

АО «СИГНАЛ-КОМ»

ПРОГРАММНЫЙ КОМПЛЕКС  
SIGNAL-COM AUTHKEY  
СЕРВЕР ЭЛЕКТРОННОЙ ПОДПИСИ  
Версия 1.0

Руководство программиста

ШКНР.00072-01 33 01  
Листов 32

---

## **АННОТАЦИЯ**

Средство имитозащиты с аутентификацией - программный комплекс «Signal-COM AuthKey» предназначено для обеспечения целостности и аутентификации данных, а также для аутентификации источника данных с использованием симметричных криптографических алгоритмов.

Настоящий документ содержит описание прикладного программного интерфейса (API) сервера электронной подписи (далее «Signal-COM AuthKey Server»), компонента программного комплекса «Signal-COM AuthKey», предназначенного для обслуживания ключей электронной подписи и для проверки электронной подписи.

---

## СОДЕРЖАНИЕ

Аннотация .....	2
Содержание .....	3
1. Назначение и условия применения .....	4
1.1. Список сокращений .....	4
1.2. Термины и определения .....	4
2. Характеристика программы .....	5
3. Обращение к программе .....	6
3.1. Методы аутентификации .....	6
3.2. Работа с токенами .....	6
3.3. Ключи диверсификации .....	6
3.3.1. Регистрация ключа диверсификации .....	6
3.3.2. Генерация ключа диверсификации .....	7
3.3.3. Активация ключа диверсификации .....	8
3.3.4. Деактивация ключа диверсификации .....	9
3.3.5. Удаление ключа диверсификации .....	9
3.3.6. Получение списка ключей диверсификации .....	10
3.3.7. Получение параметров ключа диверсификации .....	12
3.3.8. Изменение параметров ключа диверсификации .....	12
3.3.9. Удаление записи ключа диверсификации .....	13
3.4. Клиенты .....	13
3.4.1. Регистрация клиента .....	13
3.4.2. Получение списка клиентов .....	14
3.4.3. Получение параметров клиента .....	15
3.4.4. Изменение параметров клиента .....	15
3.4.5. Удаление записи клиента .....	16
3.5. Ключи ЭП клиентов .....	16
3.5.1. Создание ключа ЭП клиента .....	16
3.5.2. Получение списка ключей ЭП клиентов .....	17
3.5.3. Получение параметров ключа ЭП клиента .....	18
3.5.4. Получение ключа ЭП клиента .....	19
3.5.5. Изменение параметров ключа ЭП клиента .....	19
3.5.6. Удаление ключа ЭП клиента .....	19
3.6. Проверка ЭП документа .....	20
3.7. Получение данных о состоянии сервера ЭП .....	20
3.8. Лицензии .....	20
3.8.1. Загрузка лицензии .....	20
3.8.2. Получение списка лицензий .....	21
3.8.3. Получение параметров лицензии .....	21
3.8.4. Получение параметров актуальной лицензии .....	22
3.8.5. Удаление лицензии .....	22
3.9. Структуры и типы данных .....	22
4. Входные и выходные данные .....	28
5. Сообщения .....	29
6. Дополнительная документация .....	31
Литература .....	32

---

## 1. НАЗНАЧЕНИЕ И УСЛОВИЯ ПРИМЕНЕНИЯ

Программный комплекс «Signal-COM AuthKey» предназначен для обеспечения целостности и аутентификации данных, а также для аутентификации источника данных с использованием симметричных криптографических алгоритмов.

Программный комплекс «Signal-COM AuthKey» может использоваться в информационных системах, предназначенных для хранения и обработки персональных данных, конфиденциальной, служебной, коммерческой и другой информации, не содержащей сведений, составляющих государственную тайну, а также для обмена такой информацией и обеспечения юридической значимости электронных документов.

Сервер ЭП «Signal-COM AuthKey Server» входит в состав программного комплекса «Signal-COM AuthKey» и предназначен для выполнения следующих функций:

- ведение реестра идентификаторов клиентов;
- ведение реестра идентификаторов ключей ЭП клиентов;
- генерация симметричного ключа ЭП клиента;
- выдача по запросу ключа ЭП клиента, с возможностью парольной защиты ключа;
- выдача по запросу информации о ключе ЭП клиента (дата создания, дата последнего использования, статус и т.п.);
- проверка ЭП документа с использованием ключа ЭП клиента.

«Signal-COM AuthKey Server» реализован в виде веб-сервиса и предоставляет программный интерфейс по протоколу REST.

### 1.1. Список сокращений

В настоящем руководстве используются следующие сокращения:

- ПК – программный комплекс;
- ЭП – электронная подпись;
- API - прикладной программный интерфейс;
- HTTP - HyperText Transfer Protocol;
- JSON - JavaScript Object Notation;
- REST - Representational State Transfer;
- RFC – Request for Comments.

### 1.2. Термины и определения

В настоящем руководстве используются следующие термины:

- веб-сервис – реализация интерфейса взаимодействия между различными приложениями по протоколу REST;
- ключ электронной подписи – уникальная последовательность символов, предназначенная для создания электронной подписи;
- электронная подпись – информация в электронной форме, которая присоединена к другой информации в электронной форме (подписываемой информации) или иным образом связана с такой информацией и которая используется для определения лица, подписывающего информацию.

---

## 2. ХАРАКТЕРИСТИКА ПРОГРАММЫ

«Signal-COM AuthKey Server» реализован в виде веб-сервиса и предоставляет программный интерфейс по протоколу REST.

В качестве алгоритмов, обеспечивающих целостность и аутентификацию данных, а также аутентификацию источника данных используются алгоритмы HMAC\_GOSTR3411\_2012\_256 и HMAC\_GOSTR3411\_2012\_512, реализованные согласно RFC 2104 (HMAC: Keyed-Hashing for Message Authentication) и Рекомендациям по стандартизации Р 50.1.113 2016 «Информационная технология. Криптографическая защита информации. Криптографические алгоритмы, сопутствующие применению алгоритмов электронной цифровой подписи и функции хэширования».

Для генерации ключей ЭП клиентов используется алгоритм диверсификации KDF\_TREE\_GOSTR3411\_2012\_256 в соответствии с Рекомендациями по стандартизации Р 50.1.113 2016 «Информационная технология. Криптографическая защита информации. Криптографические алгоритмы, сопутствующие применению алгоритмов электронной цифровой подписи и функции хэширования». Для генерации исходного ключа диверсификации используется криптографически стойкий ДСЧ из состава СКЗИ. Предусмотрена опциональная возможность хранения ключа диверсификации в разделённом по пороговой схеме виде.

Алгоритмы HMAC\_GOSTR3411\_2012\_256 и KDF\_TREE\_GOSTR3411\_2012\_256 основаны на использовании хэш-функции ГОСТ Р 34.11-2012 с длиной выходного значения 256 бит. Алгоритм HMAC\_GOSTR3411\_2012\_512 основан на использовании хэш-функции ГОСТ Р 34.11-2012 с длиной выходного значения 512 бит.

Для вычисления отпечатка ключа ЭП используется хэш-функция ГОСТ Р 34.11-2012 с длиной выходного значения 256 бит.

Парольное шифрование ключей ЭП клиентов реализовано с использованием алгоритма ГОСТ Р 34.12-2015 (алгоритм блочного шифрования «Кузнечик») согласно Рекомендациям по стандартизации Р 1323565.1.040–2022 «Информационная технология. Криптографическая защита информации. Парольная защита ключевой информации».

При разработке «Signal-COM AuthKey Server» использовано средство криптографической защиты информации «Signal-COM JCP 3.1», имеющее сертификаты соответствия ФСБ России [7].

### 3. ОБРАЩЕНИЕ К ПРОГРАММЕ

Взаимодействие с «Signal-COM AuthKey Server» осуществляется с помощью программных интерфейсов по протоколу REST.

Для вызова методов сервера ЭП используются следующие типы запросов протокола HTTP: POST, GET, PUT, DELETE.

Запрос и ответ могут содержать тело в формате JSON и кодировке UTF-8 (Content-Type: "application/json; charset=UTF-8").

Бинарные данные в объектах JSON передаются закодированными в формате Base64.

Ответы со статусом ошибки (400 и выше) содержат тело в формате, основанном на RFC 7807 [2].

Все методы API, возвращающие список результатов, поддерживают разбивку на страницы: вывод результатов запросов постранично и отсортированными. Параметры разбивки на страницы («page», «size» и «sort») можно задавать в запросе. Если параметры «page» и «size» не заданы, выдается страница с полным списком результатов. Эти методы поддерживают также фильтрацию результатов.

Операции (вызов методов) веб-сервиса выполняются в синхронном режиме.

#### 3.1. Методы аутентификации

В «Signal-COM AuthKey Server» может использоваться один из следующих методов аутентификации клиента (см. [6]):

- по сертификату X.509;
- basic-аутентификация.

При использовании basic-аутентификации клиентское приложение должно задавать соответствующее значение в заголовке Authorization для каждого запроса.

##### Пример

Authorization: Basic dGVzdDoxMTEx

#### 3.2. Работа с токенами

Ключи диверсификации «Signal-COM AuthKey Server» могут храниться на токенах - отчуждаемых носителях, защищённых ПИН.

Взаимодействие с токенами осуществляется с помощью интерфейса PKCS #11.

Интерфейс PKCS #11 не предоставляет доступ к файловой системе токенов, поэтому для задания каталогов к ключевым контейнерам на токенах в «Signal-COM AuthKey Server» используются следующие правила:

- вместо имени диска задается ключевое слово «pkcs11»;
- после «pkcs11:\» следует имя файла (без расширения) соответствующего модуля интерфейса PKCS #11 (см. [6]);
- после «pkcs11» может следовать логический номер (от 0 и выше) считывающего устройства (если номер считывателя не задан, токен ищется во всех считывателях).

##### Пример

pkcs11:\rtpkcs11ecp

#### 3.3. Ключи диверсификации

##### 3.3.1. Регистрация ключа диверсификации

**POST /master\_key**

**Параметры**

*Body*

МIME	Схема	Описание
application/json	MasterKeyDto [Таблица 1]	Параметры ключа диверсификации.

Возвращаемые значения

Код	МIME	Схема	Описание
200	application/json	MasterKeyDto	Операция завершена успешно. Возвращаются параметры нового ключа.
400	application/problem+json	ProblemDetail [2]	Некорректный запрос.

Примечания

Данный метод не выполняет генерацию ключа диверсификации. Описание метода генерации ключа диверсификации приводится в п. 3.3.2.

Пример

```
{
  "id": 1000,
  "psePath": "pkcs11:/rtpkcs11ecp",
  "n": 0,
  "t": 3,
  "created": "2024-02-27T07:59:12.879Z",
  "lastModified": "2024-02-27T07:59:12.879Z",
  "notBefore": "2024-02-27T07:59:12.879Z",
  "notAfter": "2025-02-27T07:59:12.879Z",
  "status": "AVAILABLE",
  "activated": false
}
```

3.3.2. Генерация ключа диверсификации

POST /master\_key/{id}/generation

Параметры

Path

Имя	Тип	Описание
id	integer <int64>	Идентификатор ключа диверсификации.

Body

МIME	Схема	Описание
application/json	MasterKeyDto [Таблица 1]	Параметры ключа диверсификации.

Возвращаемые значения

Код	МIME	Схема	Описание
200			Метод завершен успешно.
400	application/problem+json	ProblemDetail [2]	Некорректный запрос.

404	application/problem+json	ProblemDetail	Ключ диверсификации не найден.
500	application/problem+json	ProblemDetail	Ошибка генерации ключа (или разделяемой части ключа) диверсификации.

#### Примечания

Перед генерацией ключа (или разделяемой части ключа) диверсификации он должен быть зарегистрирован (см. п. 3.3.1).  
Если ключ диверсификации должен храниться в разделенном по пороговой схеме виде, метод генерации для этого ключа необходимо выполнить  $n$  раз (т.е. сгенерировать  $n$  разделяемых частей ключа диверсификации), где  $n$  должно быть не меньше  $t$  (см. Таблица 1).  
Если ключ (или разделяемая часть ключа) диверсификации должен размещаться на отчуждаемом ключевом носителе, и этот носитель не найден в считывателе, будет возвращен код ошибки 500 и объект ProblemDetail с соответствующей диагностикой. В этом случае необходимо вставить ключевой носитель в считыватель и повторить запрос.

#### Пример

```
{
  "id": 1000,
  "psePath": "pkcs11:/rtpkcs11ecp",
  "n": 5,
  "t": 3,
  "created": "2024-02-27T07:59:12.879Z",
  "lastModified": "2024-02-27T07:59:12.879Z",
  "notBefore": "2024-02-27T07:59:12.879Z",
  "notAfter": "2025-02-27T07:59:12.879Z",
  "status": "AVAILABLE",
  "activated": false
}
```

### 3.3.3. Активация ключа диверсификации

**POST /master\_key/{id}/activation**

#### Параметры

*Path*

Имя	Тип	Описание
id	integer <int64>	Идентификатор ключа диверсификации.

*Body*

MIME	Схема	Описание
application/json	MasterKeyDto [Таблица 1]	Параметры ключа диверсификации.

#### Возвращаемые значения

Код	MIME	Схема	Описание
200			Метод завершен успешно.
400	application/problem+json	ProblemDetail [2]	Некорректный запрос.



404	application/problem+json	ProblemDetail	Ключ диверсификации не найден.
500	application/problem+json	ProblemDetail	Ошибка активации ключа диверсификации.

**Примечания**

Перед активацией ключа диверсификации он должен быть сгенерирован (см. п. 3.3.2). Если ключ диверсификации хранится в разделенном по пороговой схеме виде, метод активации для этого ключа необходимо выполнить  $t$  раз (см. Таблица 1). В этом случае необходимо контролировать значение поля `activated` возвращаемого объекта `MasterKeyDto`. Если ключ (или разделяемая часть ключа) диверсификации размещается на отчуждаемом ключевом носителе, и этот носитель не найден в считывателе, будет возвращен код ошибки 500 и объект `ProblemDetail` с соответствующей диагностикой. В этом случае необходимо вставить ключевой носитель в считыватель и повторить запрос.

**Пример**

```
{
  "id": 1000,
  "psePath": "pkcs11:/rtpkcs11ecp",
  "n": 5,
  "t": 3,
  "created": "2024-02-27T07:59:12.879Z",
  "lastModified": "2024-02-27T07:59:12.879Z",
  "notBefore": "2024-02-27T07:59:12.879Z",
  "notAfter": "2025-02-27T07:59:12.879Z",
  "status": "AVAILABLE",
  "activated": true
}
```

**3.3.4. Деактивация ключа диверсификации****DELETE /master\_key/{id}/deactivation****Параметры***Path*

Имя	Тип	Описание
id	integer <int64>	Идентификатор ключа диверсификации.

**Возвращаемые значения**

Код	MIME	Схема	Описание
200			Операция завершена успешно.
404	application/problem+json	ProblemDetail [2]	Ключ диверсификации не найден.

**3.3.5. Удаление ключа диверсификации****POST /master\_key/{id}/deleting****Параметры***Path*

Имя	Тип	Описание
id	integer <int64>	Идентификатор ключа диверсификации.

Body

MIME	Схема	Описание
application/json	MasterKeyDto [Таблица 1]	Параметры ключа диверсификации.

Возвращаемые значения

Код	MIME	Схема	Описание
200			Метод завершен успешно.
400	application/problem+json	ProblemDetail [2]	Некорректный запрос.
404	application/problem+json	ProblemDetail	Ключ диверсификации не найден.
500	application/problem+json	ProblemDetail	Ошибка удаления ключа (или разделяемой части ключа) диверсификации.

Примечания

Перед удалением ключа (или разделяемой части ключа) диверсификации он должен быть сгенерирован (см. п. 3.3.2) и активирован (см. п. 3.3.3).

Если ключ диверсификации хранится в разделенном по пороговой схеме виде, метод удаления для этого ключа необходимо выполнить  $k$  раз, где  $k$  должно быть не меньше  $t$  и не больше  $n$  (см. Таблица 1).

Если ключ (или разделяемая часть ключа) размещается на отчуждаемом ключевом носителе, и этот носитель не найден в считывателе, будет возвращен код ошибки 500 и объект ProblemDetail с соответствующей диагностикой. В этом случае необходимо вставить ключевой носитель в считыватель и повторить запрос.

Пример

```
{
  "id": 1000,
  "psePath": "pkcs11:rtpkcs11ecp",
  "n": 4,
  "t": 3,
  "created": "2024-02-27T07:59:12.879Z",
  "lastModified": "2024-02-27T07:59:12.879Z",
  "notBefore": "2024-02-27T07:59:12.879Z",
  "notAfter": "2025-02-27T07:59:12.879Z",
  "status": "AVAILABLE",
  "activated": true
}
```

### 3.3.6. Получение списка ключей диверсификации

GET /master\_key

Параметры

Query

Имя	Тип	Описание
page	integer	Индекс страницы, начиная с нуля.
size	integer	Размер возвращаемой страницы.
sort	array[string]	Критерий сортировки в формате: property,(asc desc). Порядок сортировки по умолчанию — по возрастанию. Поддерживается несколько критериев сортировки.
filter	string	Критерии фильтрации.

**Возвращаемые значения**

Код	MIME	Схема	Описание
200	application/json	PageObject [Таблица 2]	Операция завершена успешно. Возвращается объект страницы, содержащий массив объектов MasterKeyDto [Таблица 1].

**Пример**

```
{
  "content": [
    {
      "id": 1,
      "psePath": "pkcs11:/rutokenecp",
      "n": 5,
      "t": 3,
      "created": "2024-02-16T12:41:38.712+00:00",
      "lastModified": "2024-02-16T12:41:38.712+00:00",
      "notBefore": "2024-02-16T12:40:39.334+00:00",
      "notAfter": "2025-02-16T12:40:39.334+00:00",
      "status": "ACTIVE",
      "activated": true
    },
    {
      "id": 2,
      "psePath": "c:/pse_path",
      "n": null,
      "t": null,
      "created": "2024-02-16T12:42:41.078+00:00",
      "lastModified": "2024-02-16T12:42:41.078+00:00",
      "notBefore": "2024-02-16T12:40:39.334+00:00",
      "notAfter": "2025-02-16T12:40:39.334+00:00",
      "status": "ACTIVE",
      "activated": false
    }
  ],
  "pageable": {
    "sort": {
      "empty": true,
      "sorted": false,
      "unsorted": true
    },
    "offset": 0,
```

```
"pageNumber": 0,  
"pageSize": 20,  
"paged": true,  
"unpaged": false  
},  
"totalPages": 1,  
"totalElements": 2,  
"last": true,  
"size": 20,  
"number": 0,  
"sort": {  
  "empty": true,  
  "sorted": false,  
  "unsorted": true  
},  
"numberOfElements": 2,  
"first": true,  
"empty": false  
}
```

### 3.3.7. Получение параметров ключа диверсификации

**GET /master\_key/{id}**

#### Параметры

*Path*

Имя	Тип	Описание
id	integer <int64>	Идентификатор ключа.

#### Возвращаемые значения

Код	MIME	Схема	Описание
200	application/json	MasterKeyDto [Таблица 1]	Операция завершена успешно. Возвращаются параметры ключа.
404	application/problem+json	ProblemDetail [2]	Ключ не найден.

### 3.3.8. Изменение параметров ключа диверсификации

**PUT /master\_key/{id}**

#### Параметры

*Path*

Имя	Тип	Описание
id	integer <int64>	Идентификатор ключа.

*Body*

MIME	Схема	Описание
application/json	MasterKeyDto [Таблица 1]	Параметры ключа

#### Возвращаемые значения

Код	MIME	Схема	Описание
-----	------	-------	----------

200			Операция завершена успешно.
400	application/problem+json	ProblemDetail [2]	Некорректный запрос.
404	application/problem+json	ProblemDetail	Ключ не найден.

3.3.9. Удаление записи ключа диверсификации

DELETE /master\_key/{id}

Параметры

Path

Имя	Тип	Описание
id	integer <int64>	Идентификатор ключа.

Возвращаемые значения

Код	MIME	Схема	Описание
200			Операция завершена успешно.
404	application/problem+json	ProblemDetail [2]	Ключ не найден.

3.4. Клиенты

3.4.1. Регистрация клиента

POST /client

Параметры

Body

MIME	Схема	Описание
application/json	ClientDto [Таблица 5]	Параметры клиента.

Возвращаемые значения

Код	MIME	Схема	Описание
200	application/json	ClientDto	Операция завершена успешно. Возвращаются параметры зарегистрированного клиента.
400	application/problem+json	ProblemDetail [2]	Некорректный запрос.

Пример

```
{
  "id": "client1",
  "created": "2024-02-27T15:01:21.326Z",
  "lastModified": "2024-02-27T15:01:21.326Z",
  "status": "ACTIVE"
}
```

3.4.2. Получение списка клиентов

GET /client

Параметры

Query

Имя	Тип	Описание
page	integer	Индекс страницы, начиная с нуля.
size	integer	Размер возвращаемой страницы.
sort	array[string]	Критерий сортировки в формате: property,(asc desc). Порядок сортировки по умолчанию — по возрастанию. Поддерживается несколько критериев сортировки.
filter	string	Критерии фильтрации.

Возвращаемые значения

Код	MIME	Схема	Описание
200	application/json	PageObject [Таблица 2]	Операция завершена успешно. Возвращается объект страницы, содержащий массив объектов ClientDto [Таблица 5].

Пример

```
{
  "content": [
    {
      "id": " client1",
      "created": "2024-02-16T13:13:27.661+00:00",
      "lastModified": "2024-02-16T13:13:27.661+00:00",
      "status": "ACTIVE"
    },
    {
      "id": " client2",
      "created": "2024-02-16T13:13:37.986+00:00",
      "lastModified": "2024-02-16T13:13:37.986+00:00",
      "status": "ACTIVE"
    },
    {
      "id": " client4",
      "created": "2024-02-26T09:07:16.329+00:00",
      "lastModified": "2024-02-26T09:07:16.329+00:00",
      "status": "ACTIVE"
    }
  ],
  "pageable": {
    "sort": {
      "empty": true,
      "sorted": false,
      "unsorted": true
    },
    "offset": 0,
```

```
"pageNumber": 0,  
"pageSize": 20,  
"paged": true,  
"unpaged": false  
},  
"totalPages": 1,  
"totalElements": 3,  
"last": true,  
"size": 20,  
"number": 0,  
"sort": {  
  "empty": true,  
  "sorted": false,  
  "unsorted": true  
},  
"numberOfElements": 3,  
"first": true,  
"empty": false  
}
```

### 3.4.3. Получение параметров клиента

**GET /client/{id}**

#### Параметры

*Path*

Имя	Тип	Описание
id	string	Идентификатор клиента.

#### Возвращаемые значения

Код	MIME	Схема	Описание
200	application/json	ClientDto [Таблица 5]	Операция завершена успешно. Возвращаются параметры клиента.
404	application/problem+json	ProblemDetail [2]	Клиент не найден.

### 3.4.4. Изменение параметров клиента

**PUT /client/{id}**

#### Параметры

*Path*

Имя	Тип	Описание
id	string	Идентификатор клиента.

*Body*

MIME	Схема	Описание
application/json	ClientDto [Таблица 5]	Параметры клиента

#### Возвращаемые значения

Код	MIME	Схема	Описание
-----	------	-------	----------

200			Операция завершена успешно.
400	application/problem+json	ProblemDetail [2]	Некорректный запрос.
404	application/problem+json	ProblemDetail [2]	Клиент не найден.

### 3.4.5. Удаление записи клиента

**DELETE /client/{id}**

#### Параметры

*Path*

Имя	Тип	Описание
id	string	Идентификатор клиента.

#### Возвращаемые значения

Код	MIME	Схема	Описание
200			Операция завершена успешно.
404	application/problem+json	ProblemDetail [2]	Клиент не найден.

## 3.5. Ключи ЭП клиентов

### 3.5.1. Создание ключа ЭП клиента

**POST /client\_key**

#### Параметры

*Body*

MIME	Схема	Описание
application/json	ClientKeyDto [Таблица 6]	Параметры ключа ЭП клиента.

#### Возвращаемые значения

Код	MIME	Схема	Описание
200	application/json	ClientKeyDto	Операция завершена успешно. Возвращаются параметры созданного ключа ЭП клиента.
400	application/problem+json	ProblemDetail [2]	Некорректный запрос.

#### Пример

```
{
  "id": "5c0a53702dcb51feb6fc444cfcf1b188309d81b519c485fbe29af5770c71ba24",
  "clientId": "client1",
  "masterKeyId": 1,
  "notBefore": "2024-02-27T16:45:53.738Z",
  "notAfter": "2025-02-27T16:45:53.738Z",
  "created": "2024-02-27T16:45:53.738Z",
  "lastModified": "2024-02-27T16:45:53.738Z",
}
```



```
"lastUsed": null,
"status": "ACTIVE",
"deviceFingerprint":
"83971b56524562db500b217f739a1b83a2b2bbddcb75bf3f0793313dd5057a7d",
"keyLength": 256
}
```

3.5.2. Получение списка ключей ЭП клиентов

GET /client\_key

Параметры

Query

Имя	Тип	Описание
page	integer	Индекс страницы, начиная с нуля.
size	integer	Размер возвращаемой страницы.
sort	array[string]	Критерий сортировки в формате: property,(asc desc). Порядок сортировки по умолчанию — по возрастанию. Поддерживается несколько критериев сортировки.
filter	string	Критерии фильтрации.

Возвращаемые значения

Код	MIME	Схема	Описание
200	application/json	PageObject [Таблица 2]	Операция завершена успешно. Возвращается объект страницы, содержащий массив объектов ClientKeyDto [Таблица 6].

Пример

```
{
  "content": [
    {
      "id": "5c0a53702dc51feb6fc444cfcf1b188309d81b519c485fbc29af5770c71ba24",
      "clientId": "client1",
      "masterKeyId": 1,
      "notBefore": "2024-02-16T14:42:57.412+00:00",
      "notAfter": "2025-02-16T14:42:57.412+00:00",
      "created": "2024-02-16T14:47:24.883+00:00",
      "lastModified": "2024-02-16T14:47:24.883+00:00",
      "lastUsed": null,
      "status": "ACTIVE",
      "deviceFingerprint":
      "83971b56524562db500b217f739a1b83a2b2bbddcb75bf3f0793313dd5057a7d",
      "keyLength": 256
    },
    {
      "id": "2d64629527bef0e7076991bbe1b2bea760fcdeabb3a6ee1f9694b1243e7ca37",
      "clientId": "client2",
      "masterKeyId": 2,

```

```
"notBefore": "2024-02-19T12:05:59.283+00:00",
"notAfter": "2025-02-19T12:05:59.283+00:00",
"created": "2024-02-19T12:09:04.283+00:00",
"lastModified": "2024-02-19T12:09:04.283+00:00",
"lastUsed": null,
"status": "ACTIVE",
"deviceFingerprint":
"c851d17c10a9f9d40f9577acf018ce7fa6826bf942a9cd6f94138b0109b3c379",
"keyLength": null
},
],
"pageable": {
  "sort": {
    "empty": true,
    "sorted": false,
    "unsorted": true
  },
  "offset": 0,
  "pageNumber": 0,
  "pageSize": 20,
  "paged": true,
  "unpaged": false
},
"totalPages": 1,
"totalElements": 2,
"last": true,
"size": 20,
"number": 0,
"sort": {
  "empty": true,
  "sorted": false,
  "unsorted": true
},
"numberOfElements": 2,
"first": true,
"empty": false
}
```

3.5.3. Получение параметров ключа ЭП клиента

GET /client\_key/{id}

Параметры

Path

Имя	Тип	Описание
id	string	Идентификатор ключа ЭП клиента.

Возвращаемые значения

Код	MIME	Схема	Описание
200	application/json	ClientKeyDto [Таблица 6]	Операция завершена успешно. Возвращаются параметры ключа.
404	application/problem+json	ProblemDetail [2]	Ключ не найден.

### 3.5.4. Получение ключа ЭП клиента

**GET /client\_key/{id}/body**

#### Параметры

##### Path

Имя	Тип	Описание
id	string	Идентификатор ключа ЭП клиента.

##### Query

Имя	Тип	Описание
password	string	Пароль ключа ЭП клиента в кодировке UTF-8 (опционально).

#### Возвращаемые значения

Код	MIME	Схема	Описание
200	text/plain	string	Операция завершена успешно. Возвращается закодированный ключ.
404	application/problem+json	ProblemDetail [2]	Ключ не найден.

### 3.5.5. Изменение параметров ключа ЭП клиента

**PUT /client\_key/{id}**

#### Параметры

##### Path

Имя	Тип	Описание
id	string	Идентификатор ключа ЭП клиента.

##### Body

MIME	Схема	Описание
application/json	ClientKeyDto [Таблица 6]	Параметры ключа ЭП клиента

#### Возвращаемые значения

Код	MIME	Схема	Описание
200			Операция завершена успешно.
400	application/problem+json	ProblemDetail [2]	Некорректный запрос.
404	application/problem+json	ProblemDetail [2]	Ключ не найден.

### 3.5.6. Удаление ключа ЭП клиента

**DELETE /client\_key/{id}**

#### Параметры

##### Path

Имя	Тип	Описание
id	string	Идентификатор ключа ЭП клиента.

**Возвращаемые значения**

Код	MIME	Схема	Описание
200			Операция завершена успешно.
404	application/problem+json	ProblemDetail [2]	Ключ не найден.

### 3.6. Проверка ЭП документа

**POST /verification**

**Параметры**

*Body*

MIME	Схема	Описание
application/json	VerifyRequestDto [Таблица 7]	Данные для проверки ЭП.

**Возвращаемые значения**

Код	MIME	Схема	Описание
200	application/json	VerificationReportDto [Таблица 8]	Операция завершена успешно. Возвращается отчет с результатом проверки ЭП.
400	application/problem+json	ProblemDetail [2]	Некорректный запрос.

### 3.7. Получение данных о состоянии сервера ЭП

**GET /server\_state**

**Возвращаемые значения**

Код	MIME	Схема	Описание
200	application/json	ServerStateDto [Таблица 9]	Операция завершена успешно. Возвращаются данные о состоянии сервера ЭП.

### 3.8. Лицензии

#### 3.8.1. Загрузка лицензии

**POST /license**

**Параметры**

*Body*

MIME	Схема	Описание
------	-------	----------

application/json	LicenseDto [Таблица 10]	Параметры лицензии.
------------------	-------------------------	---------------------

#### Возвращаемые значения

Код	MIME	Схема	Описание
200	application/json	LicenseDto	Операция завершена успешно. Возвращаются параметры зарегистрированной лицензии.
400	application/problem+json	ProblemDetail [2]	Некорректный запрос.

### 3.8.2. Получение списка лицензий

#### GET /license

#### Параметры

Query

Имя	Тип	Описание
page	integer	Индекс страницы, начиная с нуля.
size	integer	Размер возвращаемой страницы.
sort	array[string]	Критерий сортировки в формате: property,(asc desc). Порядок сортировки по умолчанию — по возрастанию. Поддерживается несколько критериев сортировки.
filter	string	Критерии фильтрации.

#### Возвращаемые значения

Код	MIME	Схема	Описание
200	application/json	PageObject [Таблица 2]	Операция завершена успешно. Возвращается объект страницы, содержащий массив объектов LicenseDto [Таблица 10].

### 3.8.3. Получение параметров лицензии

#### GET /license/{id}

#### Параметры

Path

Имя	Тип	Описание
id	integer <int64>	Идентификатор лицензии.

#### Возвращаемые значения

Код	MIME	Схема	Описание
-----	------	-------	----------

200	application/json	LicenseDto [Таблица 10]	Операция завершена успешно. Возвращаются параметры лицензии.
404	application/problem+json	ProblemDetail [2]	Лицензия не найдена.

### 3.8.4. Получение параметров актуальной лицензии

**GET /license/current**

Возвращаемые значения

Код	MIME	Схема	Описание
200	application/json	LicenseDto [Таблица 10]	Операция завершена успешно. Возвращаются параметры лицензии.
404	application/problem+json	ProblemDetail [2]	Лицензия не найдена.

### 3.8.5. Удаление лицензии

**DELETE /license/{id}**

Параметры

Path

Имя	Тип	Описание
id	integer <int64>	Идентификатор лицензии.

Возвращаемые значения

Код	MIME	Схема	Описание
200			Операция завершена успешно.
404	application/problem+json	ProblemDetail [2]	Лицензия не найдена.

## 3.9. Структуры и типы данных

Таблица 1 MasterKeyDto

Имя	Тип	Описание
id	integer <int64>	Идентификатор ключа диверсификации (только для чтения).
psePath	string	Путь к ключевому носителю.
n	integer <int32>	Общее число сгенерированных разделяемых частей ключа диверсификации (опционально).
t	integer <int32>	Минимальное число разделяемых частей ключа диверсификации для восстановления ключа (опционально).

created	string <date-time>	Время создания ключа (только для чтения).
lastModified	string <date-time>	Время последнего изменения параметров ключа (только для чтения).
notBefore	string <date-time>	Время начала действия ключа (опционально).
notAfter	string <date-time>	Время окончания действия ключа (опционально).
status	string:: AVAILABLE, ACTIVE, BLOCKED, DELETED	Состояние ключа: AVAILABLE – ключ доступен для использования; BLOCKED – ключ временно недоступен для использования; DELETED – ключ удален.
activated	boolean	True, если ключ активирован (загружен в оперативную память сервера ЭП); false, если ключ не активирован.
password	string	Пароль (ПИН) для доступа к ключевому контейнеру (токену).

Таблица 2 PageObject

Имя	Тип	Описание
totalPages	integer <int32>	Возвращает общее количество страниц.
totalElements	integer <int64>	Возвращает общее количество элементов.
size	integer <int32>	Возвращает размер страницы.
content	array[...]	Возвращает массив объектов.
number	integer <int32>	Возвращает номер текущей страницы.
sort	SortObject [Таблица 3]	Возвращает параметры сортировки для страницы.
numberOfElements	integer <int32>	Возвращает количество элементов, находящихся в данный момент на этой странице.
pageable	PageableObject [Таблица 4]	Возвращает параметры разбивки на странице, заданные в запросе.
first	boolean	Возвращает, является ли текущая страница первой.
last	boolean	Возвращает, является ли текущая страница последней.
empty	boolean	Возвращает, является ли текущая страница пустой.

Таблица 3 SortObject

Имя	Тип	Описание
empty	boolean	Возвращает, содержит ли данный объект параметры сортировки.
sorted	boolean	Возвращает, является ли страница отсортированной.
unsorted	boolean	Возвращает значение, противоположное sorted.

Таблица 4 PageableObject

Имя	Тип	Описание
offset	integer <int64>	Возвращает смещение страницы в элементах, которое зависит от базовой страницы и размера страницы.
sort	SortObject [Таблица 3]	Возвращает параметры сортировки для страницы.
paged	boolean	Возвращает, содержит ли текущий объект информацию о разбивке на страницы.
unpaged	boolean	Возвращает значение, противоположное paged.
pageNumber	integer <int32>	Возвращает номер запрашиваемой страницы.
pageSize	integer <int32>	Возвращает размер страницы.

Таблица 5 ClientDto

Имя	Тип	Описание
id	string	Идентификатор клиента.
created	string <date-time>	Время регистрации клиента (только для чтения).
lastModified	string <date-time>	Время последнего изменения параметров клиента (только для чтения).
status	string: ACTIVE, BLOCKED, DELETED	Состояние клиента: ACTIVE – активный клиент; BLOCKED – клиент временно заблокирован; DELETED – клиент удален.

Таблица 6 ClientKeyDto

Имя	Тип	Описание
id	string	Идентификатор ключа ЭП клиента (только для чтения).
clientId	string	Идентификатор клиента.



masterKeyId	integer <int64>	Идентификатор ключа диверсификации.
notBefore	string <date-time>	Время начала действия ключа ЭП (опционально).
notAfter	string <date-time>	Время окончания действия ключа ЭП (опционально).
created	string <date-time>	Время создания ключа ЭП (только для чтения).
lastModified	string <date-time>	Время последнего изменения параметров ключа ЭП (только для чтения).
lastUsed	string <date-time>	Время последнего использования ключа ЭП (только для чтения).
status	string: ACTIVE, BLOCKED, DELETED	Состояние ключа: ACTIVE – активный ключ ЭП клиента; BLOCKED – ключ ЭП клиента временно заблокирован; DELETED – ключ ЭП клиента удален.
deviceFingerprint	string	Отпечаток устройства клиента в шестнадцатеричном виде, длиной 64 символа (опционально).
keyLength	integer <int32>	Длина ключа ЭП (опционально).

Таблица 7 VerifyRequestDto

Имя	Тип	Описание
signature	string <byte>	Подписанный документ или отсоединенная подпись в формате Base64.
document	string <byte>	Исходный документ в формате Base64. Задается только в случае проверки отсоединенной подписи.

Таблица 8 VerificationReportDto

Имя	Тип	Описание
verificationTime	string <date-time>	Время проверки ЭП.
valid	boolean	Является ли ЭП действительной.
clientId	string	Идентификатор клиента.
clientKeyId	string	Идентификатор ключа ЭП клиента.
deviceFingerprint	string	Отпечаток устройства клиента (опционально).
digestAlgorithm	string	Идентификатор хэш-алгоритма ЭП (опционально).
macAlgorithm	string	Идентификатор MAC-алгоритма ЭП (опционально).

signingTime	string <date-time>	Время создания ЭП (опционально).
messageSequenceNumber	string	Десятичное строковое представление порядкового номера сообщения (опционально).
serverNonce	string <byte>	Случайное сеансовое значение, формируемое сервером, в формате Base64 (опционально).
formatOK	boolean	Является ли корректным формат подписанного сообщения.
contentTypeOK	boolean	Является ли корректным тип содержимого подписанного сообщения.
cmsAlgorithmProtectionOK	boolean	Совпадает ли значение атрибута cmsAlgorithmProtection [5] с значением в незащищенной части.
messageDigestOK	boolean	Является ли корректным хэш-значение ЭП.
keyIdentifierOK	boolean	Совпадает ли значение атрибута clientKeyId с значением в незащищенной части.
deviceFingerprintOK	boolean	Использовался ли при проверке параметр deviceFingerprint и прошла ли проверка успешно.
macOK	boolean	Является ли корректным значение MAC-значение ЭП.
contentDigest	string <byte>	Хэш-значение ЭП.
macValue	string <byte>	MAC-значение ЭП.

Таблица 9 ServerStateDto

Имя	Тип	Описание
clientsNumber	integer <int64>	Количество клиентов.
clientKeysNumber	integer <int64>	Количество ключей ЭП клиентов.
clientDevicesNumber	integer <int64>	Количество устройств клиентов.

Таблица 10 LicenseDto

Имя	Тип	Описание
id	integer <int64>	Идентификатор лицензии (только для чтения).
file	string <byte>	Файл лицензии в формате Base64.
manufacturer	string	Издатель лицензии (опционально).
owner	string	Владелец лицензии (опционально).
label	string	Метка лицензии (опционально).
version	string	Версия лицензии (опционально).
generated	string <date-time>	Время создания лицензии (опционально).
notBefore	string <date-time>	Время начала действия лицензии (опционально).

notAfter	string <date-time>	Время окончания действия лицензии (опционально).
clientMax	integer <int64>	Максимальное количество клиентов (опционально).
clientKeyMax	integer <int64>	Максимальное количество ключей ЭП клиентов (опционально).
deviceMax	integer <int64>	Максимальное количество устройств клиентов (опционально).
clientDeviceMax	integer <int32>	Максимальное количество устройств у одного клиента (опционально).

#### **4. ВХОДНЫЕ И ВЫХОДНЫЕ ДАННЫЕ**

Входные и выходные данные методов веб-сервиса «Signal-COM AuthKey Server» представлены в виде JSON-структур, подробное описание которых приведено в настоящем документе (см. п. 3.9).

## 5. СООБЩЕНИЯ

Сообщения об ошибках при обращении к методам веб-сервиса «Signal-COM AuthKey Server» содержатся в JSON-структуре ProblemDetail (см. [2]). Список значений поля detail и их описание приведено в следующей таблице:

Значение	Описание
Объект с заданным идентификатором не найден: %s	
Значение параметра не установлено: %s	
Заданное значение параметра уже используется: %s	
Недопустимый статус клиента: %s	
Недопустимый статус ключа клиента: %s	
Недопустимый статус ключа диверсификации: %s	
Неизвестный хэш-алгоритм	
Объект с заданным идентификатором был удален: %s	
Недопустимый переход состояния: %1\$s->%2\$s	
Количество клиентских ключей не может превышать %d	Превышен лимит ключей клиентов, заданный в лицензии разработчика.
Количество клиентских устройств не может превышать %d	Превышен лимит клиентских устройств, заданный в лицензии разработчика.
Количество устройств у одного клиента не может превышать %d	Превышен лимит устройств у одного клиента, заданный в лицензии разработчика.
Количество клиентов не может превышать %d	Превышен лимит клиентов, заданный в лицензии разработчика.
Актуальная лицензия не найдена	
Срок действия лицензии истек	
Ошибка проверки подписи лицензии	
Неверный формат лицензии	
Лицензия уже существует	Попытка повторной загрузки уже загруженной лицензии.
Пользователю '%s' доступ запрещен	
Ошибка генерации ключевого контейнера	
Ошибка загрузки части разделяемого секрета	
Ошибка записи части разделяемого секрета	
Ошибка удаления ключевого контейнера	

Требуется пароль для доступа к ключевому контейнеру	
Неверный пароль ключевого контейнера	
Ключевой контейнер не инициализирован	
Вставьте ключевой носитель	
Вставьте ключевой носитель номер %d	
Ключевой контейнер уже существует	
Ключевой контейнер не найден	
Ключ диверсификации не активирован: %s	Для выполнения данной операции необходимо активировать ключ диверсификации.
Токен заблокирован	
Неверный ПИН токена	
Токен не подключен	
Ошибка токена	
Токен не поддерживает требуемый механизм	
Ошибка при восстановлении ключа диверсификации	
Часть разделяемого секрета не соответствует секрету	
Восстановленный разделяемый секрет не соответствует ожидаемому	
Дублирование части разделяемого секрета	Заданная часть разделяемого секрета уже была загружена.
Ошибка генерации ключа клиента	Внутренняя ошибка сервера.
Ошибка декодирования подписанного сообщения	Некорректный формат подписанного сообщения.
Ошибка чтения документа	Задан некорректный документ для проверки ЭП.
Ключ для проверки подписи не найден	
Ошибка проверки подписи	
Ошибка декодирования отпечатка устройства	Некорректный формат отпечатка устройства (строка в шестнадцатеричном виде, длиной 64 символа).

---

## **6. ДОПОЛНИТЕЛЬНАЯ ДОКУМЕНТАЦИЯ**

По умолчанию «Signal-COM AuthKey Server» предоставляет конечные точки для протокола OpenAPI и графического интерфейса Swagger (см. [6]).

## ЛИТЕРАТУРА

1. Housley, R., Cryptographic Message Syntax, RFC 5652, September 2009.
2. M. Nottingham, Akamai, E. Wilde, "Problem Details for HTTP APIs", RFC 7807, March 2016.
3. Федеральный закон № 63-ФЗ от 06.04.2011 «Об электронной подписи».
4. M. Nystrom, B. Kaliski, «PKCS #9: Selected Object Classes and Attribute Types Version 2.0», RFC 2985, November 2000.
5. J. Schaad, «Cryptographic Message Syntax (CMS) Algorithm Identifier Protection Attribute», RFC 6211, April 2011.
6. Signal-COM AuthKey Server. Руководство системного программиста. ШКНР.00072-01 32 01. АО «СИГНАЛ-КОМ», 2024.
7. СКЗИ «Signal-COM JCP 3.1». Формуляр. ШКНР.00049-01 30 01. ЗАО «Сигнал-КОМ», 2019.