Бастион-3 – gRPC. Руководство программиста

Версия 2025.1

(17.01.2025)

Самара, 2025

Оглавление

1. Общие сведения

Модуль «Бастион-3 — gRPC» предназначен для организации информационного взаимодействия внешних систем с ПК «Бастион-3» через gRPC-протокол. Подробнее о самом протоколе можно прочитать в официальной документации.

Ключевые сценарии использования, для которых может применяться API, включают:

1. Взаимодействие с системами учёта посетителей, кадровыми и бухгалтерскими системами – создание, изменение персональных, материальных и транспортных пропусков в ПК «Бастион-3». Выполнение операций с пропусками (выдача, блокировка / разблокировка, продление, возврат, изъятие). Получение дерева подразделений, работа с подразделениями. Получение списка точек прохода и территорий. Получение событий по запросу за период времени. Получение списка уровней доступа. Получение событий УРВ.
2. Получение событий и передача команд ПК «Бастион-3» в реальном времени. Может использоваться для разработки дополнительных АРМ-ов, для интеграции со сторонними системами, когда не требуется использование стандартных протоколов OPC UA или SNMP.

2. Условия применения

На gRPC ПК «Бастион-3» распространяются те же требования к аппаратной и программной платформе, что и для ПК «Бастион-3».

Для работы модуля «Бастион-3 — gRPC» требуется отдельная лицензия на этот модуль.

Для работы требуется ПК «Бастион-3» версии не ниже 2024.1.

3. Настройка

По умолчанию Web API ПК «Бастион-3» отключен. Для его включения и настройки необходимо воспользоваться утилитой «Локальные параметры» из комплекта поставки ПК «Бастион-3» или консольной утилитой BCnfg. Подробнее о настройках читайте в «Бастион-3. Руководство администратора».

4. Подключение к gRPC Web API

Подключение выполняется по адресу grpc://{server_addresss}:{port}, где {server_address} - адрес выделенного Web API сервера, {port} — порт для подключений посредством gRPC, выбранный в локальных настройках.

5. Взаимодействие с сервером системы через gRPC

В данном разделе приведены описания и примеры использования сервисов, доступных через gRPC Web API.

5.1. AuthorizationService

Описание сервиса приведено в файле authorization.proto. Данный сервис используется при авторизации в системе. Для выполнения почти всех запросов ко всем сервисам, доступным через gRPC, необходимо пройти авторизацию и получить токен доступа.

5.1.1. Метод Login

Метод используется для непосредственной авторизации в системе и получения токена доступа. Для авторизации необходимо в параметрах запроса (LoginRequest) передать параметры аутентификации (поле AuthenticationParameters). Сейчас доступно два типа параметров:

1. Пара логин/пароль. Пример тела запроса :

```
{
```

```
  "UserAndPassword": {
    "user": "q",
    "password": "q"
  }
}
```

2. Секретное слово. Пример тела запроса:

```
{
  "Otp": {
    "user": "q",
    "secret_word": "secret"
  }
}
```

При успешной авторизации получаем:

```
{
  "session_id": 0,
  "access_token": "",
  "access_token_expire_time": {
    "seconds": 0,
    "nanos": 0
  }
}
```

Сообщение состоит из:

- session_id – уникальный идентификатор вашей сессии;
- access_token – токен доступа, используемый для дальнейших обращений к серверу системы от имени авторизованного пользователя. Токен доступа при последующем выполнении других запросов передаётся в заголовке authorization в формате: *Bearer <токен доступа>.* Токен действителен только во время текущей сессии, если сервер системы был перезагружен, то токен перестает быть действительным;
- access_token_expire_time – время окончания действия токена доступа.

5.1.2. Метод Logout

Запрос на завершение сессии ранее авторизованного пользователя. Выполняется с пустым телом запроса, в качестве успешного результата ответ приходит с пустым телом. Запрос может быть выполнен только с использованием токена доступа.

5.1.3. Метод RefreshToken

Запрос на обновление токена доступа. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Пример ответа сервера системы:

```
{
  "access_token": "",
  "access_token_expire_time": {
    "seconds": 0,
    "nanos": 0
  }
}
```

- access_token – новый токен доступа;
- access_token_expire_time – время окончания действия нового токена доступа

После выполнения запроса, старый токен доступа, будет не действителен. Для дальнейших обращений необходимо использовать новый токен доступа, полученный в качестве ответа на запрос.

5.1.4. Метод Unauthorized

Запрос на отслеживание активной сессии. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Если сессия завершилась в результате действия пользователя (например, в результате завершения сессии пользователя путем отправки запроса *Logout*), то получаем следующий ответ:

```
{
  "Normal": {}
}
```

Если завершение сессии вызвано ошибкой, то ответ будет содержать сообщение об ошибке и ее тип:

```
{
  "Broken”: {
            “value”: “message error (Parameter ‘type error’)”
  }
}
```

5.2. ServerInfoService

Сервис предназначен для получения информации о сервере системы, идентификаторе его сессии работы, временной зоне, а также списке доступных модулей. Описание приведено в файле server_info.proto.

5.2.1. Метод GetModules

Данный метод предназначен для получения списка модулей сервера системы ПК «Бастион-3» с указанием наименования модуля и его версии. Запрос должен выполняться с использованием токена доступа. Пример ответа:

```
{
  "modules": [
      {
          "name": "Operators",
          "version": "2024.2"
      },
      {
          "name": "Persons",
          "version": "2024.1"
      },
      ...
  ]
}
```

5.2.2. Метод GetServerState

Запрос на получение информации об идентификаторе сессии работы сервера системы, а также коде его временной зоне. Выполняется с пустым телом запроса. Запрос может быть выполнен без использования токена доступа.

Пример ответа:

```
{
  "Normal": {
      "server_session_uid": "83e711ba-4a1d-4406-a1ed-9b835eacd50f",
      "time_zone_code": 3
  }
}
```

- server_session_uid – *идентификатор* сессии работы сервера системы;
- time_zone_code – код временной зоны.

Если произошла ошибка при инициализации запроса, то ответное сообщение выглядит следующим образом:

```
{
  "Broken": {
      "error_message": ""
  }
}
```

5.3. UpdateDataService

Сервис UpdateDataService служит для получения доступа к функционалу изменения данных. Описание сервиса приведено в файле update_data.proto.

5.3.1. Метод UpdateData

Запрос на изменение данных системы. Параметры запроса задаются с помощью сообщения UpdateDataRequest. В рамках одного запроса может быть выполнено несколько элементарных изменений, список которых передаётся через поле operations сообщения UpdateDataRequest в запакованном в Any виде (подробнее в приложении А Тип google.protobuf.Any). Описание типов сообщений для элементарных изменений будет приведено далее в подразделах 5.3.1. При необходимости задания связей между сущностями, добавляемыми в рамках одного запроса изменения данных, необходимо использовать механизм временных идентификаторов, алгоритм работы которого следующий:

1. На стороне клиента генерируется временное значение для идентификатора добавляемой "родительской" (той, на которую будут ссылаться другие добавляемые сущности) сущности. Временный идентификатор представляет из себя целое знаковое 4-х байтовое число, значение которого должно быть не больше -100 и должно быть уникальным в рамках всех элементарных операций, передаваемых в одном запросе (если есть другие новые "родительские" сущности, добавляемые в рамках этого же запроса, то для них должны быть сгенерированы другие временные идентификаторы.

2. В элементарной операции добавления "дочерней" сущности в поле, в котором через идентификатор указывается ссылка на "родительскую" сущность (далее поле-ссылка), указывается значение временного идентификатора, сгенерированного на первом шаге.

3. При обработке на сервере системы элементарной операции добавления "родительской" сущности, происходит генерация окончательного идентификатора, при этом строится общая карта соответствия временных и окончательных идентификаторов для всех добавляемых в рамках одного запроса "родительских" сущностей.

4. При обработке операции добавления "дочерней" сущности в случае, если в поле-ссылке задан временный идентификатор (значение не больше -100, все окончательные идентификаторы имеют неотрицательные значения) с помощью карты, заполняемой на 3-м шаге, происходит разрешение временного идентификатора в окончательный: в поле подставляется значение соответствующего окончательного идентификатора.

5. В результатах запроса (сообщение UpdateDataResponse) на клиент возвращается карта соответствия временных и окончательных идентификаторов (поле temp_ids_map). Эта информация может быть использована клиентом для дальнейшей обработки данных.

Запрос изменения данных может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
  "operations": [
      {
          "type_url": "type.googleapis.com/esprom.taurus.grpc.v1.persons.DeleteAccessLevel",
          "value": "CPEB"
      },
      {
          "type_url": "type.googleapis.com/esprom.taurus.grpc.v1.persons.DeleteAccessLevel",
          "value": "CPEB"
      }
  ]
}
```

- operations – список элементарны операций изменения данными, которые необходимо выполнить.

Пример ответа на успешное выполнение запроса:

```
{
  "temp_ids_map": {
      "22": 321,
      "25": 322
  }
}
```

Если при изменении данных никакие временные идентификаторы не использовались, то поле temp_ids_map будет пустым:

```
{
  "temp_ids_map": {}
}
```

В последующих подразделах будет приведено описание сообщений соответствующих элементарным операциям изменения данных, которые поддерживаются системой.

5.3.1.1. Редактирование уровней доступа

Описание элементарных операций редактирования уровней доступа приведено в файле persons/access_level_update.proto. Подробнее про сами уровни доступа можно прочитать в разделе 5.4. AccessLevelService.

esprom.taurus.grpc.v1.persons.AddAccessLevel
Добавление уровня доступа. Поля:

- access_level (тип esprom.taurus.grpc.v1.persons.AccessLevel) — описание добавляемого уровня доступа.

esprom.taurus.grpc.v1.persons.DeleteAccessLevel
Удаление уровня доступа. Поля:

- access_level_id – идентификатор удаляемого уровня доступа.

esprom.taurus.grpc.v1.persons.UpdateAccessLevel
Обновление уровня доступа. Поля:

- access_level (тип esprom.taurus.grpc.v1.persons.AccessLevel) — описание обновляемого уровня доступа.

esprom.taurus.grpc.v1.persons.SetAccessLevelContent
Задание состава уровня доступа. Поля:

- access_level_id – идентификатор удаляемого уровня доступа;
- entries – перечисление элементов состава уровня доступа.

esprom.taurus.grpc.v1.persons.AddWeakAccessLevel
Создание автоматического уровня доступа. Автоматические уровни доступа имеют следующие важные отличия от обычных уровней доступа:

1. Они не могут существовать, если на них нет ссылающихся пропусков. При удалении последнего ссылающегося на пропуска, будет также удалён автоматический уровень доступа.
2. При их добавлении в систему, сначала происходит поиск уже существующего аналогичного по составу автоматического уровня доступа. Если такой уровень будет найден, то добавление не будет фактически выполнено, а переданный временный идентификатор для автоматического уровня доступа будет разрешён в идентификатор уже существующего аналогичного автоматического уровня доступа.
3. У автоматического уровня доступа нет явного имени. Оно генерируется автоматически при добавлении сервером системы.

Поля:

- temp_access_level_id — временный идентификатор для автоматического уровня доступа, если в системе будет найден аналогичный по составу другой уже существующий автоматический уровень доступа, то его значение будет разрешено в значение идентификатора аналогичного уровня.
- main_time_block_id — идентификатор основного временного блока для уровня доступа.
- entries — перечисление элементов состава уровня доступа. При добавлении параметров и состава уровня доступа, создается уровень доступа (с параметром is_weak = true), если он используется в пропусках или расписаниях.

5.3.1.2. Редактирование карт доступа

Описание элементарных операций редактирования карт доступа приведено в файле persons/card_update.proto. Подробнее про сами карты доступа можно прочитать в разделе 5.5. CardService.

esprom.taurus.grpc.v1.persons.AddCard
Добавление новой карты доступа в систему. Поля:

- card (тип esprom.taurus.grpc.v1.persons.Card) — описание добавляемой карты доступа.

esprom.taurus.grpc.v1.persons.UpdateCard
Редактирование существующей карты доступа. Поля:

- card (тип esprom.taurus.grpc.v1.persons.Card) — описание редактируемой карты доступа.

esprom.taurus.grpc.v1.persons.DeleteCard
Удаление существующей карты доступа. Поля:

- card_id — идентификатор удаляемой карты доступа.

5.3.1.3. Редактирование словарных значений

Описание элементарных операций редактирования словарных значений приведено в файле persons/dictionaries_update.proto. Подробнее про сами словари и их значения можно прочитать в разделе 5.10. DictionariesService.

esprom.taurus.grpc.v1.persons.SetDictionaryValue
Операция условного добавления словарного значения. Поля:

- record (тип esprom.taurus.grpc.v1.persons.DictionaryRecord) — описание словарного значения.

При обработке этой операции сервер системы сначала попробует найти уже существующее аналогичное словарное значение, и, если такое найдётся, добавление не будет выполнено, при этом указанный для значения временный идентификатор будет разрешён в идентификатор найденного аналогичного значения. Например, в системе имеется словарь «Гражданство (id = 3)» со значениями "Россия" и "Белоруссия". После выполнения следующих операций (для наглядности в поле value используется не запакованное в Any представление):

```
{
  "operations": [
    {
      "type_url": "type.googleapis.com/esprom.taurus.grpc.v1.persons.SetDictionaryValue",
      "value": {
        "record": {
          "header_id": 3,
          "id": -101,
          "is_system": false,
          "value": "Белоруссия"
        }
      }
    },
    {
      "type_url": "type.googleapis.com/esprom.taurus.grpc.v1.persons.SetDictionaryValue",
      "value": {
        "record": {
          "header_id": 3,
          "id": -102,
          "is_system": false,
          "value": "Казахстан"
        }
      }
    }
  ]
}
```

Будет добавлено только значение «Казахстан», так как такого значения ещё нет в системе.

esprom.taurus.grpc.v1.persons.AddDictionaryValue
Операция добавления словарного значения. Если в системе уже есть такое значение, то операция завершится с ошибкой. Поля:

- record (тип esprom.taurus.grpc.v1.persons.DictionaryRecord) — описание словарного значения.

esprom.taurus.grpc.v1.persons.UpdateDictionaryValue
Операция редактирования словарного значения. Если в системе уже есть значение, совпадающее с новым значение, то операция завершится с ошибкой. Поля:

- record (тип esprom.taurus.grpc.v1.persons.DictionaryRecord) — описание словарного значения.

esprom.taurus.grpc.v1.persons.DeleteDictionaryValue
Операция удаления словарного значения. Поля:

- record_id — идентификатор удаляемого словарного значения.

5.3.1.4. Редактирование пропусков

Описание элементарных операций редактирования пропусков приведено в файле persons/pass_update.proto. Подробнее про пропуска можно прочитать в разделе 5.11. PassService.

esprom.taurus.grpc.v1.persons.AddPass
Операция добавления пропуска. Поля:

- pass (тип esprom.taurus.grpc.v1.persons.Pass) — описание добавляемого пропуска.

esprom.taurus.grpc.v1.persons.UpdatePass
Операция редактирования пропуска. Поля:

- pass (тип esprom.taurus.grpc.v1.persons.Pass) — описание редактируемого пропуска.

esprom.taurus.grpc.v1.persons.DeletePass
Операция удаления пропуска. Поля:

- pass_id — идентификатор удаляемого пропуска.

esprom.taurus.grpc.v1.persons.UpdatePassSecurityControlGroupSetting
Операция задания для пропуска группы управления охраной. Поля:

- group_settings (тип esprom.taurus.grpc.v1.persons.PassSecurityControlGroupSetting) — настройка привязки пропуска к группе управления охраной.

esprom.taurus.grpc.v1.persons.DeletePassSecurityControlGroupSetting
Операция сброса для пропуска группы управления охраны. Поля:

- pass_id — идентификатор пропуска.

5.3.1.5. Редактирование персон

Описание элементарных операций редактирования персон приведено в файле persons/person_update.proto. Подробнее про персон можно прочитать в разделе 5.13. PersonService.

esprom.taurus.grpc.v1.persons.AddPerson
Операция добавления персоны. Поля:

- person (тип esprom.taurus.grpc.v1.persons.Person) — описание добавляемой персоны.

esprom.taurus.grpc.v1.persons.UpdatePerson
Операция редактирования персоны. Поля:

- person (тип esprom.taurus.grpc.v1.persons.Person) — описание редактируемой персоны.

esprom.taurus.grpc.v1.persons.DeletePerson
Операция удаления персоны. Поля:

- person_id — идентификатор удаляемой персоны.

esprom.taurus.grpc.v1.persons.UpdatePersonalInfo
Операция обновления персональных данных персоны. Поля:

- personal_info (тип esprom.taurus.grpc.v1.persons.PersonalInfo) — персональные данные/реквизиты персоны.

esprom.taurus.grpc.v1.persons.SetPersonPhoto
Операция задания фото для персоны. Поля:

- person_id — идентификатор персоны;
- photo_data — фото персоны.

esprom.taurus.grpc.v1.persons.DeletePersonPhoto
Операция сброса фото для персоны. Поля:

- person_id — идентификатор персоны.

5.3.1.6. Редактирование временных блоков

Описание элементарных операций редактирования временных блоков приведено в файле persons/time_block_update.proto. Подробнее про временные блоки можно прочитать в разделе 5.18. TimeBlockService.

esprom.taurus.grpc.v1.persons.AddTimeBlock
Операция добавления временного блока. Поля:

- time_block (тип esprom.taurus.grpc.v1.persons.TimeBlock) — описание добавляемого временного блока.

esprom.taurus.grpc.v1.persons.UpdateTimeBlock
Операция редактирования временного блока. Поля:

- time_block (тип esprom.taurus.grpc.v1.persons.TimeBlock) — описание редактируемого временного блока.

esprom.taurus.grpc.v1.persons. DeleteTimeBlock
Операция редактирования временного блока. Поля:
- time_block_id — идентификатор удаляемого временного блока.

esprom.taurus.grpc.v1.persons.AddWeakTimeBlock
Создание автоматического временного блока. Автоматические временные блоки имеют следующие важные отличия от обычных временных блоков:
1. Они не могут существовать, если на них нет ссылающихся сущностей (уровни доступа и/или их элементы, территории). При удалении последней ссылки на автоматический временной блок, он будет удалён.
2. При их добавлении в систему, сначала происходит поиск уже существующего аналогичного автоматического временного блока. Если такой блок будет найден, то добавление не будет фактически выполнено, а переданный временный идентификатор для автоматического временного блока будет разрешён в идентификатор уже существующего аналогичного автоматического временного блока.
3. У автоматического временного блока нет явного имени. Оно автоматически генерируется сервером системы при добавлении.
4. Автоматические временные блоки всегда являются недельными и учитывают праздники.

Поля:
- temp_time_block_id — временный идентификатор для автоматического временного блока, если в системе будет найден аналогичный по составу другой уже существующий автоматический временной блок, то его значение будет разрешено в значение идентификатора аналогичного блока.
- time_zones — перечисление временных зон входящих во временной блок.

5.3.1.7. Редактирование дополнительных полей устройств

Описание элементарных операций редактирования значений дополнительных полей устройств приведено в файле drivers/additional_field_update.proto. Подробнее про временные блоки можно прочесть в разделе 5.19. AdditionalFieldService.

esprom.taurus.grpc.v1.persons.UpdateAdditionalFieldValue
Операция задания временного блока. Поля:
- value (тип esprom.taurus.grpc.v1.drivers.AdditionalFieldValue) — описание задаваемого значения дополнительного поля устройства.

5.3.1.8. Редактирование организационной структуры

Описание элементарных операций редактирования организационной структуры приведено в файле persons/organization_structure_update.proto. Подробнее про организационную структуру можно прочитать в разделе 5.8. OrganizationStructureService.

esprom.taurus.grpc.v1.persons.AddOrganizationNode
Операция добавления узла дерева организационной структуры. Поля:
- node (тип esprom.taurus.grpc.v1.persons.OrganizationNode) — описание добавляемого узла дерева организационной структуры.

esprom.taurus.grpc.v1.persons.UpdateOrganizationNode
Операция редактирования узла дерева организационной структуры. Поля:
- node (тип esprom.taurus.grpc.v1.persons.OrganizationNode) — описание редактируемого узла дерева организационной структуры.

esprom.taurus.grpc.v1.persons.DeleteOrganizationNode
Операция удаления узла дерева организационной структуры. Поля:
- node_id — идентификатор удаляемого узла дерева организационной структуры.

esprom.taurus.grpc.v1.persons.SetOrganizationNode
Операция условного добавления узла дерева организационной структуры. Поля:

- node (тип esprom.taurus.grpc.v1.persons.OrganizationNode) — описание добавляемого узла дерева организационной структуры.

При обработке этой операции сервер системы сначала попробует найти уже существующий аналогичный узел дерева организационной структуры, и, если такой найдётся, добавление не будет выполнено, при этом указанный для узла временный идентификатор будет разрешён в идентификатор найденного аналогичного узла. При поиске аналогичного узла выполняется сравнение по трём полям: родительскому узлу, типу и имени узла.

Например, если текущее дерево организационной структуры выглядит следующим образом:
- Все (id: 0)
  - ООО Организация 1 (id: 101)
    - Департамент 1 (id: 103)
    - Департамент 2 (id: 104)
  - ООО Организация 2 (id: 102)

То после выполнения следующих операций (для наглядности в поле value используется не запакованное в Any представление):

```
{
  "operation": [
    {
      "type_url": "type.googleapis.com/esprom.taurus.grpc.v1.persons.SetOrganizationNode",
      "value": {
        "node": {
          "id": -101,
          "name": "ООО Организация 2",
          "node_type": "ORGANIZATION_NODE_TYPE_ORGANIZATION",
          "parent_id": {
            "value": 0
          }
        }
      }
    },
    {
      "type_url": "type.googleapis.com/esprom.taurus.grpc.v1.persons.SetOrganizationNode",
      "value": {
        "node": {
          "id": -102,
          "name": "ООО Департамент 1",
          "node_type": "ORGANIZATION_NODE_TYPE_DEPARTMENT",
          "parent_id": {
            "value": -101
          }
        }
      }
    }
  ]
}
```

Систему будет добавлен только «Департамент 2», который является дочерним в организации «ООО Организация 2». Организация не будет добавлена, так как она уже есть в системе.

esprom.taurus.grpc.v1.persons.AllowPassCategoryForOrganizationNode

Операция разрешения использования категории пропусков для узла дерева организационной структуры. Поля:
- organization_node_id — идентификатор узла организационной структуры, для которого разрешают использование категории пропусков.

- pass_category_id — идентификатор категории пропусков, которая должна стать доступной для использования в узле организационной структуры. Если значение не задано, то для узла организационной структуры станут доступны все категории пропусков.

esprom.taurus.grpc.v1.persons.RestrictPassCategoryForOrganizationNode

Операция изменения данных, которая запрещает использование категории пропусков для узла организационной структуры. Поля:
- organization_node_id — идентификатор узла организационной структуры, для которого запрещают использование категории пропусков.

- pass_category_id — идентификатор категории пропусков, которую запрещают для использования в узле организационной структуры. Если значение не задано, то для узла организационной структуры будут запрещены все категории пропусков.

5.4. AccessLevelService

Файл access_level.proto предназначен для получения информации об уровнях доступа.

5.4.1. Метод GetAccessLevels

Запрос на получение списка уровней доступа. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Пример ответа на запрос:

```
{
  "access_levels": [
      {
         "id": 141,
         "physical_number": {
            "value": 4
         },
         "is_weak": true,
         "name": "Уровень доступа №1 (авто)",
         "main_time_block_id": 4
      },
      {
         "id": 1,
         "physical_number": null,
         "is_weak": false,
         "name": "По умолчанию",
         "main_time_block_id": 4
      },
      {
         "id": 121,
         "physical_number": null,
         "is_weak": false,
         "name": "Уровень доступа 3",
         "main_time_block_id": 1
      },
      ...
  ]
}
```

Ответное сообщение содержит список всех уровней доступа в системе. О каждом уровне доступа представлена следующая информация:

- id – уникальный идентификатор;
- physical_number – физический номер. Если уровень доступа не используется значение будет *null*;
    - value – значение физического номера;
- is_weak – булево значение указывающий на то что уровень доступа задан был сгенерирован автоматически в ходе ручного задания состава уровня доступа;
- name – наименование уровня доступа, максимальная длина 100 символов;
- main_time_block_id – идентификатор временного блока используемого в данном уровне доступа.

5.4.2. Метод GetAccessLevel

Запрос на получение информации об уровне доступа. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
   "access_level_id": 101
}
```

В тело запроса необходимо записать в поле *access_level_id* уникальный идентификатор уровня доступа, информацию о котором необходимо получить.

Пример ответа:

```
{
```

```
    "access_level": {
        "id": 101,
        "physical_number": {
            "value": 4
        },
        "is_weak": true,
        "name": "Уровень доступа №1 (авто)",
        "main_time_block_id": 4
    }
}
```


В теле *access_level* указаны та же информация об уровне доступа, что и в ответном сообщении на запрос *GetAccessLevels*.

5.4.3. Метод GetAccessLevelContents

Запрос на получение информации о составе уровня доступа. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:


```
{
    "access_level_id": 101
}
```


В тело запроса необходимо записать в поле *access_level_id* уникальный идентификатор уровня доступа, информацию о составе которого необходимо получить. Если в состав уровня доступа пуст, то ответные сообщения не будут получены, при этом запрос выполнится без ошибок.

Ответ разделен на несколько сообщений. Сообщения разделены на несколько категорий:

Сообщение о вложенном считывателе:


```
{
    "entry": {
        "Reader": {
            "reader_sdn": 350,
            "custom_time_block_id": {
                "value": 8
            }
        }
    }
}
```


В сообщении содержится:

- reader_sdn – уникальный идентификатор считывателя;
- custom_time_block_id – уникальный идентификатор используемого временного блока. Если к считывателю привязан автоматический временной блок, то значение будет равное *null*.

Сообщение о вложенной территории:


```
{
    "entry": {
        ""ControlArea": {
            "control_area_id": 100,
            "custom_time_block_id": {
                "value": 5
            }
        }
    }
}
```


- reader_sdn – уникальный идентификатор территории;
- custom_time_block_id – уникальный идентификатор используемого временного блока. Если к считывателю привязан автоматический временной блок, то значение будет равное *null*.

Сообщение о вложенном уровне доступа:


```
{
    "entry": {
        "NestedLevel": {
            "access_level_id": 126,
            "order": 4
        }
    }
}
```


- access_level_id – уникальный идентификатор вложенного уровня доступа;
- order – порядковый номер.

5.4.4. Метод GetAccessLevelContentCompositions

Запрос на получение информации о считывателях входящих в состав уровня доступа. Запрос может быть выполнен только с использованием токена доступа. Если в состав уровня доступа пуст, то ответные сообщения не будут получены, при этом запрос выполнится без ошибок.

Пример запроса:

```
{
   "access_level_id": 101
}
```

В тело запроса необходимо записать в поле *access_level_id* уникальный идентификатор уровня доступа, информацию о вложенных считывателях которых необходимо получить.

В качестве ответа будут получены несколько сообщений, в зависимости от количества вложенных считывателя.

Пример ответа:

```
{
   "reader_sdn": 350,
   "time_block_id": 8
}
```

- reader_sdn – уникальный идентификатор считывателя;
- custom_time_block_id – уникальный идентификатор используемого временного блока.

5.4.5. Метод AccessLevelChanged

Запрос на отслеживание изменений в уровнях доступа. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Пример сообщений:

```
{
   "access_level_id": 122,
   "change_type": "ENTITY_CHANGE_TYPE_UPDATE"
}
```

- access_level_id – уникальный идентификатор уровня доступа, которого затронули изменения;
- change_type – тип произведенных изменений. Это может быть добавление (ENTITY_CHANGE_TYPE_ADD), обновление (ENTITY_CHANGE_TYPE_UPDATE), удаление (ENTITY_CHANGE_TYPE_DELETE) или неизвестное изменение (ENTITY_CHANGE_TYPE_UNSPECIFIED).

5.4.6. Метод AccessLevelContentChanged

Запрос на отслеживание изменений в составах уровней доступа. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Пример сообщения:

```
{
   "access_level_id": 122,
}
```

- access_level_id – уникальный идентификатор уровня доступа, состав которого был изменен.

5.5. CardService

Файл card.proto предназначен для получения информации о картах доступа.

5.5.1. Метод GetCard

Запрос на получение информации о карте доступа. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса с использованием идентификатора доступа:

```
{
   "by_card_id": {
```

```
        "card_id": 313
    },
}
```

Если карта не была найдена, то ответ будет следующим:

```
{
    "card": null
}
```

Если карта с таким номером существует, то ответ содержит информацию о карте:

```
{
    "card": {
        "id": 313,
        "full_card_code": "1104801",
        "serial_number": {
            "value": "345"
        },
        "create_date": {
            "seconds": "1712853501",
            "nanos": 340667000
        },
        "return_date": {
            "seconds": "1712903024",
            "nanos": 121046000
        },
        "status": "CARD_STATUS_ISSUED",
        "identifier_type": 0,
        "mifare_security_level": 0,
        "pre_issued": false
    }
}
```

- id – уникальный идентификатор карты доступа;
- full_card_code – полный номер карты;
- serial_number – содержит числовое значение серийного номера карты;
- create_date – дата создания карты доступа, значение хранит время в секундах *seconds* и наносекундах *nanos*;
- return_date – дата возврата карты доступа, значение хранит время в секундах *seconds* и наносекундах *nanos*;
- status – указывает текущий статус пропуска (выпущен CARD_STATUS_ISSUED, потерян CARD_STATUS_LOST и другие, подробнее о статусах написано в документации «Бастион-3 – Бюро пропусков. Руководство оператора»;
- identifier_type – тип идентификатора карты, указывающий об используемом считывателе. Подробнее о типах идентификации написано в документации «Бастион-3 – Бюро пропусков. Руководство оператора»;
- mifare_security_level – номер уровня безопасности для карт доступа MIFARE;
- pre_issued – показывает, была ли карта эмитирована в ПК «Бастион-3».

Помимо поиска по идентификатору карты, можно отправить запрос на получение информации о карте с использованием серийного номера карты:

```
{
    "by_serial_number": {
        "serial_number": "345"
    }
}
```

Ответное сообщение аналогично запросу карты доступа по идентификатору карты.

5.5.2. Метод GetCards

Запрос на получение списка карт доступа. Запрос может быть выполнен только с использованием токена доступа.

Для получения списка карты есть несколько запросов.

Запрос с использованием кода карт:

```
{
    "by_card_code": {
        "card_code": "4"
    }
}
```

В запросе требуется указать код карты доступа. Ответ содержит список всех карт, у которых указан искомый код карты:

```
{
    "cards": [
```

```
      {
         "id": 413,
         "full_card_code": "4",
         "serial_number": null,
         "create_date": {
            "seconds": "1712906246",
            "nanos": 787102000
         },
         "return_date": {
            "seconds": "1712906783",
            "nanos": 432439000
         },
         "status": "CARD_STATUS_UNUSABLE",
         "identifier_type": 0,
         "mifare_security_level": 0,
         "pre_issued": false
      },
      {
         "id": 433,
         "full_card_code": "4",
         "serial_number": null,
         "create_date": {
            "seconds": "1712906808",
            "nanos": 882140000
         },
         "return_date": null,
         "status": "CARD_STATUS_ISSUED",
         "identifier_type": 0,
         "mifare_security_level": 0,
         "pre_issued": false
      },
      ...
   ]
}
```


Для того, чтобы получить список конкретных карт доступа, можно использовать запрос по идентификаторам карт доступа:


```
{
   "by_card_ids": {
      "card_ids": [
         313,
         433
      ]
   }
}
```


В тело *card_ids* требуется указать список идентификаторов карт доступа, информацию о которых требуется получить.

Если карт доступа с указанными идентификаторами нет, то ответ будет содержать пустой список:


```
{
   "cards": []
}
```


При наличии искомых карт, ответное сообщение содержит список карт доступа:


```
{
   "cards": [
      {
         "id": 313,
         "full_card_code": "55328922",
         "serial_number": null,
         "create_date": {
            "seconds": "1712906246",
            "nanos": 787102000
         },
         "return_date": {
            "seconds": "1712906783",
            "nanos": 432439000
         },
         "status": "CARD_STATUS_UNUSABLE",
         "identifier_type": 0,
         "mifare_security_level": 0,
         "pre_issued": false
      },
      {
         "id": 433,
         "full_card_code": "438175",
         "serial_number": null,
         "create_date": {
            "seconds": "1712906808",
            "nanos": 882140000
```

```
            },
            "return_date": null,
            "status": "CARD_STATUS_ISSUED",
            "identifier_type": 0,
            "mifare_security_level": 0,
            "pre_issued": false
        },
        ...
    ]
}
```

Для получения списка всех карт требуется отправить запрос:

```
{
    "empty": {}
}
```

Для получения списка свободных карт нужно использовать запрос:

```
{
    "free_cards": {
        "packet_size": 3
    }
}
```

Ответ посылается несколькими пакетами информации, каждый пакет содержит количество карт, указанные в поле *packet_size*. В качестве завершающего пакета данных, будет получен пустой список карт:

```
{
    "cards": []
}
```

5.5.3. Метод CardChanged

Подписка на событие изменение карт доступа. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

При изменении карты доступа в системе, придет следующее сообщение:

```
{
    "card_id": 433,
    "change_type": "ENTITY_CHANGE_TYPE_UPDATE"
}
```

- card_id – идентификатор карты доступа, с которой произошло изменение;
- change_type – тип изменения карты доступа (добавление, удаление, обновление или неизвестное).

5.6. DriverPassProfileService

Файл driver_pass_profile.proto предназначен для получения информации о профилей пропусков.

5.6.1. Метод GetDriverPassProfileTypes

Запрос на получение списка всех типов профилей пропусков установленных в системе. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Пример ответа:

```
{
    "profile_types": [
    {
      "db_id": 1,
      "driver_id": 100,
      "internal_id": 3,
      "name": {
        "value": "профиль 1"
      },
      "pass_link_type": 0,
      "is_required": false
    },
    {
      "db_id": 2,
      "driver_id": 101,
      "internal_id": 3,
      "name": {
        "value": "профиль 2"
      },
      "pass_link_type": 0,
```

```
    "is_required": false
   },
   ...
  ]
}
```

- db_id – идентификатор профиля;
- driver_id – идентификатор типа драйвера;
- internal_id – внутренний для драйвера идентификатор типа профилей;
- name – наименование профиля;
- pass_link_type – тип связи пропуска к экземплярам данного типа профилей;
- is_required – должен ли в наборах пропусков обязательно быть выбран данный профиль.

5.6.2. Метод GetDriverPassProfiles

Запрос на получение всех добавленных в систему профилей пропусков для указанного типа профилей и типа драйвера. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
   "driver_id": 0,
   "profile_type_id": 2
}
```

- driver_id – идентификатор типа драйвера;
- profile_type_id – идентификатор типа профиля.

Пример ответа:

```
{
   "profiles": [
   {
     "profile_type_db_id": 2,
     "id": 2,
     "is_predefined": false,
     "name": "профиль 2",
     "description": {
       "value": "описание"
     },
     "is_default": false
   },
   {
     "profile_type_db_id": 2,
     "id": 4,
     "is_predefined": false,
     "name": "профиль 1",
     "description": {
       "value": "описание"
     },
     "is_default": false
   },
   ...
  ]
}
```

- profile_type_db_id – идентификатор типа профиля настроек персонала;
- id – идентификатор профиля;
- is_predefined – признак предопределенного профиля;
- name – наименование профиля;
- description – описание профиля;
- is_default – используется ли данный профиль как значение по умолчанию.

5.6.3. Метод GetDriverPassProfileForPass

Запрос на получение списка всех назначенных для пропуска профилей пропусков. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
   "pass_id": 124
}
```

- pass_id – идентификатор пропуска, профили которого нужно получить.

Ответ:

```
{
  "linked_profiles": [
  {
    "profile_id": 20,
    "identification_type": "PASS_IDENTIFICATION_TYPE_BY_PIN_CODE"
  },
  {
    "profile_id": 21,
    "identification_type": "PASS_IDENTIFICATION_TYPE_BY_CARD"
  },
  ...
 ]
}
```

- profile_id – идентификатор профиля драйвера;
- identification_type – тип идентификации профиля (по номеру карты, по ПИН-коду, по ПИН-коду и номеру карты).

5.7. ExternalIntegration

Файл external_integration.proto предназначен для отслеживания запросов на подтверждение доступа.

5.7.1. Метод ExternalAccessConfirmRequired

Запрос на подписку к внешней системе запросов подтверждения доступа. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса с идентификатором внешней системы, от которого будем получить запросы на подтверждения доступа:

```
{
  "external_id": {
     "value": "43"
  }
}
```

При появлении запроса на подтверждение доступа, получим следующее сообщение:

```
{
  "request_id": 3,
  "pass_id": 129,
  "pass_point_sdn": 345,
  "pass_direction": "PASS_DIRECTION_TYPE_IN",
  "external_id": {
    "value": "43"
  }
}
```

- request_id – идентификатор запроса, используемый при последующей отправке в систему подтверждения или отказа в доступе;
- pass_id – идентификатор пропуска, для которого выполняется запрос на подтверждение доступа;
- pass_point_id – идентификатор точки прохода, через которую предполагается доступ;
- pass_direction – направление прохода, в котором предполагается выполнение доступа. Возможные значения:
  - PASS_DIRECTION_TYPE_UNSPECIFIED – не указано:
  - PASS_DIRECTION_TYPE_IN – вход;
  - PASS_DIRECTION_TYPE_OUT – выход.
- external_id – идентификатор внешней системы, в которую отправляется запрос на подтверждение доступа.

5.7.2. Метод ConfirmAccess

Запрос на подтверждение доступа по ранее полученному идентификатору запроса. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
  "request_id": 3
}
```

- request_id – идентификатор ранее полученного запроса на подтверждение доступа во внешней системе.

При успешном подтверждении доступа придет пустое ответное сообщение.

5.7.3. Метод DenyAccess

Запрос на отказ в доступе для ранее полученного идентификатора запроса с возможностью указать причину. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
   "reason": {
      "value": "ушел"
   },
   "request_id": 3
}
```

- request_id – идентификатор ранее полученного запроса на подтверждение доступа во внешней системе;
- reason – причина отказа доступа.

При успешном отказе доступа придет пустое ответное сообщение.

5.8. OrganizationStructureService

Файл organization_structure.proto предназначен для получения информации об организациях и департаментах.

5.8.1. Метод GetOrganizationStructureNodes

Запрос получение организаций и департаментов. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Ответные сообщения будут содержать информацию об узле организации/департамента:

```
{
   "node": {
      "id": 2,
      "name": "Отдел не указан",
      "parent_id": {
         "value": 1
      },
      "node_type": "ORGANIZATION_NODE_TYPE_DEPARTMENT"
   }
}
```

- id – уникальный идентификатор организации/департамента;
- name – наименовании организации/департамента (ограничение 255 символов);
- parent_id – содержит значение уникального идентификатора родительской организации. Для корневой организации «Все» значение родительской организации равно *null* для остальных обязательно должно быть значение не равное *null,* указывающий на родительскую организацию;
- node_type – тип узла:
  - ORGANIZATION_NODE_TYPE_UNSPECIFIED – значение не определено;
  - ORGANIZATION_NODE_TYPE_ORGANIZATION – организация;
  - ORGANIZATION_NODE_TYPE_DEPARTMENT – департамент.

5.8.2. Метод OrganizationStructureChanged

Запрос отслеживания изменений дерева организаций/департаментов. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Ответ:

```
{
   "organization_node_id": 0,
   "change_type": "ENTITY_CHANGE_TYPE_UNSPECIFIED"
}
```

- organization_node_id – идентификатор организации/департамента, который изменился;
- change_type – тип изменения организации/департамента:
  - ENTITY_CHANGE_TYPE_UNSPECIFIED – тип изменения не указан;
  - ENTITY_CHANGE_TYPE_ADD – запись создана;
  - ENTITY_CHANGE_TYPE_UPDATE – запись обновлена;
  - ENTITY_CHANGE_TYPE_DELETE – запись удалена.

5.9. ControlAreaService

Файл control_area.proto предназначен для получения информации о территориях.

5.9.1. Метод GetControlArea

Для получения информации о необходимой территории, требуется отравить запрос с указанием идентификатора территории. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
   "ctrl_area_id": 101
}
```

В поле *ctrl_area_id* нужно указать идентификатор территории, информацию которой желаете получить.

Ответное сообщение:

```
{
   "control_area": {
      "id": 101,
      "name": "Территория",
      "parent_id": {
         "value": 2
      },
      "time_block_id": {
         "value": 5
      }
   }
}
```

- id – уникальный идентификатор территории;
- name – наименовании территории;
- parent_id – содержит числовое значение идентификатора родительской территории. Если территория не принадлежит другой территории, то данное поле будет содержать значение *null*;
- time_block_id – идентификатор временного блока, которого принадлежит территория. Если у территории не указан временной блок, то данное поле содержит значение *null*.

При отправки с пустым телом запроса, ответное сообщение будет содержать территорию «Не указано».

5.9.2. Метод GetControlAreas

Запрос на получение списка всех территорий. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Ответ:

```
{
   "control_areas": [
      {
         "id": 0,
         "name": "Не указано",
         "parent_id": null,
         "time_block_id": null
      },
      {
         "id": 1,
         "name": "Вне территории",
         "parent_id": null,
         "time_block_id": null
      },
      ...
   ]
}
```

5.9.3. Метод GetControlAreaPassPointLinks

Запрос на получение точек прохода с привязанными территориями. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Пример пакета сообщения:

```
{
   "control_area_pass_point_links": {
      "pass_point_sdn": 349,
      "source_area_id": 100,
      "destination_area_id": 1
   }
}
```


- pass_point_sdn – уникальный идентификатор точки прохода;
- source_area_id – идентификатор территории откуда ведет точка проходу;
- destination_area_id – идентификатор территории куда ведет точка прохода.

Точки прохода, которые изначально не были привязаны ни к одной территории, в данный пакет ответных сообщений не попадут.

5.9.4. Метод ControlAreaChanged

Запрос на отслеживание изменений территорий. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Пример ответа:


```
{
   "control_area_id": 100,
   "change_type": "ENTITY_CHANGE_TYPE_UPDATE"
}
```


- control_area_id – идентификатор территорий, которая изменилась;
- change_type – тип изменения карты доступа (добавление, удаление, обновление или неизвестное).

5.9.5. Метод ControlAreaPassPointLinkUpdated

Запрос на отслеживание изменений направлений точек прохода. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Пример ответа:


```
{
   "pass_point_sdn": 351,
   "source_control_area_id": 1,
   "destination_control_area_id": 101
}
```


- pass_point_sdn – уникальный идентификатор точки прохода;
- source_area_id – идентификатор территории откуда ведет точка проходу;
- destination_area_id – идентификатор территории куда ведет точка прохода.

5.10. DictionariesService

Файл dictionaries.proto предназначен для работы со словарями и их записями.

5.10.1. Метод GetDictionaryHeaders

Запрос получение списка словарей. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Ответ:


```
{
   "dictionary_headers": [
      {
         "id": 2,
         "name": "Виды документов",
         "is_system": false
      },
      {
         "id": 3,
         "name": "Гражданство",
         "is_system": false
      },
      ...
   ]
}
```

- id – уникальный идентификатор словаря;
- name – наименование словаря ;
- is_system – указание на то что словарь является системным.

5.10.2. Метод GetDictionaryRecords

Запрос на получение всех записей в словаре. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
   "header_id": 2
}
```

- header_id – идентификатор словаря.

Ответ содержит все записи указанного словаря:

```
{
   "dictionary_records": [
      {
         "id": 4,
         "header_id": 2,
         "value": "Паспорт",
         "is_system": false
      },
      {
         "id": 5,
         "header_id": 2,
         "value": "Удостоверение военнослужащего",
         "is_system": false
      }
   ]
}
```

- id – идентификатор записи;
- header_id – идентификатор принадлежности к словарю;
- value – значение записи (ограничение 255 символов);
- is_system – указание, является ли запись системной.

5.10.3. Метод GetDictionaryRecord

Запрос на получение записи словаря. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
   "id": 2
}
```

- id – идентификатор записи.

Ответ содержит информацию о записи в словаре:

```
{
   "dictionary_record": {
      "id": 5,
      "header_id": 2,
      "value": "Удостоверение военнослужащего",
      "is_system": false
   }
}
```

- id – идентификатор записи;
- header_id – идентификатор принадлежности к словарю;
- value – значение записи;
- is_system – указание, является ли запись системной.

5.10.4. Метод DictionaryRecordChanged

Запрос на отслеживание изменений в словарях. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Ответ:

```
{
  "record_id": 78,
  "header_id": 5,
  "change_type": "ENTITY_CHANGE_TYPE_ADD"
}
```


- record_id – идентификатор записи;
- header_id – идентификатор принадлежности к словарю;
- change_type – тип изменения записи словаря:
  - ENTITY_CHANGE_TYPE_UNSPECIFIED – тип изменения не указан;
  - ENTITY_CHANGE_TYPE_ADD – запись создана;
  - ENTITY_CHANGE_TYPE_UPDATE – запись обновлена;
  - ENTITY_CHANGE_TYPE_DELETE – запись удалена.

Так же можно указать в теле запроса идентификатор словаря, чтобы отслеживать изменения связанные только с указанным словарем:


```
{
  "header_id": {
    "value": 5
  }
}
```


5.11. PassService

Файл pass.proto предназначен для работы с пропусками персон. Подробнее о пропусках в документации «Бастион-3 – Бюро пропусков. Руководство оператора»

5.11.1. Метод GetPass

Запрос на получение информации о пропуске. Запрос может быть выполнен только с использованием токена доступа.

Запрос:


```
{
  "pass_id": 147
}
```


Ответ:


```
{
  "pass": {
    "id": 147,
    "person_id": 142,
    "pass_category_id": 1,
    "status": "PASS_STATUS_RETURNED",
    "card_id": {
      "value": 413
    },
    "access_level_id": {
      "value": 121
    },
    "priority": 1,
    "create_date": {
      "seconds": "1712906231",
      "nanos": 595189000
    },
    "activation_time": null,
    "issue_date": {
      "seconds": "1712906246",
      "nanos": 799649000
    },
    "start_date": null,
    "end_date": null,
    "visit_goal_id": null,
    "confirm_person_id": null,
    "return_reason_id": {
      "value": 19
    },
    "return_date": {
      "seconds": "1712906691",
      "nanos": 877843000
    },
    "disable_anti_pass_back_check": false,
    "accept_person_id": null,
    "accept_department_id": null,
    "block_reason_id": null,
```

```
      "blocked_date": null,
      "unblocked_date": null,
      "comment": null,
      "number": null
}
```


- id – уникальный идентификатор пропуска;
- person_id – идентификатор привязанной персоны;
- pass_category_id – идентификатор категории пропуска;
- status – статус пропуска (полный список статусов можно посмотреть в файле common.proto);
- card_id – числовое значение идентификатора карты доступа, привязанный к пропуску;
- access_level_id – числовое значение идентификатора уровня доступа;
- priority – приоритет пропуска;
- create_date – дата создания пропуска;
- activation_time – дата активации пропуска;
- issue_date – дата выдачи пропуска.
- start_date – дата начала действия пропуска;
- end_date – дата окончания действия пропуска;
- visit_goal_id – идентификатор значения словаря «Цель посещения»;
- confirm_person_id – идентификатор персоны принявший заявку;
- return_reason_id – идентификатор значения словаря «Причина возврата пропуска»;
- return_date – дата возврата пропуска;
- disable_anti_pass_back_check – указатель, может ли пропуск быть повторно выдан;
- accept_person_id – идентификатор принимающего лица;
- accept_department_id – идентификатор принимающего подразделения;
- block_reason_id – идентификатор значения словаря «Причина блокировки»;
- blocked_date – дата блокировки пропуска;
- unblocked_date – дата разблокировки пропуска;
- comment – комментарий к пропуску (ограничение 100 символов);
- number – номер пропуска (ограничение 16 символов).

5.11.2. Метод GetPasses

Если необходимо получить информацию не об одном пропуске, а о нескольких, можно использовать следующий запрос, где необходимо перечислить идентификаторы искомых пропусков:


```
{
   "pass_ids": [
      147,
      110,
      109
   ]
}
```


Запрос может быть выполнен только с использованием токена доступа.

В качестве результата получаем список пропусков:


```
{
   "passes": [
      {
         "id": 109,
         "person_id": 106,
         "pass_category_id": 1,
         "status": "PASS_STATUS_ACTIVE",
         "card_id": {
            "value": 261
         },
         "access_level_id": {
            "value": 121
         },
         "priority": 1,
         "create_date": {
            "seconds": "1712223312",
            "nanos": 820706000
         },
         "activation_time": null,
         "issue_date": {
            "seconds": "1712904199",
            "nanos": 906148000
```

```
            },
            "start_date": null,
            "end_date": null,
            "visit_goal_id": null,
            "confirm_person_id": null,
            "return_reason_id": null,
            "return_date": null,
            "disable_anti_pass_back_check": false,
            "accept_person_id": null,
            "accept_department_id": null,
            "block_reason_id": null,
            "blocked_date": null,
            "unblocked_date": null,
            "comment": null
            "number": null
        },
        ...
    ]
}
```

5.11.3. Метод GetPassesForPerson

Запрос на получение всех пропусков у персоны. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
    "person_id": 107
}
```

- person_id – идентификатор персоны, пропуска которого необходимо получить.

В ответе получаем список аналогично запросу GetPasses, но в список входят только пропуска, которые относятся к указанной персоне.

5.11.4. Метод GetPassPinCode

Запрос на получение ПИН-кода пропуска. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
    "pass_id": 110
}
```

- pass_id – идентификатор пропуска, ПИН-код которого необходимо получить.

Ответ приходит с числовым значением ПИН-кода:

```
{
    "pin_code": {
        "value": "0001"
    }
}
```

5.11.5. Метод GetPassPinCodes

Запрос на получение списка ПИН-кодов у пропусков. Запрос может быть выполнен только с использованием токена доступа.

В запросе необходимо перечислить все идентификаторы пропусков, ПИН-кодов которых требуется получить:

```
{
    {
    "pass_ids": [
        124,
        110,
        143,
        144,
        109
    ]
}
```

Ответ будет содержать список указывающий, какой ПИН-код принадлежит какому пропуску:

```
{
    "pin_code_infos": [
        {
```

```
            "pass_id": 109,
            "pin_code": "0572"
        },
        {
            "pass_id": 110,
            "pin_code": "9801"
        }
    ]
}
```

- pass_id – идентификатор пропуска, которому принадлежит ПИН-код;
- pin_code – ПИН-код пропуска.

Если в запросе указаны идентификаторы пропусков, у которых нет ПИН-кода, то в ответном сообщение информации о них не будет.

5.11.6. Метод GetCardReplacementInfo

Запрос на получение списка информации о замене карты доступа. Запрос может быть выполнен только с использованием токена доступа.

Для запроса необходимо указать идентификатор пропуска, информацию о которой необходимо получить:

```
{
    "pass_id": 110
}
```

Ответное сообщение содержит список карты замены:

```
{
    "card_replacements": [
        {
            "id": 1,
            "pass_id": 142,
            "previous_card_id": 333,
            "replacement_date": {
                "seconds": "1713093000",
                "nanos": 957140000
            },
            "reason": "CARD_REPLACEMENT_REASON_RETURN",
            "return_reason_id": {
                "value": 120
            }
        },
        {
            "id": 1,
            "pass_id": 142,
            "previous_card_id": 333,
            "replacement_date": {
                "seconds": "1713093000",
                "nanos": 957140000
            },
            "reason": "CARD_REPLACEMENT_REASON_WITHDRAW",
            "withdrawal_reason": "PASS_WITHDRAWAL_REASON_UNUSABLE"
        },
        ...
    ]
}
```

- id – идентификатор карты возврата;
- pass_id – идентификатор пропуска;
- previous_card_id – идентификатор предыдущей карты доступа;
- reason – причина возврата. Карта доступа может быть либо возвращена *CARD_REPLACEMENT_REASON_RETURN*, либо изъята *CARD_REPLACEMENT_REASON_WITHDRAW;*
- return_reason_id – идентификатор записи словаря «Причины возврата пропусков». Не указывается, если карта была изъята;
- withdrawal_reason – причина изъятия пропуска. Не указывается, если карта была возвращена.

5.11.7. Метод GetPassSecurityControlGroupSettings

Запрос на получение информации об управлении охраной для пропуска. Запрос может быть выполнен только с использованием токена доступа.

Для запроса необходимо указать идентификатор пропуска, информацию о которой необходимо получить:

```
{
   "pass_id": 110
}
```
Ответное сообщение содержит информацию об управлении охраной указанного пропуска:

```
{
   "setting": {
      "pass_id": 149,
      "ops_group_id": 2,
      "pass_identification_type": "PASS_IDENTIFICATION_TYPE_BY_PIN_CODE"
   }
}
```

- pass_id – идентификатор пропуска;
- ops_grou_id – идентификатор группы управления охраной;
- pass_identification_type – тип идентификации пропуска (по номеру карты, по ПИН-коду, по ПИН-коду и номеру карты).

Если к пропуску не привязано управление охраной, то результатом запроса будет:

```
{
   "setting": null
}
```

5.11.8. Метод IssuePass

Запрос на выдачу пропуска. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
   "card_id": 373,
   "issue_date": {
      "nanos": 728198743,
      "seconds": 542916
   },
   "pass_id": 150
}
```

- pass_id – идентификатор пропуска, который нужно выдать;
- card_id – идентификатор карты доступа для выдачи;
- issue_date – дата выдачи пропуска.

Если выдача прошла успешно, то придет сообщение с пустым телом.

5.11.9. Метод ReturnPass

Запрос на возврат пропуска. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
   "pass_id": 110,
   "return_reason_id": {
      "value": 5
   }
}
```

- pass_id – идентификатор пропуска, который нужно вернуть;
- return_reason_id – идентификатор значения словаря «Причины возврата пропусков».

Если возврат прошла успешно, то придет сообщение с пустым телом.

5.11.10. Метод WithdrawPass

Запрос на изъятия пропуска. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
   "pass_id": 142,
   "reason": "PASS_WITHDRAWAL_REASON_CHARGED_OFF"
}
```

- pass_id – идентификатор пропуска, который нужно изъять;
- reason – причина изъятия:
    - PASS_WITHDRAWAL_REASON_UNSPECIFIED – не указанная причина;

- PASS_WITHDRAWAL_REASON_UNUSABLE – пропуск пришел в негодность;
- PASS_WITHDRAWAL_REASON_CHARGED_OFF – пропуск списан;
- PASS_WITHDRAWAL_REASON_LOST – пропуск утерян.

Если изъятие прошло успешно, то придет сообщение с пустым телом.

5.11.11. Метод ProlongPass

Запрос на продление пропуска. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
   "new_end_time": {
      "nanos": 244543620,
      "seconds": 369878714
   },
   "pass_id": 138,
   "prolong_reason": {
      "value": "Не указано"
   }
}
```

- pass_id – идентификатор пропуска, который нужно продлить;
- new_end_time – новая дата окончания действия пропуска;
- prolong_reason – причина продления пропуска.

Если продление прошло успешно, то придет сообщение с пустым телом.

5.11.12. Метод ReplaceCardWithReturn

Запрос на замену карты доступа, в качестве по причине возврата карты. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
   "new_card_id": 2,
   "pass_id": 110,
   "return_reason_id": 5
}
```

- pass_id – идентификатор пропуска, карту которого необходимо заменить;
- new_card_id – идентификатор карты доступа, на которую надо заменить;
- return_reason_id – идентификатор значение словаря «Причины возврата пропуска».

Если замена прошла успешно, то придет сообщение с пустым телом.

5.11.13. Метод ReplaceCardWithWithdraw

Запрос на замену карты доступа, причина которого изъятие. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
   "new_card_id": 2,
   "pass_id": 110,
   "withdraw_reason": "PASS_WITHDRAWAL_REASON_UNUSABLE"
}
```

- pass_id – идентификатор пропуска, карту которого необходимо заменить;
- new_card_id – идентификатор карты доступа, на которую надо заменить;
- withdraw_reason – причина изъятия:
  - PASS_WITHDRAWAL_REASON_UNSPECIFIED – не указанная причина;
  - PASS_WITHDRAWAL_REASON_UNUSABLE – пропуск пришел в негодность;
  - PASS_WITHDRAWAL_REASON_CHARGED_OFF – пропуск списан;
  - PASS_WITHDRAWAL_REASON_LOST – пропуск утерян.

5.11.14. Метод PassChanged

Запрос на отслеживание изменений в пропусках. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Пример ответного сообщения от события изменения пропуска:

```
{
   "pass_id": 150,
   "change_type": "ENTITY_CHANGE_TYPE_UPDATE"
}
```

- pass_id – идентификатор пропуска, который изменился;
- change_type – тип изменения пропуска:
    - ENTITY_CHANGE_TYPE_UNSPECIFIED – тип изменения не указан;
    - ENTITY_CHANGE_TYPE_ADD – пропуск создан;
    - ENTITY_CHANGE_TYPE_UPDATE – пропуск обновлен;
    - ENTITY_CHANGE_TYPE_DELETE – пропуск удален.

5.12. PassCategoryService

Файл pass_category.proto предназначен для управления категориями пропусков.

5.12.1. Метод GetPassCategory

Запрос на получение информации о категории пропуска. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
   "pass_category_id": 2
}
```

- pass_category_id – идентификатор категории пропуска, информацию о которой необходимо получить.

Ответ:

```
{
   "pass_category": {
      "id": 2,
      "name": "Контрагенты",
      "time_restriction_rule": "TIME_RESTRICTION_RULE_END",
      "time_restriction_unit": "TIME_RESTRICTION_UNIT_YEAR",
      "time_restriction_value": 1,
      "photo_identification_form_id": {
         "value": 2
      },
      "is_change_pass_end_date_allowed": true,
      "is_pass_prolongation_allowed": true,
      "numeration_id": {
         "value": 3
      },
      "inactive_days_count_before_auto_block_pass": {
         "value": 2
      },
      "activation_time_limit": {
         "value": 2
      }
   }
}
```

- id – идентификатор категории пропуска;
- name – наименование категории;
- time_restriction_rule – правило ограничения срока действия для пропусков, входящих в категорию:
    - TIME_RESTRICTION_RULE_UNSPECIFIED – не указан;
    - TIME_RESTRICTION_RULE_NONE – нет ограничения по сроку действия пропуска;
    - TIME_RESTRICTION_RULE_FULL – срок действия пропуска ограничивается заданное количество временных интервалов с момента начала действия пропуска;
    - TIME_RESTRICTION_RULE_END – срок действия пропуска ограничивается до конца временного интервала, в который попадает начало действия пропуска;

- time_ restriction_uint – тип интервала времени, применяемого при расчёте срока действия для пропусков, входящих в категорию:
  - TIME_RESTRICTION_UNIT_UNSPECIFIED – не указан;
  - TIME_RESTRICTION_UNIT_DAY – день;
  - TIME_RESTRICTION_UNIT_WEEK – неделя;
  - TIME_RESTRICTION_UNIT_MONTH – месяц;
  - TIME_RESTRICTION_UNIT_YEAR – год;
- time_restriction_value – количество интервалов времени применяемое при расчёте срока действия для пропусков, входящих в категорию;
- photo_identification_form_id – идентификатор формы фото идентификации, отображаемой для пропусков, входящих в категорию;
- is_change_pass_end_date_allowed – признак возможности ручного изменения даты окончания действия для пропусков, входящих в категорию;
- is_pass_prolongation_allowed – признак возможности пролонгации срока действия для пропусков, входящих в категорию;
- numeration_id – идентификатор нумерации пропусков;
- inactive_days_count_before_auto_block_pass – количество дней отсутствия активности пропуска до его автоматической блокировки;
- activation_time_limit – ограничение на время активации пропусков.

5.12.2. Метод GetPassCategory

Запрос на получения списка всех категорий пропусков. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Пример ответа:


```
{
   "pass_categories": [
      {
         "id": 1,
         "name": "Сотрудники",
         "time_restriction_rule": "TIME_RESTRICTION_RULE_NONE",
         "time_restriction_unit": "TIME_RESTRICTION_UNIT_DAY",
         "time_restriction_value": 1,
         "photo_identification_form_id": {
            "value": 1
         },
         "is_change_pass_end_date_allowed": true,
         "is_pass_prolongation_allowed": true,
         "numeration_id": null,
         "inactive_days_count_before_auto_block_pass": null,
         "activation_time_limit": null
      },
      {
         "id": 3,
         "name": "Посетители",
         "time_restriction_rule": "TIME_RESTRICTION_RULE_END",
         "time_restriction_unit": "TIME_RESTRICTION_UNIT_DAY",
         "time_restriction_value": 1,
         "photo_identification_form_id": {
            "value": 2
         },
         "is_change_pass_end_date_allowed": false,
         "is_pass_prolongation_allowed": false,
         "numeration_id": null,
         "inactive_days_count_before_auto_block_pass": null,
         "activation_time_limit": null
      },
      ...
   ]
}
```


5.12.3. Метод GetPassCategoriesAvailabilityInfo

Запрос на получения списка активных подразделений для категорий. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Пример ответа:

```
{
  "availability_info": [
    {
      "organization_node_ids": [
        0,
        104
      ],
      "pass_category_id": 1
    },
    {
      "organization_node_ids": [
        0,
        101,
        102,
        103,
        104,
        105
      ],
      "pass_category_id": 2
    },
    {
      "organization_node_ids": [
        100,
        103
      ],
      "pass_category_id": 3
    },
    ...
  ]
}
```

Availability_info содержит список идентификаторов доступных организаций *organization_node_ids*, для определенной категории *pass_category_id.*

5.12.4. Метод GetOrganizationAllowedPassCategories

Запрос на получение о привязанных категория пропуска к подразделению. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
  "organization_node_id": 100
}
```

- organization_node_id – идентификатор подразделения;

Ответ содержит список идентификаторов категорий пропусков, которые доступны на указанном подразделении:

```
{
  "pass_category_ids": [
    3,
    1
  ]
}
```

5.12.5. Метод PassCategoryChanged

Запрос на подписку событий изменения изменений категорий пропусков. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Ответ:

```
{
  "pass_category_id": 101,
  "change_type": "ENTITY_CHANGE_TYPE_ADD"
}
```

- pass_category_id – идентификатор категории пропуска, которая изменилась;
- change_type – тип изменения пропуска:
  - ENTITY_CHANGE_TYPE_UNSPECIFIED – тип изменения не указан;
  - ENTITY_CHANGE_TYPE_ADD – категория создана;
  - ENTITY_CHANGE_TYPE_UPDATE – категория обновлена;
  - ENTITY_CHANGE_TYPE_DELETE – категория удалена.

5.13. PersonService

Файл person.proto предназначен для работы с персонами.

5.13.1. Метод GetPerson

Запрос на получение информации о персоне. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
  "person_id": 107
}
```

- person_id – идентификатор персоны.

Пример ответа:

```
{
  "person": {
    "id": 107,
    "name": "Иванов",
    "first_name": {
      "value": "Иван"
    },
    "second_name": {
      "value": "Иванович"
    },
    "table_no": {
      "value": "aug65"
    },
    "comments": {
      "value": "Работает только до августа"
    },
    "organization_node_id": 0,
    "position_id": {
      "value": 18
    },
    "add_field_1": null,
    "add_field_2": null,
    "add_field_3": null,
    "add_field_4": {
      "value": "Женат"
    },
    "add_field_5": {
      "value": "2 детей"
    },
    "add_field_6": null,
    "add_field_7": null,
    "add_field_8": null,
    "add_field_9": null,
    "add_field_10": null,
    "add_field_11": null,
    "add_field_12": null,
    "add_field_13": null,
    "add_field_14": null,
    "add_field_15": null,
    "add_field_16": null,
    "add_field_17": null,
    "add_field_18": null,
    "add_field_19": null,
    "add_field_20": null,
    "create_date": {
      "seconds": "1712227017",
      "nanos": 748900000
    }
  }
}
```

- id – идентификатор персоны;
- name – фамилия персоны (ограничение 100 символов);
- first_name – имя персоны (ограничение 100 символов);
- second_name – отчество персоны (ограничение 100 символов);
- table_no – табельный номер персоны (ограничение 20 символов);
- comments – комментарий к персоне (ограничение 2000 символов);
- organization_node_id – идентификатор подразделения, к которой принадлежит персона;
- position_id – идентификатор должности персоны;
- add_field[1..20] – значение дополнительных полей (ограничение 255 символов);

- create_date – дата создания персоны.

5.13.2. Метод GetPersons

Запрос на получение списка персон. Запрос может быть выполнен только с использованием токена доступа.

Запрос формируется из списка идентификаторов персон, информацию о которых необходимо получить:


```
{
   "person_ids": [
      139,
      144,
      80,
      101
   ]
}
```


Ответ содержит список только тех персон, которые были найдены по указанным идентификаторам:


```
{
   "persons": [
      {
         "id": 139,
         "name": "Иван",
         "first_name": null,
         "second_name": null,
         "table_no": null,
         "comments": null,
         "organization_node_id": 0,
         "position_id": null,
         "add_field_1": null,
         "add_field_2": null,
         "add_field_3": null,
         "add_field_4": {
            "value": "Холост"
         },
         "add_field_5": {
            "value": ""
         },
         "add_field_6": null,
         "add_field_7": null,
         "add_field_8": null,
         "add_field_9": null,
         "add_field_10": null,
         "add_field_11": null,
         "add_field_12": null,
         "add_field_13": null,
         "add_field_14": null,
         "add_field_15": null,
         "add_field_16": null,
         "add_field_17": null,
         "add_field_18": null,
         "add_field_19": null,
         "add_field_20": null,
         "create_date": {
            "seconds": "1712853484",
            "nanos": 103671000
         }
      },
      {
         "id": 144,
         "name": "Дмитрий",
         "first_name": null,
         "second_name": null,
         "table_no": null,
         "comments": null,
         "organization_node_id": 0,
         "position_id": null,
         "add_field_1": null,
         "add_field_2": null,
         "add_field_3": null,
         "add_field_4": null,
         "add_field_5": null,
         "add_field_6": null,
         "add_field_7": null,
         "add_field_8": null,
         "add_field_9": null,
         "add_field_10": null,
         "add_field_11": null,
         "add_field_12": null,
         "add_field_13": null,
         "add_field_14": null,
```

```
            "add_field_15": null,
            "add_field_16": null,
            "add_field_17": null,
            "add_field_18": null,
            "add_field_19": null,
            "add_field_20": null,
            "create_date": {
                "seconds": "1712906764",
                "nanos": 685790000
            }
        }
    ]
}
```


Если ни одна персона не была найдена по идентификаторам, в ответ придет пустой список:


```
{
    "persons": []
}
```


5.13.3. Метод GetPersonalInfo

Запрос на получение персональных данных о персоне. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:


```
{
    "person_id": 107
}
```


- person_id – идентификатор персоны.

Ответ:


```
{
    "personal_info": {
        "person_id": 107,
        "birth_place": {
            "value": "Москва"
        },
        "birth_date": {
            "year": 2024,
            "month": 4,
            "day": 4
        },
        "document_type_id": {
            "value": 4
        },
        "document_number": {
            "value": "2222222"
        },
        "document_series": {
            "value": "1111111"
        },
        "document_issue_organ": {
            "value": 9
        },
        "document_issue_date": {
            "year": 2024,
            "month": 4,
            "day": 3
        },
        "phone": {
            "value": "88888888888"
        },
        "address": {
            "value": "Москва"
        },
        "citizenship_id": {
            "value": 3
        },
        "gender": "GENDER_MALE",
        "email": {
            "value": "email@email"
        },
        "additional_phone": null
    }
}
```


- person_id – идентификатор персоны;
- birth_place – место рождения (ограничение 60 символов);
- birth_date – дата рождения;

- document_type_id – идентификатор значения словаря «Виды документов»;
- document_number – номер документа (ограничение 12 символов);
- document_series – серия документа (ограничение 12 символов);
- document_issue_organ – орган выдавший документ;
- document_issue_date – дата выдачи документа;
- phone – телефон (ограничение 15 символов);
- address – адрес проживания (ограничение 255 символов);
- citizenship_id – идентификатор значения словаря «Гражданство»;
- gender – пол;
- email – электронная почта (ограничение 70 символов);
- additional_phone – дополнительный телефон (ограничение 15 символов).

Данные поля могут быть не заполнены:


```
{
   "personal_info": {
      "person_id": 107,
      "birth_place": null,
      "birth_date": null,
      "document_type_id": null,
      "document_number": null,
      "document_series": null,
      "document_issue_organ": null,
      "document_issue_date": null,
      "phone": null,
      "address": null,
      "citizenship_id": null,
      "gender": "GENDER_UNSPECIFIED",
      "email": null,
      "additional_phone": null
   }
}
```


5.13.4. Метод GetPersonalInfos

Запрос на получение списка персональных данных персон. Запрос может быть выполнен только с использованием токена доступа.

В запросе необходимо указать список идентификаторов персон, о которых требуется получить персональные данные:


```
{
   "person_ids": [
      107,
      144,
      101
   ]
}
```


Ответ содержит список с персональными данными о найденных персонах:


```
{
   "personal_infos": [
      {
         "person_id": 107,
         "birth_place": {
            "value": "Москва"
         },
         "birth_date": {
            "year": 2024,
            "month": 3,
            "day": 27
         },
         "document_type_id": {
            "value": 4
         },
         "document_number": null,
         "document_series": null,
         "document_issue_organ": {
            "value": 9
         },
         "document_issue_date": {
            "year": 2024,
            "month": 4,
            "day": 8
         },
         "phone": {
```

```
            "value": "888888"
        },
        "address": {
            "value": "Москва"
        },
        "citizenship_id": {
            "value": 3
        },
        "gender": "GENDER_MALE",
        "email": {
            "value": "email@email"
        },
        "additional_phone": null
    },
    {
        "person_id": 144,
        "birth_place": null,
        "birth_date": null,
        "document_type_id": null,
        "document_number": null,
        "document_series": null,
        "document_issue_organ": null,
        "document_issue_date": null,
        "phone": null,
        "address": null,
        "citizenship_id": null,
        "gender": "GENDER_UNSPECIFIED",
        "email": null,
        "additional_phone": null
    },
    ...
]
}
```


5.13.5. Метод GetPersonalPhoto

Запрос для получения фото персоны. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
    "person_id": 107
}
```

- person_id – идентификатор персоны.

Ответ содержит значение фотографии:

```
{
    "photo": {
        "value": "QEBAQEBAQEBAQEBAQEBAQE...”
    }
}
```

Если фото у персоны не задана придет значение null:

```
{
    "photo": null
}
```

5.13.6. Метод GetPersonPhotoHash

Запрос на получение хеша фотографии персоны. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
    "person_id": 107
}
```

- person_id – идентификатор персоны.

Ответ содержит значение хеша фотографии:

```
{
    "hash": {
        "value": "b3ed4a19a4bc556ebffe680e1cdddd61”
    }
}
```

Если фото у персоны не задана придет значение null:

```
{
    "hash": null
```

```
}
```

5.13.7. Метод GetPersonPhotoThumbnail

Запрос на получение превью фотографии. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
   "person_id": 107
}
```

- person_id – идентификатор персоны.

Ответ содержит значение хеша фотографии:

```
{
   "thumbnail": {
      "value": "/9j/4AAQSkZJRgABAQAAAQABAAD/2wCEAAEBAQEB…"
   }
}
```

Если фото у персоны не задана придет значение null:

```
{
   "thumbnail": null
}
```

5.13.8. Метод PersonChanged

Запрос на подписку изменения персон. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Ответ:

```
{
   "person_id": 101,
   "change_type": "ENTITY_CHANGE_TYPE_ADD"
}
```

- person_id – идентификатор персоны, которая изменилась;
- change_type – тип изменения персоны:
  - ENTITY_CHANGE_TYPE_UNSPECIFIED – тип изменения не указан;
  - ENTITY_CHANGE_TYPE_ADD – персона создана;
  - ENTITY_CHANGE_TYPE_UPDATE – персона обновлена;
  - ENTITY_CHANGE_TYPE_DELETE – персона удалена.

5.13.9. Метод PersonPhotoChanged

Запрос на подписку изменения фото персоны. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Ответ:

```
{
   "person_id": 101,
   "change_type": "ENTITY_CHANGE_TYPE_ADD"
}
```

- person_id – идентификатор персоны, фото которой изменилось;
- change_type – тип изменения фото:
  - ENTITY_CHANGE_TYPE_UNSPECIFIED – тип изменения не указан;
  - ENTITY_CHANGE_TYPE_ADD – фото создано;
  - ENTITY_CHANGE_TYPE_UPDATE – фото обновлено;
  - ENTITY_CHANGE_TYPE_DELETE – фото удалено.

5.13.10. Метод PersonInfoChanged

Запрос на подписку изменения персональных данных персон. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Ответ:

```
{
  "person_id": 101,
  "change_type": "ENTITY_CHANGE_TYPE_ADD"
}
```


- person_id – идентификатор персоны, персональные данные которые изменились;
- change_type – тип изменения:
  - ENTITY_CHANGE_TYPE_UNSPECIFIED – тип изменения не указан;
  - ENTITY_CHANGE_TYPE_ADD – создано;
  - ENTITY_CHANGE_TYPE_UPDATE – обновлено;
  - ENTITY_CHANGE_TYPE_DELETE – удалено.

5.14. SearchPassService

Файл search_pass.proto предназначен для поиска пропусков.

5.14.1. SearchPasses

Запрос на поиск пропусков с определенными критериями. Запрос может быть выполнен только с использованием токена доступа.

При поиске пропусков в параметрах запроса (SearchPassesRequest) можно указать дополнительные условия поиска (в поле terms), при этом сами условия поиска запаковываются в тип Any (см. Приложение А).

5.14.2. Фильтры пропусков

Поиск по атрибутам персон-владельцев пропусков:

- type_url – type.googleapis.com/esprom.taurus.grpc.v1.persons.PassByPersonAttributesSearchTerm;
- value – сообщение:


```
{
  "position": {
  "assigned": {
    "is_assigned": false
  },
  "ids": {
    "ids": [
      0
    ]
  }
 },
 "organization_node_ids": {
  "ids": [
    0
  ]
 },
 "create_date": {
  "from": {
    "seconds": 0,
    "nanos": 0
  },
  "to": {
    "seconds": 0,
    "nanos": 0
  }
 }
}
```


- position – фильтр по должности персон:
  - is_assigned – должен ли поиск осуществляться на этих должностях, или они должны быть исключены из выборки;
  - ids – идентификаторы значений словаря «Должности»;
- organization_node_ids – фильтр по указанным идентификаторам подразделения;
  - ids – список идентификаторов подразделения;
- create_date – фильтр по дате создания пропуска:
  - from – дата нижней границы поиска пропуска;
  - to – дата верхней границы поиска пропуска.

Поиск по подразделению персон-владельцев пропусков:

- type_url – type.googleapis.com/esprom.taurus.grpc.v1.persons.PassByOrganizationSearchTerm;
- value – сообщение:

```
{
  "organization_node_id": 0,
  "include_nested_nodes": false
}
```

- organization_node_id – идентификатор подразделения для поиска;
- include_nested_nodes – включать ли вложенные подразделения.

Поиск по основным атрибутам персоны:

- type_url – type.googleapis.com/esprom.taurus.grpc.v1.persons.PassByPassAttributesSearchTerm;
- value – сообщение:

```
{
 "priority": {
  "from": {
   "value": 0
  },
  "to": {
   "value": 0
  }
 },
 "statuses": {
  "statuses": [
   "PASS_STATUS_UNSPECIFIED"
  ]
 },
 "pass_category_ids": {
  "ids": [
   0
  ]
 },
 "create_date": {
  "from": {
   "seconds": 0,
   "nanos": 0
  },
  "to": {
   "seconds": 0,
   "nanos": 0
  }
 },
 "issue_date": {
  "assigned": {
   "is_assigned": false
  },
  "range": {
   "from": {
    "seconds": 0,
    "nanos": 0
   },
   "to": {
    "seconds": 0,
    "nanos": 0
   }
  }
 },
 "activation_date": {
  "assigned": {
   "is_assigned": false
  },
  "range": {
   "from": {
    "seconds": 0,
    "nanos": 0
   },
   "to": {
    "seconds": 0,
    "nanos": 0
   }
  }
 },
 "start_date": {
  "assigned": {
   "is_assigned": false
  },
  "range": {
   "from": {
```

```
      "seconds": 0,
      "nanos": 0
    },
    "to": {
      "seconds": 0,
      "nanos": 0
    }
  }
},
"end_date": {
  "assigned": {
    "is_assigned": false
  },
  "range": {
    "from": {
      "seconds": 0,
      "nanos": 0
    },
    "to": {
      "seconds": 0,
      "nanos": 0
    }
  }
},
"blocked_date": {
  "assigned": {
    "is_assigned": false
  },
  "range": {
    "from": {
      "seconds": 0,
      "nanos": 0
    },
    "to": {
      "seconds": 0,
      "nanos": 0
    }
  }
},
"unblocked_date": {
  "assigned": {
    "is_assigned": false
  },
  "range": {
    "from": {
      "seconds": 0,
      "nanos": 0
    },
    "to": {
      "seconds": 0,
      "nanos": 0
    }
  }
},
"return_date": {
  "assigned": {
    "is_assigned": false
  },
  "range": {
    "from": {
      "seconds": 0,
      "nanos": 0
    },
    "to": {
      "seconds": 0,
      "nanos": 0
    }
  }
},
"last_passage_date": {
  "assigned": {
    "is_assigned": false
  },
  "range": {
    "from": {
      "seconds": 0,
      "nanos": 0
    },
    "to": {
      "seconds": 0,
      "nanos": 0
    }
  }
},
"return_reason": {
```

```
    "assigned": {
      "is_assigned": false
    },
    "ids": {
      "ids": [
        0
      ]
    }
  },
  "block_reason": {
    "assigned": {
      "is_assigned": false
    },
    "ids": {
      "ids": [
        0
      ]
    }
  }
}
```


- priority – фильтр по приоритету пропуска:
  - from – нижняя граница приоритета;
  - to – верхняя граница приоритета;
- statuses – фильтр по включенным статусам:
  - statuses – список статусов для включения в поиск. Возможные статусы пропусков:
    - PASS_STATUS_UNSPECIFIED – не указан;
    - PASS_STATUS_NOT_ACTIVE – не активен;
    - PASS_STATUS_ACTIVE – активен;
    - PASS_STATUS_EXPIRED – срок действия истек;
    - PASS_STATUS_RETURNED – возвращен;
    - PASS_STATUS_UNUSABLE – пришел в негодность;
    - PASS_STATUS_CHARGED_OFF – списан;
    - PASS_STATUS_LOST – утерян;
    - PASS_STATUS_DENIAL – отказ в утверждении;
    - PASS_STATUS_NOT_CONFIRM – принят в утверждении;
- pass_category_ids – фильтр по категориям пропуска:
  - ids – список идентификаторов категорий пропусков для поиска;
- create_date – фильтр по дате создания пропуска:
  - from – нижняя граница поиска по дате;
  - to – верхняя граница поиска по дате;
- issue_date – фильтр по дате выдачи пропуска:
  - is_assigned:
    - если поле имеет значение *null*, то для фильтрации используется диапазон дат;
    - если поле имеет значение *true*, то будут возвращены пропуска, у которых задана дата выдачи;
    - если поле имеет значение false, то будут возвращены пропуска, которых не задана дата выдачи;
  - range – диапазон даты:
    - from – нижняя граница даты;
    - to – верхняя граница даты;
- activation_date – фильтр по дате активации пропуска:
  - is_assigned:
    - если поле имеет значение *null*, то для фильтрации используется диапазон дат;
    - если поле имеет значение *true*, то будут возвращены пропуска, у которых задана дата активации;
    - если поле имеет значение false, то будут возвращены пропуска, которых не задана дата активации;
  - range – диапазон даты:
    - from – нижняя граница даты;
    - to – верхняя граница даты;
- start_date – фильтр по дате начала действия пропуска:
  - is_assigned:
    - если поле имеет значение *null*, то для фильтрации используется диапазон дат;
    - если поле имеет значение *true*, то будут возвращены пропуска, у которых задана дата начала действия;

        - если поле имеет значение false, то будут возвращены пропуска, которых не задана дата начала действия;
    - range – диапазон даты:
        - from – нижняя граница даты;
        - to – верхняя граница даты;
- end_date – фильтр по дате окончания действия пропуска:
    - is_assigned:
        - если поле имеет значение *null*, то для фильтрации используется диапазон дат;
        - если поле имеет значение *true*, то будут возвращены пропуска, у которых задана дата окончания действия;
        - если поле имеет значение false, то будут возвращены пропуска, которых не задана дата окончания действия;
    - range – диапазон даты:
        - from – нижняя граница даты;
        - to – верхняя граница даты;
- blocked_date – фильтр по дате блокировки пропуска:
    - is_assigned:
        - если поле имеет значение *null*, то для фильтрации используется диапазон дат;
        - если поле имеет значение *true*, то будут возвращены пропуска, у которых задана дата блокировки;
        - если поле имеет значение false, то будут возвращены пропуска, которых не задана дата блокировки;
    - range – диапазон даты:
        - from – нижняя граница даты;
        - to – верхняя граница даты;
- unblocked_date – фильтр по дате разблокирования пропуска:
    - is_assigned:
        - если поле имеет значение *null*, то для фильтрации используется диапазон дат;
        - если поле имеет значение *true*, то будут возвращены пропуска, у которых задана дата разблокировки;
        - если поле имеет значение false, то будут возвращены пропуска, которых не задана дата разблокировки;
    - range – диапазон даты:
        - from – нижняя граница даты;
        - to – верхняя граница даты;
- return_date – фильтр по дате возврата пропуска:
    - is_assigned:
        - если поле имеет значение *null*, то для фильтрации используется диапазон дат;
        - если поле имеет значение *true*, то будут возвращены пропуска, у которых задана дата возврата;
        - если поле имеет значение false, то будут возвращены пропуска, которых не задана дата возврата;
    - range – диапазон даты:
        - from – нижняя граница даты;
        - to – верхняя граница даты;
- last_passage_date – фильтр по дате последнего использования пропуска:
    - is_assigned:
        - если поле имеет значение *null*, то для фильтрации используется диапазон дат;
        - если поле имеет значение *true*, то будут возвращены пропуска, у которых задана дата последнего использования;
        - если поле имеет значение false, то будут возвращены пропуска, которых не задана дата последнего использования;
    - range – диапазон даты:
        - from – нижняя граница даты;
        - to – верхняя граница даты;
- return_reason – фильтр по причинам возврата:
    - is_assigned:

        - если поле имеет значение *null*, то для фильтрации используется диапазон дат;
        - если поле имеет значение *true*, то будут возвращены пропуска, у которых задана причина возврата;
        - если поле имеет значение false, то будут возвращены пропуска, которых не задана причина возврата;
    - ids – список идентификаторов записей словаря «Причины возврата пропусков»;
- block_reason – фильтр по причинам блокировки:
    - is_assigned:
        - если поле имеет значение *null*, то для фильтрации используется диапазон дат;
        - если поле имеет значение *true*, то будут возвращены пропуска, у которых задана причина блокировки;
        - если поле имеет значение false, то будут возвращены пропуска, которых не задана причина блокировки;
    - ids – список идентификаторов записей словаря «Причины блокировки пропусков».

Поиск по статусам пропусков:

- type_url – type.googleapis.com/esprom.taurus.grpc.v1.persons.PassStatusesSearchTermEntry ;
- value – сообщение:

```
{
   "statuses": [
   "PASS_STATUS_NOT_ACTIVE",
   “PASS_STATUS_CHARGED_OFF”
  ]
}
```

- statuses – фильтр по включенным статусам. Возможные статусы пропусков:
    - PASS_STATUS_UNSPECIFIED – не указан;
    - PASS_STATUS_NOT_ACTIVE – не активен;
    - PASS_STATUS_ACTIVE – активен;
    - PASS_STATUS_EXPIRED – срок действия истек;
    - PASS_STATUS_RETURNED – возвращен;
    - PASS_STATUS_UNUSABLE – пришел в негодность;
    - PASS_STATUS_CHARGED_OFF – списан;
    - PASS_STATUS_LOST – утерян;
    - PASS_STATUS_DENIAL – отказ в утверждении;
    - PASS_STATUS_NOT_CONFIRM – принят в утверждении.

Поиск по вхождению заданных подстрок в основные параметры: ФИО, табельный номер, номер пропуска:

- type_url – type.googleapis.com/esprom.taurus.grpc.v1.persons.PassByCommonPropertiesContainWordsSearchTerm:
- value – сообщение:

```
{
   "words": [
   "Иванов",
   “Петров”
  ]
}
```

- words – фильтр ключевые слова для поиска пропусков.

Поиск пропусков по правам доступа:

- type_url – type.googleapis.com/esprom.taurus.grpc.v1.persons.PassByAccessPermissionsSearchTerm:
- value – в качестве фильтра нужно указать одно из следующих сообщений:

для поиска пропусков по заданным уровням доступа:

```
{
   "access_levels": {
   "assigned": {
     "is_assigned": false
   },
   "ids": {
```

```
    "ids": [
      0
    ]
  }
}
```

- is_assigned:
  - если поле имеет значение *null*, то для фильтрации используется список идентификаторов;
  - если поле имеет значение *true*, то будут возвращены пропуска, у которых задан идентификатор уровня доступа;
  - если поле имеет значение false, то будут возвращены пропуска, которых не задан идентификатор уровня доступа;
- ids – список идентификаторов уровней доступа.

Для поиска пропусков обладающих правом доступа на указанных территориях:

```
{
  "accessible_areas": {
    "ids": [
      0
    ]
  },
}
```

- ids – список идентификаторов территорий.

Для поиска пропусков обладающих правом прохода через указанные считыватели:

```
{
  "accessible_readers": {
    "ids": [
      0
    ]
  }
}
```

- ids – список идентификаторов считывателей (точек прохода).

Фильтр для поиска пропусков с использованием дополнительных полей:

- type_url – type.googleapis.com/esprom.taurus.grpc.v1.persons.PassByPersonAdditionalFieldsSearchTerm:
- value – сообщение:

```
{
  "additional_field_1": {
    "assigned": {
      "is_assigned": false
    },
    "substring": {
      "value": ""
    }
  },
  "additional_field_2": {
    "assigned": {
      "is_assigned": false
    },
    "substring": {
      "value": ""
    }
  },
  "additional_field_3": {
    "assigned": {
      "is_assigned": false
    },
    "substring": {
      "value": ""
    }
  },
  "additional_field_4": {
    "assigned": {
      "is_assigned": false
    },
    "substring": {
      "value": ""
    }
  },
  "additional_field_5": {
    "assigned": {
      "is_assigned": false
    },
    "substring": {
      "value": ""
    }
```

```
  },
  "additional_field_6": {
    "assigned": {
      "is_assigned": false
    },
    "substring": {
      "value": ""
    }
  },
  "additional_field_7": {
    "assigned": {
      "is_assigned": false
    },
    "substring": {
      "value": ""
    }
  },
  "additional_field_8": {
    "assigned": {
      "is_assigned": false
    },
    "substring": {
      "value": ""
    }
  },
  "additional_field_9": {
    "assigned": {
      "is_assigned": false
    },
    "substring": {
      "value": ""
    }
  },
  "additional_field_10": {
    "assigned": {
      "is_assigned": false
    },
    "substring": {
      "value": ""
    }
  },
  "additional_field_11": {
    "assigned": {
      "is_assigned": false
    },
    "substring": {
      "value": ""
    }
  },
  "additional_field_12": {
    "assigned": {
      "is_assigned": false
    },
    "substring": {
      "value": ""
    }
  },
  "additional_field_13": {
    "assigned": {
      "is_assigned": false
    },
    "substring": {
      "value": ""
    }
  },
  "additional_field_14": {
    "assigned": {
      "is_assigned": false
    },
    "substring": {
      "value": ""
    }
  },
  "additional_field_15": {
    "assigned": {
      "is_assigned": false
    },
    "substring": {
      "value": ""
    }
  },
  "additional_field_16": {
    "assigned": {
      "is_assigned": false
    },
```

```
    "substring": {
      "value": ""
    }
  },
  "additional_field_17": {
    "assigned": {
      "is_assigned": false
    },
    "substring": {
      "value": ""
    }
  },
  "additional_field_18": {
    "assigned": {
      "is_assigned": false
    },
    "substring": {
      "value": ""
    }
  },
  "additional_field_19": {
    "assigned": {
      "is_assigned": false
    },
    "substring": {
      "value": ""
    }
  },
  "additional_field_20": {
    "assigned": {
      "is_assigned": false
    },
    "substring": {
      "value": ""
    }
  }
}
```

- additional_field[1..20] – дополнительное поле пропуска для поиска:
  - is_assigned:
    - если поле имеет значение *null*, то для фильтрации используется поле *substring*;
    - если поле имеет значение *true*, то будут возвращены пропуска, у которых задано дополнительное поле соответствующего номера;
    - если поле имеет значение false, то будут возвращены пропуска, которых не задано дополнительное поле соответствующего номера;
  - substring – текстовое значение для поиска вхождения в значения дополнительных полей.

Фильтр для поиска пропусков по персональным данным обладателей пропусков:

- type_url – type.googleapis.com/esprom.taurus.grpc.v1.persons.PassByPersonalInfoSearchTerm:
- value – сообщение:

```
{
  "gender": {
    "assigned": {
      "is_assigned": false
    },
    "gender": "GENDER_UNSPECIFIED"
  },
  "birth_date": {
    "assigned": {
      "is_assigned": false
    },
    "range": {
      "from": {
        "year": 0,
        "month": 0,
        "day": 0
      },
      "to": {
        "year": 0,
        "month": 0,
        "day": 0
      }
    }
  },
  "birth_place": {
    "assigned": {
      "is_assigned": false
```

```
  },
  "substring": {
    "value": ""
  }
},
"citizenship": {
  "assigned": {
    "is_assigned": false
  },
  "ids": {
    "ids": [
      0
    ]
  }
},
"address": {
  "assigned": {
    "is_assigned": false
  },
  "substring": {
    "value": ""
  }
},
"phone": {
  "assigned": {
    "is_assigned": false
  },
  "substring": {
    "value": ""
  }
},
"email": {
  "assigned": {
    "is_assigned": false
  },
  "substring": {
    "value": ""
  }
},
"document_type": {
  "assigned": {
    "is_assigned": false
  },
  "ids": {
    "ids": [
      0
    ]
  }
},
"document_authority": {
  "assigned": {
    "is_assigned": false
  },
  "ids": {
    "ids": [
      0
    ]
  }
},
"document_serial": {
  "assigned": {
    "is_assigned": false
  },
  "substring": {
    "value": ""
  }
},
"document_number": {
  "assigned": {
    "is_assigned": false
  },
  "substring": {
    "value": ""
  }
},
"document_issue_date": {
  "assigned": {
    "is_assigned": false
  },
  "range": {
    "from": {
      "year": 0,
      "month": 0,
      "day": 0
    },
```

```
      "to": {
        "year": 0,
        "month": 0,
        "day": 0
      }
    }
  }
}
```

- gender – фильтр по гендеру владельца:
  - is_assigned:
    - если поле имеет значение *null*, то для фильтрации используется поле *gender*;
    - если поле имеет значение *true*, то будут возвращены пропуска, у которых задан гендер;
    - если поле имеет значение false, то будут возвращены пропуска, которых не задан гендер;
  - gender – значение гендера владельца:
    - GENDER_UNSPECIFIED – не указан;
    - GENDER_MALE – мужской;
    - GENDER_FEMALE – женский;
- birth_date – фильтр по дате рождения:
  - is_assigned:
    - если поле имеет значение *null*, то для фильтрации используется диапазон дат;
    - если поле имеет значение *true*, то будут возвращены пропуска, у которых задана дата рождения;
    - если поле имеет значение false, то будут возвращены пропуска, которых не задана дата рождения;
  - range – диапазон даты:
    - from – нижняя граница даты;
    - to – верхняя граница даты;
- birth_place – фильтр по месту рождения:
  - is_assigned:
    - если поле имеет значение *null*, то для фильтрации используется поле *substring*;
    - если поле имеет значение *true*, то будут возвращены пропуска, у которых задано место рождения;
    - если поле имеет значение false, то будут возвращены пропуска, которых не задано место рождения;
  - substring – текстовое значение для поиска вхождения в значения места рождения;
- citizenship – фильтр по гражданству:
  - is_assigned:
    - если поле имеет значение *null*, то для фильтрации используется список идентификаторов значений словаря «Гражданство»;
    - если поле имеет значение *true*, то будут возвращены пропуска, у которых задано гражданство;
    - если поле имеет значение false, то будут возвращены пропуска, которых не задано гражданство;
  - ids – список идентификаторов значений словаря «Гражданство»;
- birth – фильтр по адресу проживания:
  - is_assigned:
    - если поле имеет значение *null*, то для фильтрации используется поле *substring*;
    - если поле имеет значение *true*, то будут возвращены пропуска, у которых задан адрес проживания;
    - если поле имеет значение false, то будут возвращены пропуска, которых не задан адрес проживания;
  - substring – текстовое значение для поиска вхождения в значения адреса проживания;
- phone – фильтр по номеру телефона:
  - is_assigned:
    - если поле имеет значение *null*, то для фильтрации используется поле *substring*;
    - если поле имеет значение *true*, то будут возвращены пропуска, у которых задан номер телефона;
    - если поле имеет значение false, то будут возвращены пропуска, которых не задан номер телефона;
  - substring – текстовое значение для поиска вхождения в значения номера телефона;
- email – фильтр по электронная почте:
  - is_assigned:
    - если поле имеет значение *null*, то для фильтрации используется поле *substring*;
    - если поле имеет значение *true*, то будут возвращены пропуска, у которых задана электронная почта;

        - если поле имеет значение false, то будут возвращены пропуска, которых не задана электронная почта;
    - substring – текстовое значение для поиска вхождения в значения электронной почты;
- document_type – фильтр по типу документа:
    - is_assigned:
        - если поле имеет значение *null*, то для фильтрации используется список идентификаторов значений словаря «Виды документов»;
        - если поле имеет значение *true*, то будут возвращены пропуска, у которых задан тип документа;
        - если поле имеет значение false, то будут возвращены пропуска, которых не задано тип документа;
    - ids – список идентификаторов значений словаря «Виды документов»;
- document_authority – фильтр по типу документа:
    - is_assigned:
        - если поле имеет значение *null*, то для фильтрации используется список идентификаторов значений словаря «Орган, выдавший документ»;
        - если поле имеет значение *true*, то будут возвращены пропуска, у которых задан орган, выдавший документ;
        - если поле имеет значение false, то будут возвращены пропуска, которых не задан орган, выдавший документ;
    - ids – список идентификаторов значений словаря «Орган, выдавший документ»;
- document_serial – фильтр по серии документа:
    - is_assigned:
        - если поле имеет значение *null*, то для фильтрации используется поле *substring*;
        - если поле имеет значение *true*, то будут возвращены пропуска, у которых задана серия документа;
        - если поле имеет значение false, то будут возвращены пропуска, которых не задана серия документа;
    - substring – текстовое значение для поиска вхождения в значения серии документа;
- document_number – фильтр по номеру документа:
    - is_assigned:
        - если поле имеет значение *null*, то для фильтрации используется поле *substring*;
        - если поле имеет значение *true*, то будут возвращены пропуска, у которых задан номер документа;
        - если поле имеет значение false, то будут возвращены пропуска, которых не задан номер документа;
    - substring – текстовое значение для поиска вхождения в значения номера документа;
- document_issue_date – фильтр по дате выдачи документа:
    - is_assigned:
        - если поле имеет значение *null*, то для фильтрации используется диапазон дат;
        - если поле имеет значение *true*, то будут возвращены пропуска, у которых задана дата выдачи документа;
        - если поле имеет значение false, то будут возвращены пропуска, которых не задана дата выдачи документа;
    - range – диапазон даты:
        - from – нижняя граница даты;
        - to – верхняя граница даты.

Фильтр для поиска пропусков по указанному гендеру:

- type_url – type.googleapis.com/esprom.taurus.grpc.v1.persons.GenderSearchTermEntry:
- value – сообщение:

В результате поиска будут возвращены только те пропуска, гендер владельцев которых задан/не задан.

```
{
  "assigned": {
    "is_assigned": false
  }
}
```

- is_assigned:
    - если поле имеет значение *true*, то будут возвращены пропуска, у которых задан гендер;

- если поле имеет значение false, то будут возвращены пропуска, которых не задан гендер.

Для поиска с определенным гендером, используется следующее сообщение:

```
{
  "gender": "GENDER_UNSPECIFIED"
}
```

- gender – значение гендера владельца:
  - GENDER_UNSPECIFIED – не указан;
  - GENDER_MALE – мужской;
  - GENDER_FEMALE – женский.

Фильтр для поиска пропусков по атрибутам привязанной к ним карте доступа:

- type_url – type.googleapis.com/esprom.taurus.grpc.v1.persons.PassByCardAttributesSearchTerm:
- value – сообщение:

```
{
  "card_code": {
   "value": 1111
  }
}
```

- card_code – значение привязанной карты доступа. При сопоставлении учитывается системная настройка минимальной длины кода идентификации.

При использовании соответствующих фильтров, результатом будут сообщения с информацией о пропусках с указанными значениями фильтров подробнее о пропусках читайте в соответствующем разделе:

```
{
  "pass": {
     "id": 142,
     "person_id": 107,
     "pass_category_id": 1,
     "status": "PASS_STATUS_CHARGED_OFF",
     "card_id": {
        "value": 493
     },
     "access_level_id": null,
     "priority": 1,
     "create_date": {
        "seconds": "1712853460",
        "nanos": 148463000
     },
     "activation_time": null,
     "issue_date": {
        "seconds": "1712853851",
        "nanos": 446661000
     },
     "start_date": null,
     "end_date": null,
     "visit_goal_id": null,
     "confirm_person_id": null,
     "return_reason_id": null,
     "return_date": {
        "seconds": "1713094974",
        "nanos": 420667000
     },
     "disable_anti_pass_back_check": false,
     "accept_person_id": null,
     "accept_department_id": null,
     "block_reason_id": {
        "value": -3
     },
     "blocked_date": {
        "seconds": "1713110412",
        "nanos": 571347000
     },
     "unblocked_date": {
        "seconds": "1713110332",
        "nanos": 996133000
     },
     "comment": {
        "value": ""
     },
     "number": null
  }
}
```

5.15. PersonLocationService

Файл person_location.proto предназначен для отслеживания персон на территории.

5.15.1. Метод GetPersonLocationCounters

Запрос на получение информации количестве персон на определенной территории. Запрос может быть выполнен только с использованием токена доступа.

```
{
   "control_area_id": 101
}
```

- control_area_id – идентификатор территории.

Ответ:

```
{
   "counters": [
   {
     "control_area_id": 101,
     "pass_category_id": 2,
     "people_count": 34
   },
   {
     "control_area_id": 101,
     "pass_category_id": 3,
     "people_count": 2
   },
   ...
  ]
}
```

- control_area_id – идентификатор территории отслеживания;
- pass_category_id – идентификатор категории пропусков, которая находятся на территории;
- people_count – количество персон данной категории на указанной территории.

5.15.2. Метод GetPersonLocations

Запрос на получение информации о проходах категорий пропусков, на определенную территорию под определенным подразделением. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
  "pass_category_id": {
    "value": 2
  },
  "control_area_id": {
    "value": 101
  },
  "organization_node_id": {
    "value": 2
  }
}
```

- pass_category_id – идентификатор категорий пропусков;
- control_area_id – территория, на которой проходили;
- organization_node_id – подразделение к которому принадлежит проходящая персона.

Ответ:

```
{
   "locations": [
   {
     "person_id": 126,
     "pass_id": 222,
     "control_area_id": 101,
     "pass_point_sdn": {
       "value": 367
     },
     "pass_time": {
       "seconds": 0,
       "nanos": 0
     },
     "message_type_id": {
       "value": 22
     }
   },
   {
     "person_id": 34,
```

```
    "pass_id": 123,
    "control_area_id": 101,
    "pass_point_sdn": {
     "value": 0
    },
    "pass_time": {
     "seconds": 0,
     "nanos": 0
    },
    "message_type_id": {
     "value": 22
    }
   },
   ...
  ]
}
```

- person_id – идентификатор проходящей персоны;
- pass_id – идентификатор используемого пропуска;
- control_area_id – идентификатор области контроля;
- pass_point_sdn – идентификатор точки прохода;
- pass_time – время прохода;
- message_type_id – идентификатор типа события о проходе.

5.15.3. Метод PersonLocationChanged

Запрос на подписку отслеживания прохода. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Ответ:

```
{
  "locations": [
   {
    "person_id": 126,
    "pass_id": 222,
    "control_area_id": 101,
    "pass_point_sdn": {
     "value": 367
    },
    "pass_time": {
     "seconds": 0,
     "nanos": 0
    },
    "message_type_id": {
     "value": 22
    }
   },
   {
    "person_id": 34,
    "pass_id": 123,
    "control_area_id": 101,
    "pass_point_sdn": {
     "value": 0
    },
    "pass_time": {
     "seconds": 0,
     "nanos": 0
    },
    "message_type_id": {
     "value": 22
    }
   },
   ...
  ]
}
```

5.15.4. Метод PersonLocationCounterChanged

Запрос на подписку изменения количества персон на территориях. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Ответ:

```
{
  "counters": [
   {
```

```
      "control_area_id": 101,
      "pass_category_id": 2,
      "people_count": 34
    },
    {
      "control_area_id": 101,
      "pass_category_id": 3,
      "people_count": 2
    },
    ...
  ]
}
```

5.16. SecurityControlGroupService

Файл security_control_group.proto предназначен для получения информации о групп управления охраной.

5.16.1. Метод GetSecurityControlGroups

Запрос на получения полного списка групп управления охраной. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Пример ответа:

```
{
  "groups": [
    {
      "id": 1,
      "name": "Программная ГУО"
    },
    {
      "id": 2,
      "name": "Программная ГУО 2"
    },
    {
      "id": 3,
      "name": "Программная ГУО 3"
    },
    ...
  ]
}
```

- id – идентификатор группы управления охраной;
- name – наименование группы управления охраной.

5.17. StopListService

Файл stop_list.proto предназначен для управления стоп-листа.

5.17.1. Метод GetBlockedPersons

Запрос на получение персон, находящихся в стоп-листе. Запрос может быть выполнен только с использованием токена доступа.

Для получения определенного списка персон требуется использовать запрос с указанием идентификаторов персон:

```
{
  "by_person_ids": {
    "person_ids": [
      141,
      108
    ]
  }
}
```

В ответ войдут все персоны находящиеся в стоп-листе с указанными идентификаторами:

```
{
  "persons": [
    {
      "person_id": 107,
      "block_date": {
        "seconds": "1713109392",
        "nanos": 815363000
      },
      "reason": {
        "value": "в отпуске"
```

```
            }
        },
        {
            "person_id": 141,
            "block_date": {
                "seconds": "1713109400",
                "nanos": 74101000
            },
            "reason": {
                "value": "не указано"
            }
        }
    ]
}
```


- person_id – идентификатор персоны;
- block_date – дата добавления в стоп-лист;
- reason – причина добавления в стоп-лист.

Для получения полного списка персон, требуется отправить следующий запрос:


```
{
    "empty": {}
}
```


5.17.2. Метод GetBlockedPersonByPersonId

Запрос на получение информации о блокировки персоны с помощью идентификатора. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:


```
{
    "person_id": 107
}
```


Ответ:


```
{
    "person": {
        "person_id": 107,
        "block_date": {
            "seconds": "1713109887",
            "nanos": 373867000
        },
        "reason": {
            "value": "в отпуске"
        }
    }
}
```


5.17.3. Метод AddPersonToStopList

Запрос на добавление персоны в стоп-лист. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:


```
{
    "person_id": 107
    "reason": {
        "value": "длительный отпуск"
    }
}
```


- person_id – идентификатор персоны, которую нужно отправить в стоп-лист;
- reason – причина добавления в стоп-лист.

При успешной отправки персоны в стоп-лист, получим пустое сообщение.

5.17.4. Метод RemovePersonFromStopList

Запрос на удаление персоны из стоп-листа. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:


```
{
    "person_id": 107
}
```

- person_id – идентификатор персоны, которую нужно удалить из стоп-листа.

При успешном удалении персоны из стоп-листа, получим пустое сообщение.

5.17.5. Метод BlockedPersonChanged

Запрос на подписку об изменениях в стоп-листе. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

При изменении в стоп-листе получим следующее сообщение:

```
{
   "person_id": 107,
   "change_type": "ENTITY_CHANGE_TYPE_UPDATE"
}
```

- person_id – идентификатор персоны, которую добавили или удалили из стоп-листа;
- change_type – тип изменения стоп-листа (всегда будет иметь значение ENTITY_CHANGE_TYPE_UPDATE – обновлен).

5.18. TimeBlockService

Файл time_block.proto для получения информации о временных блоках.

5.18.1. Метод GetTimeBlock

Запрос на получение информации о временном блоке. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
   "time_block_id": 1
}
```

- time_block_id – идентификатор временного блока.

Ответное сообщение содержит всю информацию о временном блоке:

```
{
   "time_block": {
      "time_zones": [
         {
            "start_time": {
               "hours": 0,
               "minutes": 0,
               "seconds": 0,
               "nanos": 0
            },
            "end_time": {
               "hours": 23,
               "minutes": 59,
               "seconds": 59,
               "nanos": 0
            },
            "work_days_mask": 127,
            "holiday_mask": 3
         }
      ],
      "id": 11,
      "physical_number": 10,
      "is_weak": false,
      "name": "Скользящий блок",
      "consider_holidays": false,
      "period": 2,
      "start_date": {
         "year": 2024,
         "month": 4,
         "day": 14
      }
   }
}
```

- time_zones – содержит список временных зон:
  - start_time – время начала действия временной зоны:
    - hours – час;
    - minutes – минута;

        - seconds – секунда;
        - nanos – миллисекунда;
    - end_time – время окончания действия временной зоны;
    - work_days_mask – рабочие дни в рабочей неделе;
    - holiday_mask – рабочие дни в выходные;
- id – идентификатор временного блока;
- physical_number – физический номер;
- is_weak – указатель на автоматическое создание;
- name – наименование временного блока (ограничение 100 символов);
- consider_holidays – учитывать ли праздничные дни;
- peroid – периодичность, относится ко скользящему временному блоку;
- start_date – дата с которой будет вестись расчет периодичности скользящего блока.

5.18.2. Метод GetTimeBlocks

Запрос на получение полного списка временных блоков. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Пример ответа:


```
{
  "time_blocks": [
    {
      "time_zones": [
        {
          "start_time": {
            "hours": 15,
            "minutes": 5,
            "seconds": 0,
            "nanos": 0
          },
          "end_time": {
            "hours": 23,
            "minutes": 59,
            "seconds": 59,
            "nanos": 0
          },
          "work_days_mask": 49,
          "holiday_mask": 1
        }
      ],
      "id": 1,
      "physical_number": 1,
      "is_weak": false,
      "name": "Круглосуточно (№1)",
      "consider_holidays": false,
      "period": 7,
      "start_date": null
    },
    {
      "time_zones": [
        {
          "start_time": {
            "hours": 0,
            "minutes": 0,
            "seconds": 0,
            "nanos": 0
          },
          "end_time": {
            "hours": 23,
            "minutes": 59,
            "seconds": 59,
            "nanos": 0
          },
          "work_days_mask": 71,
          "holiday_mask": 3
        }
      ],
      "id": 3,
      "physical_number": 2,
      "is_weak": true,
      "name": "Блок №2 (авто)",
      "consider_holidays": true,
      "period": 7,
      "start_date": null
```

```
        },
        ...
    ]
}
```

5.18.3. Метод TimeBlockChanged

Запрос на подписку об изменениях во временных блоках. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Пример ответа:

```
{
    "time_block_id": 101,
    "change_type": "ENTITY_CHANGE_TYPE_ADD"
}
```

- time_block_id – идентификатор временного блока, который изменился;
- change_type – тип изменения:
    - ENTITY_CHANGE_TYPE_UNSPECIFIED – тип изменения не указан;
    - ENTITY_CHANGE_TYPE_ADD – временной блок создан;
    - ENTITY_CHANGE_TYPE_UPDATE – временной блок обновлен;
    - ENTITY_CHANGE_TYPE_DELETE – временной блок удален.

5.19. AdditionalFieldService

Файл additional_field.proto предназначен для получения информации о дескрипторах дополнительных полей драйвера.

5.19.1. GetAdditionalFieldDescriptor

Запрос на получение информации о дескрипторе. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
    "descriptor_id": 1
}
```

- descriptor_id – идентификатор дескриптора;

Ответ:

```
{
    "field_descriptor": {
        "id": 1,
        "name": "деск",
        "description": {
            "value": "порт"
        }
    }
}
```

- id – идентификатор дескриптора;
- name – наименование дескриптора;
- description – назначение дескриптора.

5.19.2. GetAdditionalFieldDescriptors

Запрос на получение полного списка дескрипторов. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Ответ содержит список дескрипторов в системе:

```
{
    "descriptors": [
        {
            "id": 1,
            "name": "деск 1",
            "description": {
                "value": "порт"
            }
```

```
        },
        {
            "id": 2,
            "name": "деск 2",
            "description": {
                "value": "наличие доступа"
            }
        },
        ...
    ]
}
```

5.19.3. GetAdditionalFieldValues

Запрос для получения значения дескриптора. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса, если значения дескриптора нужно только от одного устройства:

```
{
    "by_device": {
        "sdn": 101
    }
}
```

- sdn – идентификатор устройства.

Если необходимо значения дескрипторов от нескольких устройств, используется следующий запрос, где необходимо перечислить идентификаторы устройств:

```
{
    "by_devices": {
        "sdns": [
            101,
            365
        ]
    }
}
```

Так же можно получить значения дескрипторов для всех устройств экземпляра драйвера:

```
{
    "by_driver": {
        "din": 100
    }
}
```

- din – идентификатор драйвера.

Результат любого из запросов будет список значений дескрипторов:

```
{
    "values": [
        {
            "sdn": 365,
            "descriptor_id": 1,
            "boolean": true
        },
        {
            "sdn": 365,
            "descriptor_id": 2,
            "integer": "5432"
        }
    ]
}
```

- sdn – идентификатор устройства.
- descriptor_id – идентификатор дескриптора;
- Следующее поле может отличаться в зависимости от того, какой тип значений указан в дескрипторе:
    - boolean – булево значение;
    - integer – целое числовое;
    - float – дробное число;
    - string – текст;
    - date_time – дата и время;
    - date – дата;
    - time_span – время.

5.19.4. GetAdditionalFieldValueChanged

Запрос на подписку об изменениях дескрипторах. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Пример ответа:

```
{
    "value": {
    "sdn": 345,
    "descriptor_id": 101,
    "boolean": false,
  },
  "change_type": "ENTITY_CHANGE_TYPE_ADD"
}
```

- sdn – идентификатор устройства связанного с дескриптором;
- descriptor_id – идентификатор дескриптора, который изменился;
- change_type – тип изменения:
    - ENTITY_CHANGE_TYPE_UNSPECIFIED – тип изменения не указан;
    - ENTITY_CHANGE_TYPE_ADD – дескриптор создан;
    - ENTITY_CHANGE_TYPE_UPDATE – дескриптор обновлен;
    - ENTITY_CHANGE_TYPE_DELETE – дескриптор удален.

5.20. DriverInfoService

Файл driver_info.proto предназначен для получения информации о драйверах и устройствах.

5.20.1. Глоссарий

5.20.1.1. Тип драйвера

Класс **esprom.taurus.grpc.v1.drivers.DriverType**.

Соответствует отдельной интегрируемой в Бастион-3 системе, например: Elsys, С2000, Заря, Macroscop и т.п. Получения списка установленных/поддерживаемых системой типов драйверов доступно через метод **GetDriverTypes**.

Основные поля:

- driver_Id – уникальный идентификатор, константа, задаётся при начале разработки интеграции со сторонней системой;
- name – название типа драйвера;
- version – версия типа драйвера.

5.20.1.2. Базовый тип устройства

Сейчас отсутствует в gRPC, по сути является статичным справочником. Описывает абстрактный тип устройства, например: дверь, турникет, контроллер, считыватель, камера и т.п. Список доступных типов можно просмотреть в БД в таблице *driver.device_types*;

5.20.1.3. Сервер (хост) оборудования

Класс **esprom.taurus.grpc.v1.drivers.DriverHost**.

Информация об отдельном компьютере/сервере в системе, который может выполнять роль серверов оборудования для конкретного типа драйверов. Получение списка доступно через метод **GetDriverHosts**. Список серверов оборудования всегда содержит хост, на котором работает сервер системы.

Основные поля:

- socket_id – уникальный идентификатор сервера оборудования, генерируется при добавлении сервера оборудования в систему;
- name – сетевое имя или IP-адрес сервера оборудования.

5.20.1.4. Экземпляр (инстанс) драйвера

Класс **esprom.taurus.grpc.v1.drivers.DriverInstance**.

Экземпляр конкретного типа драйвера; в зависимости от типа драйвера в систему может быть добавлен один, один на сервер оборудования и несколько экземпляров (для масштабирования) драйвера. Получение списка доступно через метод **GetDriverInstances**.

Основные поля:

- din – уникальный идентификатор экземпляра драйвера, генерируется при добавлении экземпляра в систему;
- socket_id – идентификатор сервера оборудования, к которому относится (на котором должен работать) экземпляр драйвера; Если поле не задано, то экземпляр всегда относится к хосту - серверу системы (в основном это экземпляры системных типов драйверов);
- driver_id – идентификатор типа драйвера, за интеграция устройств которого отвечает данный экземпляр драйвера;
- name – имя экземпляра драйвера.

5.20.1.5. Устройство

Класс **esprom.taurus.grpc.v1.drivers.Device**.

Устройство интегрируемое в систему рамках конкретного экземпляра драйвера, может являться непосредственным источником событий или выполнять команды. Получение списка доступно через метод **GetDriverInstanceDevices**, либо через сервис поиска устройств (**esprom.taurus.grpc.v1.drivers.SearchDeviceService**).

Основные поля:

- sdn – уникальный идентификатор устройства, генерируется при добавлении устройства в систему;
- parent_sdn – идентификатор родительского устройства, используется для построения иерархии устройств. В рамках одного экземпляра драйвера всегда есть только одно корневое устройство – логическое устройство экземпляра драйвера, для которого не задан parent_sdn, для остальных устройств экземпляра драйвера это поле обязательно имеет значение;
- din – идентификатор экземпляра драйвера, к которому относиться устройство;
- name – имя устройства.

5.20.1.6. Тип события устройства

Класс **esprom.taurus.grpc.v1.drivers.MessageType**.

Описание конкретного типа событий, который может генерироваться устройствами заданного типа драйвера с заданным базовым типом. Получение списка типов событий, относящихся к конкретному типу драйверов, доступно через метод **GetDriverMessageTypes,** либо через анализ ddt-файлов.

Основные поля:

- driver_id – идентификатор типа драйвера, к которому относится устройство, которое может генерировать данный тип события (далее устройство-источник)

- device_type – код базового типа устройства, к которому относится устройство-источник;
- code – уникальный в рамках типа драйвера и базового типа устройства код для типа событий;
- text – шаблон текста сообщения о событии;
- message_profile_id – идентификатор профиля событий (ссылка на **MessageProfile**).

5.20.1.7. Тип команды/действия устройства

Класс **esprom.taurus.grpc.v1.drivers.ActionType**.

Описание конкретного типа команды, которое поддерживается устройствами заданных базового типа и типа драйвера. Получение списка команд, относящихся к конкретному типу драйверов, доступно через метод **GetDriverActionTypes**, либо через анализ ddt-файлов.

Основные поля:

- driver_id – идентификатор типа драйвера, к которому относится устройство, которое может выполнять данный тип команды (далее целевое устройство)
- device_type – код базового типа устройства, к которому относится целевое устройство;
- code – уникальный в рамках типа драйвера и базового типа устройства код для типа событий;
- name – имя/название команды.

5.20.2. Метод GetDriverHosts

Запрос на получение списка серверов оборудования. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Пример ответа:

```
{
   "driver_hosts": [
      {
         "socket_id": 100,
         "name": "Server"
      },
      {
         "socket_id": 110,
         "name": "DESKTOPLinux"
      },
      ...
   ]
}
```

- socket_id – идентификатор сервера оборудования;
- name – наименование сервера.

5.20.3. Метод GetDriverTypes

Запрос получение списка о типах драйверов. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
   "include_system_types": false
}
```

- include_system_types – стоит ли включать в список типы системных драйверов.

Пример ответа:

```
{
   "driver_types": [
      {
         "driver_id": 12,
         "name": "Бастион-3 - Face",
```

```
            "version": {
                "value": "2.0.1"
            }
        },
        {
            "driver_id": 33,
            "name": "Бастион-3 - Алкорамка",
            "version": {
                "value": "2024.1"
            }
        },
        {
            "driver_id": 43,
            "name": "Бастион-3 - Сириус",
            "version": {
                "value": "1.0.1"
            }
        }
    ]
}
```

- driver_id – идентификатор драйвера;
- name – наименование драйвера;
- version – версия драйвера.

5.20.4. Метод GetDriverInstances

Запрос для получения экземпляра драйверов. Запрос может быть выполнен только с использованием токена доступа.

Для получения списка экземпляров типов драйвера с указанного сервера оборудования, нужно использовать запрос с указание идентификатора сервера оборудования:

```
{
    "socket_id": {
        "value": 100
    }
}
```

Для получения экземпляров драйверов определенного типа, нужно использовать запрос с указанием идентификатора драйвера:

```
{
    "driver_id": {
        "value": 88888
    }
}
```

Эти запросы можно объединять, таким образом получить экземпляры драйверов определенного типа на указанном сервере оборудования:

```
{
    "driver_id": {
        "value": 88888
    },
    "socket_id": {
        "value": 100
    }
}
```

Результатом всегда будет список типов экземпляров драйвера:

```
{
    "driver_instances": [
        {
            "din": 100,
            "socket_id": {
                "value": 100
            },
            "driver_id": 12,
            "name": "Face"
        },
        {
            "din": 101,
            "socket_id": {
                "value": 100
            },
            "driver_id": 33,
            "name": "Алкорамка"
        },
```

```
        {
            "din": 102,
            "socket_id": {
                "value": 100
            },
            "driver_id": 43,
            "name": "Сириус"
        }
    ]
}
```

- din – идентификатор экземпляра драйвера;
- socket_id – идентификатор сервера оборудования;
- driver_id – идентификатор типа драйвера;
- name – наименование экземпляра драйвера.

5.20.5. Метод GetDriverInstanceDevice

Запрос на получение устройств экземпляра драйвера. Запрос может быть выполнен только с использованием токена доступа. Указание идентификатора экземпляра драйвера является обязательным для всех запросов.

Если нужно найти все устройства определенного типа, то нужно использовать следующий запрос:

```
{
    "by_deice_type": {
        "value": 101
    },
    "din": 100
}
```

- din – идентификатор экземпляра драйвера;
- by_device_type – поиск с использованием идентификатора типа драйвера.

Поиск можно осуществить с использованием идентификаторов устройств:

```
{
    "by_device_sdns": {
        "ids": [
            345,
            101,
            23,
            34
        ]
    },
    "din": 100
}
```

- by_device_sdns – поиск с использованием идентификаторов устройств.

Если необходимо найти устройства различных типов драйвера, то нужно указать список типов драйверов для поиска:

```
{
    "by_device_types": {
        "ids": [
            22,
            33
        ]
    },
    "din": 100
}
```

- by_device_types – поиск с использованием идентификаторов типов драйверов.

Результатом будет набор сообщений содержащий в себе информацию об устройстве:

```
{
    "device": {
        "sdn": 376,
        "din": 100,
        "parent_sdn": {
            "value": 201
        },
        "name": "Сервер внешней системы 4",
        "type": 26,
        "address": 0,
        "is_active": true,
        "time_zone_code": 0
    }
}
```

- sdn – идентификатор устройства;
- din – идентификатор экземпляра драйвера;
- parent_sdn – идентификатор родительского устройства;

- name – наименование устройства;
- type – тип устройства;
- address – адрес устройства;
- is_active – активно ли устройство;
- time_zone_code – код временной зоны, где находится устройство.

5.20.6. Метод GetDevice

Запрос на получение информации об устройстве. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
   "sdn": 364
}
```

- sdn – идентификатор устройства.

Ответ содержит информацию об устройстве:

```
{
   "device": {
      "sdn": 364,
      "din": 100,
      "parent_sdn": {
         "value": 201
      },
      "name": "Виртуальная точка прохода 1",
      "type": 3,
      "address": 0,
      "is_active": true,
      "time_zone_code": 0
   }
}
```

5.20.7. Метод GetDriverActionTypes

Запрос на получение всех типов действия драйверов. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
   "driver_id": 22
}
```

- driver_id – идентификатор типа драйвера.

Вами будут получены сообщения о возможных действиях данного типа драйвера:

```
{
   "action_type": {
      "driver_id": 22,
      "device_type": 3,
      "code": 2,
      "name": "Нормальный режим"
   }
}
```

- driver_id – идентификатор типа драйвера.
- device_type – тип устройства;
- code – код действия;
- name – наименование действия.

5.20.8. Метод GetDriverMessageTypes

Запрос на получение о возможных событиях от экземпляров типа драйвера. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
   "driver_id": 22
}
```

- driver_id – идентификатор типа драйвера.

Ответные сообщения содержат информацию о возможном событиях устройства указанного типа драйвера:


```
{
   "message_type": {
      "driver_id": 22,
      "device_type": 3,
      "code": 3,
      "text": "Штатный выход %s1",
      "message_profile_id": 5,
      "pass_direction": "PASS_DIRECTION_TYPE_UNSPECIFIED",
      "credential_check_result": "CREDENTIAL_CHECK_RESULT_UNSPECIFIED",
      "credential_check_result_group": null
   }
}
```


- driver_id – идентификатор типа драйвера;
- device_type – тип устройства;
- code – код типа события;
- text – информационное сообщение;
- message_profile_id – идентификатор профиля сообщения;
- pass_direction – направление прохода:
  - PASS_DIRECTION_TYPE_UNSPECIFIED – не указано;
  - PASS_DIRECTION_TYPE_IN – вход;
  - PASS_DIRECTION_TYPE_OUT – выход.
- credential_check_result – результат проверки полномочий на доступ пользователя СКУД через точку прохода:
  - CREDENTIAL_CHECK_RESULT_UNSPECIFIED – не определен;
  - CREDENTIAL_CHECK_RESULT_ACCESS_VERIFIED – пропуск имеет полномочия на доступ;
  - CREDENTIAL_CHECK_RESULT_ACCESS_DENIED – пропуск не имеет полномочия на доступ;
  - CREDENTIAL_CHECK_RESULT_UNKNOWN_CREDENTIAL – карта неизвестна;
  - CREDENTIAL_CHECK_RESULT_BLOCKED_PASS – пропуск заблокирован;
  - CREDENTIAL_CHECK_RESULT_EXPIRED_PASS – пропуск просрочен;
  - CREDENTIAL_CHECK_RESULT_TIME_ZONE_VIOLATION – пропуск не имеет полномочия на доступ из-за нарушения временной зоны;
  - CREDENTIAL_CHECK_RESULT_PENDING_INITIALIZATION – пропуск имеет полномочие, но не был доставлен в контроллер.
- credential_check_result_group – строковый ключ для группы результатов проверки полномочий.

Примечание! Для определения успешного прохода необходимо смотреть на поле *credential_check_result*, оно должно иметь значение *CREDENTIAL_CHECK_RESULT_ACCESS_VERIFIED*. Проход считается не успешным если значение *credential_check_result* не равно значению *CREDENTIAL_CHECK_RESULT_ACCESS_VERIFIED* и не равно значению *CREDENTIAL_CHECK_RESULT_UNSPECIFIED*. При этом проход возможен только если значение поля *pass_direction* не равно *PASS_DIRECTION_TYPE_UNSPECIFIED*.

5.20.9. Метод ExecuteDeviceCommand

Запрос на выполнение действия. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:


```
{
   "action_type_code": 13978182,
   "parameters": {
      "value": ""
   },
   "sdn": 354
}
```


- action_type_code – код типа события;
- parameters – строковое значение для передачи параметров события;
- sdn – идентификатор устройства, на котором произошло событие.

При успешном выполнении придет пустое сообщение.

5.20.10. Метод GetDeviceState

Запрос на получение состояния устройства. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
   "sdn": 354
}
```

- sdn – идентификатор устройства, состояние которого запрашивается.

Пример ответа:

```
{
   "state": {
      "simple": {
         "id": 0,
         "code": 0,
         "time": {
            "seconds": "1713131098",
            "nanos": 524790300
         }
      }
   }
}
```

- id – идентификатор состояния;
- code – код состояния;
- time – время наступления текущего состояния.

5.20.11. Метод ConfigurationChanged

Запрос на отслеживание изменений конфигураций устройств. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Пример ответного сообщения:

```
{
   "Changes": {
      "889222": {
         "Changes": {
            "100": {
               "new_devices": [
                  377,
                  378,
                  379,
                  380
               ],
               "updated_device": [],
               "deleted_device": [],
               "din": 100
            }
         },
         "driver_id": 889222
      }
   },
   "driver_instances_changed": false
}
```

- Changes – содержит информацию какие драйвера затронули изменения:
  - Changes – содержит информацию об изменениях экземпляров драйвера:
    - new_device – содержит идентификаторы новых устройств драйвера;
    - updated_device – содержит идентификаторы обновленных устройств;
    - deleted_device – содержит информацию об удаленных устройствах;
    - din – идентификатор типа экземпляра драйвера;
  - driver_id – идентификатор экземпляра драйвера;
- driver_instances_changed – был ли изменен экземпляр драйвера.

5.21. SearchDevicesService

Файл search_device.proto предназначен для поиска устройств сервера оборудования.

5.21.1. SearchDevices

Запрос на поиск устройств с определенными критериями. Запрос может быть выполнен только с использованием токена доступа.

Для поиска устройств требуется указать список *url* указывающий, какой используется фильтр и его значение:

```
{
   "terms": [
      {
         "value": "Ql6GWYh9YSqhbZld+CQeSDC2xYiA",
         "type_url": "type.googleapis.com/esprom.taurus.grpc.v1.drivers.DeviceByMainPropertiesSearchTerm"
      },
      {
         "value": "JpwDNsSJDX/kpbs+qENitO",
         "type_url": "type.googleapis.com/esprom.taurus.grpc.v1.drivers.DeviceByAdditionalFieldSearchTerm"
      },
      ...
   ]
}
```

- type_url – url для указания типа фильтрации пропусков;
- value – закодированное в Base64 значение фильтров.

5.21.2. Фильтры устройств

Поиск устройств по их основным атрибутам:

- type_url – type.googleapis.com/esprom.taurus.grpc.v1.drivers.DeviceByMainPropertiesSearchTerm;
- value – сообщение:

Запрос поиска по типам драйвера:

```
{
   "device_types": {
    "ids": [
      0
    ]
   },
   "by_types": {
    "ids": [
      0
    ]
   }
}
```

- device_types – фильтр по типам устройств:
  - ids – идентификаторы типов для поиска;
- by_types – фильтр по типам драйвера;
  - ids – список идентификаторов типов драйвера.

Если необходимо найти устройства по экземплярам драйвера, то в запросе вместо типов драйверов используется идентификаторы экземпляров драйверов:

```
{
   "device_types": {
    "ids": [
      0
    ]
   },
   "by_instances": {
    "ids": [
      0
    ]
   }
}
```

- by_instance – фильтр по экземплярам драйвера:
  - ids – список идентификаторов экземпляров драйверов.

Поиск устройств по вхождению заданных подстрок в имя устройства:

- type_url – type.googleapis.com/esprom.taurus.grpc.v1.drivers.DeviceByCommonPropertiesContainWordsSearchTerm;
- value – сообщение:

```
{
   "words": [
   "дверь на кухню",
   “ворота”
  ]
}
```

- words – список текстовых значений, которые будут проверятся на вхождение в имя устройств.

Поиск устройств по их дополнительным полям:

- type_url – type.googleapis.com/esprom.taurus.grpc.v1.drivers.DeviceByAdditionalFieldSearchTerm;
- value – сообщение:

Для поиска по значению дополнительного поля целочисленного типа:

```
{
  "descriptor_id": 0,
  "integer_value": {
    "assigned": {
      "is_assigned": false
    },
    "range": {
      "from": {
        "value": 0
      },
      "to": {
        "value": 0
      }
    }
  }
}
```

- descriptor_id – идентификатор дескриптора дополнительного поля;
- integer_value – фильтр по значению дополнительного поля целочисленного типа:
  - is_assigned:
    - если поле имеет значение *null*, то для фильтрации используется диапазон значения;
    - если поле имеет значение *true*, то будут возвращены пропуска, у которых задано числовое значение;
    - если поле имеет значение false, то будут возвращены пропуска, которых не задан числовое значение;
  - range – диапазон числовых значений:
    - from – нижняя граница значения;
    - to – верхняя граница значения.

Для поиска по значению дополнительного поля для чисел с плавающей точкой:

```
{
  "descriptor_id": 0,
  "float_value": {
  "assigned": {
    "is_assigned": false
  },
  "range": {
    "from": {
      "value": 0
    },
    "to": {
      "value": 0
    }
  }
 }
}
```

- descriptor_id – идентификатор дескриптора дополнительного поля;
- integer_value – фильтр по значению дополнительного поля для чисел с плавающей точкой:
  - is_assigned:
    - если поле имеет значение *null*, то для фильтрации используется диапазон значения;
    - если поле имеет значение *true*, то будут возвращены пропуска, у которых задано числовое значение;
    - если поле имеет значение false, то будут возвращены пропуска, которых не задан числовое значение;
  - range – диапазон числовых значений:
    - from – нижняя граница значения;
    - to – верхняя граница значения.

Для поиска по значению дополнительного поля строчного типа:

```
{
  "descriptor_id": 0,
```

```
  "float_value": {
  "assigned": {
    "is_assigned": false
  },
  "range": {
    "from": {
      "value": 0
    },
    "to": {
      "value": 0
    }
  }
 }
}
```

- descriptor_id – идентификатор дескриптора дополнительного поля;
- integer_value – фильтр по значению дополнительного поля строчного типа:
  - is_assigned:
    - если поле имеет значение *null*, то для фильтрации используется параметр *substring*;
    - если поле имеет значение *true*, то будут возвращены пропуска, у которых задано строчное значение;
    - если поле имеет значение false, то будут возвращены пропуска, которых не задано строчное значение;
  - substring – строчное значение, которое будет использоваться для поиска вхождения в строковое поле.

Для поиска по значению дополнительного поля логического типа:

```
{
  "descriptor_id": 0,
  "boolean_value": {
    "assigned": {
      "is_assigned": false
    },
    "value": false
  },
}
```

- descriptor_id – идентификатор дескриптора дополнительного поля;
- boolean_value – фильтр по значению дополнительного поля логического типа:
  - is_assigned:
    - если поле имеет значение *null*, то для фильтрации используется параметр *value*;
    - если поле имеет значение *true*, то будут возвращены пропуска, у которых задано логическое значение;
    - если поле имеет значение false, то будут возвращены пропуска, которых не задано логическое значение;
  - value – логическое значение поля.

Для поиска по значению дополнительного поля даты и времени:

```
{
  "descriptor_id": 0,
  "date_time_value": {
  "assigned": {
    "is_assigned": false
  },
  "range": {
    "from": {
      "seconds": 0,
      "nanos": 0
    },
    "to": {
      "seconds": 0,
      "nanos": 0
    }
  }
 }
}
```

- descriptor_id – идентификатор дескриптора дополнительного поля;
- date_time_value – фильтр по дополнительному полю даты и времени:
  - is_assigned:

        - если поле имеет значение *null*, то для фильтрации используется диапазон дат;
        - если поле имеет значение *true*, то будут возвращены пропуска, у которых задана дата;
        - если поле имеет значение false, то будут возвращены пропуска, которых не задана дата;
    - range – диапазон даты:
        - from – нижняя граница даты;
        - to – верхняя граница даты.

Для поиска по значению дополнительного поля даты:

```
{
  "descriptor_id": 0,
  "date_value": {
  "assigned": {
    "is_assigned": false
  },
  "range": {
    "from": {
      "year": 0,
      "month": 0,
      "day": 0
    },
    "to": {
      "year": 0,
      "month": 0,
      "day": 0
    }
  }
 }
}
```

- descriptor_id – идентификатор дескриптора дополнительного поля;
- date_value – фильтр по дополнительному полю даты:
    - is_assigned:
        - если поле имеет значение *null*, то для фильтрации используется диапазон дат;
        - если поле имеет значение *true*, то будут возвращены пропуска, у которых задана дата;
        - если поле имеет значение false, то будут возвращены пропуска, которых не задана дата;
    - range – диапазон даты:
        - from – нижняя граница даты;
        - to – верхняя граница даты.

Ответное сообщение содержит информацию об устройствах, которые подошли под указанные фильтры [подробнее об устройствах читайте в соответствующем разделе]:

```
{
  "device": {
  "sdn": 0,
  "din": 0,
  "parent_sdn": {
    "value": 0
  },
  "name": "",
  "type": 0,
  "address": 0,
  "is_active": false,
  "time_zone_code": 0
 }
}
```

5.22. MessageInfoService

Файл message_info.proto предназначен для получения информации о сообщениях сервера системы.

5.22.1. Глоссарий

5.22.1.1. Профиль события

Класс **esprom.taurus.grpc.v1.messages.MessageProfile**.

Описывает общие характеристики того или иного класса событий в системе. Получение списка всех профилей в системе доступно через метод **GetMessageProfiles**.

Основные поля:

- id – уникальный идентификатор профиля; для системных профилей является предустановленной константой, для пользовательских профилей генерируется при добавлении в систему.
- priority – приоритет событий, относящихся к этому профилю;
- kind – вид событий, относящихся к профилю, перечисление, основные значения: 1 – штатное событие, 2 – тревога, 3 – неисправность;
- name – имя профиля событий;
- is_system – признак системного предустановленного профиля.

5.22.2. Метод GetMessageProfile

Запрос на получение информации о профиле сообщения. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
   "profile_id": 35
}
```

- profile_id – идентификатор профиля сообщений.

Ответ:

```
{
   "profile": {
      "id": 35,
      "priority": 35,
      "kind": "MESSAGE_KIND_FAULT",
      "name": "Нарушение связи",
      "is_system": true
   }
}
```

- id – идентификатор профиля сообщений;
- priority – приоритет сообщения;
- kind – тип сообщения;
- name – сообщение;
- is_system – является ли сообщение системным.

5.22.3. Метод GetMessageProfiles

Запрос на получение списка профилей сообщения. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Ответ содержит список всех профилей сообщения:

```
{
   "profiles": [
      {
         "id": 1,
         "priority": 1,
         "kind": "MESSAGE_KIND_NORMAL",
         "name": "Не показывать",
         "is_system": true
      },
      {
         "id": 35,
         "priority": 35,
         "kind": "MESSAGE_KIND_FAULT",
         "name": "Нарушение связи",
         "is_system": true
      },
      ...
```

```
  ]
}
```

5.22.4. Метод GetUnconfirmedMessages

Запрос на получение списка неподтверждённых сообщений. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Пример ответного сообщения:

```
{
  "message": {
   "id": 0,
   "session_id": 0,
   "server_session_id": 0,
   "event": {
    "sdn": 0,
    "message_code": 0,
    "time": {
     "seconds": 0,
     "nanos": 0
    },
    "card_code": 0,
    "detected_string": {
     "value": ""
    },
    "ext_int": 0,
    "ext_double": 0,
    "ext_string_1": {
     "value": ""
    },
    "ext_string_2": {
     "value": ""
    },
    "attached_image": {
     "value": []
    }
   },
   "profile_id": 0,
   "can_be_confirmed": false,
   "linked_cameras": [
    0
   ],
   "pass_direction": "PASS_DIRECTION_TYPE_UNSPECIFIED",
   "can_be_passage": false,
   "base_text": "",
   "prepared_text": "",
   "is_actual": false,
   "details": [
    {
     "type_url": "",
     "value": []
    }
   ]
  }
}
```

- id – идентификатор сообщения;
- session_id – идентификатор сессии;
- server_session_id – идентификатор сессии сервера;
- event – пришедшее событие:
  - sdn – идентификатор устройства, от которого было получено событие;
  - message_code – код события;
  - time – время события;
  - card_code – код карты, полученный в событии:
  - detected_string – распознанная строка. Константа подстановки %nb;
  - ext_int – дополнительное целое знаковой 4-х байтовое число. Используется при вычислении значений констант подстановки: %tv, %up, %ra, %rb, %av, %dl, %sc и %rr;
  - ext_double – дополнительное число двойной точности с плавающей запятой. Константа подстановки %ef;
  - ext_string_1 – первая дополнительная строка. Константа подстановки %s1;
  - ext_string_2 – вторая дополнительная строка. Константа подстановки %s2;
  - attached_image – привязанное к событию изображение;

5.22.5. Метод ConfirmMessages

Запрос на подтверждение оператором списка тревожных событий. Запрос может быть выполнен только с использованием токена доступа.

Для подтверждения тревожных событий требуется отправить список идентификаторов всех событий, которые требуется подтвердить:

```
{
   "confirm_parameters": [
      {
         "message_session_id": 101,
         "message_id": 345,
         "message_server_session_id": 7293ba11-1b76-4418-8798-c89ba12d9725,
         "sdn": 157
      },
      {
         "message_session_id": 101,
         "message_server_session_id": 7293ba11-1b76-4418-8798-c89ba12d9725,
         "message_id": 346,
         "sdn": 156
      },
      ...
   ]
}
```

- message_session_id – идентификатор сессии, где произошло событие;
- message_server_session_id – идентификатор сессии сервера оборудования, где произошло событие;
- message_id – идентификатор тревожного сообщения;
- sdn – идентификатор устройства, от которого произошло событие.

В качестве ответа придет сообщение с пустым телом.

5.22.6. Метод MessageProfileChanged

Запрос на получение уведомлений об изменения списка профилей событий. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

5.22.7. Метод NewMessageProcessed

Запрос на получение уведомлений об обработке системой списка новых сообщений. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Пример ответа содержит список значений новых сообщений:

```
{
   "messages": [
   {
     "id": 0,
     "session_id": 0,
     "server_session_id": 0,
     "event": {
       "sdn": 0,
       "message_code": 0,
       "time": {
         "seconds": 0,
         "nanos": 0
       },
       "card_code": 0,
       "detected_string": {
         "value": ""
       },
       "ext_int": 0,
       "ext_double": 0,
       "ext_string_1": {
         "value": ""
       },
       "ext_string_2": {
         "value": ""
       },
       "attached_image": {
         "value": []
       }
     },
     "profile_id": 0,
     "can_be_confirmed": false,
     "linked_cameras": [
```

```
      0
    ],
    "pass_direction": "PASS_DIRECTION_TYPE_UNSPECIFIED",
    "can_be_passage": false,
    "base_text": "",
    "prepared_text": "",
    "is_actual": false,
    "details": [
     {
      "type_url": "",
      "value": []
     }
    ]
   }
  ]
}
```

5.22.8. Метод MessagesConfirmed

Запрос на получение уведомлений о подтверждении тревожных сообщений. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Пример ответа со списком параметров тревожных сообщений:

```
{
  "confirm_parameters": [
   {
    "sdn": 0,
    "message_id": 0,
    "message_session_id": 0,
    "message_server_session_id": 0
   }
  ]
}
```

- sdn – идентификатор устройство, от которого поступило тревожное сообщение;
- message_id – идентификатор сообщения;
- message_session_id – идентификатор сессии, когда поступило тревожное сообщение;
- message_server_session_id – идентификатор сессии сервера, когда поступило тревожное сообщение.

5.23. ProtocolInfoService

Файл protocol_info.proto предназначен для получения информации о данных из протокола событий системы.

5.23.1. Метод GetMessages

Запрос на получение данных из протокола событий. Запрос может быть выполнен только с использованием токена доступа.

Пример запроса:

```
{
  "terms": [
    {
      "type_url": "type.googleapis.com/esprom.taurus.grpc.v1.protocol.ProtocolMessageByTimeSearchTerm",
      "value": "ChQKCAjQytiKBhAAEggI0KGajQYQAA=="
    },
    {
      "type_url": "type.googleapis.com/esprom.taurus.grpc.v1.protocol.ProtocolMessageByDeviceInstancesSearchTerm",
      "value": "CggKBtMhmx6tEw=="
    }
  ],
  "detail_codes": [
    "AttachedPass", "AttachedCard", "AttachedCameras"
  ],
  "query_description": {
    "limit": {
      "value": 5
    },
    "offset": {
      "value": 4
    },
    "sort_descriptions": [
      {
        "field_name": "ProtocolMessage.gid",
```

```
            "sort_type": "SORT_TYPE_ASCENDING"
        }
    ]
  },
  "include_removed": false
}
```


- terms – список запакованных в тип *google.protobuf.Any* фильтров, применяемых к событиям протокола. В списке обязательно должен быть указан как минимум один фильтр. Описание поддерживаемых фильтров приведено в разделе "Фильтры событий".
- detail_codes – набор строк-ключей для включения в результаты запроса дополнительной информации, связанной с событиями. Описание поддерживаемой дополнительной информации приведено в разделе "Дополнительная информация";
- query_description – дополнительные параметры запроса:
  - limit – ограничение на количество возвращаемых событий;
  - offset – значение сдвига, начиная с которого требуется получить события;
  - sort_descriptions – описание сортировки данных:
    - field_name – имя поля, по которому необходимо выполнить сортировку. Сортировка доступна по полям "ProtocolMessage.gid" и "ProtocolMessage.time";
    - sort_type – необходимый порядок сортировки:
      - SORT_TYPE_UNSPECIFIED — не указан;
      - SORT_TYPE_ASCENDING — по возрастанию;
      - SORT_TYPE_DESCENDING — по убыванию;
- include_removed – учитывать ли в результате запроса информацию об удалённых объектах.

Результатом будут сообщения с информацией о событиях из протокола, которые удовлетворяют условиям фильтрации, заданным в параметре terms. Пример:


```
{
  "message": {
    "details": [
      {
        "key": "AttachedCard",
        "value": {
          "type_url": "type.googleapis.com/esprom.taurus.grpc.v1.persons.Card",
          "value": "CKnKAxD/8qqBwA4aBwj/8qqBwA4iDAiTmfOEBhCgloilATAEOAE="
        }
      },
      {
        "key": "AttachedPass",
        "value": {
          "type_url": "type.googleapis.com/esprom.taurus.grpc.v1.persons.Pass",
          "value":
"CN7QBxDDugQYZyAEKgQIqcoDMgIIDjgBQgwImJjzhAYQ8ImbgwJKDAiN8PeEBhCw0NLRAIIMCJOZ84QGELCqv6cBegMIpH6CAQsI1NeBjQ
YQ+JSraA=="
        }
      }
    ],
    "gid": 33720117,
    "id": 422972,
    "session_id": 6,
    "server_session_id": 4654,
    "time": {
      "seconds": "1633035790",
      "nanos": 0
    },
    "source_device_id": 2477,
    "device_type_id": 22,
    "message_type_id": 2236,
    "message_profile_id": 4,
    "message_kind": "MESSAGE_KIND_NORMAL",
    "priority": 4,
    "message_text": {
      "value": "Предоставление доступа на выход"
    },
    "comments": {
      "value": ""
    },
    "utc_offset": null,
    "actual_time": {
      "seconds": "1633035790",
      "nanos": 0
    }
  }
}
```

- details – дополнительная информацию по событию в виде списка пар ключ-значение, где ключ — это код дополнительной информации, привязанной к событию, а значение — это сама дополнительная информация, запакованная в тип *google.protobuf.Any*;
- gid – глобальный идентификатор события;
- id – идентификатор события, в рамках клиентской сессии;
- session_id – идентификатор клиентской сессии;
- server_session_id – идентификатор сессии сервера;
- time – время события;
- source_device_id – идентификатор устройства-источника события;
- device_type_id – идентификатор типа устройства-источника;
- message_type_id – идентификатор типа события;
- message_profile_id – идентификатор профиля события;
- message_kind – вид события. Возможные виды событий:
    - MESSAGE_KIND_UNSPECIFIED — не указан;
    - MESSAGE_KIND_NORMAL — штатные события;
    - MESSAGE_KIND_ALARM — тревожные события;
    - MESSAGE_KIND_FAULT — неисправности;
- priority – приоритет события;
- message_text – текст события;
- comments – комментарий к событию;
- utc_offset – сдвиг часового пояса относительно UTC на момент фиксации события,
- actual_time – время фактического добавления события в базу данных.

Пример запроса получения событий проходов за период с «01.10.2021» по «01.12.2021» сотрудников подразделения с идентификатором 111:

```
{
   "terms": [
      {
         "type_url": "type.googleapis.com/esprom.taurus.grpc.v1.protocol.ProtocolMessageByTimeSearchTerm",
         "value": "ChQKCAjQytiKBhAAEggI0KGajQYQAA=="
      },
      {
         "type_url": "type.googleapis.com/esprom.taurus.grpc.v1.protocol.ProtocolMessageByIsPassageSearchTerm",
         "value": ""
      },
      {
         "type_url": "type.googleapis.com/esprom.taurus.grpc.v1.protocol.ProtocolMessageByPersonParametersSearchTerm",
         "value": "GgMKAW8="
      }
   ],
   "detail_codes": [
      "AttachedPerson"
   ],
   "query_description": {
      "limit": {
         "value": 20
      }
   },
   "include_removed": true
}
```

Где значение фильтра ProtocolMessageByTimeSearchTerm:

```
{
  "time": {
    "from": {
      "seconds": 1633035600,
      "nanos": 0
    },
    "to": {
      "seconds": 1638306000,
      "nanos": 0
    }
  }
}
```

Значение фильтра ProtocolMessageByPersonParametersSearchTerm:

```
{
  "organization_node_ids": {
    "ids": [ 111 ]
  }
```

```
}
```

Пример запроса получения всех событий-тревог начиная с последнего идентификатора 34120041:

```
{
   "terms": [
      {
         "type_url": "type.googleapis.com/esprom.taurus.grpc.v1.protocol.ProtocolMessageByLastGlobalIdSearchTerm",
         "value": "COnCohA="
      },
      {
         "type_url": "type.googleapis.com/esprom.taurus.grpc.v1.protocol.ProtocolMessageByMessageParametersSearchTerm",
         "value": "GgMKAQA="
      }
   ],
   "include_removed": false
}
```

Где значение фильтра ProtocolMessageByLastGlobalIdSearchTerm:

```
{
  "last_gid": 34120041
}
```

Значение фильтра ProtocolMessageByMessageParametersSearchTerm:

```
{
  "message_kinds": {
    "message_kinds": [ "MessageKind.MESSAGE_KIND_ALARM" ]
  }
}
```

5.23.1.1. Фильтры событий

При запаковке в тип *google.protobuf.Any* значение поля *type_url* формируется по следующему шаблону: type.googleapis.com/[имя пакета].[Тип сообщения-фильтра], где имя пакета соответствует значению указанному в поле package proto-файла, в котором приведено описание фильтра. Например для фильтра событий ProtocolMessageByDriverTypesSearchTerm, описанного следующим proto-файлом:

```
syntax = "proto3";

package esprom.taurus.grpc.v1.protocol;

message ProtocolMessageByDriverTypesSearchTerm {
  IdsSearchTermEntry driver_ids = 1;
}
```

при запаковке в Any поле type_url получит значение "type.googleapis.com/esprom.taurus.grpc.v1.protocol.ProtocolMessageByDriverTypesSearchTerm".

Файл protocol_info.proto (package esprom.taurus.grpc.v1.protocol)

ProtocolMessageByLastGlobalIdSearchTerm - получение событий из протокола, глобальный идентификатор которых больше, чем указанное значение. Параметры:

- last_gid (int32) – нижняя граница для глобального идентификатора события.

ProtocolMessageByTimeSearchTerm - получение событий, время возникновения которых попадает в указанный диапазон. Параметры:

- time (esprom.taurus.grpc.v1.TimestampSearchTermEntry) — описание диапазона времени (обязательно должна быть указана хотя бы одна из границ),
    - from (google.protobuf.Timestamp) – нижняя граница времени (включается в результаты запроса);
    - to (google.protobuf.Timestamp) – верхняя граница времени (не включается в результаты запроса).

Пример фильтра для получения событий с «01.10.2021» по «01.12.2021»:

```
{
  "time": {
    "from": {
      "seconds": 1633035600,
      "nanos": 0
    },
    "to": {
```

```
    "seconds": 1638306000,
    "nanos": 0
   }
  }
}
```

ProtocolMessageByTimeOfDaySearchTerm - получение событий, дневная составляющая времени возникновения которых попадает в указанны диапазон. Параметры:

- timeof_day (esprom.taurus.grpc.v1.TimeOfDaySearchTermEntry) — описание диапазона времени дня (обязательно должна быть указана хотя бы одна из границ диапазона. Если не указана нижняя то берутся события произошедшие с начала дня до верхней границы, если не указана верхняя граница, то берутся все события начиная с нижней границы до конца дня):
    - from – нижняя граница времени дня (включается в результаты запроса);
    - to – верхняя граница времени дня (не включается в результаты запроса).

Пример фильтра для получения событий с 12:00 до 19:00:

```
{
  "time_of_day": {
    "from": {
      "hours": 12,
      "minutes": 0,
      "seconds": 0,
      "nanos": 0
    },
    "to": {
      "hours": 19,
      "minutes": 0,
      "seconds": 0,
      "nanos": 0
    }
  }
}
```

ProtocolMessageByHostInfosSearchTerm — получение событий, источниками возникновения которых являются устройства, принадлежащие заданным серверам оборудования. Параметры:

- host_info_ids (esprom.taurus.grpc.v1.IdsSearchTermEntry) – описание фильтра по идентификаторам:
    - ids (repeated int32) – список идентификаторов серверов оборудования.

ProtocolMessageByDriverInstancesSearchTerm — получение событий, источниками возникновения которых являются устройства, относящиеся к указанным экземплярам драйверов. Параметры:

- dins (esprom.taurus.grpc.v1.IdsSearchTermEntry) – описание фильтра по идентификаторам:
    - ids (repeated int32) – список идентификаторов экземпляров драйверов.

ProtocolMessageByDeviceTypesSearchTerm — получение событий, источниками возникновения которых являются устройства с заданным типом;

- device_types (esprom.taurus.grpc.v1.IdsSearchTermEntry) — описание фильтра по идентификаторам:
    - ids (repeated int32) – список кодов типов устройств.

Пример фильтра для получения событий от точек прохода (дверь, ворота, турникет):

```
{
  "device_types": {
    "ids": [ 3, 4, 22 ]
  }
}
```

ProtocolMessageBySourceDevicesSearchTerm — получение событий от указанных источников-устройств. Параметры:

- sdns (esprom.taurus.grpc.v1.IdsSearchTermEntry) — описание фильтра по идентификаторам:
    - ids (repeated int32) – список идентификаторов устройств.

ProtocolMessageByDriverTypesSearchTerm — получение событий, источниками возникновения которых являются устройства, относящиеся к указанным типам драйверов. Параметры:

- driver_ids (esprom.taurus.grpc.v1.IdsSearchTermEntry) — описание фильтра по идентификаторам:
    - ids (repeated int32) – список идентификаторов типов драйверов.

Пример фильтра для получения событий от устройств драйвера Elsys (дверь, ворота, турникет):

```
{
  "driver_ids": {
    "ids": [ 356 ]
  }
}
```

ProtocolMessageByMessageParametersSearchTerm — получение событий с фильтрацией по критериям, относящимся к типу события. Параметры:

- message_types (esprom.taurus.grpc.v1.IdsSearchTermEntry) – фильтр по типам событий:
    - ids (repeated int32) – список идентификаторов типов событий;
- message_profiles (esprom.taurus.grpc.v1.IdsSearchTermEntry) – фильтр по профилям событий:
    - ids (repeated int32) – список идентификаторов профилей событий;
- message_kinds (esprom.taurus.grpc.v1.protocol.MessageKindsSearchTermEntry) – фильтр по видам событий:
    - message_kinds (repeated esprom.taurus.grpc.v1.messages.MessageKind) – список видов событий;
- priority (esprom.taurus.grpc.v1.Int32SearchTermEntry) – фильтр по приоритету событий:
    - from (google.protobuf.Int32Value) – нижняя граница приоритета;
    - to (google.protobuf.Int32Value) – верхняя граница приоритета.

Пример фильтра для получения событий с профилем «СКУД. Отказ в доступе» с приоритетом от 1 до 5:

```
{
  "message_profiles": {
    "ids": [ 45]
  },
  "priority": {
    "from": {
      "value": 1
    },
    "to": {
      "value": 5
    }
  }
}
```

ProtocolMessageByIsPassageSearchTerm — получение событий, которые являются событиями прохода. Параметры:

- include_access_denied (bool) – включать или нет в ответ на запрос события об отказе в доступе.

ProtocolMessageByActualTimeSearchTerm — получение событий, время записи в протокол которых попадает в указанный диапазон. Параметры:

- actual_time (esprom.taurus.grpc.v1.TimestampSearchTermEntry) — описание диапазона времени записи в протокол (обязательно должна быть указана хотя бы одна из границ),
    - from (google.protobuf.Timestamp) – нижняя граница времени (включается в результаты запроса);
    - to (google.protobuf.Timestamp) – верхняя граница времени (не включается в результаты запроса).

Пример фильтра для получения событий, записанных в протокол с «01.01.2025» по «01.10.2025»:

```
{
  "actual_time": {
    "from": {
      "seconds": 1735689601,
      "nanos": 0
    },
    "to": {
      "seconds": 1736467201,
      "nanos": 0
    }
  }
}
```

Файл protocol_persons.proto (package esprom.taurus.grpc.v1.protocol)

ProtocolMessageByPersonParametersSearchTerm — получение событий с фильтрацией по параметрам, которые связаны со СКУД:

- person_ids (esprom.taurus.grpc.v1.IdsSearchTermEntry) – фильтр по привязанным к событиям персонам:
  - ids (repeated int32) – список идентификаторов персон;
- organization_node_ids (esprom.taurus.grpc.v1.IdsSearchTermEntry) – фильтр по организациям и подразделениям привязанных персон:
  - ids (repeated int32) – список идентификаторов организаций и подразделений;
- pass_category_ids (esprom.taurus.grpc.v1.IdsSearchTermEntry) – фильтр по категории пропусков, привязанных к событиям:
  - ids (repeated int32) – список идентификаторов категорий пропусков;
- pass_priority (esprom.taurus.grpc.v1.Int32SearchTermEntry) – фильтр по приоритету привязанных пропусков:
  - from (google.protobuf.Int32Value) – нижняя граница приоритета;
  - to (google.protobuf.Int32Value) – верхняя граница приоритета;
- card_code (google.protobuf.UInt64Value) – фильтр по коду идентификации карт, привязанных к событиям.

ProtocolMessageByControlAreasSearchTerm — получение событий, источниками возникновения которых являются устройства привязанные к указанным территориям:

- control_area_ids (esprom.taurus.grpc.v1.IdsSearchTermEntry) – фильтр по территориям:
  - ids (repeated int32) – список идентификаторов территорий.

5.23.1.2. Дополнительная информация

Код дополнительной информации, строка	Описание	Тип protobuf-сообщения	Пример

AttachedCameras	Привязанные к событию камеры.	esprom.taurus.grpc.v1.protocol.AttachedCameras	{
			"camera_ids": [
			5351,
			5352
			]
			}

AttachedCoordinate	Привязанное к событию географическое положение.	esprom.taurus.grpc.v1.drivers.GeoCoordinate	{
			"altitude": 25,
			"latitude": 33,
			"longitude": 53,
			"horizontal_accuracy": 76,
			"vertical_accuracy": 98,
			"speed": 6,
			"course": 3
			}

AttachedImage	Привязанное к событию изображение.	esprom.taurus.grpc.v1.protocol.AttachedImage	{
			"image": [
			67,
			58,
			47,
			85,
			115...
			]
			}

SourceDevice	Информация об устройстве-источнике события.	esprom.taurus.grpc.v1.drivers.Device	{
			"sdn": 6025,
			"din": 144,
			"parent_sdn": {
			"value": 6006
			},
			"name": "ППЦ - Калитка",
			"type": 4,
			"address": 1,
			"is_active": true,
			"time_zone_code": 0
			}

Код дополнительной информации, строка	Описание	Тип protobuf-сообщения	Пример

MessageType	Информация о типе события.	esprom.taurus.grpc.v1.drivers.MessageType	{
			"driver_id": 256,
			"device_type": 4,
			"code": 1,
			"text": "Штатный вход %nm %n1",
			"message_profile_id": 5,
			"pass_direction": "PASS_DIRECTION_TYPE_IN",
			"credential_check_result": "CREDENTIAL_CHECK_RESULT_ACCESS_VERIFIED",
			"credential_check_result_group": null
			}

AttachedPerson	Привязанной к событию информация о персоне.	type.googleapis.com/esprom.taurus.grpc.v1.persons.Person	{
			"id": 33696,
			"name": "Иванов",
			"first_name": {
			"value": "Иван"
			},
			"second_name": {
			"value": "Иванович"
			},
			"table_no": {
			"value": "000000078298305"
			},
			"comments": null,
			"organization_node_id": 97,
			"position_id": {
			"value": 4616
			},
			"create_date": {
			"seconds": 1572620123,
			"nanos": 0
			}
			}

AttachedPass	Привязанная к событию информация о пропуске.	type.googleapis.com/esprom.taurus.grpc.v1.persons.Pass	{
			"id": 41852,
			"person_id": 33696,
			"pass_category_id": 103,
			"status": "PASS_STATUS_ACTIVE",
			"card_id": {
			"value": 34081
			},
			"access_level_id": {
			"value": 14
			},
			"priority": 1,
			"create_date": {
			"seconds": 1347013357,
			"nanos": 0
			},
			"activation_time": {
			"seconds": 1572782465,
			"nanos": 856021000
			},
			"issue_date": {
			"seconds": 1347013468,
			"nanos": 0
			},
			"disable_anti_pass_back_check": false
			}

Код дополнительной информации, строка	Описание	Тип protobuf-сообщения	Пример
AttachedCard	Привязанная к событию информация о карте доступа.	type.googleapis.com/ esprom.taurus.grpc.v1.persons.Card	{ "id": 34081, "full_card_code": 304958821244, "create_date": { "seconds": 1572620019, "nanos": 0 }, "status": "CARD_STATUS_ISSUED", "identifier_type": 0, "mifare_security_level": 0, "pre_issued": false }
DestinationArea	Информация о территории назначения события прохода (куда был осуществлен вход).	esprom.taurus.grpc.v1.persons.ControlArea	{ "id": 48, "name": "ППЦ", "parent_id": { "value": 2 }, "time_block_id": null }
SourceArea	Информация об исходной территории события прохода (откуда был осуществлен вход).	esprom.taurus.grpc.v1.persons.ControlArea	{ "id": 1, "name": "Вне территории", "parent_id": null, "time_block_id": null }
PassCategory	Информация о категории пропуска, привязанного к событию.	esprom.taurus.grpc.v1.persons.PassCategory	{ "id": 103, "name": "Для служащих", "time_restriction_rule": "TIME_RESTRICTION_RULE_NONE", "time_restriction_unit": "TIME_RESTRICTION_UNIT_DAY", "time_restriction_value": 0, "photo_identification_form_id": { "value": 1 }, "is_change_pass_end_date_allowed": true, "is_pass_prolongation_allowed": true, "numeration_id": null, "inactive_days_count_before_auto_block_pass": null, "activation_time_limit": null }
AttachedPersonPosition	Информация о должности персоны, привязанной к событию.	esprom.taurus.grpc.v1.persons.DictionaryRecord	{ "id": 47814, "header_id": 9, "value": "Машинист бульдозера", "is_system": false }
AttachedPersonDepartment	Информация о подразделении персоны, привязанной к событию.	esprom.taurus.grpc.v1.persons.OrganizationNode	{ "id": 11711, "name": "Московский Офис", "parent_id": { "value": 11710 }, "node_type": "ORGANIZATION_NODE_TYPE_DEPARTMENT" }

Код дополнительной информации, строка	Описание	Тип protobuf-сообщения	Пример

AttachedPassAccessLevel	Информация об уровне доступа пропуска, привязанного к событию.	esprom.taurus.grpc.v1.persons.AccessLevel	{
			"id": 9874,
			"physical_number": null,
			"is_weak": false,
			"name": "Служебный",
			"main_time_block_id": 0
			}

5.23.2. Метод GetLastProtocolMessageInfo

Запрос на получение информации о последнем записанном в протокол событии. Выполняется с пустым телом запроса. Запрос может быть выполнен только с использованием токена доступа.

Ответ:


```
{
    "gid": {
        "value": 63230991
    }
}
```


- gid – глобальный идентификатор последнего записанного в протокол события.

Приложение А. Тип *google.protobuf.Any*

Тип Any может служить для описания любого типа сообщения в protobuf, что позволяет расширять возможности протокола, не изменяя описания базовых сообщений. Поле сообщения с типом Any представляет из себя по сути вложенную protobuf-сериализацию другого сообщения. Для передачи через сообщение Any любого другого сообщения его необходимо сначала "запаковать". Для получения сообщения из Any его нужно "распаковать" явно указав тип сообщения, которое должно быть десериализовано. Поля типа Any:

- type_url – строка, идентифицирующая тип запакованного сообщения. Значение в общем случае формируется по следующему шаблону: type.googleapis.com/[имя пакета].[Тип запакованного сообщения], где имя пакета соответствует значению указанному в поле package proto-файла, в котором приведено описание типа запакованного сообщения.
- value – base64-строка, в которой содержится массив байт соответствующий protobuf-сериализации запакованного сообщения.

Приложение В. Ошибки, возвращаемые ПК «Бастион-3»

Полный перечень возможных кодов ошибок gRPC, возвращаемых ПК «Бастион-3», и их описание приведены ниже. Ошибки со стороны сервера системы, не приведенные в данной таблице, по умолчанию будут возвращены с кодом 13 INTERNAL.

Код ошибки gRPC	Код ошибки на сервере системы	Описание
3 INVALID_ARGUMENT	System.ArgumentException	Некорректное значение параметра метода.
	-17 IncorrectDataError	Некорректные данные или состояние объекта.
	-19 UnsupportedError	Неизвестный фильтр поиска объектов или некорректные параметры фильтра.
5 NOT_FOUND	-10 NotFoundError	Запрошенные данные не были найдены.
7 PERMISSION_DENIED	-14 Forbidden	Недостаточно прав для запрошенной информации.
13 INTERNAL	-1 ConnectionError	Ошибка подключения.
	-2 ProtocolError	Ошибка протокола передачи данных.
	-3 WrongFormatError	Неправильный формат представления данных.
	-5 LogicError	Ошибка бизнес-логики.
	-6 DbError	Ошибка базы данных.
	-7 RejectedError	Запрос или команда была отброшена.
	-13 UndefinedError	Неопределённая ошибка.
	-16 LicenseError	Ошибка лицензирования.
	-19 UnsupportedError	Запрос выполнения действия, которое не поддерживается.
	-20 TimeoutError	Тайм-аут операции.
	-21 InitializationError	Ошибка инициализации приложения.
	-22 ExternalSystemError	Ошибка внешней системы.
16 UNAUTHENTICATED	-9 AuthorizationError	Ошибка авторизации.
	-12 UnauthorizedAccessError	Доступ к защищённым данным или попытка выполнения

		действия без авторизации.