает аутентификацию по протоколу OAuth2. Получение токена с учетными данными Active Directory или протоколу Kerberos.

Для авторизации предполагается наличие у пользователя группы AD - printService. При отсутствии токена или данной группы сервис будет возвращать HTTP-код 401.

Основные этапы взаимодействия отражены на Рисунок 5.2:

* получение микросервисом печати открытого ключа для проверки JWT токенов.
* получение приложением потребителем JWT токена в сервисе Keycloak. Возможно получение токена по логину/паролю доменой учетной запси или по протоколу Kerberos.
* запрос приложением потребителем сервиса-шлюза(Zuul).
* запрос сервиом-шлюза (Zuul) микросевриса печати.
* подключение микросервиса печати к БД PostgresSQL для сохранения/получения шаблонов печати.

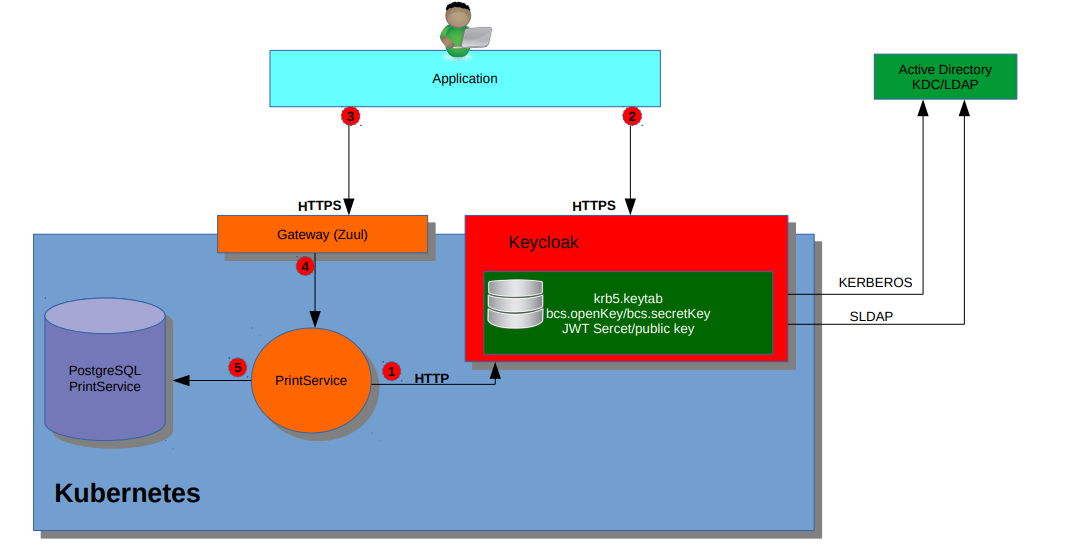


Рисунок .. Архитектура сервиса печати.

## Описание API

Для формирования печатной формы на основе шаблона и формирования запроса на печать с данными для заполнения плейсхолдеров шаблона и необходимо:

1. Создать шаблон и сохранить его с помощью операции Save.
2. Сгенерировать документ (-ы) на основании шаблона (-ов) с помощью операции report-single (report).

**Описание запросов для печати документов в формате base64**

(функция base-64-print-controller)

|  |  |
| --- | --- |
| ***Наименование запроса*** | ***Описание*** |
| POST/v1/base64/print/report | Печать нескольких документов, каждый из одного шаблона. Сколько шаблонов на вход, столько документов получается на выходе. Результат в формате base64. |
| POST/v1/base64/print/report-list | Печать нескольких документов, каждый из нескольких шаблонов. Самый гибкий метод, но требующий максимальной аккуратности. Результат в формате base64. |
| POST/v1/base64/print/report-single | Печать одного документа из нескольких шаблонов. Один или несколько шаблонов объединяются в один документ на выходе. Результат в формате base64. |

**Описание запросов для работы с шаблонами в формате base64**

(Функция base-64-template-controller)

|  |  |
| --- | --- |
| ***Наименование запроса*** | ***Описание*** |
| POST/v1/base64/template | Сохранение нового шаблона, новой версии шаблона. Шаблон в формате base64. |
| GET/v1/base64/template/find | Поиск всех шаблонов. |
| GET/v1/base64/template/find/{owner}/{code} | Поиск всех версий шаблона. Результат в формате base64. |
| GET/v1/base64/template/findOne/{owner}/{code} | Поиск последней версии шаблона. Результат в формате base64. |
| GET/v1/base64/template/findOne/{owner}/{code}/{version} | Поиск конкретной версии шаблона. Результат в формате base64. |

**Описание запросов для печати документов в формате MultipartFile**

(Функция print-controller)

|  |  |
| --- | --- |
| ***Наименование запроса*** | ***Описание*** |
| POST/v1/print/report | Печать нескольких документов, каждый из одного шаблона. Сколько шаблонов на входе, столько документов получается на выходе. Результат в виде ссылки на ZIP файл (Resource). |
| POST/v1/print/report-list | Печать нескольких документов, каждый из нескольких шаблонов. Самый гибкий метод, но требующий максимальной аккуратности. Результат в виде ссылки на ZIP файл (Resource). |
| POST/v1/print/report-single | Печать одного документа из нескольких шаблонов. Один или несколько шаблонов объединяются в один документ на выходе. Результат в виде ссылки (Resource). |

**Описание запросов для работы с шаблонами в формате MultipartFile**

(Функция print-controller)

|  |  |
| --- | --- |
| ***Наименование запроса*** | ***Описание*** |
| POST/v1/template | Сохранение нового шаблона, новой версии шаблона. Шаблон в формате MultipartFile. |
| DELETE/v1/template | Удаление одной версии шаблона. |
| GET /v1/template/find/{owner} | Поиск шаблона по владельцу. |
| GET /v1/template/find/{owner}/{code} | Поиск всех версий шаблона. |
| GET /v1/template/findOne/{owner}/{code} | Поиск последней версии шаблона. |
| GET /v1/template/findOne/{owner}/{code}/{version} | Поиск конкретной версии шаблона. |

## Примеры заполнения шаблонов и вызова на печать документов.

**Формат шаблонов и результирующих документов представлен на** Таблица 5.1**.**

Таблица 5.1.Формат шаблонов.

|  |  |
| --- | --- |
| **Шаблон** | **Документ** |
| \*docx | \*pdf |
| \*html | \*docx |
|  | \*html |

### Опитсание плейфсхолдеров

Основные функции строковых плейсхолдеров:

**<<[field]>>** - простой строковый плейсхолдер

**<<[BCS.substring(field1, 0, 1)]>>** - получение символа (-ов) указанной строки

**<<[field]:caps>>** - первая буква во всех словах заглавная

**<<[field]:firstCap>>** - первая буква в первом слове заглавная

**<<[field]:upper>>** - все буквы заглавными

**<<[field]:lower>>** - все буквы маленькие

**<<[field]:roman>>** - формат для цифр. Переводит все цифры в римские (1 -> I; 2 -> II и т.д.)

**<<[field]:arabicDash>>** - формат для цифр. К арабским цифрам добавляет тире (1 -> - 1 - ; 2 -> - 2 - и т.д.)

### Плейсхолдеры в формате даты

Основные функции:

**<<[dateFormat(EndDate, «dd MMMM yyyy», «RU»)]>>** - месяц прописью на русском;

**<<[dateFormat(EndDate, «dd MMMM yyyy», «RU»)]:upper>> -** месяц прописью на русском, заглавными буквами;

**<<[dateFormat(EndDate, «dd MMM yyyy», «RU»)]>> -** месяц прописью, сокращенно;

**<<[dateFormat(EndDate, «dd MMMM yyyy», «eng»)]>> -** месяц прописью на английском;

**<<[dateFormat(EndDate, «dd.MM.yyyy»)]>> -** дата в формате день.месяц.год;

**<<[dateFormat(EndDate, «dd.MM.yyyy HH:mm», «RU»)]>> -** дата в формате день.месяц.год часы:минуты;

**<<[dateFormat(LocalDateTime.now(), "dd.MM.yyyy HH:mm", "RU**")]>> - подставляет текущую дату и времяю

### Условия

Основные функции:

**<<if [!items.any()]>>no items<<else>><<[items.count()]>> item(s)<</if>>** - проверка на пустой массив;

**<<if [limit2 == null]>> Лимит не задан<<else>> <<[limit2]>> <</if>>** - проверка на пустую строку;

**<<[test]>> <<if [test == «234»]>>равно<<else>>не равно<</if>> 234** - проверка на совпадение значений строки;

**<<[test]>> <<if [test == «345»]>>равно 345 <<elseif [test == «234»]>> равно 234 <<else>>не равно ничему<</if>>** - двойная проверка;

### Массив

Основные функции:

<<if [listString.any()]>>**<<[listString.get(0)]>>**<</if>> - получение значения первого элемента массива;

**<<foreach [ls in listString]>><<[ls]>>; <</foreach>> -** вывод значений массива через точку с запятой;

**<< var [Peg = 0]>><<foreach [ls in listString]>><< var [Peg = Peg + 1]>><<[ls]>>;<</foreach>>** - вывод пронумерованных по порядку значений массива;

**<<[join(listString)]>>** - вывод массива в виде строки через запятую;

**<<[join(«\n», listString)]>>** - вывод массива с переносом строк;

### Склонения по падежам

Основные функции:

**<<[nameCase(test1, «Р»)]>>** - склонение существительных;

**<<[fioCase(surname, name, patronymic, «И»)]>>** - склонении ФИО в именительном падеже;

**<<[fioCase(surname, name, patronymic, «Р»)]>>** - склонении ФИО в родительном падеже;

**<<[fioCase(surname, name, patronymic, «Д»)]>>** - склонении ФИО в дательном падеже;

**<<[fioCase(surname, name, patronymic, «В»)]>>** - склонении ФИО в винительном падеже;

**<<[fioCase(surname, name, patronymic, «Т»)]>>** - склонении ФИО в творительном падеже;

**<<[fioCase(surname, name, patronymic, «П»)]>>** - склонении ФИО в предложном падеже.

### Суммы прописью и склонение валютных единиц (целой и дробной частей)

Возможные коды валют: **RUB, RUR, EUR, USD, HKD, CAD, AUD, TWD, XCD, GBP, JPY, CHF, PLN, UAH, KZ, UZS, BYN, PHP, XOF**

Сумма передается в шаблон цифрами через точку.

Основные функции:

**<<[formatSumInWords(sum, code)]>>** - всё прописью;

**<<[formatSumInWords(sum, code, «eng»)]>> -**всё прописью на английском;

**<<[formatSumMajorInWords(sum, code)]>> -**рубли прописью копейки цифрами;

**<<[formatSum(sum, code)]>> -**цифры цифрами, валюта прописью.

### Числа прописью

Основные функции:

**<<[formatNumberInWords(num)]>>** - число прописью (пример: 1010 - одна тысяча десять);

**<<[formatNumberInOrdinalWords(num)]>>** - склонение числительных (пример: 01 - первое, 02 - второе и т.д.). Как правило, применяется вместе с получением дня из даты:  **<< var [num = dateFormat(PayDate, «dd»)]>>**

### Валюта прописью и знаком

Возможные коды валют:

**RUB, RUR, EUR, USD, HKD, CAD, AUD, TWD, XCD, GBP, JPY, CHF, PLN, UAH, KZT, UZS, BYN, PHP, XOF**

**Основные функции:**

**<<[currencyFormat(code, «rus»)]>>** - прописью (Российский рубль, Доллар США и пр.);

**<<[currencyFormat(code, «sign»)]>>** - знаком (₽, $, € и пр.).

### Передача картинок

Картинка передаётся в запросе в формате base64. Обратите внимание, что тег **image** необходимо расположить внутри контейнера.

## Дополнительные функции при печати документа

Количество копий (copies) - сколько раз один шаблон будет повторяться в результирующем документе.

Признак редактируемости (readOnly) - только для docx. Результирующий документ не редактируется.

Дата действия документа (deleteDate) - дата, после которой шаблон невозможно использовать для генерации документа.

## Безопасность

Все запросы защищены аутентификацией. Аутентификация осуществляется по стандарту Oauth2 c JWT. Токен должен передаваться в заголовке Authorization

# Сервис получения справочника БИКов

## Общие сведения

Актуальный справочник БИК (ED807) ежедневно размещается на сайте Центрального Банка Российской Федерации. Изменения по справочнику БИК в течении дня (каждый день в 01:00 по Мск) присылаются по электронному взаимодействию от Центробанка в Банк организации в формате ED807.

Сервис отвечает за выполнение задач:

* получение полного списка всех БИК;
* получение полного списка действующих БИК и кор. счёта;
* получение полного списка удалённых БИК и кор. счёта;
* получение списка изменений по БИК за прошедший операционный день;
* получение списка с ограничением;
* поиск по БИК;
* поиск по кор. счёту;
* поиск по названию.

## ****Маппинг значений справочника БИК****

При загрузке справочника БИК, полученного с сайта Центробанка, происходят следующие действия:

* если БИК уже был заведен, в системе обновляется версия;
* если БИК не был заведён ранее, в системе он добавляется с датой создания предыдущим операционным днем;
* если БИК не найден в новой загрузке, то у такого БИКа проставляется дата удаления предыдущим операционным днём;
* если счёт не был заведён ранее, он добавляется с датой создания предыдущим операционным днем;
* если счёт не найден в новой загрузке, то у такого счёта проставляется дата удаления предыдущим операционным днём;
* если обновляются данные по БИК, то заполняется дата изменения предыдущим операционным днём.

## Применение интеграционной шины в интеграции сервиса

Для интеграции сервиса получения справочника БИК-ов с другими приложениями и сервисами применяется интеграционная шина (Рисунок 6.1). В качестве шины, может использоваться, например BizTalk Server.

После получения данных по справочникам формируются следующие сообщения и отправляются через шину:

* «Справочник БИК – Справочник действующих банков» - **#bnkseek;**
* «Справочник БИК – Справочник ликвидированных банков» - **#bnkdel.**

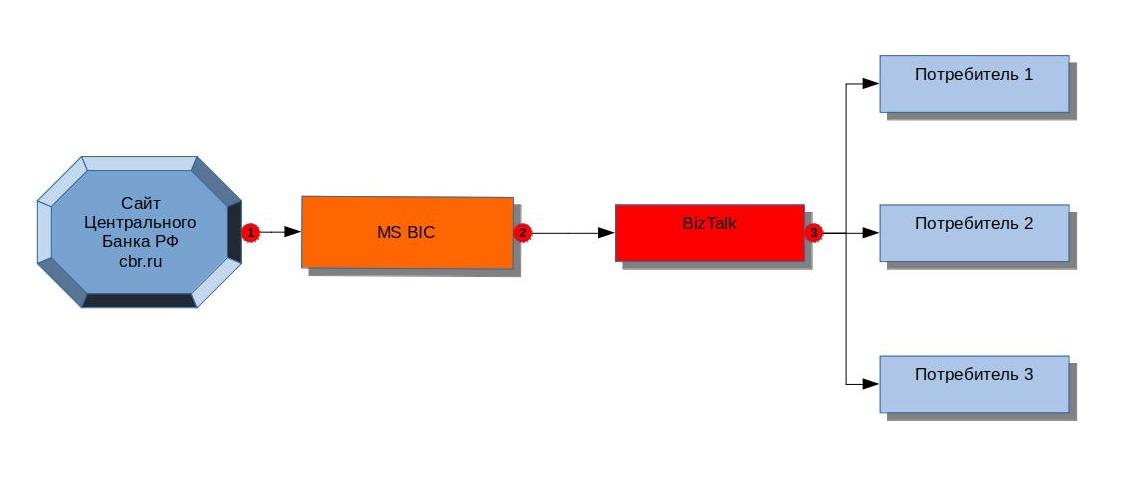


Рисунок .. Интеграционная схема сервиса.

### Ограничения передаваемых данных через интеграционную шину

1. Передаются только БИК, у которых есть кор. счет. Признак:

regulation\_account\_type = CRSA – Корреспондентский счет.

1. Не для всех типов населённых пунктов имеется соответствие. Например, появились такие значения как: ст. , жд\_ст., и т. д., которых ранее не было. Для них будет передаваться код 02 соответствующий типу пункта.

## Rest запросы

**Сервис позволяет получить следующие данные:**

* **получение полного списка всех БИК -** по запросу без входных данных;
* **получение полного списка действующих БИК и кор. счёт -** по запросу "active": true;
* **получение полного списка удалённых БИК и кор. счёт -** по запросу "active": false;
* **получение списка изменений по БИК за прошедший опер. день -** по запросу "lastChange": true;
* **получение списка с ограничением (например: только БИК по кредитным организациям) -** по запросу "ptType": "20";
* **поиск по БИК -** по запросу "bic": "044525095";
* **поиск по кор. счёту -** по запросу "lastChange": "30101810545250000095";
* **поиск по названию (поиск по like - минимум 3 символа) -** по запросу "name": **"**realm**";**
* **поиск по БИК головной организации -** по запросу "prntBic": "044537002";
* **поиск по СВИФТ -**по запросу "swbic": "MJSBRUMMXXX".

## Маппинг данных

### «Справочник БИК – Справочник действующих банков»

Маппинг данных «Справочник БИК – Справочник действующих банков» (bnkseek) приведен в Таблица 6.1.

Таблица 6.1. Маппинг данных «Справочник БИК – Справочник действующих банков».

|  |  |  |
| --- | --- | --- |
| **Переменная по xsd схеме** | **Переменная из БД** | **Описание** |
| VKEY | bic.id | Идентификатор записи |
| REAL | nil | Код ограничений участия в расчетах. Кроме соответствия с REAL по LWRS = 0009 (ОТЗВ) |
| PZN | bic.pt\_type | Код типа участника расчетов |
| UER | nil | Код участника системы электронных расчетов |
| RGN | bic.rgn | Код региона |
| IND | bic.ind | Индекс |
| TNP | bic.tnp | Код типа населенного пункта |
| NNP | bic.nnp | Наименование населенного пункта |
| ADR | bic.adr | Адрес |
| RKC | nil | БИК РКЦ |
| NAMEP | bic.name\_p | Полное наименование банка |
| NAMEN | bic.name\_p | Сокращенное наименование банка |
| NEWNUM | bik.bik | БИК |
| NEWKS | nil | БИК, действовавший до перехода на новый план счетов |
| PERMFO | nil | Номер МФО |
| SROK | nil | Срок доставки расчетно-денежных документов от кредитных организаций до РКЦ |
| AT1 | nil | Абонентский телеграф 1 |
| AT2 | nil | Абонентский телеграф 2 |
| TELEF | nil | Телефоны |
| REGN | bic.reg\_n | Регистрационный номер |
| OKPO | nil | Код ОКПО |
| DT\_IZM | bic.date\_change | Дата последнего изменения записи |
| CKS | nil | Номер установки центра коммутации сообщений |
| KSNP | аccount.аccount | Номер корреспондентского счета |
| DATE\_IN | bic.date\_in | Дата включения информации о банке в справочник |
| DATE\_CH | nil | Дата контроля |
| VKEYDEL | nil | Идентификатор преемника |
| DT\_IZMR | nil |  |

### «Справочник БИК – Справочник ликвидированных банков»

Маппинг данных «Справочник БИК – Справочник ликвидированных банков» (bnkdel) представлен в Таблица 6.2.

### Таблица 6.2. Маппинг данных «Справочник БИК – Справочник ликвидированных банков» .

|  |  |  |
| --- | --- | --- |
| **Переменная по xsd схеме** | **Переменная из БД** | **Описание** |
| VKEY | bic.id | Идентификатор записи |
| VKEYDEL | nil | Идентификатор преемника |
| PZN | bic.pt\_type | Код типа участника расчетов |
| UER | nil | Код участника системы электронных расчетов |
| RGN | bic.rgn | Код региона |
| IND | bic.ind | Индекс |
| TNP | bic.tnp | Код типа населенного пункта |
| NNP | bic.nnp | Наименование населенного пункта |
| ADR | bic.adr | Адрес |
| RKC | nil | БИК РКЦ |
| NAMEP | bic.name\_p | Полное наименование банка |
| NAMEN | bic.name\_p | Сокращенное наименование банка |
| NEWNUM | bik.bik | БИК |
| NEWKS | nil | БИК, действовавший до перехода на новый план счетов |
| PERMFO | nil | Номер МФО |
| SROK | nil | Срок доставки расчетно-денежных документов от кредитных организаций до РКЦ |
| AT1 | nil | Абонентский телеграф 1 |
| AT2 | nil | Абонентский телеграф 2 |
| TELEF | nil | Телефоны |
| REGN | bic.reg\_n | Регистрационный номер |
| OKPO | nil | Код ОКПО |
| DATEDEL | bic.date\_out | Дата удаления записи из справочника БИК |
| DT\_IZM | bic.date\_change | Дата последнего изменения записи |
| CKS | nil | Номер установки центра коммутации сообщений |
| KSNP | аccount.аccount | Номер корреспондентского счета |
| R\_CLOSE | nil | Код причины закрытия счета |
| DATE\_IN | bic.date\_in | Дата включения информации о банке в справочник |

## Структура базы данных справочника БИК

Структура базы данных справочника БИК дана в Таблица 6.3.

Таблица 6.3. Таблица ed807

|  |  |  |  |
| --- | --- | --- | --- |
| **Параметр** | **Тип** | **Обязательность** | **Описание** |
| ed\_no | int[9] | да | Номер ЭС в течение опер. дня. |
| ed\_date | date | да | Дата составления ЭС. |
| ed\_author | int[10] | да | Уникальный идентификатор составителя ЭС - УИС. |
| ed\_receiver | int[10] | нет | Уникальный идентификатор получателя ЭС. |

## Описание API сервиса хранения БИК-ов

|  |  |
| --- | --- |
| ***Наименование запроса*** | ***Описание запроса*** |
| POST/v1/bic-list | отчет |

# Сервис управления IP-телефонией

## Общие сведения

Сервис использует JTAPI, который предоставляет Cisco Unified Communications Manager (CUCM).

Сервис дает возможность выполнения следующих операций:

* совершение звонка с телефона сотрудником организации (как на телефон другого сотрудника, так и на внешние номера мобильных и городских телефонов клиентов);
* завершение звонка с телефона сотрудника организации;
* удержание звонка;
* снятие звонка с удержания;
* перевод звонка на другого сотрудника организации;
* подключение к звонку дополнительного абонента (конференция);
* ответ на входящий звонок.

## Безопасность

Все запросы защищены аутентификацией. Аутентификация осуществляется по стандарту Oauth2 c JWT. Токен должен передаваться в заголовке Autorization.

## Ограничения

Работа сервиса предполагает, что номер сотрудника организации состоит из 5 цифр. Номера, состоящие из другого количества цифр, считаются ошибочными.

Сервис предоставляет пользователю возможность совершения звонков только со своего рабочего IP номера (атрибут токена **ipPhone**).

Управление звонком может осуществлять только создатель.

Если звонок находится в статусе START и был инициирован дублирующий его звонок (с теми же номерами вызывающего и вызываемого), то первый звонок будет завершен автоматически.

## Авторизация

Авторизация осуществляется под учетной записью домена организации.

## Интеграционная схема

На Рисунок 7.1 представлена схема интеграции сервиса управления звонками с приложениями и другими сервисами.

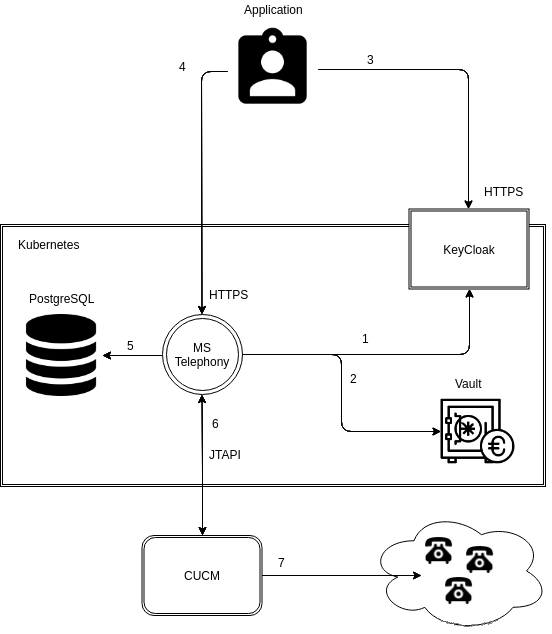


Рисунок .. Интеграционная схема сервиса.

1. Микросервис IP телефонии получает из сервиса аутентификации Keycloak открытый ключ для проверки JWT токенов.
2. Микросервис получает данные для подключения к CUCM из хранилище секретов Vault.
3. Приложение - потребитель получает JWT токена в сервисе Keycloak.
4. Приложение - потребитель направляет запрос в сервис IP Телефонии.
5. Микросервис подключается к БД для сохранения/получения данных о звонках.
6. Микросервис IP телефонии отправляет запрос в менеджер звонков CUCM.
7. Менеджер звонков CUCM осуществляет необходимые действия.

## Аутентификация пользователей

Запросы защищены аутентификацией. Аутентификация осуществляется по стандарту Oauth2 c JWT. Токен должен передаваться в заголовке Autorization.

**realm**: AD

**client\_id**: Telephony-service

**Группа AD**: APP\_MicroServices\_TelephonyService\_User

## Формирование звонка

Идентификатор звонка формируется следующим образом:

callId = globalCallID\_callId-globalCallID\_callManagerId

## Операции

**makeCall** - звонок с телефона сотрудника организации.

При вызове операции звонок будет осуществляться с рабочего телефона сотрудника, который содержится в токене при авторизации в микросервисе.

Данные запроса представлены в Таблица 7.1.

Таблица 7.1.Описание запроса звонка с телефона сотрудника организации.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Параметр** | **Тип** | **Описание** | **Обязательность** | **Пример** |
| phoneTo | string | Номер телефона, на который будет осуществляться звонок. Допустимы:   * Внутренние номера сотрудников организации * Мобильные номера * Городские номера | да | 19202  8913145263  +7913145263 |

Данные ответа представлены в Таблица 7.2.

Таблица 7.2. Описание ответов на запрос о звонке с телефона сотрудника организации.

|  |  |  |  |
| --- | --- | --- | --- |
| **Код** | **Поле** | **Описание** | **Пример** |
| 200 | callId | Идентификатор звонка. | 21073-1 |
| 422 |  | Управление номером ipPhone недоступно, так как это короткий номер корпоративного мобильного. | При попытке позвонить с номера вида  169хх, 186хх и 189хх |

**endCall** - завершение звонка.

Данные запроса представлены в Таблица 7.3.

Таблица 7.3. Описание запроса на завершение звонка.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Параметр** | **Тип** | **Описание** | **Обязательность** | **Пример** |
| callId | string | Идентификатор звонка. | да | 21073-1 |

Данные ответа: отсутствуют.

**holdCall** - удержание звонка.

Данные запроса представлены в Таблица 7.4.

Таблица 7.4. Описание запроса на удержание звонка.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Параметр** | **Тип** | **Описание** | **Обязательность** | **Пример** |
| callId | string | Идентификатор звонка. | да | 21073-1 |

Данные ответа: отсутствуют.

**returnToCall** - активация звонка после удержания.

Данные запроса представлены в Таблица 7.5.

Таблица 7.5. Описание запроса на активизацию звонка после удержания.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Параметр** | **Тип** | **Описание** | **Обязательность** | **Пример** |
| callId | string | Идентификатор звонка. | да | 21073-1 |

Данные ответа: отсутствуют.

**transferCall** - перевод звонка с телефона сотрудника организации.

Данные запроса представлены в Таблица 7.6.

Таблица 7.6. Описание запросов на перевод звонка с телефона сотрудника организации.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Параметр** | **Тип** | **Описание** | **Обязательность** | **Пример** |
| callId | string | Идентификатор звонка. | да | 21073-1 |
| newPhone | string[5] | Номер телефона сотрудника организации, на который нужно перевести звонок.  По результату выполнения этот номер будет подключен к звонку. | да | 19202 |

Данные ответа: отсутствуют.

**addCall** - подключение к звонку дополнительного участника.

Данные запроса представлены в Таблица 7.7.

Таблица 7.7. Описание запросов на подключение к звонку дополнительного участника.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Параметр** | **Тип** | **Описание** | **Обязательность** | **Пример** |
| callId | string | Идентификатор звонка. | да | 21073-1 |
| addPhone | string[5] | Номер телефона, который нужно присоединить к звонку. | да | 19202 |

Данные ответа: отсутствуют.

**answerCall** - ответ на входящий звонок.

Данные запроса представлены в Таблица 7.8.

Таблица 7.8. Описание запроса на ответ входящего звонка.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Параметр** | **Тип** | **Описание** | **Обязательность** | **Пример** |
| callId | string | Идентификатор звонка. | да | 21073-1 |

Данные ответа: отсутствуют.

## Описание API сервиса

**Контроллер конференц - связи**

|  |  |
| --- | --- |
| ***Наименование запроса*** | ***Описание*** |
| POST/v1/make2AbonentsConference | Создать конференцию на 2-х абонентов через звонки с CTI-порта (виртуального номера). |

**Контроллер телефонии**

|  |  |
| --- | --- |
| ***Наименование запроса*** | ***Описание*** |
| POST/v1/addCall/{callId} | Подключение к звонку дополнительного участника. |
| POST/v1/answerCall/{callId} | Ответ на звонок. |
| POST/v1/endCall/{callId} | Завершение звонка. |
| POST/v1/holdCall/{callId} | Удержание звонка. |
| POST/v1/makeCall | Звонок с телефона сотрудника организации. |
| POST/v1/returnToCall/{callId} | Возобновление звонка. |
| POST/v1/transferCall/{callId} | Перевод звонка с телефона сотрудника организации. |

# Сервис хранения файлов

## Общие сведения

Сервис предназначен для хранения файлов в едином информационном пространстве.

Может использоваться для передачи больших файлов между отделами, путем отправки идентификатора файла в хранилище вместо самого файла.

Хранение файлов организовано с помощью разбивки на бакеты.

## Авторизация

Запросы защищены аутентификацией. Аутентификация осуществляется по стандарту Oauth2 c JWT. Токен должен передаваться в заголовке Autorization.

realm: **<REALM>**

role (группа в AD):

APP\_MicroServices\_ BCSFS \_READ\_BUCKET - Чтение файлов в бакете.

APP\_MicroServices\_ BCSFS \_WRITE\_BUCKET - Запись, изменение, удаление файлов в бакете.

APP\_MicroServices\_ BCSFS \_WRITE\_ACL\_BUCKET - Изменение прав доступа к объекту.

APP\_MicroServices\_ BCSFS \_READ\_ACL\_BUCKET - Чтение списка прав доступа к объекту.

## Информационная архитектура

На Рисунок 8.1 представлена схема интеграции сервиса хранения файлов с приложениями. В Таблица 8.1 перечислены запросы, осуществляемые к микросервису, и дано их описание.

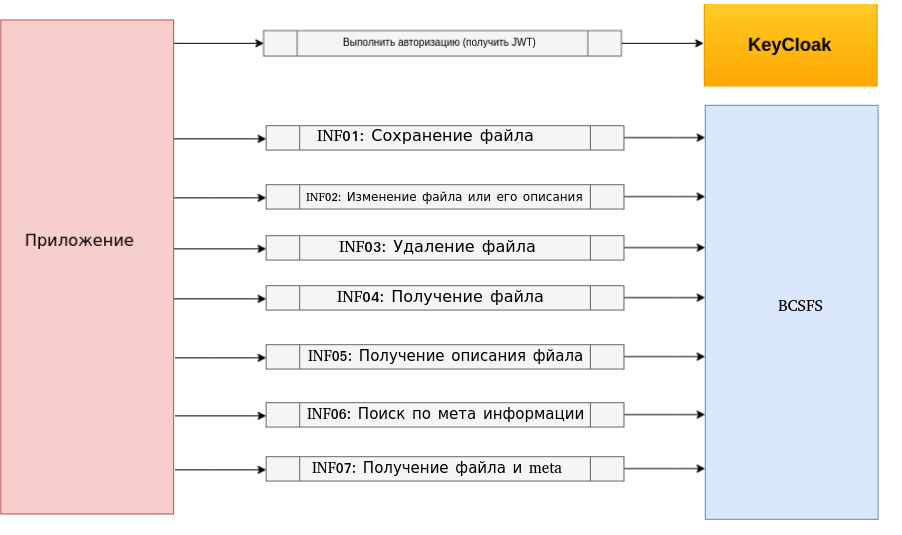


Рисунок .. Интеграция сервиса хранения файлов.

Таблица 8.1. Описание запросов.

|  | **Код** | **Название** | **Отправитель** | **Получатель** | **Описание** |
| --- | --- | --- | --- | --- | --- |
| 1 | INF01 | Сохранение файла | Приложение | BCSFS | POST запрос на сохранения файла и его описания в бакет. |
| 2 | INF02 | Изменение файла или его описания | Приложение | BCSFS | PUT для изменения файла или его описания. |
| 3 | INF03 | Удаление файла | Приложение | BCSFS |  |
| 4 | INF04 | Получение файла | Приложение | BCSFS | GET запрос для получения файла по его идентификатору. |
| 5 | INF05 | Получение описания файла | Приложение | BCSFS | GET запрос для получения информации по файлу по его идентификатору. |
| 6 | INF06 | Поиск по meta информации | Приложение | BCSFS | GET запрос информации о файлах по мета информации (внутри бакета и по всем бакетам). |
| 7 | INF07 | Получения файла с meta | Приложение | BCSFS | GET запрос для получения файла и метаинформации по его идентификатору. |

## Компонентная архитектура

На Рисунок 8.2 представлена компонентная архитектура сервиса хранения файлов.

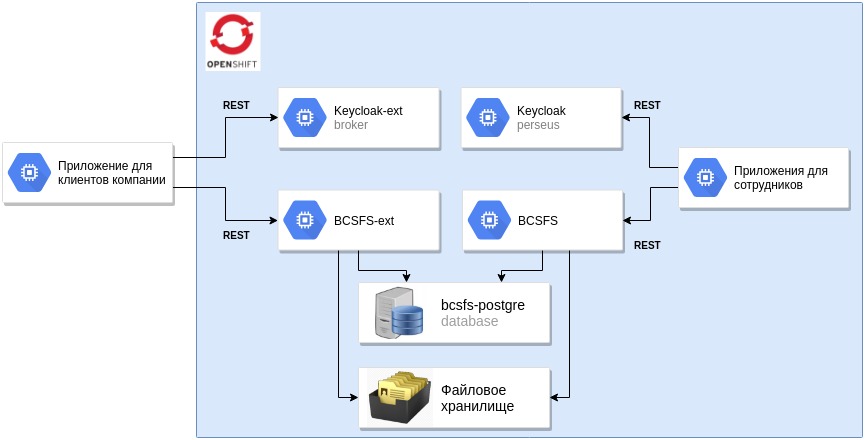


Рисунок .. Компоненты сервиса хранения файлов.

## Описание API

File - controller

|  |  |
| --- | --- |
| ***Наименование запроса*** | ***Описание*** |
| GET[​/v1​/file​/{bucket-name}​/{id}](https://bcsfs.okd.t-global.bcs/swagger-ui/index.html?configUrl=/v3/api-docs/swagger-config#/file-controller/getFile) | Получение файла |
| PUT[​/v1​/file​/{bucket-name}​/{id}](https://bcsfs.okd.t-global.bcs/swagger-ui/index.html?configUrl=/v3/api-docs/swagger-config#/file-controller/replaceFile) | Замена файла |
| DELETE[​/v1​/file​/{bucket-name}​/{id}](https://bcsfs.okd.t-global.bcs/swagger-ui/index.html?configUrl=/v3/api-docs/swagger-config#/file-controller/deleteFile) | Удаление файла |
| GET[​/v1​/file​/{bucket-name}​/{id}​/with-meta](https://bcsfs.okd.t-global.bcs/swagger-ui/index.html?configUrl=/v3/api-docs/swagger-config#/file-controller/getFileWithMeta) | Получение файла вместе с метаданными |
| POST[​/v1​/file​/{bucket-name}](https://bcsfs.okd.t-global.bcs/swagger-ui/index.html?configUrl=/v3/api-docs/swagger-config#/file-controller/uploadNewFile) | Отправка файла |

[File-info-controller](https://bcsfs.okd.t-global.bcs/swagger-ui/index.html?configUrl=/v3/api-docs/swagger-config#/file-info-controller)

|  |  |
| --- | --- |
| ***Наименование запроса*** | ***Описание*** |
| GET[​/v1​/file-info​/{bucket-name}​/{id}](https://bcsfs.okd.t-global.bcs/swagger-ui/index.html?configUrl=/v3/api-docs/swagger-config#/file-info-controller/getFileMeta) | Информация о файле в каталоге |
| POST[​/v1​/file-info​/search](https://bcsfs.okd.t-global.bcs/swagger-ui/index.html?configUrl=/v3/api-docs/swagger-config#/file-info-controller/searchFileMeta) | Поиск информации о файлах в перечисленных бакетах по метаданным, если для них есть доступ. |
| GET[​/v1​/file-info​/{bucket-name}](https://bcsfs.okd.t-global.bcs/swagger-ui/index.html?configUrl=/v3/api-docs/swagger-config#/file-info-controller/searchFileMeta_1) | Поиск информации о файлах в каталоге. |

# Сервис ЭЦП

## Общие сведения

Сервис предназначен для формирования ЭЦП (электронно - цифровой подписи) для файлов формата pdf, используемых в организации. Сервис использует стандартные методы шифрования на основе алгоритмов MD5 и RSA.

Сервис используется для подписания файлов в формате pdf размером до 20МБ (возможно расширение по согласованию). Сервис предоставляет возможность подписания как невидимой, так и видимой подписью (штампом).

Штамп подписи содержит информацию:

* номер сертификата;
* срок действия сертификата;
* ФИО владельца;
* дата и время подписания.

Сервис предоставляет возможность выполнения операций:

* подписать файл в формате pdf электронной подписью;
* получить хэш последней невидимой подписи;
* получить список ЭЦП доступных пользователю.

## Ограничения

Перед использованием сервиса ЭЦП сотруднику необходимо убедится, что файлы:

1. являются файлами в формате pdf;
2. размер файла не превышает 20 Мб;
3. файлы не содержат пароля;
4. документ с видимой подписью имеет формат книжной ориентации;
5. файл содержит видимую подпись не более чем на 30 страницах.

Сервис предназначен для подписания файлов:

* В формате pdf
* Размером не более 20 Мб
* Не защищенных паролем

Для корректного отображения видимой подписи необходимо помнить:

* подпись на каждом листе добавляется на документы не более 30 страниц;
* штамп подписи корректно отображается на документах с книжной ориентацией;

## Безопасность

Все запросы защищены аутентификацией. Аутентификация осуществляется по стандарту OAuth2 c JWT. Токен передается в заголовке Autorization.

Токен выдается на основе учетных данных пользователя в домене организации с помощью сервиса Keycloak. Группы AD в которых содержится пользователь напрямую представляются в виде ролей в токене.

Право пользователя на использование подписи **sign\_name** подтверждается следующим образом:

1. пользователь добавлен в группу **ACTIVE Directory** (роль в токене)с именем**sign\_name;**
2. пароль для подписи **sign\_name** помещается в систему управления секретами Vault и запрошевается сервисом ЭЦП при получении запроса на подписание документа.

## Получение хэш-а последней невидимой подписи

Хэш - функция последней невидимой подписи передается в заголовке **md5-hash.**

## Проверка подлинности документа

Проверка подписи в документе осуществляется средствами Adobe Reader версии 11 и выше. С более подробной информацией о проверке подлинности документа вы можете ознакомится по ссылке <https://helpx.adobe.com/ru/acrobat/using/validating-digital-signatures.html>.

Для проверки подписей, выданных организацией, на компьютере должен быть установлен корневой сертификат удостоверяющего центра (УЦ) этой организации. Необходимо добавить данный сертификат в  Adobe Reade к надежным сертификатам (Рисунок 9.1).

Пример проверок на подлинность подписи в документах приведен на Рисунок 9.2 и Рисунок 9.3.

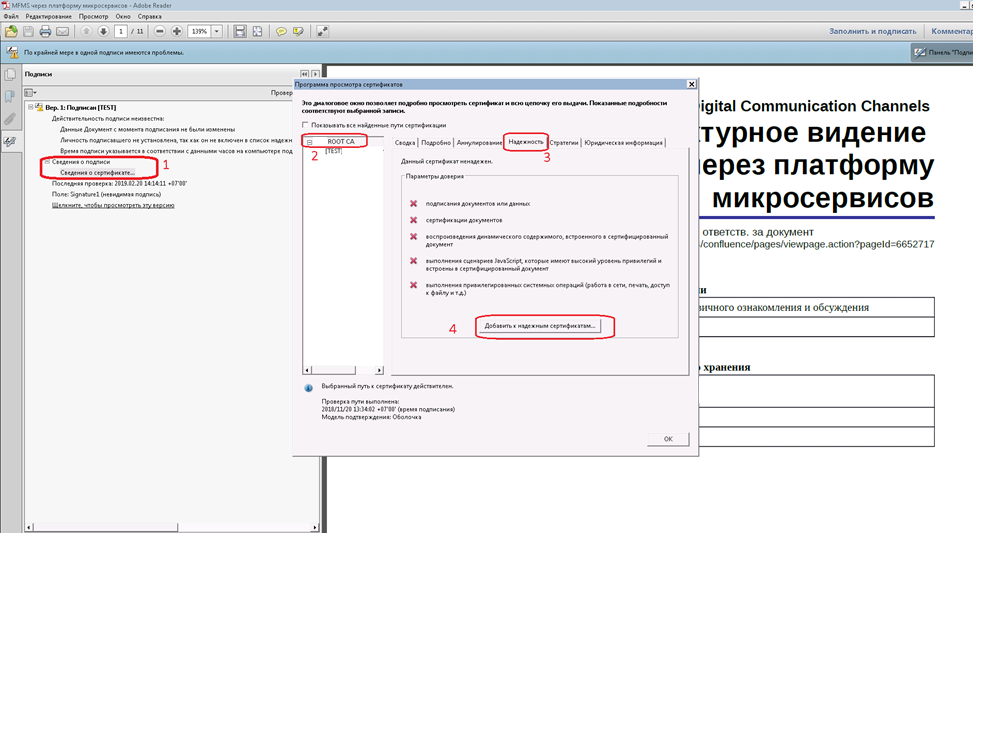


Рисунок .. Добавление корневого сертификата.

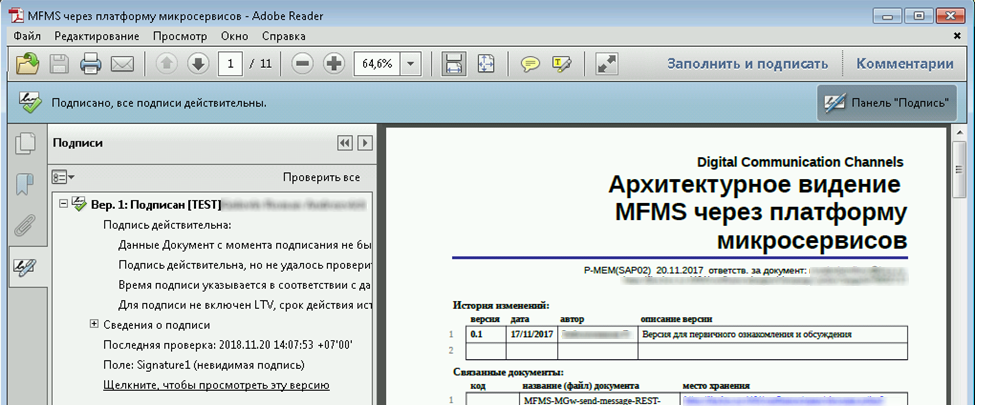


Рисунок .. Подпись действительна.

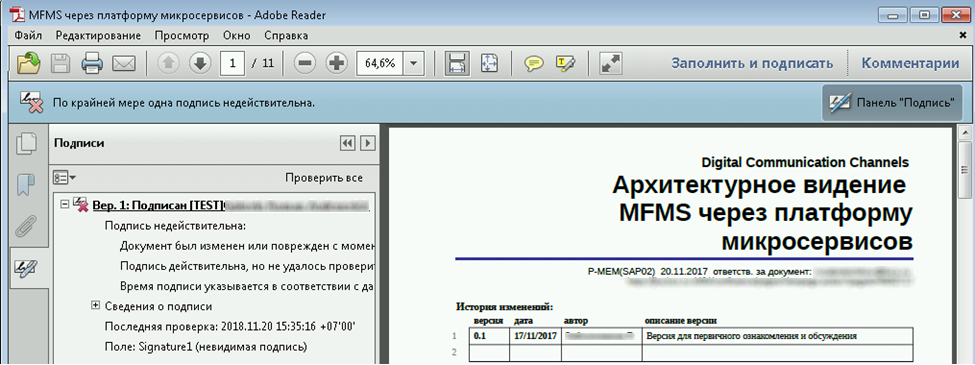


Рисунок .. Подпись недействительна. Документ был изменен после подписания.

## Интеграционная схема сервиса ЭЦП

Интеграционная схема сервиса ЭЦП представлена на Рисунок 9.4.

Интеграция сервиса ЭЦП с другими сервисами и приложениями построена следующим образом:

1. подписант обращается в Удостоверяющий Центр организации с просьбой о выпуске цифровой подписи. УЦ организации выдает цифровую подпись (pfx файл  **sign\_name**.pfx и пароль);
2. pfx файл цифровой подписи загружается в сервис ЭЦП;
3. пароль цифровой подписи загружается в защищенное хранилище Vault;
4. приложение - подписант получает JWT токен в сервисе KeyCloak;
5. приложение - подписант отправляет в сервис ЭЦП запрос на подписание документа. В составе запроса передаются:

* файл, который необходимо подписать (файл в формате pdf размером до 20 Мб);
* идентификатор подписи **sign\_name;**
* JWT токен. При разборе токена сервис ЭЦП удостоверяет право пользователя на использование подписи **sign\_name**. Проверяет наличие роли **sign\_name;**

1. сервис ЭЦП запрашивает пароль для подписи **sign\_name** из хранилища секретов Vault;
2. сервис ЭЦП возвращает инициатору подписания документ, заверенный подписью **sign\_name.**

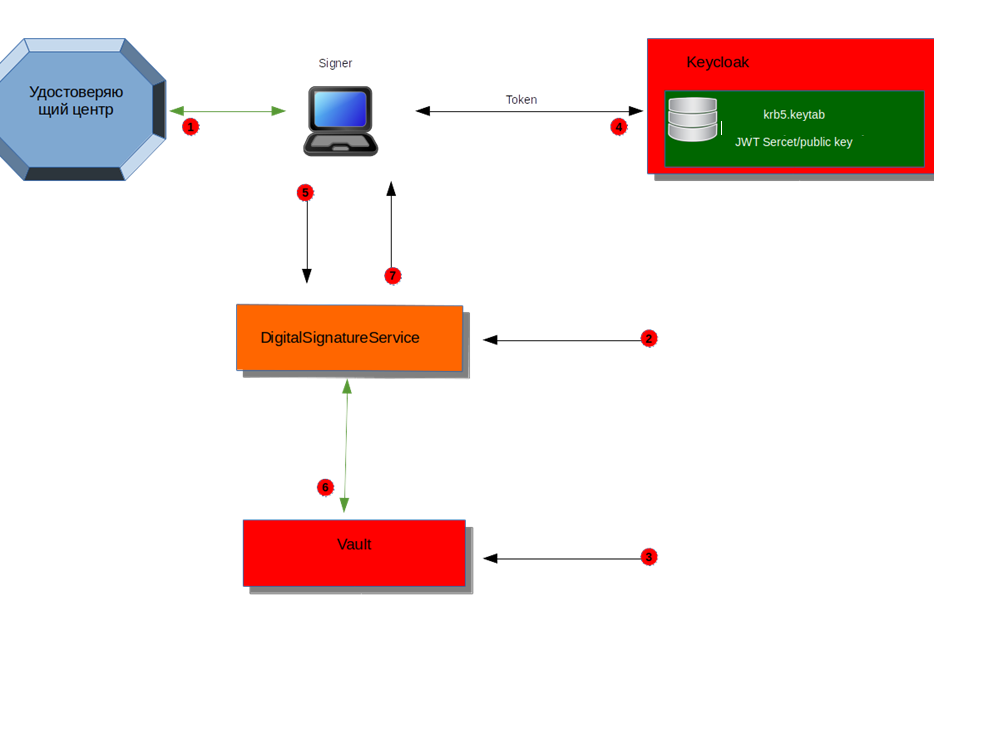


Рисунок .. Интеграционная схема сервиса ЭЦП.

## Описание API сервиса подписания файлов

|  |  |
| --- | --- |
| ***Наименование запроса*** | ***Описание запроса*** |
| GET/api/v2/get-set-of-roles | Получение списка доступных ролей |
| POST/api/v2/invisibleSignature | Невидимая подпись |
| POST/api/v2/signature-hash | Hash подписи |
| POST/api/v2/signatures-info | Получить информацию о сигнатурах |
| POST/api/v2/visibleSignature | Видимая подпись |