

API аутентификации

Оглавление

- # Общие замечания по работе с API
- # Автоматический вход по технологии HTTP Header Enrichment
- # Аутентификация по логину-паролю
 - # Старт сценария
 - # Передача логина и пароля
 - # Отображение captcha
 - # Передача логина, пароля и символов captcha
 - # Двухфакторная аутентификация
 - # Валидация OTP
- # Аутентификация по номеру телефона и OTP коду
 - # Ввод номера телефона и отправка СМС сообщения
 - # Валидация введенного кода
- # Обработка ошибок и особых случаев
 - # Блокировка пользователя
 - # Принудительная смена пароля пользователя при входе
 - # Блокировка доступа абонента к конкретному ресурсу по client_id
 - # Возможные ограничения на поля формы
 - # Возможные коды ошибок
- # Аутентификация от имени абонента
 - # Формат запроса
- # Валидация токена доступа
 - # Формат запроса
- # Повышение уровня авторизации WebSSO с помощью OTP
 - # Схема повышения уровня авторизации
 - # Формат запроса на повышение уровня авторизации
 - # Формат успешного ответа на повышение уровня авторизации
 - # Формат запроса на повторную отправку OTP абоненту
 - # Формат успешного ответа на отправку OTP абоненту
 - # Формат запроса на валидацию OTP
 - # Формат успешного ответа на валидацию OTP
 - # Формат ошибки валидации OTP
 - # Формат ответа при блокировке пользователя

- # Возможные коды ошибок
 - # Получение токена под операцию
 - # Проверка необходимости получения токена под операцию
 - # Получение одноразового токена под операцию
 - # Запрос валидации введенного кода
 - # Формат ошибки валидации OTP
 - # Формат ответа при блокировке пользователя
 - # Возможные коды ошибок
 - # Смена данных аутентификации (логин и пароль) пользователя
 - # Запрос на смену пары пароль/логин пользователя
 - # Выход через API
 - # Формат запроса
 - # Вызов callback URL при инвалидации токенов
 - # Инвалидация токена пользователя
 - # Блокирование сервиса целиком
 - # Понижение уровня авторизации токена
 - # Использование параметра scope
 - # Диаграммы переходов для работы с UIDM
 - # Авторизация в UIDM при существующей сессии
 - # Аутентификация в UIDM, отсутствие сессии, успешный автоход
 - # Аутентификация и авторизация в UIDM
 - # Повышение уровня авторизации
 - # Отзыв токена
-

API предназначено для аутентификации через UIDM из SPA или мобильных приложений.

Поддерживаются следующие сценарии входа:

- через логин-пароль, в том числе усиленный вторым фактором - OTP через SMS
- через долгоживущий токен, сохраненный в приложении (используется для аутентификации через ПИН-код или биометрию)
- автоход по технологии HTTP Header Enrichment
- Kerberos

Общие замечания по работе с API

- Все параметры запросов и ответов, атрибуты пользователей и прочие параметры являются регистрозависимыми
- Все параметры запросов и ответов являются обязательными, если явно не указано обратное
- Переносы строк в некоторых примерах запросов добавлены для удобства чтения, реальная строка запроса должна быть без них
- Клиент должен поддерживать HTTP cookies

- При запросах к API, ошибки со статусом 503 всегда приходят в HTML.

Автоматический вход по технологии HTTP Header Enrichment

НАПОМИНАЕМ

Данный шаг является опциональным и может быть пропущен, если автоматический вход не требуется или используется технология, отличная от HTTP Header Enrichment.

Для возможности автоматического входа по технологии HTTP Header Enrichment необходимо получить токен автоматического входа отдельным POST-запросом.

Запрос должен обязательно идти по протоколу HTTP, а не HTTPS - это ограничение технологии.

Формат запроса

```
POST /sso/auth/autologin
Host: <sso_host>
Accept: application/json
```

- <sso_host> - базовый адрес сервера UIDM, например sso.rooxteam.com

Формат ответа

В ответе всегда возвращается токен автоматического входа, который нужно передать в запросе [начала аутентификации](#):

```
HTTP/1.1 200 OK
```

```
{
  "auto-login-jwt": "eyJhbGciOiJSU0EtT0FFUCIsImVuYyI6IkJhbmRlbnQ00ifQ.enSg9mzML5y-RE_tFDMh1Ixwrb8yYX-bPN8X-irz9nHFQFhb1yGyQ5RvfEDF12hpfDPbMTi5WzHaC79FFwkHwWdneDYMCnFlGBMTpZGNRrQxu6yItycAY_GEAURpW3_3BT2Dkj78Q-faD4XoeREodhaPsfQrA0TunML0DCwia5Rs.6r2M6pqz47qe5aGQ6L6n5I-U55qZ5ISd4aik4Ywa.j09EB1wfaD0NNn6BZRo0dNgndUejGWQxpeLljnxY.SJ8AtbswxqHFf6Wb_gd5BQ"
}
```

Поля ответа

- <auto-login-jwt> - токен автоматического входа

Аутентификация по логину-паролю

Старт сценария

Аутентификация выполняется по протоколу OAuth2.0 с передачей специального значения параметра grant_type для SPA и мобильных приложений.

Для попытки автоматического входа через технологию HTTP Header Enrichment может быть передан параметр auto-login-jwt, полученный в ответе на шаге [Автоматический вход](#).

Формат запроса

```
POST /sso/oauth2/access_token
Host: <sso_host>
Accept: application/json
Content-Type: application/x-www-form-urlencoded

client_id=<client_id>&
client_secret=<client_secret>&
scope=<scope>&
grant_type=urn:roox:params:oauth:grant-type:m2m&
realm=<realm>&
service=dispatcher&
response_type=token&
auto-login-jwt=<auto-login-jwt>&
accessToken=<accessToken>&
mpt=<mpt>
```

Параметры

- <sso_host> - базовый адрес сервера UIDM, например sso.rooxteam.com
- <realm> - реалм в UIDM, например, %2Fcustomer, которое является uri-encoded значением /customer
- <client_id> - идентификатор клиента, например selfcare
- <client_secret> - пароль клиента
- <scope> - OAuth2.0 scope, **опционально, регистрозависимо**
- <grant_type> - способ авторизации пользователя, для данного сценария всегда urn:roox:params:oauth:grant-type:m2m
- <service> - имя цепочки аутентификации, для данного сценария всегда dispatcher
- <response_type> - способ авторизации пользователя; в данном сценарии всегда token
- <auto-login-jwt> - токен автоматического входа, **опционально**
- <mpt> - мультиплатформенный токен, **опционально**

В случае успешного автовхода, в ответе будет сразу возвращен access_token.

В случае неуспешного автовхода, ответ будет содержать данные о форме для входа по логину и паролю.

Формат ответа при успешном автовходе

```
HTTP/1.1 200 OK
```

```
{
  "access_token": "fe8a0976-972c-4773-9ba8-662c92869639",
  "refresh_token": "863a4983-2424-4cb3-9ed2-18f9357b64c0",
  "refresh_expires_in": 1599,
  "scope": [
    "cn"
  ],
}
```



```
{
  "token_type": "Bearer",
  "expires_in": 599,
  "old_token": "fe8a0976-972c-4773-9ba8-662c92869639",
  "JWTToken": "eyJhZG90YV9sa19tMm
0iXSwed3ViIjoiYmlzX19fX18yRpg3NjU0MzIxIiwiaABUoIjoxNjAxOTgzNTc4LCJhenAiOiJ5b3RhT5trX20ybSIs
ImFtciI6ImRldmljZV9zdG9yZWVfdG9rZW4iLCJ0b2t1bk5hbWUiOiFrCF90b2t1biIsImIzcyI6InlvdSWwbGtfbT
JtIiwicmVhbG0iOiUuY3VzdG9tZXIiLCJ0b2t1b1R5cGUiOiJCaVJlZGRlZEpXVFRva2VuIiwiaXhwIjoxNjAxOTgz
MTc4LCJpN0ZiOiJlE2MDE5ODM1Nzh9._kCRnEw0qI9K1JVCVbwofCqsZzzEm7PJBqfJCAG19bG"
}
```

Параметры запроса

- score - список score (в формате JSON Array) разрешенных для использования от имени пользователя, возможные значения задаются конфигурацией
- token_type - тип выданного токена. Всегда Bearer
- refresh_token - выданный refresh token
- refresh_expires_in - время до истечения срока действия refresh токена в секундах
- access_token - выданный access token
- expires_in - время до истечения срока действия токена в секундах
- old_token - токен, выданный внешней системой, строка
- JWTToken - JWT токен для длительного хранения и последующей аутентификации

Формат ответа сервера после неуспешного автовхода

Если автовход был пропущен, либо закончился неуспешно, UIDM отправляет описание формы ввода учетных данных пользователем (таких как логин/пароль):

UI должен отрисовать форму согласно описанию.

```
HTTP/1.1 200 OK
```

```
{
  "form": {
    "errors": [],
    "name": "loginForm",
    "fields": {
      "username": {
        "constraints": [
          {
            "name": "NotNull"
          },
          {
            "name": "Size",
            "attributes": {
              "min": 10,
              "max": 25
            }
          }
        ]
      },
      {
        "name": "FilteredSize",
        "attributes": {
```

```

        "skip": "([^\9]+)|([^\0-9])",
        "min": 10,
        "max": 10
    }
}
],
{
    "password": {
        "constraints": [
            {
                "name": "Size",
                "attributes": {
                    "min": 4,
                    "max": 1024
                }
            },
            {
                "name": "NotNull"
            }
        ]
    }
}
},
{
    "view": {
        "blockedFor": null,
        "isBlocked": false
    },
    "serverUrl": "<ignore>",
    "step": "auth_form",
    "execution": "dec7dacf-as2d-4345-aae5-d08410cb8f95_H4sIAAAAAAAAAAN1WTWwcRRYu27GdxE5wEnYF"
}
}

```

Поля ответа

- form - форма получения параметров аутентификации пользователя
- errors - список ошибок, сообщение может быть привязано к полю (field)
- form.name - имя формы
- fields - список полей формы
- username - атрибуты, относящиеся к имени пользователя
- password - атрибуты, относящиеся к паролю
- constraints - список ограничений на поле
- constraints.name - имя ограничения
- constraints.attributes - перечень атрибутов ограничений
- constraints.attributes.min - минимальное значение для данного атрибута
- constraints.attributes.max - максимальное значение для данного атрибута
- constraints.attributes.skip - регулярное выражение валидации значения для поля
- view - дополнительные данные текущего шага
- isBlocked - признак блокировки пользователя
- blockedFor - время до разблокировки в секундах
- serverUrl - URL для запросов на сервер
- step - идентификатор шага
- execution - идентификатор предсессии аутентификации

Передача логина и пароля

В этом запросе используется значение параметра execution из предыдущего ответа сервера. В случае успешной аутентификации будет возвращен access_token. В случае неуспешной аутентификации будет возвращен ответ, аналогичный ответу сервера после неуспешного автохода, содержащий описание ошибок.

Возможна отправка дополнительных параметров для внешних антифрод систем. Например: extIp, innerIp, mac. Поля несут информативный характер так как безопасность передачи значений не гарантируется.

Формат запроса

```
POST /sso/oauth2/access_token HTTP/1.1
Host: <sso_host>
Accept: application/json
Cache-Control: no-cache
Content-Type: application/x-www-form-urlencoded

client_id=<client_id>&
client_secret=<client_secret>&
scope=<scope>&
grant_type=urn:roox:params:oauth:grant-type:m2m&
realm=%2Fcustomer&
service=dispatcher&
response_type=token&
execution=<execution>&
username=<username>&
password=<password>&
_eventId=next
```

Параметры

- <sso_host> - базовый адрес сервера UIDM, например sso.rooxteam.com
- <client_id> - идентификатор клиента, например selfcare
- <client_secret> - пароль клиента
- <scope> - OAuth2.0 scope в кодировке UTF-8, **опционально, регистрозависимо**
- <grant_type> - способ авторизации пользователя, в данном сценарии всегда используется значение urn:roox:params:oauth:grant-type:m2m
- <realm> - группа пользователей UIDM, всегда используется значение %2Fcustomer, которое является uri-encoded значением /customer
- <execution> - идентификатор сессии аутентификации, значение берется из предыдущего ответа сервера
- <service> - имя цепочки аутентификации, всегда dispatcher
- <response_type> - способ авторизации пользователя; в данном сценарии всегда token
- <username> - логин пользователя
- <password> - пароль пользователя
- <_eventId> - идентификатор действия, всегда next
- <innerIp> - внутренний IP-адрес пользователя, **опционально**
- <mac> - MAC-адрес сетевого адаптера, **опционально**
- <extIp> - внешний IP-адрес пользователя, **опционально**

Формат успешного ответа

```
HTTP/1.1 200 OK
```

```
{
  "access_token": "fe8a0976-972c-4773-9ba8-662c92869639",
  "refresh_token": "863a4983-2424-4cb3-9ed2-18f9357b64c0",
  "refresh_expires_in": 1599,
  "scope": [
    "cn"
  ],
  "token_type": "Bearer",
  "expires_in": 599,
  "JWTToken": "eyJY3R5IjogRhlXVCIsICJ0eXZu5iAiand0IiwgImFsGuM6ICJIUzI1NiIgfQ.eyJhdWQiOi0lsieW90YV9sa19tMm0iXSw3ViIjoiYmlzX19fX18yRpg3NjU0MzIxIiwiaABUoIjoxNjAxOTgzNTc4LCJhenAiOiJ5b3RhT5trX20ybSIsImFtciI6ImRldmljZV9zdG9yZWZldG9rZW4iLCJ0b2t1bk5hbWUiOiFrCF90b2t1biIsImIzcyI6InlvdSWwbGtfbTJtIiwicmVhbG0iOi0uUvY3VzdG9tZXIiLCJ0b2t1b1R5cGUiOiJCavJlZGRlZEpxVFRva2VuIiwiaXhwIjoxNjAxOTgzMTc4LCJpN0ZiOjE2MDE5ODM1Nzh9._kCRnEwOqI9K1JVCVbwofCqsZzzEm7PJBqfJCAG19bG",
  "mpt": "4b527cee-af96-455f-902c-eb14f13ed3fb"
}
```

Поля ответа

- scope - список scope (в формате JSON Array) разрешенных для использования от имени пользователя, возможные значения задаются конфигурацией
- token_type - тип выданного токена. Всегда Bearer
- refresh_token - выданный refresh token
- refresh_expires_in - время до истечения срока действия refresh токена в секундах
- access_token - выданный access token
- expires_in - время до истечения срока действия токена в секундах
- JWTToken - JWT токен для длительного хранения и последующей аутентификации в сценарии [Автоматический вход](#)
- mpt - мультиплатформенный токен. Служит для замены ввода логина/пароля/OTP в сценарии аутентификации; наличие этого токена в ответе зависит от конфигурации сервиса UIDM

Формат ответа при неверном пароле

```
HTTP/1.1 200 OK
```

```
{
  "form": {
    "errors": [
      {
        "message": "invalid_credentials"
      }
    ],
    "name": "loginForm",
    "fields": {}
  },
  "message": "invalid_credentials"
}
```

```

"view": {
  "blockedFor": null,
  "isBlocked": false
},
"serverUrl": "https://sso.rooxteam.com/sso/auth/login-widget-router",
"step": "auth_form",
"execution": "dec7dacf-as2d-4345-aae5-d08410cb8f95_H4sIAAAAAAAAAAN1WTWwcRRYu27GdxE5wEnYF"
}

```

Поля ответа

- form - форма получения параметров аутентификации пользователя
- errors - список ошибок, сообщение может быть привязано к полю (field)
- message - сообщение об ошибке
- form.name - имя формы
- fields - список полей формы; в данном примере значение удалено для удобства чтения
- view - дополнительные данные текущего шага
- isBlocked - признак блокировки пользователя
- blockedFor - время до разблокировки в секундах
- serverUrl - URL для запросов на сервер
- step - идентификатор шага
- execution - идентификатор предсессии аутентификации

Формат ответа при ошибке аутентификации через внешнюю систему

При использовании аутентификации через внешнюю систему возможен проброс ошибок внешней системы в форму. Значение поля `form.errors.message` и формат данных `view.externalAuthError` определяется разработчиком при реализации конкретного модуля.

HTTP/1.1 200 OK

```
{
  "form": {
    "errors": [
      {
        "message": "user_blocked"
      }
    ],
    "name": "loginForm",
    "fields": {}
  },
  "view": {
    "isBlocked": false,
    "externalAuthError": {
      "error_type": {
        "additionalParams": {},
        "message": "Отказ аутентификации: user_blocked"
      }
    }
  }
},
```

```
{
  "serverUrl": "https://sso.rooxteam.com/sso/auth/login-widget-router",
  "step": "auth_form",
  "execution": "dec7dacf-as2d-4345-aae5-d08410cb8f95_H4sIAAAAAAAAAAN1WTWwcRRYu27GdxE5wEnYF"
}
```

Поля ответа

- form - форма получения параметров аутентификации пользователя
- errors - список ошибок, сообщение может быть привязано к полю (field)
- message - сообщение об ошибке (определяется интеграционным сценарием)
- form.name - имя формы
- fields - список полей формы; в данном примере значение удалено для удобства чтения
- view - дополнительные данные текущего шага
- isBlocked - признак блокировки пользователя
- blockedFor - время до разблокировки в секундах
- externalAuthError - модель данных содержащая ошибки внешней системы (определяется интеграционным сценарием)
- serverUrl - URL для запросов на сервер
- step - идентификатор шага
- execution - идентификатор предсессии аутентификации

Отображение captcha

После превышения заданного числа попыток входа UIDM передает captcha и требует в следующем запросе передать символы этого изображения.

Состояние может отрисовываться как в сценарии входа по логину-пароллю, так и в сценарии создания привязки с соцсетями.

Таблица 1. Входные данные (от сервера к виджету)

имя	обязательный	описание	констрейнты
serverUrl	да	URL для следующего запроса	
execution	да	Идентификатор предсессии аутентификации	
step	да	Код состояния	всегда captcha_auth_form
view.recaptchaSiteKey	да	Ключ для формирования response для последующей валидации ReCaptcha	
form.errors[].message	нет	Сообщение об ошибке валидации	
form.errors[].field	нет	Поле в котором найдена ошибка	

Таблица 2. Данные, передающиеся только в сценарии создания привязки

имя	обязательный	описание	констрейнты
avatarUrl	нет	Адрес фотографии аккаунта соц. сети для привязки	
firstName	нет	Имя пользователя аккаунта соц. сети для привязки	
fullName	нет	Полное имя пользователя аккаунта соц. сети для привязки	
socialNetworkId	да	Идентификатор соц. сети (vkontakte, odnoklassniki) для привязки	

Пример

```
{
  "form": {
    "errors": [
      {
        "field": "captchaCode",
        "message": "invalid_captcha"
      }
    ],
    "name": "captchaLoginForm",
    "fields": {
      "username": {
        "constraints": [
          {
            "name": "NotNull"
          },
          {
            "name": "Size",
            "attributes": {
              "min": 10,
              "max": 25
            }
          },
          {
            "name": "FilteredSize",
            "attributes": {
              "skip": "([^9]*)|(^\\d)",
              "min": 10,
              "max": 10
            }
          }
        ]
      },
      "password": {
        "constraints": [
          {
            "name": "Size",
            "attributes": {
```

```
        "min": 4,
        "max": 1024
    },
    {
        "name": "NotNull"
    }
]
}
},
"view": {
    "recaptchaSiteKey": "6Lf5CQkUAAAAA3shbFPdp0b7HEWJivKgKFuAcNtR"
},
"serverUrl": "https://sso.rooxteam.com/sso/auth/websso",
"step": "captcha_auth_form",
"execution": "dec7dacf-as2d-4345-aae5-d08410cb8f95_H4sIAAAAAAAAAAN1WTWwcRRYu27GdxE5wEnYF..."
}
```

Таблица 3. Поля ответа

имя	обязательный	описание	констрейнты
username	да	Номер телефона	набор из символов, минимум 10, максимум 25
password	да	Пароль	4-1024 символа
gRecaptchaResponse	да	Response сформированный рекапчей для валидации	

Передача логина, пароля и символов captcha

В этом запросе используется значение параметра execution из неуспешного ответа с требованием ввода символов captcha. В случае успешной аутентификации будет возвращен access_token. В случае неуспешной аутентификации будет возвращен ответ, содержащий описание ошибок.

Ввод пользователем символов с captcha

```
POST /sso/oauth2/access_token
Host: <sso_host>
Accept: application/json
Cache-Control: no-cache
Content-Type: application/x-www-form-urlencoded

client_id=<client_id>&
client_secret=<client_secret>&
scope=<scope>&
grant_type=urn:roox:params:oauth:grant-type:m2m&
realm=%2Fcustomer&
service=dispatcher&
execution=<execution>&
username=<username>&
password=<password>&
```



```
captchaCode=<captcha_code>&
_eventId=next
```

Параметры запроса

- `<realm>` - группа пользователей WebSSO, всегда используется значение `%2Fcustomer`, которое является uri-encoded значением `/customer`
- `<sso_host>` - базовый адрес сервера WebSSO, например `sso.rooxteam.com`
- `<client_id>` - идентификатор клиента, например `selfcare`
- `<service>` - имя цепочки аутентификации, всегда `dispatcher`
- `<client_secret>` - пароль клиента
- `<scope>` - OAuth2.0 scope в кодировке UTF-8, **опционально, регистрозависимо**
- `<execution>` - идентификатор предсессии аутентификации, значение берется из предыдущего ответа сервера
- `<grant_type>` - способ авторизации пользователя, всегда используется значение `urn:roox:params:oauth:grant-type:m2m`
- `<username>` - имя пользователя (номер телефона)
- `<password>` - пароль пользователя
- `<captcha_code>` - символы captcha
- `<_eventId>` - идентификатор действия, всегда `next`

Формат успешного ответа

```
HTTP/1.1 200 OK
```

```
{
  "access_token": "fe8a0976-972c-4773-9ba8-662c92869639",
  "refresh_token": "863a4983-2424-4cb3-9ed2-18f9357b64c0",
  "refresh_expires_in": 1599,
  "scope": [
    "cn"
  ],
  "token_type": "Bearer",
  "expires_in": 599,
  "old_token": "fe8a0976-972c-4773-9ba8-662c92869639",
  "JWTToken": "eyJhZGQlOiIsieW90YV9sa19tMm\n0iXSwed3ViIjoiYmlzX19fX18yRpg3NjU0MzIxIiwiaBUoIjoxNjAxOTgzNTc4LCJhenAiOiJ5b3RhT5trX20ybSIs\nImFtcCI6ImRldmlljZV9zdG9yZWRFdG9rZW4iLCJ0b2t1bk5hbWUiOiFrCF90b2t1biIsImIzcyciOiJldvSWwbGtfbT\nJtIiwicmVhbG0iOiUvY3VzdG9tZXIiLCJ0b2t1blR5cGUiOiJCavJlZGRlZEpxVFRva2VuIiwiaXhwIjoxNjAxOTg0\nMTc4LCJpNOZiojE2MDE5ODM1Nzh9._kCRnEwOqI9K1JVCVbwofCqsZzzEm7PJBqfJCAG19bG"
```

Параметры запроса

- score - список score (в формате JSON Array) разрешенных для использования от имени пользователя, возможные значения задаются конфигурацией
- token_type - тип выданного токена. Всегда Bearer

- refresh_token - выданный refresh token
- refresh_expires_in - время до истечения срока действия refresh токена в секундах
- access_token - выданный access token
- expires_in - время до истечения срока действия токена в секундах
- old_token - токен, выданный внешней системой, строка
- JWTToken - JWT токен для длительного хранения и последующей аутентификации

Формат неуспешного ответа при неверном значении captcha

```
HTTP/1.1 200 OK
```

```
{
  "form": {
    "errors": [
      {
        "field": "captchaCode",
        "message": "invalid_captcha"
      }
    ],
    "name": "captchaLoginForm",
    "fields": {}
  },
  "view": {
    "captchaUrl": "https://sso.rooxteam.com/sso/api/captchas/e1bd0719-f8cd-4aa9-9bc6-d8c1c834932e"
  },
  "serverUrl": "https://sso.rooxteam.com/sso/auth/websso",
  "step": "captcha_auth_form",
  "execution": "dec7dacf-as2d-4345-aae5-d08410cb8f95_H4sIAAAAAAAAAAN1WTWwcRRYu27GdxE5wEnYF"
}
```

Поля ответа

- form - форма получения параметров аутентификации пользователя
- errors - список ошибок, сообщение может быть привязано к полю (field)
- field - поле, к которому привязано сообщение об ошибке
- form.name - имя формы
- fields - список полей формы; в данном примере значение удалено для удобства чтения
- view - дополнительные данные текущего шага
- isBlocked - признак блокировки пользователя
- blockedFor - время до разблокировки в секундах
- serverUrl - URL для запросов на сервер
- step - идентификатор шага
- execution - идентификатор предсессии аутентификации

Двухфакторная аутентификация

Если на окружении и у пользователя включена настройка, требующая подтверждение аутентификации с помощью OTP, то будет запрошен код на шаге "step": "enter_otp_form".

Формат ответа, требующий ввода OTP

```
HTTP/1.1 200 OK
```

```
{
  "step": "enter_otp_form",
  "execution": "30cfc0f9-b1d4-4820-a476-06056f140e12_AAABFQAAQBFRxdnz6ys",
  "serverUrl": "https://sso.rooxteam.com/sso/auth/_otp-sms",
  "form": {
    "fields": {
      "otpCode": {
        "constraints": [
          {
            "name": "NotNull"
          },
          {
            "attributes": {
              "max": 4,
              "min": 4
            },
            "name": "Size"
          },
          {
            "attributes": {
              "regexp": "^[0-9]+$",
              "flags": []
            },
            "name": "Pattern"
          }
        ]
      }
    },
    "errors": [],
    "name": "otpForm"
  },
  "view": {
    "msisdn": "79876543210",
    "isBlocked": false,
    "nextOtpCodePeriod": 29,
    "blockedFor": 0,
    "expireOtpCodeTime": 59,
    "otpCodeAvailableAttempts": 4
  }
}
```

Поля ответа

- form - форма получения параметров аутентификации пользователя
- errors - список ошибок, сообщение может быть привязано к полю (field)
- form.name - имя формы
- fields - список полей формы
- otpCode - атрибуты, относящиеся к коду OTP

- constraints - список ограничений на поле
- constraints.name - имя ограничения
- constraints.attributes - перечень атрибутов ограничений
- constraints.attributes.min - минимальное значение для данного атрибута
- constraints.attributes.max - максимальное значение для данного атрибута
- constraints.attributes.skip - регулярное выражение валидации значения для поля
- view - дополнительные данные текущего шага
- view.msisdn - MSISDN, на который было отправлено сообщение с одноразовым кодом
- view.nextOtpCodePeriod - время в секундах до наступления возможности отправки нового OTP кода
- view.otpCodeAvailableAttempts - количество попыток ввода OTP кода
- view.isBlocked - признак блокировки пользователя
- view.blockedFor - время до разблокировки в секундах
- serverUrl - URL для запросов на сервер
- step - идентификатор шага
- execution - идентификатор предсессии аутентификации

Валидация OTP

Данный запрос валидирует введенный OTP код пользователем. И результатом выполнения данного запроса будет выдача access token.

```
POST /sso/oauth2/access_token
Host: <sso_host>
Accept: application/json
Cache-Control: no-cache
Content-Type: application/x-www-form-urlencoded

client_id=<client_id>&
client_secret=<client_secret>&
scope=<scope>&
grant_type=urn:roox:params:oauth:grant-type:m2m&
realm=%2Fcustomer&
service=dispatcher&
response_type=token&
execution=<execution>&
otpCode=<otpCode>&
_eventId=start
```

Параметры

- execution - идентификатор предсессии аутентификации
- _eventId - идентификатор действия, определяется отдельно для каждого состояния
- otpCode - полученный OTP код

Формат успешного ответа на валидацию OTP

```
HTTP/1.1 200 OK
```

```
{
  "access_token": "fe8a0976-972c-4773-9ba8-662c92869639",
  "refresh_token": "863a4983-2424-4cb3-9ed2-18f9357b64c0",
  "refresh_expires_in": 1599,
  "scope": [
    "cn"
  ],
  "token_type": "Bearer",
  "expires_in": 599,
  "old_token": "fe8a0976-972c-4773-9ba8-662c92869639",
  "JWTToken": "eyJhZDQwI0lsiew90YV9sa19tMm0iXSwed3ViIjoiYmlzX19fX18yRpg3NjU0MzIxIiwiaWUoIjoxNjAxOTgzNTc4LCJhenAiOiJ5b3RhT5trX20ybSIiImFtciI6ImRldmIjZV9zdG9yZWZldG9rZW4iLCJ0b2t1bk5hbWUiOiFrCF90b2t1biIsImIzcyI6InlvdSWwbGtfbTJtIiwicmVhbG0iOiUvY3VzdG9tZXIiLCJ0b2t1b1R5cGUiOiJCavJlZGRlZEpXVFRva2VuIiwiaXhwIjoxNjAxOTgzMTc4LCJpN0ZiOjE2MDE5ODM1Nzh9._kCRnEw0qI9K1JVCVbwofCqsZzzEm7PJBqfJCAG19bG"
}
```

Параметры запроса

- scope - список scope (в формате JSON Array) разрешенных для использования от имени пользователя, возможные значения задаются конфигурацией
- token_type - тип выданного токена. Всегда Bearer
- refresh_token - выданный refresh token
- refresh_expires_in - время до истечения срока действия refresh токена в секундах
- access_token - выданный access token
- expires_in - время до истечения срока действия токена в секундах
- old_token - токен, выданный внешней системой, строка
- JWTToken - JWT токен для длительного хранения и последующей аутентификации
- JWTToken - может быть использован для аутентификации в сценарии [Автоматический вход](#)

Формат ошибки валидации OTP

```
HTTP/1.1 200 OK
Content-Type: application/json
```

```
{
  "form": {
    "errors": [
      {
        "field": "otpCode",
        "message": "invalid_otp"
      }
    ],
    "name": "otpForm",
    "fields": {
      "otpCode": {
        "constraints": [
```

```

    {
      "name": "NotNull"
    }
  ]
}
},
"view": {
  "otpCodeAvailableAttempts": 2,
  "msisdn": "79876543210",
  "nextOtpPeriod": 120,
  "blockedFor": 0,
  "isBlocked": false
},
"serverUrl": "http://sso.ocb.hosted:8080/sso/auth/otp-sms",
"step": "otp_form",
"execution": "dec7dacf-as2d-4345-aae5-d08410cb8f95_H4sIAAAAAAAAAAN1WTWwcRRYu27GdxE5wEnYF"
}

```

Поля ответа

- execution - идентификатор предсессии аутентификации
- _eventId - идентификатор действия, определяется отдельно для каждого состояния
- view.msisdn - номер телефона, на который будет отправлен OTP SMS
- view.otpCodeAvailableAttempts - кол-во попыток ввода OTP кода
- view.nextOtpPeriod - время в секундах до наступления возможности отправки нового OTP кода
- view.isBlocked - признак блокировки пользователя
- view.blockedFor - время до разблокировки в секундах
- form.errors[].message - сообщение об ошибке валидации, **опционально**
- form.errors[].field - поле в котором найдена ошибка, **опционально**

Аутентификация по номеру телефона и OTP коду

Если в UIDM и у пользователя включена возможность аутентификации через OTP, после первого шага можно передать _eventId=login-by-otp и execution.

ЗАМЕТКА

Это самостоятельный способ входа, а не двухфакторная аутентификация

Формат запроса

```

POST /sso/oauth2/access_token HTTP/1.1
Host: <sso_host>
Accept: application/json
Cache-Control: no-cache
Content-Type: application/x-www-form-urlencoded

client_id=<client_id>&
client_secret=<client_secret>&
scope=<scope>&
grant_type=urn:roox:params:oauth:grant-type:m2m&
realm=<realm>&

```

```
service=dispatcher&
response_type=token&
execution=<execution>&
_eventId=login-by-otp
```

Параметры запроса

- <sso_host> - базовый адрес сервера UIDM, например sso.rooxteam.com
- <client_id> - идентификатор клиента, например selfcare
- <client_secret> - пароль клиента
- <scope> - OAuth2.0 scope в кодировке UTF-8, **опционально, регистрозависимо**
- <grant_type> - способ авторизации пользователя, всегда используется значение urn:roox:params:oauth:grant-type:m2m
- <realm> - группа пользователей UIDM, всегда используется значение %2Fcustomer, которое является uri-encoded значением /customer
- <execution> - идентификатор предсессии аутентификации, значение берется из предыдущего ответа сервера
- <service> - имя цепочки аутентификации, всегда dispatcher
- <response_type> - способ авторизации пользователя; в данном сценарии всегда token
- <_eventId> - идентификатор действия, всегда login-by-otp

Формат успешного ответа

```
HTTP/1.1 200 OK
```

```
{
  "step": "login-by-otp-form",
  "execution": "2787d6ac-d397-435a-a6f5-b2dd72f11725",
  "serverUrl": "http://sso.demo.rooxteam.com:8080/sso/auth/_login-by-otp",
  "form": {
    "fields": {
      "msisdn": {
        "constraints": [
          {
            "name": "NotNull"
          },
          {
            "attributes": {
              "max": 10,
              "message": "symbols {skip} should be filtered out, and resulting string should have length between {min} and {max}",
              "skip": "([^9+)]|([0-9])",
              "min": 10
            },
            "name": "FilteredSize"
          },
          {
            "attributes": {
              "max": 25,
              "min": 10
            },
            "name": "Size"
          }
        ]
      }
    }
  }
}
```

```
{
  }
]
},
"errors": [],
"name": "form"
}
}
```

Поля ответа

- form - форма получения параметров аутентификации пользователя
- errors - список ошибок, сообщение может быть привязано к полю (field)
- form.name - имя формы
- fields - список полей формы
- constraints - список ограничений на поле
- constraints.name - имя ограничения
- constraints.attributes - перечень атрибутов ограничений
- constraints.attributes.min - минимальное значение для данного атрибута
- constraints.attributes.max - максимальное значение для данного атрибута
- constraints.attributes.skip - регулярное выражение валидации значения для поля
- serverUrl - URL для запросов на сервер
- step - идентификатор шага
- execution - идентификатор предсессии аутентификации

Ввод номера телефона и отправка СМС сообщения

После выполнения данного запроса будет отправлено смс на введенный номер телефона.

```
POST /sso/oauth2/access_token HTTP/1.1
Host: <sso_host>
Accept: application/json
Content-Type: application/x-www-form-urlencoded

client_id=<client_id>&
client_secret=<client_secret>&
scope=<scope>&
grant_type=urn:roox:params:oauth:grant-type:m2m&
realm=<realm>&
service=dispatcher&
response_type=token&
execution=<execution>&
msisdn=<msisdn>&
_eventId=next
```

Параметры запроса

- <sso_host> - базовый адрес сервера UIDM, например sso.rooxteam.com
- <client_id> - идентификатор клиента, например selfcare
- <client_secret> - пароль клиента

- <scope> - OAuth2.0 scope в кодировке UTF-8, **опционально, регистрозависимо**
- <grant_type> - способ авторизации пользователя, всегда используется значение urn:roox:params:oauth:grant-type:m2m
- <realm> - группа пользователей UIDM, всегда используется значение %2Fcustomer, которое является uri-encoded значением /customer
- <execution> - идентификатор предсессии аутентификации, значение берется из предыдущего ответа сервера
- <service> - имя цепочки аутентификации, всегда dispatcher
- <response_type> - способ авторизации пользователя; в данном сценарии всегда token
- <msisdn> - msisdn аккаунта
- <_eventId> - идентификатор действия, всегда next

Формат успешного ответа

```
HTTP/1.1 200 OK
```

```
{
  "step": "enter_otp_form",
  "execution": "c0b47945-ed8a-4cf7-a0aa-781dfe5988e1",
  "serverUrl": "http://sso.demo.rooxteam.com:8080/sso/auth/_otp-sms",
  "form": {
    "fields": {
      "otpCode": {
        "constraints": [
          {
            "name": "NotNull"
          },
          {
            "attributes": {
              "max": 4,
              "min": 4
            },
            "name": "Size"
          },
          {
            "attributes": {
              "regexp": "^[0-9]+$",
              "flags": []
            },
            "name": "Pattern"
          }
        ]
      }
    },
    "errors": [],
    "name": "otpForm"
  },
  "view": {
    "msisdn": "9876543210",
    "isBlocked": false,
    "nextOtpCodePeriod": 29,
    "nextOtpPeriod": 29,
    "blockedFor": 0,
    "expireOtpCodeTime": 59,
    "otpCodeAvailableAttempts": 4,
```

```
"otpCodeNumber": 0
```

```
}
```

```
}
```

Поля ответа

- form - форма получения параметров аутентификации пользователя
- errors - список ошибок, сообщение может быть привязано к полю (field)
- form.name - имя формы
- fields - список полей формы
- constraints - список ограничений на поле
- constraints.name - имя ограничения
- constraints.attributes - перечень атрибутов ограничений
- constraints.attributes.min - минимальное значение для данного атрибута
- constraints.attributes.max - максимальное значение для данного атрибута
- constraints.attributes.skip - регулярное выражение валидации значения для поля
- view.nextOtpPeriod - время в секундах до наступления возможности отправки нового OTP кода
- view.otpCodeAvailableAttempts - кол-во попыток ввода OTP кода
- view.msisdn - номер телефона, на который будет отправлен OTP SMS
- view.blockedFor - время до разблокировки в секундах
- view.isBlocked - признак блокировки пользователя
- view.nextOtpPeriod - время в секундах до наступления возможности отправки нового OTP кода
- view.expireOtpCodeTime - время жизни OTP кода в секундах
- serverUrl - URL для запросов на сервер
- step - идентификатор шага
- execution - идентификатор предсессии аутентификации

Валидация введенного кода

```
POST /sso/oauth2/access_token HTTP/1.1
Host: <sso_host>
Accept: application/json
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Host: <sso_host>

realm=<realm>&
client_id=<client_id>&
client_secret=<client_secret>&
grant_type=urn%3Aroox%3Aparams%3Aoauth%3Agrant-type%3Am2m&
service=otp_operation_token&
response_type=token&
execution=<execution>&
_eventId=validate&
otpCode=<code>
```

Параметры запроса

- <sso_host> - базовый адрес сервера UIDM, например sso.rooxteam.com, может содержать порт, например sso.rooxteam.com:8080
- <realm> - группа пользователей UIDM
- <client_id> - идентификатор клиента, например selfcare, возможные значения зависят от конфигурации
- <client_secret> - пароль клиента, возможные значения зависят от конфигурации
- <grant_type> - способ авторизации пользователя, в данном сценарии всегда используется значение urn:roox:params:oauth:grant-type:m2m
- <service> - используемый сервис. В данном сценарии всегда otp_operation_token.
- <response_type> - способ авторизации пользователя; в данном сценарии всегда token
- <execution> - идентификатор сессии аутентификации из ответа предыдущего запроса
- <_eventId> - действие в контексте данного шага. В данном сценарии всегда validate
- <code> - код из отправленного сообщения

Успешный ответ

HTTP/1.1 200 OK

```
{
  "access_token": "fe8a0976-972c-4773-9ba8-662c92869639",
  "refresh_token": "863a4983-2424-4cb3-9ed2-18f9357b64c0",
  "refresh_expires_in": 1599,
  "scope": [
    "cn"
  ],
  "token_type": "Bearer",
  "expires_in": 599,
  "old_token": "fe8a0976-972c-4773-9ba8-662c92869639",
  "JWTToken": "eyJAiY3R5IjogRhlXVCIsICJ0eXZu5iAiand0IiwgImFsGuM6ICJlUzI1NiIgfgQ.eyJhdWQiOi0lsiew90YV9sa19tMm0iXSwed3ViIjoiYmlzX19fX18yRpg3NjU0MzIxIiwiaBUoIjoxNjAxOTgzNTc4LCJhenAiOiJ5b3RhT5trX20ybSIsImFtciI6ImRldmljZV9zdG9yZWRFdG9rZW4iLCJ0b2t1bWU0IiwiaWUiOiJCaVJlZGRlZEpxVFRva2VuIiwiaXhwIjoxNjAxOTg0MTc4LCJpbn0Zi0jE2MDE5ODM1Nzh9._kCRnEwOqI9K1JVCVbwofCqsZzzEm7PJBqfJCAG19bG"
}
```

Параметры запроса

- scope - список scope (в формате JSON Array) разрешенных для использования от имени пользователя, возможные значения задаются конфигурацией
- token_type - тип выданного токена. Всегда Bearer
- refresh_token - выданный refresh token
- refresh_expires_in - время до истечения срока действия refresh токена в секундах
- access_token - выданный access token
- expires_in - время до истечения срока действия токена в секундах
- old_token - токен, выданный внешней системой, строка

- JWTToken - JWT токен для длительного хранения и последующей аутентификации

Обработка ошибок и особых случаев

Блокировка пользователя

Формат ответа при блокировке пользователя

```
HTTP/1.1 200 OK
Content-Type: application/json
```

```
{
  "form": {
    "errors": [
      {
        "message": "too_many_wrong_code"
      }
    ]
  },
  "view": {
    "blockedTo": "2018-02-18T12:00:00.000+00:00"
  },
  "serverUrl": "http://sso.rooxteam.com/sso/auth/otp-sms",
  "step": "enter_otp_form",
  "execution": "dec7dacf-as2d-4345-aae5-d08410cb8f95_H4sIAAAAAAAAAAN1WTWwcRRYu27GdxE5wEnYF"
}
```

Поля ответа

- execution - идентификатор предсессии аутентификации
- _eventId - идентификатор действия, определяется отдельно для каждого состояния
- view.blockedTo - дата, время в формате UTC определяет срок блокировки.
- form.errors[].message - сообщение об ошибке валидации, **опционально**
- form.errors[].field - поле в котором найдена ошибка, **опционально**

После заданного количества неверных попыток ввода пароля для одного и того же логина требуется ввод символов captcha. После заданного количества неверных попыток ввода пароля или символов captcha для одного и того же логина происходит временная блокировка по логину. Подсчет ошибочных попыток входа, captcha и блокировка происходят и для логинов, которые не зарегистрированы в системе. После заданного количества неверных попыток ввода логина/пароля/captcha с одного и того же IP-адреса в течение определенного количества секунд, происходит блокировка этого адреса по IP.

Формат ответа при временной блокировке по логину

```
HTTP/1.1 200 OK
```

```
{
  "form": {
    "errors": [
      {
        "message": "user_blocked"
      }
    ],
    "name": "loginForm",
    "fields": {}
  },
  "view": {
    "blockedFor": 3000,
    "isBlocked": true
  },
  "serverUrl": "https://sso.rooxteam.com/sso/auth/login-widget-router",
  "step": "auth_form",
  "execution": "dec7dacf-as2d-4345-aae5-d08410cb8f95_H4sIAAAAAAAAAAN1WTWwcRRYu27GdxE5wEnYF"
}
```

Поля ответа

- form - форма получения параметров аутентификации пользователя
- errors - список ошибок, сообщение может быть привязано к полю (field)
- form.name - имя формы
- fields - список полей формы; в данном примере значение удалено для удобства чтения
- view - дополнительные данные текущего шага
- isBlocked - признак блокировки пользователя
- blockedFor - время до разблокировки в секундах
- serverUrl - URL для запросов на сервер
- step - идентификатор шага
- execution - идентификатор предсессии аутентификации

Принудительная смена пароля пользователя при входе

Если для текущей пары логин/пароль пользователя установлен флаг passwordMustChange будет затребована смена пароля пользователя.

Подробнее о форматах запросов и ответов см. в разделе [Смена данных аутентификации \(логин и пароль\) пользователя](#)

Блокировка доступа абонента к конкретному ресурсу по client_id

Абоненту может быть заблокирован доступ к конкретному ресурсу по client_id, в этом случае, после передачи логина и пароля вернется ответ с HTTP кодом 400 "Bad Request"

Формат ответа

```
HTTP/1.1 400 Bad Request
```

```
{
  "error_description": "The resource owner or authorization server denied the request.",
  "error": "access_denied"
}
```

Поля ответа

- error_description - текстовое описание ошибки
- error - код ошибки согласно спецификации OAuth 2.0 (<https://tools.ietf.org/html/rfc6749#section-5.2>[RFC 6749 пункт 5.2])

Возможные ограничения на поля формы

название	описание	атрибуты	описание атрибута
NotNull	Обязательность поля		
Size	Длина строкового параметра	min	Минимальная длина
		max	Максимальная длина
Pattern	Regex для строкового параметра	regex	Регулярное выражение
Min	Минимальное значение целого числового параметра	value	Минимальное значение
Max	Максимальное значение целого числового параметра	value	Максимальное значение
название	описание	атрибуты	описание атрибута
FilteredSize	Ограничения для номера телефона	min	Минимальная длина
		max	Максимальная длина
		skip	Регулярное выражение для фильтрации. Перед отправкой нужно удалять все, что ему удовлетворяет. Например, происходит очищение введенных данных ото всех символов кроме цифр, далее удаляются все символы слева до первой девятки.

Возможные коды ошибок

Код ошибки	Описание причины возникновения
------------	--------------------------------

invalid_credentials	Пользователь ввел неправильные данные учетной записи и не был аутентифицирован.
expired_password	Время действия пароля, введенного пользователем, истекло. Пользователь не был аутентифицирован.
user_blocked	Учетная запись пользователя заблокирована.
ip_blocked	IP адресс, с которого пользователь обращается в WebSSO, заблокирован.
invalid_captcha	Введенная пользователем CAPTCHA неверная.
need_captcha	Для продолжения сценария пользователь должен ввести CAPTCHA.
error	Пользователь не был аутентифицирован в результате неожиданного ответа от внешней системы.
login-by-otp-disabled	Пользователь попытался аутентифицироваться через сценарий с аутентификацией по OTP, но это невозможно, поскольку данный функционал отключен конфигурационным ключем.
error_sending_otp	Невозможно отправить OTP пользователю.
too_many_sms	Невозможно отправить OTP пользователю, поскольку превышено число допустимых повторных запросов отправки OTP. Примечание: OTP может быть отправлен пользователю произвольным способом, а не только через sms сообщение.
too_many_wrong_code	Превышено число допустимых попыток ввода OTP.
invalid_otp	Введенный OTP неверный.
validate-otp-fail	Введенный OTP неверный.
otp_expired	Время действия введенного OTP истекло.
otp-expired	Время действия введенного OTP истекло.
otp-error	Сценарий ввода OTP завершился неуспешно.
external-api-error	Выполнение сценария невозможно, поскольку получен неожиданный ответ от внешней системы или внешняя система не доступна.
external-system-bad-response	Выполнение сценария невозможно, поскольку получен неожиданный ответ от внешней системы или внешняя система не доступна.
msisdn-is-not-b2b	Введенный msisdn не принадлежит реалму b2b.

too_many_attempts	Превышено число допустимых попыток изменения пароля.
user-is-not-allowed	Данному пользователю запрещено продолжать данный сценарий.
error_password_change	Сценарий изменения пароля не был успешно завершен. Пароль не был изменен.
no_email_found	Пользователь не найден или у пользователя не существует email.
msisdn-not-exists	Пользователя с заданным msisdn не существует.
principal-not-exists	Пользователь не существует.
no-principal-found	Пользователь не существует.
user-not-found	Пользователь не существует.
login-exists	Невозможно зарегистрировать пользователя с заданным login, поскольку пользователь с заданным login уже существует.
login_already_exists	Невозможно изменить login пользователя на заданный, поскольку пользователь с заданным login уже существует.
email-exists	Пользователь с заданным email уже существует.
user-exists	Невозможно зарегистрировать пользователя с заданными msisdn или email или login, поскольку пользователь с такими параметрами уже существует.
email_verification_failed	Невозможно подтвердить email.
reset_required	Пароль или не был задан либо был сброшен в результате каких-то действий, необходимо создать новый пароль. Отличается от expired_password, поскольку в том случае пароль существует и был провалидирован, но требуется смена. В данном случае пароль не провалидирован, поскольку не задан.

Аутентификация от имени абонента

Аутентификация от имени абонента выполняется по протоколу OAuth2.0 с передачей специального значения параметра grant_type.

Формат запроса


```
POST /sso/oauth2/access_token
Host: <sso_host>
Accept: application/json
Content-Type: application/x-www-form-urlencoded
```

```
client_id=<client_id>&
client_secret=<client_secret>&
scope=<scope>&
grant_type=urn:roox:params:oauth:grant-type:m2m&
realm=%2Fcustomer&
service=impersonate-auth&
jwt=<jwt>
```

Параметры

- <sso_host> - базовый адрес сервера WebSSO, например sso.rooxteam.com
- <realm> - группа пользователей WebSSO, например, значение %2Fcustomer, которое является uri-encoded значением /customer
- <client_id> - идентификатор клиента, например selfcare
- <client_secret> - пароль клиента
- <scope> - OAuth2.0 scope в кодировке UTF-8, **опционально, регистрозависимо**
- <grant_type> - способ авторизации пользователя, в данном сценарии всегда используется значение urn:roox:params:oauth:grant-type:m2m
- <service> - имя цепочки аутентификации, в данном сценарии всегда dispatcher
- <jwt> - токен, разрешающий аутентификацию от имени абонента

Формат ответа при успешном автовоходе

```
HTTP/1.1 200 OK
```

```
{
  "access_token": "fe8a0976-972c-4773-9ba8-662c92869639",
  "refresh_token": "863a4983-2424-4cb3-9ed2-18f9357b64c0",
  "refresh_expires_in": 1599,
  "scope": [
    "cn"
  ],
  "token_type": "Bearer",
  "expires_in": 599,
  "old_token": "fe8a0976-972c-4773-9ba8-662c92869639",
  "JWTToken":
  "eyJhY3R5IjogRhlXVCIsICJ0eXZu5iAiand0IiwgImFsGuM6ICJIUzI1NiIgQ.eyJhdWQiOi0lsieW90YV9sa19tMm
  0iXSwed3ViIjoiYmlzX19fX18yRpg3NjU0MzIxIiwiaBUoIjoxNjAxOTgzNTc4LCJhenAiOiJ5b3RhT5trX20ybSIs
  ImFtciI6ImRldmljZV9zdG9yZWZldG9rZW4iLCJ0b2t1bk5hbWUiOiFrCF90b2t1biIsImIzcyI6InlvdSWwbGtfbT
  JtIiwicmVhbG0iOi0uUvY3VzdG9rZXIiLCJ0b2t1blR5cGUiOiJCaVJlZGRlZEpxVFRva2VuIiwiaXhwIjoxNjAxOTgz
  MTc4LCJpNOZiOjE2MDE5ODM1Nzh9._kCRnEw0qI9K1JVCVbwofCqsZzzEm7PJBqfJCAG19bG"
}
```

Параметры запроса

- scope - список scope (в формате JSON Array) разрешенных для использования от имени пользователя, возможные значения задаются конфигурацией
- token_type - тип выданного токена. Всегда Bearer
- refresh_token - выданный refresh token
- refresh_expires_in - время до истечения срока действия refresh токена в секундах
- access_token - выданный access token
- expires_in - время до истечения срока действия токена в секундах
- old_token - токен, выданный внешней системой, строка
- JWTToken - JWT токен для длительного хранения и последующей аутентификации

Формат ответа сервера после неуспешного входа

Если JWT токен, разрешающий аутентификацию от имени абонента, не действителен, будет возвращена ошибка:

```
HTTP/1.1 200 OK
```

```
{
  "error_description": "The provided access grant is invalid, expired, or revoked.",
  "error": "invalid_grant"
}
```

Валидация токена доступа

Ранее полученный токен может оказаться просрочен или инвалидирован, поэтому перед выполнением клиентом действий от имени пользователя, необходимо выполнять валидацию токена. Для валидации токена нужно выполнить запрос на соответствующий URL UIDM с указанием access token. Результатом будет информация о переданном токене либо сообщение об ошибке.

Помимо простой валидации токена можно запрашивать разрешение на доступ к защищаемому ресурсу. Для этого нужно передать в запрос на валидацию токена дополнительный параметр scope. Можно передать любой scope, не обязательно указанный при аутентификации. Если доступ разрешен, то ответ будет аналогичен ответу без указания метода. Если нет, ответ будет содержать информацию о требуемом уровне авторизации для получения доступа к запрашиваемому методу.

Если требуется передать дополнительные параметры для аудита доступа к защищенным ресурсам (см. документацию "Спецификация API для ЛК"), необходимо выполнить POST-запрос и передать параметры в теле запроса в виде json-объекта.

Формат запроса

```
POST /sso/oauth2/tokeninfo?access_token=<access_token>&scope=<scope>
Host: <sso_host>
Content-Type: application/json
Accept: application/json
```

```
{
  "httpMethod": "POST",
  "url": "http://example.com/some/url",
  "headers": {
    "User-Agent": [
      "Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/40.0.2214.115 Safari/537.36"
    ],
    "Referer": [
      "https://www.google.com/"
    ],
    "X-Provider-Net": [

```

```

    "1"
  ],
  "X-Nokia-MSISDN": [
    "9876543210"
  ],
  "X-Forwarded-For": [
    "10.20.30.40",
    "10.10.35.46",
    "192.168.12.74"
  ]
}

```

Параметры

- <sso_host> - базовый адрес сервера UIDM, например sso.rooxteam.com, может содержать порт, например sso.rooxteam.com:8080
- access_token - полученный в результате аутентификации access token, например 7bdaeacc-3d80-415c-920f-a7c30ca5e743
- scope - идентификатор ресурса защищаемого сервиса в кодировке UTF-8, см. подробнее в пункте [Использование параметра scope](#), **опционально, регистрозависимо**
- httpMethod - HTTP метод, для которого запрашивается доступ; **опционально**
- url - URL, для которого запрашивается доступ, **опционально**
- headers - список HTTP заголовков из запроса пользователя к защищаемому сервису, передаются без дополнительной обработки или фильтрации, **опционально**

Формат успешного ответа

```
HTTP/1.1 200 OK
```

```

{
  "scope": [
    "cn",
    "networkAuthenticationType"
  ],
  "cn": "9263752235",
  "realm": "/customer",
  "token_type": "Bearer",
  "JWTToken": "eyJAi...9lEuA",
  "expires_in": 28,
  "access_token": "7bdaeacc-3d80-415c-920f-a7c30ca5e743",
  "auth_level": "2",
  "networkAuthenticationType": "AUTO",
  "client_id": "selfcare"
}

```

Параметры

- scope - список scope (в формате JSON Array) разрешенных для использования от имени пользователя, возможные

значения задаются конфигурацией

- realm - группа пользователей UIDM, всегда возвращается значение /customer
- token_type - тип выданного токена. Всегда Bearer
- expires_in - время до истечения срока действия токена в секундах
- access_token - проверяемый access token
- auth_level - выданный уровень авторизации пользователя
- client_id - идентификатор клиента, которому выдан токен, возможные значения зависят от конфигурации
- cn - номер телефона пользователя
- networkAuthenticationType - разрешен ли автоматический вход

Набор возможных значений scope зависит от конфигурации.

Формат неуспешного ответа

При передаче невалидного токена или токена с истекшим сроком действия, а также при блокировке учетной записи абонента и при блокировке абоненту доступа к конкретному ресурсу по client_id будет возвращен ответ:

```
HTTP/1.1 401 Unauthorized
```

```
{
  "error": "expired_token",
  "error_description": "The request contains a token no longer valid."
}
```

Параметры

- error - код ошибки согласно спецификации OAuth 2.0 (<https://tools.ietf.org/html/rfc6749#section-5.2> [RFC 6749 пункт 5.2])
- error_description - текстовое описание ошибки

Формат неуспешного ответа при недостаточном уровне авторизации

```
HTTP/1.1 403 Forbidden
```

```
{
  "scope": [
    "cn"
  ],
  "cn": "9876543210",
  "realm": "/customer",
  "token_type": "Bearer",
  "JWTToken": "eyJAi...9lEuA",
  "expires_in": 31,
```

```
{
  "advices": {
    "required_auth_level": "5"
  },
  "auth_level": "2",
  "access_token": "5fdfeafd-3061-4b1c-9076-0fe460f91fc8",
  "client_id": "selfcare"
}
```

Параметры

- scope - список scope (в формате JSON Array) разрешенных для использования от имени пользователя, возможные значения задаются конфигурацией
- realm - группа пользователей UIDM, всегда возвращается значение /customer
- token_type - тип выданного токена. Всегда Bearer
- expires_in - время до истечения срока действия токена в секундах
- advices - перечень необходимых условий, которые требуется удовлетворить для получения доступа к защищаемому ресурсу
- required_auth_level - требуемый уровень авторизации для данного scope
- access_token - проверяемый access token
- auth_level - выданный уровень авторизации пользователя
- client_id - идентификатор клиента, которому выдан токен, возможные значения зависят от конфигурации

Формат ответа при блокировке OAuth 2.0 клиента

```
HTTP/1.1 403 Forbidden
```

```
{
  "error": "client_blocked",
  "error_description": "Client is blocked."
}
```

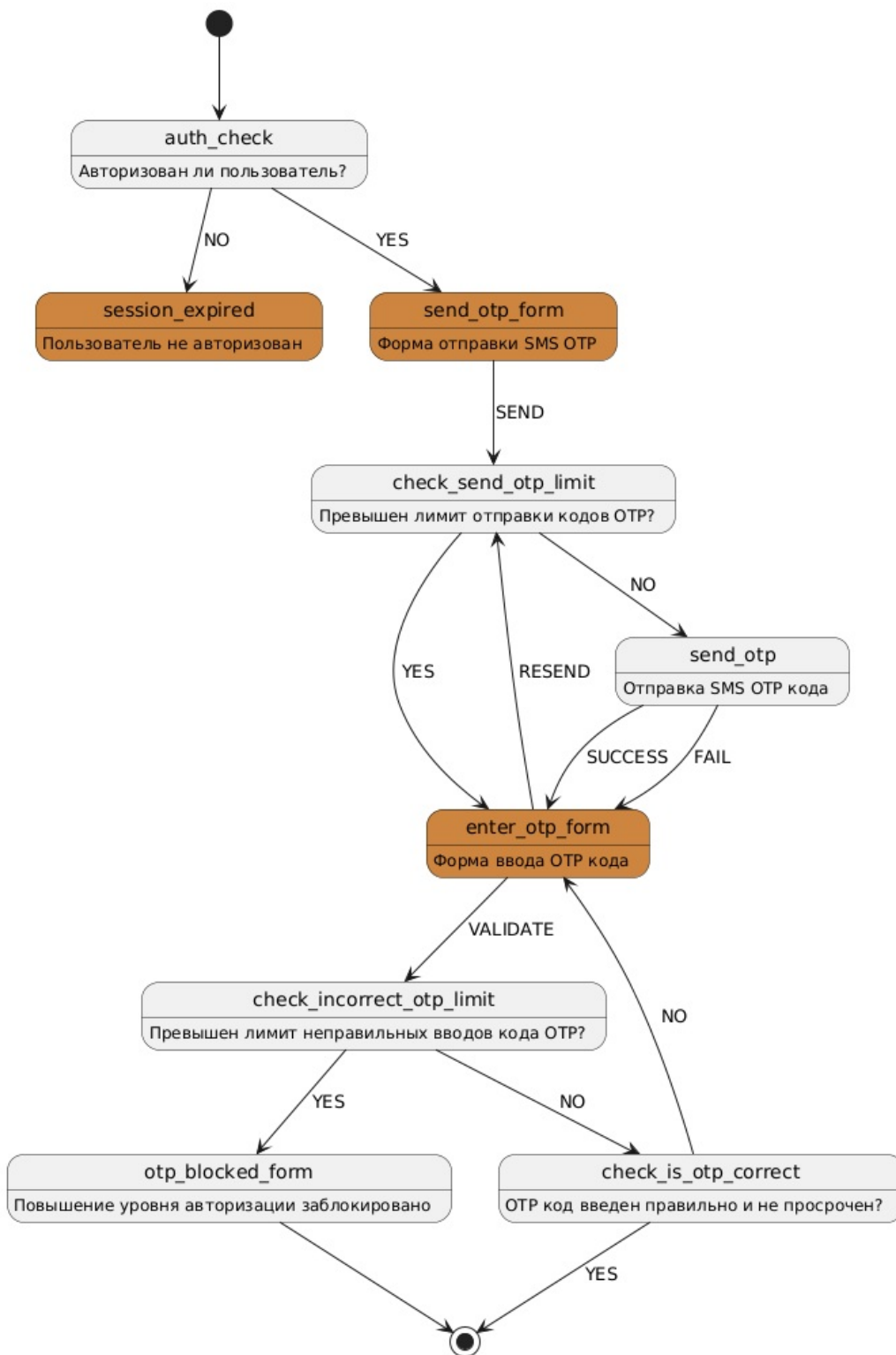
Параметры

- error - код ошибки согласно спецификации OAuth 2.0 Token Revocation (<http://tools.ietf.org/html/rfc7009#section-4.1.1>[RFC 7009 пункт 4.1.1])
- error_description - текстовое описание ошибки

Повышение уровня авторизации WebSSO с помощью OTP

Схема повышения уровня авторизации

Цветом выделены состояния которые передают управление виджету.



Повышение уровня авторизации происходит через OAuth 2.0 с передачей параметра `auth_level`. Если уровень авторизации пользователя меньше запрашиваемого, он будет перенаправлен на виджет повышения уровня по OTP коду.

Если SMS-код был успешно подтвержден, будет возвращен `access_token` с повышенным уровнем авторизации `auth_level`

Повышенный уровень авторизации действует фиксированное время (по-умолчанию 3 минуты), по истечении которого уровень авторизации возвращается к исходному значению.

ЗАМЕТКА

Следует заметить, что старый `access token` продолжает действовать и для запросов, где не нужен повышенный уровень авторизации, нужно использовать его.

ЗАМЕТКА

Время действия нового access token меньше чем у старого

ВАЖНО

Возможные ограничения на поля формы описаны в пункте [Возможные ограничения на поля формы](#)

ЗАМЕТКА

После успешного повышения уровня авторизации OAuth 2.0 /tokeninfo вернет auth_level соответствующий запрашиваемому

ЗАМЕТКА

В каждый запрос нужно подставлять текущий execution - идентификатор предсессии аутентификации.

Формат запроса на повышение уровня авторизации

```
POST /sso/oauth2/access_token
Host: <sso_host>
Accept: application/json
Content-Type: application/x-www-form-urlencoded

client_id=<client_id>&
client_secret=<client_secret>&
scope=<scope>&
grant_type=urn:roox:params:oauth:grant-type:m2m&
realm=/customer&
service=dispatcher&
auth_level=<auth_level>&
access_token=<access_token>&
method=<method>
```

Параметры

- sso_host - базовый адрес сервера WebSSO, например sso.rooxteam.com, может содержать порт, например sso.rooxteam.com:8080
- client_id - идентификатор клиента, например selfcare, возможные значения зависят от конфигурации
- client_secret - пароль клиента, возможные значения зависят от конфигурации
- scope - список запрашиваемых scope через пробел в кодировке UTF-8. **опционально, регистрозависимо**
- grant_type - способ авторизации пользователя, всегда используется значение urn:roox:params:oauth:grant-type:m2m
- service - имя цепочки аутентификации, всегда dispatcher
- realm - группа пользователей WebSSO, всегда используется значение %2Fcustomer, которое является uri-encoded значением /customer
- auth_level - желаемый уровень авторизации для текущего пользователя.
- method - предпочтительный способ повышения уровня авторизации, например otp_sms. **опционально**

Результатом выполнения запроса, будет валидация существующего токена и отображение номера телефона абонента на который будет отправлена SMS.

Формат успешного ответа на повышение уровня авторизации

После получения результата надо отобразить пользователю форму для отправки OTP кода.

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
```

```
{
  "view": {
```

```
{
  "msisdn": "79876543210",
  "serverUrl": "http://sso.rooxteam.com/sso/auth/otp-sms",
  "step": "send_otp_form",
  "execution": "dec7dacf-as2d-4345-aae5-d08410cb8f95_H4sIAAAAAAAAAAN1WTWwcRRYu27GdxE5wEnYF"
}
```

Параметры

- serverUrl - URL для следующего запроса, например sso.rooxteam.com, может содержать порт, например sso.rooxteam.com:8080
- execution - идентификатор предсессии аутентификации
- step - код состояния
- view.msisdn - номер телефона, на который будет отправлен OTP SMS

Формат запроса на повторную отправку OTP абоненту

Данный запрос отправит OTP код абоненту.

```
POST /sso/oauth2/access_token
Content-Type: application/x-www-form-urlencoded
Accept: application/json

client_id=<client_id>&
client_secret=<client_secret>&
scope=<scope>&
grant_type=urn:roox:params:oauth:grant-type:m2m&
realm=/customer&
auth_level=<auth_level>&
execution=dec7dacf-as2d-4345-aae5-d08410cb8f95_H4sIAAAAAAAAAAN1WTWwcRRYu27GdxE5wEnYF&
_eventId=send
```

Параметры

- client_id - идентификатор клиента, например selfcare, возможные значения зависят от конфигурации
- client_secret - пароль клиента, возможные значения зависят от конфигурации
- scope - список запрашиваемых scope через пробел в кодировке UTF-8. **опционально, регистрозависимо**
- grant_type - способ авторизации пользователя, всегда используется значение urn:roox:params:oauth:grant-type:m2m
- realm - группа пользователей WebSSO, всегда используется значение %2Fcustomer, которое является uri-encoded значением /customer
- auth_level - желаемый уровень авторизации для текущего пользователя.
- execution - идентификатор предсессии аутентификации
- _eventId - идентификатор действия, определяется отдельно для каждого состояния

Формат успешного ответа на отправку OTP абоненту

Полученный ответ содержит в себе описание параметров с ограничениями на них, которые WebSSO будет ждать в следующем запросе.

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
```

```
{
  "form": {
    "fields": {
      "otpCode": {
        "constraints": [
          {
            "name": "NotNull"
          }
        ]
      }
    }
  },
  "errors": [],
  "name": "otpForm"
},
"serverUrl": "http://sso.rooxteam.com/sso/auth/otp-sms",
"step": "enter_otp_form",
"execution": "dec7dacf-as2d-4345-aae5-d08410cb8f95_H4sIAAAAAAAAAAN1WTWwcRRYu27GdxE5wEnYF",
"view": {
  "otpCodeAvailableAttempts": 2,
  "msisdn": "79876543210",
  "nextOtpPeriod": 120,
  "blockedFor": 0,
  "isBlocked": false
}
}
```

Параметры

- serverUrl - URL для следующего запроса, например sso.rooxteam.com, может содержать порт, например sso.rooxteam.com:8080
- execution - идентификатор предсессии аутентификации
- step - код состояния
- form.fields[] - поля, ожидаемые для отправки.
- view.msisdn - номер телефона, на который будет отправлен OTP SMS
- view.otpCodeAvailableAttempts - кол-во попыток ввода OTP кода
- view.nextOtpPeriod - время в секундах до наступления возможности отправки нового OTP кода
- view.isBlocked - признак блокировки пользователя
- view.blockedFor - время до разблокировки в секундах

Формат запроса на валидацию OTP

Данный запрос валидирует введенный OTP код пользователем. И результатом выполнения данного запроса будет окончание повышения авторизации пользователя с выдачей нового access token с повышенным уровнем авторизации.

```
POST /sso/oauth2/access_token
Content-Type: application/x-www-form-urlencoded
Accept: application/json

client_id=<client_id>&
client_secret=<client_secret>&
scope=<scope>&
grant_type=urn:roox:params:oauth:grant-type:m2m&
realm=/customer&
auth_level=<auth_level>&
execution=dec7dacf-as2d-4345-aae5-d08410cb8f95_H4sIAAAAAAAAAAN1WTWwcRRYu27GdxE5wEnYF&
_eventId=validate&
otpCode=<otp_code>&
```

Параметры

- client_id - идентификатор клиента, например selfcare, возможные значения зависят от конфигурации
- client_secret - пароль клиента, возможные значения зависят от конфигурации
- scope - список запрашиваемых scope через пробел в кодировке UTF-8. **опционально, регистрозависимо**
- grant_type - способ авторизации пользователя, всегда используется значение urn:roox:params:oauth:grant-type:m2m
- realm - группа пользователей WebSSO, всегда используется значение %2Fcustomer, которое является uri-encoded значением /customer
- auth_level - желаемый уровень авторизации для текущего пользователя.
- execution - идентификатор предсессии аутентификации
- _eventId - идентификатор действия, определяется отдельно для каждого состояния
- otpCode - полученный OTP код

Формат успешного ответа на валидацию OTP

Полученный ответ содержит в себе новый access token с повышенным уровнем авторизации.

ЗАМЕТКА

Следует заметить, что старый access token продолжает действовать и для запросов, где не нужен повышенный уровень авторизации, нужно использовать его.

ЗАМЕТКА

Время действия нового access token меньше чем у старого

```
HTTP/1.1 200 OK
Content-Type: application/json
```

```
{
  "scope": [
    "cn"
  ],
  "expires_in": 59,
  "token_type": "Bearer",
  "access_token": "89fcd81f-e72b-4dbe-a2aa-71b57c519de2"
}
```

Параметры

- scope - список scope (в формате JSON Array) разрешенных для использования от имени пользователя, возможные значения задаются конфигурацией
- expires_in - время до истечения срока действия токена в секундах
- token_type - тип выданного токена. Всегда Bearer
- access_token - выданный access token

Формат ошибки валидации OTP

```
HTTP/1.1 200 OK
Content-Type: application/json
```

```
{
  "form": {
    "errors": [
      {
        "field": "otpCode",
        "message": "invalid_otp"
      }
    ],
    "name": "otpForm",
    "fields": {
      "otpCode": {
        "constraints": [
          {
            "name": "NotNull"
          }
        ]
      }
    }
  },
  "view": {
    "otpCodeAvailableAttempts": 2,
    "msisdn": "79876543210",
    "nextOtpPeriod": 120,
    "blockedFor": 0,
    "isBlocked": false
  },
  "serverUrl": "http://sso.hosted:8080/sso/auth/otp-sms",
  "step": "otp_form",
  "execution": "dec7dacf-as2d-4345-aae5-d08410cb8f95_H4sIAAAAAAAAAAN1WTWwcRRYu27GdxE5wEnYF"
}
```

Параметры

- execution - идентификатор предсессии аутентификации
- _eventId - идентификатор действия, определяется отдельно для каждого состояния
- view.msisdn - номер телефона, на который будет отправлен OTP SMS
- view.otpCodeAvailableAttempts - кол-во попыток ввода OTP кода
- view.nextOtpPeriod - время в секундах до наступления возможности отправки нового OTP кода
- view.isBlocked - признак блокировки пользователя
- view.blockedFor - время до разблокировки в секундах
- form.errors[].message - сообщение об ошибке валидации, **опционально**
- form.errors[].field - поле в котором найдена ошибка, **опционально**

Формат ответа при блокировке пользователя

```
HTTP/1.1 200 OK
Content-Type: application/json
```

```
{
  "form": {
    "errors": [
      {
        "message": "too_many_wrong_code"
      }
    ]
  },
  "view": {
    "blockedTo": "2015-02-18T12:00:00.000+00:00"
  },
  "serverUrl": "http://sso.rooxteam.com/sso/auth/otp-sms",
  "step": "otp_blocked_form",
  "execution": "dec7dacf-as2d-4345-aae5-d08410cb8f95_H4sIAAAAAAAAAAN1WTWwcRRYu27GdxE5wEnYF"
}
```

Параметры

- execution - идентификатор предсессии аутентификации
- _eventId - идентификатор действия, определяется отдельно для каждого состояния
- view.blockedTo - дата, время в формате UTC определяет срок блокировки.
- form.errors[].message - сообщение об ошибке валидации, **опционально**
- form.errors[].field - поле в котором найдена ошибка, **опционально**

Возможные коды ошибок

код	описание	комментарий
invalid_otp	Неверный OTP код	
error_sending_otp	Ошибка приотправке SMS с кодом	
too_many_sms	Превышен лимит отправки SMS	
too_many_wrong_code	Превышен лимит попыток ввода OTP кода	
may not be null	Поле не может быть пустым	Ошибка валидации
size must be between {min} and {max}	Значение должно быть в заданном диапазоне	Ошибка валидации
required on {field name}	Обязательное поле	Ошибка валидации

Получение токена под операцию

Получение токена под операцию происходит через OAuth 2.0 с передачей контекста операции, для которой требуется токен.

Если SMS-код был успешно подтвержден, будет возвращен `access_token` для выполнения запрошенной операции.

Проверка необходимости получения токена под операцию

Пример запроса

```
POST /sso/api/policyEvaluation/isAllowed HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer 0ff6618c-6dfd-4a59-840a-cea9170e7e76
Content-Length: 159
Host: <sso_host>
```

```
{
  "serviceName": "iPlanetAMWebAgentService",
  "actionName": "<action>",
  "resourceName": "<resource_name>",
  "envParams": <params>,
  "realm": "<realm>"
}
```

Параметры

- <sso_host> - базовый адрес сервера WebSSO, например sso.rooxteam.com, может содержать порт, например sso.rooxteam.com:8080
- <action> - действие, указанное в политиках SSO, которое требуется совершить (например, GET, POST)
- <resource_name> - имя ресурса, к которому осуществляется доступ. Должно совпадать с именем ресурса, указанным в соответствующей политике SSO.
- <params> - Произвольные параметры (в виде key-value пар из двух строк), которые требуются для вычисления политики. По умолчанию, пустой набор.
- <realm> - Контекстный realm.

Пример ответа

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 94
```

```
{
  "advice": {
    "PerOperationTokenConditionAdvice": "PerOperationTokenRequired"
  },
  "decision": "Deny"
}
```

Параметры

- <advices> - перечень необходимых условий, которые требуется удовлетворить для получения доступа к защищаемому ресурсу
- <decision> - решение о предоставлении доступа к ресурсу с текущим токеном

Получение одноразового токена под операцию

Пример запроса

```
POST /sso/oauth2/access_token HTTP/1.1
Accept: application/json
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Host: <sso_host>

realm=<realm>&
client_id=<client_id>&
client_secret=<client_secret>&
grant_type=urn%3Aroox%3Aparams%3Aoauth%3Agrant-type%3Am2m&
service=otp_operation_token&
access_token=0ff6618c-6dfd-4a59-840a-cea9170e7e76&
operation=%7B%22serviceName%22%3A%22iPlanetAMWebAgentService%22%2C%22actionName%22%3A%22GET%22%2C%22resourceName%22%3A%22%2Fotp-
settings%2F%3Aid%2Fotp%2Ftest%22%2C%22envParams%22%3A%7B%22principalId%22%3A%22%40me%22%7D%22%2C%22realm%22%3A%22%2Fcustomer%22%7D
```

Параметры

- <sso_host> - базовый адрес сервера WebSSO, например sso.rooxteam.com, может содержать порт, например sso.rooxteam.com:8080
- realm - группа пользователей WebSSO
- client_id - идентификатор клиента, например selfcare, возможные значения зависят от конфигурации
- client_secret - пароль клиента, возможные значения зависят от конфигурации
- grant_type - способ авторизации пользователя, всегда используется значение urn:roox:params:oauth:grant-type:m2m
- service - используемый сервис. Всегда otp_operation_token.
- access_token - текущий токен пользователя
- operation - контекст операции в json формате. Совпадает с телом запроса POST /sso/api/policyEvaluation/isAllowed

Пример ответа

Полученный ответ содержит в себе описание параметров с ограничениями на них, которые WebSSO будет ждать в следующем запросе.

```
HTTP/1.1 200 OK
Set-Cookie: execution=4e20650e-d4f5-405a-8144-fcceb939f2f5;Version=1;Path=/; HttpOnly
Content-Type: application/json; charset=UTF-8
```

```

{
  "form": {
    "errors": [],
    "name": "otpForm",
    "fields": {
      "otpCode": {
        "constraints": [
          {
            "name": "NotNull"
          },
          {
            "name": "Size",
            "attributes": {
              "min": 4,
              "max": 4
            }
          },
          {
            "name": "Pattern",
            "attributes": {
              "flags": [],
              "regexp": "^[0-9]+$"
            }
          }
        ]
      }
    }
  },
  "view": {
    "nextOtpCodePeriod": 29,
    "otpCodeAvailableAttempts": 4,
    "msisdn": "9876543210",
    "blockedFor": 0,
    "isBlocked": false,
    "nextOtpPeriod": 29,
    "expireOtpCodeTime": 59
  },
  "serverUrl": "http://sso.rooxteam.com/sso/auth/_otp-sms",
  "step": "enter_otp_form",
  "execution": "dec7dacf-as2d-4345-aae5-d08410cb8f95_H4sIAAAAAAAAAAN1WTWwcRRYu27GdxE5wEnYF"
}

```

Параметры

- form - структура, формы ввода отправленного пользователю в сообщении OTP кода
- form.errors - список ошибок. Всегда пустой.
- form.name - имя формы
- form.fields - список полей формы
- form.fields.otpCode - поле для ввода OTP кода
- form.fields.*.constraints - список ограничений на поле
- form.fields.*.constraints[].name - имя ограничения
- form.fields.*.constraints[].attributes - перечень атрибутов ограничений
- form.fields.*.constraints[].attributes.min - минимальное значение для данного атрибута
- form.fields.*.constraints[].attributes.max - максимальное значение для данного атрибута

- `form.fields.*.constraints[[]].attributes.regexp` - регулярное выражение валидации значения для поля
- `view.nextOtpPeriod` - время в секундах до наступления возможности отправки нового OTP кода
- `view.otpCodeAvailableAttempts` - кол-во попыток ввода OTP кода
- `view.msisdn` - номер телефона, на который будет отправлен OTP SMS
- `view.blockedFor` - время до разблокировки в секундах
- `view.isBlocked` - признак блокировки пользователя
- `view.nextOtpPeriod` - время в секундах до наступления возможности отправки нового OTP кода
- `view.expireOtpCodeTime` - время жизни OTP кода в секундах
- `serverUrl` - url для следующего запроса
- `step` - идентификатор шага
- `execution` - идентификатор предсессии аутентификации

Запрос валидации введенного кода

Данный запрос валидирует введенный OTP код пользователем. И результатом выполнения данного запроса будет выдача нового access token с правами на выполнение запрошенной операции.

Пример запроса

```
POST /sso/oauth2/access_token HTTP/1.1
Accept: application/json
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Host: <sso_host>

realm=<realm>&
client_id=<client_id>&
client_secret=<client_secret>&
grant_type=urn%3Aroox%3Aparams%3Aoauth%3Agrant-type%3Am2m&
service=otp_operation_token&
execution=<execution>&
_eventId=validate&
otpCode=<code>
```

Параметры

- `<sso_host>` - базовый адрес сервера WebSSO, например `sso.rooxteam.com`, может содержать порт, например `sso.rooxteam.com:8080`
- `<realm>` - группа пользователей WebSSO
- `<client_id>` - идентификатор клиента, например `selfcare`, возможные значения зависят от конфигурации
- `<client_secret>` - пароль клиента, возможные значения зависят от конфигурации
- `<grant_type>` - способ авторизации пользователя, всегда используется значение `urn:roox:params:oauth:grant-type:m2m`
- `<service>` - используемый сервис. Всегда `otp_operation_token`.
- `<execution>` - идентификатор предсессии аутентификации из ответа предыдущего запроса
- `<_eventId>` - действие в контексте данного шага. Всегда `validate`
- `<code>` - код из отправленного сообщения

Пример ответа

```
HTTP/1.1 200 OK
X-Node-Id: sso
X-Context-Id: R60DXELARPTIV1
Cache-Control: no-store
Date: Tue, 21 Nov 2017 14:13:50 GMT
Accept-Ranges: bytes
Server: Restlet-Framework/2.1.1
Vary: Accept-Encoding
Pragma: no-cache
Content-Type: application/json;charset=UTF-8
Content-Encoding: gzip
Connection: close
```

```
{
  "scope": [
    "cn",
    "impersonator"
  ],
  "expires_in": 59,
  "token_type": "Bearer",
  "refresh_token": "01a9d5f3-a797-4d04-be89-706f3692b182",
  "access_token": "aa784831-b585-44b7-ae96-cf0a6711d584"
}
```

Параметры

- scope - список scope (в формате JSON Array) разрешенных для использования от имени пользователя, возможные значения задаются конфигурацией
- expires_in - время до истечения срока действия токена в секундах
- token_type - тип выданного токена. Всегда Bearer
- refresh_token - выданный refresh token
- access_token - выданный access token

Формат ошибки валидации OTP

```
HTTP/1.1 200 OK
Content-Type: application/json
```

```
{
  "form": {
    "errors": [
      {
        "field": "otpCode",
        "message": "invalid_otp"
      }
    ],
    "name": "otpForm",
    "fields": {
      "otpCode": {
        "constraints": [
          {
            "name": "NotNull"
          }
        ]
      }
    }
  },
  "view": {
    "otpCodeAvailableAttempts": 2,
    "msisdn": "79876543210",
    "nextOtpPeriod": 120,
    "blockedFor": 0,
    "isBlocked": false
  },
  "serverUrl": "http://sso.rooxteam.com/sso/auth/otp-sms",
  "step": "otp_form",
  "execution": "dec7dacf-as2d-4345-aae5-d08410cb8f95_H4sIAAAAAAAAAAN1WTWwcRRYu27GdxE5wEnYF"
}
```

Параметры

- execution - идентификатор предсессии аутентификации
- _eventId - идентификатор действия, определяется отдельно для каждого состояния
- view.msisdn - номер телефона, на который будет отправлен OTP SMS
- view.otpCodeAvailableAttempts - кол-во попыток ввода OTP кода
- view.nextOtpPeriod - время в секундах до наступления возможности отправки нового OTP кода
- view.isBlocked - признак блокировки пользователя
- view.blockedFor - время до разблокировки в секундах
- form.errors[].message - сообщение об ошибке валидации, **опционально**
- form.errors[].field - поле в котором найдена ошибка, **опционально**

Формат ответа при блокировке пользователя

```
HTTP/1.1 200 OK
Content-Type: application/json
```

```
{
  "form": {
    "errors": [
      {
        "message": "too_many_wrong_code"
      }
    ]
  },
  "view": {
    "blockedTo": "2015-02-18T12:00:00.000+00:00"
  },
  "serverUrl": "http://sso.rooxteam.com/sso/auth/otp-sms",
  "step": "otp_blocked_form",
  "execution": "dec7dacf-as2d-4345-aae5-d08410cb8f95_H4sIAAAAAAAAAAN1WTWwcRRYu27GdxE5wEnYF"
}
```

Параметры

- execution - идентификатор предсессии аутентификации
- _eventId - идентификатор действия, определяется отдельно для каждого состояния
- view.blockedTo - дата, время в формате UTC определяет срок блокировки.
- form.errors[].message - сообщение об ошибке валидации, **опционально**
- form.errors[].field - поле в котором найдена ошибка, **опционально**

Возможные коды ошибок

код	описание	комментарий
invalid_otp	Неверный OTP код	
error_sending_otp	Ошибка приотправке SMS с кодом	
too_many_sms	Превышен лимит отправки SMS	
too_many_wrong_code	Превышен лимит попыток ввода OTP кода	
may not be null	Поле не может быть пустым	Ошибка валидации
size must be between {min} and {max}	Значение должно быть в заданном диапазоне	Ошибка валидации
required on {field name}	Обязательное поле	Ошибка валидации

Смена данных аутентификации (логин и пароль) пользователя

Авторизованный пользователь может сменить логин или пароль используемые для аутентификации. Смена логина может быть заблокирована через конфигурацию, если это не допускается сценариями интеграции.

Формат запроса

```
POST /sso/oauth2/access_token
Host: <sso_host>
Content-Type: application/x-www-form-urlencoded
Accept: application/json

realm=<realm>&
client_id=<client_id>&
client_secret=<client_secret>&
grant_type=urn%3Aroox%3Aparams%3Aoauth%3Agrant-type%3Am2m&
service=change-credentials&
access_token=<access_token>
```

Параметры запроса

- <sso_host> - базовый адрес сервера UIDM, например sso.rooxteam.com, может содержать порт, например sso.rooxteam.com:8080
- access_token - полученный в результате аутентификации access token, например 7bdaeacc-3d80-415c-920f-a7c30ca5e743
- service - используемый сервис; для данного сценария всегда change-credentials

Формат ответа - форма смены логина-пароля пользователя

Данный ответ будет также возвращен из флоу аутентификации если для пары логин-пароль установлен флаг требования смены пароля: passwordMustChange.

```
HTTP/1.1 200 OK
```

```
{
  "form": {
    "errors": [],
    "name": "credentialsForm",
    "fields": {
      "password": {
        "constraints": [
          {
            "name": "Size",
            "attributes": {
              "min": 4,
              "max": 1024
            }
          }
        ]
      }
    }
  }
}
```

```

    ],
    },
    "newPasswordBody": {
      "constraints": [
        {
          "name": "Size",
          "attributes": {
            "min": 4,
            "max": 1024
          }
        }
      ]
    },
    "newUsername": {
      "constraints": [
        {
          "name": "Size",
          "attributes": {
            "min": 4,
            "max": 1024
          }
        }
      ]
    },
    "view": {
      "username": "<username>"
    },
    "serverUrl": "https://sso.rooxteam.com/sso/auth/_common-change-credentials",
    "step": "enter_credentials",
    "execution": "dec7dacf-as2d-4345-aae5-d08410cb8f95_H4sIAAAAAAAAAAN1WTWwcRRYu27GdxE5wEnYF"
  }
}

```

Поля ответа

- form - форма получения параметров смены логина/пароля пользователя
- errors - список ошибок, сообщение может быть привязано к полю (field)
- form.name - имя формы
- fields - список полей формы
- password - атрибуты, относящиеся к полю ввода текущего пароля
- newPasswordBody - атрибуты, относящиеся к полю ввода нового пароля
- newUsername - атрибуты, относящиеся к полю ввода нового имени пользователя
- constraints - список ограничений на поле
- constraints.name - имя ограничения
- constraints.attributes - перечень атрибутов ограничений
- constraints.attributes.min - минимальное значение для данного атрибута
- constraints.attributes.max - максимальное значение для данного атрибута
- constraints.attributes.skip - регулярное выражение валидации значения для поля
- view - дополнительные данные текущего шага
- username - текущий логин пользователя
- serverUrl - URL для запросов на сервер

- step - идентификатор шага
- execution - идентификатор предсессии аутентификации

Запрос на смену пары пароль/логин пользователя

Данный запрос меняет пару/логин пароля для пользователя. Успешным результатом выполнения данного запроса будет выдача access token.

```
POST /sso/oauth2/access_token
Host: <sso_host>
Accept: application/json
Cache-Control: no-cache
Content-Type: application/x-www-form-urlencoded

client_id=<client_id>&
client_secret=<client_secret>&
scope=<scope>&
grant_type=urn:roox:params:oauth:grant-type:m2m&
realm=%2Fcustomer&
service=dispatcher&
response_type=token&
execution=<execution>&
password=<password>&
newPasswordBody=<newPasswordBody>&
newUsername=<newUsername>&
_eventId=next
```

Параметры запроса

- execution - идентификатор предсессии аутентификации
- _eventId - идентификатор действия, определяется отдельно для каждого состояния
- password - текущий пароль пользователя
- newPasswordBody - новый пароль пользователя
- newUsername - новый логин пользователя

В зависимости от конфигурации UIDM поля password и newUsername могут не использоваться.

Формат успешного ответа с выдачей access token

```
HTTP/1.1 200 OK
```

```
{
  "access_token": "fe8a0976-972c-4773-9ba8-662c92869639",
  "refresh_token": "863a4983-2424-4cb3-9ed2-18f9357b64c0",
  "refresh_expires_in": 1599,
  "scope": [
    "cn"
  ],
  "token_type": "Bearer",
  "expires_in": 599,
  "old_token": "fe8a0976-972c-4773-9ba8-662c92869639",
```

```
{
  "JWTToken": "eyJhZGQlOi0lsieW90YV9sa19tMm
0iXSwed3ViIjoiYmlzX19fX18yRpg3NjU0MzIxIiwiABUoIjoxNjAxOTgzNTc4LCJhenAiOiJ5b3RhT5trX20ybSIzImFtciI6ImRldmljZV9zdG9yZWZldG9rZW4iLCJ0b2t1bk5hbWUiOiFrCF90b2t1biIsImZcyI6InlvdSWwbGtfbTJtIiwicmVhbG0iOi0uUyV3VzdG9tZXIiLCJ0b2t1b1R5cGUiOiJCavJlZGRlZEpxVFRva2VuIiwiaXhwIjoxNjAxOTgzMTc4LCJpbn0iOi0jE2MDE5ODM1Nzh9._kCRnEw0qI9K1JVCVbwofCqsZzzEm7PJBqfJCAG19bG"
}
```

Параметры запроса

- score - список score (в формате JSON Array) разрешенных для использования от имени пользователя, возможные значения задаются конфигурацией
- token_type - тип выданного токена. Всегда Bearer
- refresh_token - выданный refresh token
- refresh_expires_in - время до истечения срока действия refresh токена в секундах
- access_token - выданный access token
- expires_in - время до истечения срока действия токена в секундах
- old_token - токен, выданный внешней системой, строка
- JWTToken - JWT токен для длительного хранения и последующей аутентификации

Формат успешного ответа с повторной аутентификацией

Если смена логина-пароля была вызвана из флоу аутентификации, то существует возможность запроса повторной аутентификации с измененными значениями логина-пароля. Данное поведение определяется параметром конфигурации com.rooxteam.sso.credentials.change.need_relogin_on_success

В этом случае будет следующий формат ответа:

```
HTTP/1.1 200 OK
```

```
{
  "form": {
    "errors": [
      {
        "message": "need_relogin"
      }
    ],
    "name": "loginForm",
    "fields": {
      "username": {
        "constraints": [
          {
            "name": "NotNull"
          },
          {
            "name": "Size",
            "attributes": {
              "min": 10,
              "max": 25
            }
          }
        ]
      }
    }
  }
}
```

```

        "name": "FilteredSize",
        "attributes": {
          "skip": "([^\d]*)|([^\d])",
          "min": 10,
          "max": 10
        }
      },
    ],
  },
  "password": {
    "constraints": [
      {
        "name": "Size",
        "attributes": {
          "min": 4,
          "max": 1024
        }
      },
      {
        "name": "NotNull"
      }
    ]
  },
  "view": {
    "isBlocked": false,
    "blockedFor": null
  },
  "serverUrl": "https://sso.rooxteam.com/sso/auth/login-widget-router",
  "step": "auth_form",
  "execution": "dec7dacf-as2d-4345-aae5-d08410cb8f95_H4sIAAAAAAAAAAN1WTWwcRRYu27GdxE5wEnYF"
}

```

Параметры

- form - форма получения параметров аутентификации пользователя
- errors - список ошибок, сообщение может быть привязано к полю (field)
- message - сообщение об ошибке
- form.name - имя формы
- fields - список полей формы
- username - атрибуты, относящиеся к имени пользователя
- password - атрибуты, относящиеся к паролю
- constraints - список ограничений на поле
- constraints.name - имя ограничения
- constraints.attributes - перечень атрибутов ограничений
- constraints.attributes.min - минимальное значение для данного атрибута
- constraints.attributes.max - максимальное значение для данного атрибута
- constraints.attributes.skip - регулярное выражение валидации значения для поля
- view - дополнительные данные текущего шага
- isBlocked - признак блокировки пользователя
- blockedFor - время до разблокировки в секундах

- serverUrl - URL для запросов на сервер
- step - идентификатор шага
- execution - идентификатор предсессии аутентификации

Формат ответа при ошибке совпадения нового логина с существующим в системе

```
HTTP/1.1 200 OK
Content-Type: application/json
```

```
{
  "view": {
    "blockedFor": 0,
    "attempts": 5
  },
  "form": {
    "errors": [
      {
        "message": "login_already_exists"
      }
    ]
  }
}
```

Параметры

- view.blockedFor - количество секунд, в течение которых операция смены логина заблокирована; в данном случае можно игнорировать
- view.attempts - количество попыток выполнения операции до блокировки

Формат ответа при блокировке операции смены логина

```
HTTP/1.1 200 OK
Content-Type: application/json
```

```
{
  "view": {
    "blockedFor": 3590,
    "attempts": 0
  },
  "form": {
    "errors": [
      {
        "message": "too_many_attempts"
      }
    ]
  }
}
```

Параметры

- view.blockedFor - количество секунд, в течение которых операция смены логина заблокирована
- view.attempts - количество попыток выполнения операции до блокировки
- form.errors - содержит ошибку блокировки операции смены логина: `too_many_attempts`

Конфигурация UIDM для запроса смены пары логин-пароль

Ключ	Описание	По умолчанию
com.rooxteam.sso.webflow.credentialsForm.password.pattern	Regex шаблон для валидации ввода текущего пароля	не задан
com.rooxteam.sso.webflow.credentialsForm.password.min_length	Минимальная длина значения текущего пароля	не задан
com.rooxteam.sso.webflow.credentialsForm.password.max_length	Максимальная длина значения текущего пароля	не задан
com.rooxteam.sso.webflow.credentialsForm.newPasswordBody.pattern	Regex шаблон для валидации ввода нового пароля	не задан
com.rooxteam.sso.webflow.credentialsForm.newPasswordBody.min_length	Минимальная длина значения нового пароля	не задан
com.rooxteam.sso.webflow.credentialsForm.newPasswordBody.max_length	Максимальная длина значения нового пароля	не задан
com.rooxteam.sso.webflow.credentialsForm.newUsername.pattern	Regex шаблон для валидации ввода нового имени пользователя	не задан
com.rooxteam.sso.webflow.credentialsForm.newUsername.min_length	Минимальная длина значения нового имени пользователя	не задан
com.rooxteam.sso.webflow.credentialsForm.newUsername.max_length	Максимальная длина значения нового имени пользователя	не задан
com.rooxteam.sso.credentials.change.check_used_password	Включение проверки историчности пароля. Проверяет использовался ли устанавливаемый пароль пользователем раньше	false - не проверяется
com.rooxteam.sso.credentials.change.check_same_password.depth	Глубина историчности пароля. Количество	10

последних паролей,
которые не должны
совпадать с
устанавливаемым новым
паролем

com.rooxteam.sso.credentials.change.need_relogin_on_success	Требовать повторной аутентификации после смены логина-пароля пользователю (только для флоу аутентификации)	false
com.rooxteam.uidm.login.change.limit	Количество разрешённых попыток смен логина перед блокировкой этой операции для текущего пользователя. Распространяется как на успешные смены логина, так и на операции, завершившиеся ошибкой.	2
com.rooxteam.uidm.login.change.block.seconds	Количество секунд, в течение которых операция смены логина для текущего пользователя будет заблокирована	86400 секунд (24 часа)

Выход через API

Выданный ранее токен может быть инвалидирован отправкой запроса на соответствующий адрес UIDM. Дополнительная аутентификация для этого метода не требуется, достаточно передать access token.

Для инвалидации выданного токена клиентское приложение защищаемого сервиса должно сделать аякх запрос на сервер защищаемого ресурса, после чего серверное приложение защищаемого ресурса должно выполнить запрос на отзыв токена в UIDM, передав access_token пользователя.

Формат запроса

```
POST /sso/oauth2/revoke HTTP/1.1
Host: <sso_host>
Content-Type: application/x-www-form-urlencoded
Accept: application/json

token=<token>&token_type_hint=<token_type_hint>
```

Параметры

- < sso_host > - базовый адрес сервера UIDM, например sso.rooxteam.com, может содержать порт, например sso.rooxteam.com:8080
- < token > - токен для инвалидации, например 5fdfeafd-3061-4b1c-9076-0fe460f91fc8

- <token_type_hint> - тип токена, всегда access_token

Формат успешного ответа

```
HTTP/1.1 200 OK
```

Формат неуспешного ответа

Пример ответа при передаче невалидного типа токена:

```
HTTP/1.1 400 Bad Request
```

```
{  
  "error_description": "Requested token type is not supported.",  
  "error": "unsupported_token_type"  
}
```

Параметры

- error - код ошибки согласно спецификации OAuth 2.0 Token Revocation (<http://tools.ietf.org/html/rfc7009#section-4.1.1>[RFC 7009 пункт 4.1.1])
- error_description - текстовое описание ошибки

Вызов callback URL при инвалидации токенов

UIDM поддерживает отправку запроса на URL защищаемого сервиса с оповещением о факте инвалидации токена пользователя или инвалидации всех токенов защищаемого сервиса.

Защищаемый сервис может подписаться на события инвалидации токенов, выданных этому сервису ранее. Для этого необходимо сообщить один или несколько callback URL, на которые UIDM будет отправлять оповещения. Список URL для оповещения настраивается администратором UIDM.

Инвалидация токена пользователя

При инвалидации токена конкретного пользователя UIDM отправляет соответствующее оповещение на callback URL. Защищаемый сервис, получив такое оповещение, должен немедленно инвалидировать сессию пользователя. Если защищаемый сервис использует кэширование результатов авторизации, кеш результатов авторизации данного пользователя также должен быть инвалидирован.

Формат оповещения о инвалидации токена

```
POST <callback_url>  
Cache-Control: no-cache  
Content-Type: application/x-www-form-urlencoded
```

```
event=token_revoked&  
global=false&  
cn=<cn>&  
access_token=<access_token>
```

Параметры

- callback_url - URL для оповещения из списка, заданного администратором UIDM
- event - имя события
- global - флаг полной блокировки сервиса, в этом оповещении всегда false
- cn - номер телефона пользователя (msisdn)
- access_token - инвалидированный access token

Блокирование сервиса целиком

При блокировании сервиса целиком UIDM отправляет соответствующее оповещение на callback URL. Защищаемый сервис, получив такое оповещение, должен немедленно инвалидировать сессии всех пользователей. Если защищаемый сервис использует кеширование результатов авторизации, кеш результатов авторизации всех пользователей также должен быть инвалидирован.

Формат оповещения о блокировке сервиса

```
POST <callback_url>  
Cache-Control: no-cache  
Content-Type: application/x-www-form-urlencoded  
  
event=service_blocked&  
global=true
```

Параметры

- callback_url - URL для оповещения из списка, заданного администратором UIDM
- event - имя события
- global - флаг полной блокировки сервиса, в этом оповещении всегда true

Понижение уровня авторизации токена

При понижении уровня авторизации токена конкретного пользователя UIDM отправляет соответствующее оповещение на callback URL. Если защищаемый сервис использует кеширование результатов авторизации, кеш результатов авторизации всех пользователей также должен быть инвалидирован.

Формат оповещения о понижении уровня авторизации токена

```
POST <callback_url>  
Cache-Control: no-cache  
Content-Type: application/x-www-form-urlencoded
```

```
event=token_auth_level_decreased&
global=false&
cn=<cn>&
access_token=<access_token>
```

Параметры

- callback_url - URL для оповещения из списка, заданного администратором UIDM
- event - имя события
- global - флаг полной блокировки сервиса, в этом оповещении всегда false
- cn - номер телефона пользователя (msisdn)
- access_token - измененный access token

Использование параметра scope

Параметр scope (<https://tools.ietf.org/html/rfc6749#section-3.3> [RFC 6749 пункт 3.3]) может применяться в UIDM для двух разных случаев:

- определение набора данных пользователя, к которым сервис получает доступ. По умолчанию любой запрос авторизации содержит scope cn, предоставляющий информацию о номере телефона. Для получения дополнительных атрибутов необходимо передать соответствующий список scope через пробел при аутентификации, например scope=networkAuthenticationType displayName. В этом случае при валидации токена будет возвращен расширенный список атрибутов. Возможные для запрашивания атрибуты конфигурируются администратором UIDM отдельно для каждого сервиса.

Атрибуты могут включать:

название	описание
cn	Номер телефона пользователя (msisdn)
telephoneNumber	Номер телефона пользователя (msisdn), alias на cn
networkAuthenticationType	Тип аутентификации по сетевому устройству. AUTO - разрешен автоматический вход, NONE - запрещен
displayName	ФИО пользователя
contactEmail	Контактный email пользователя, указанный в контракте

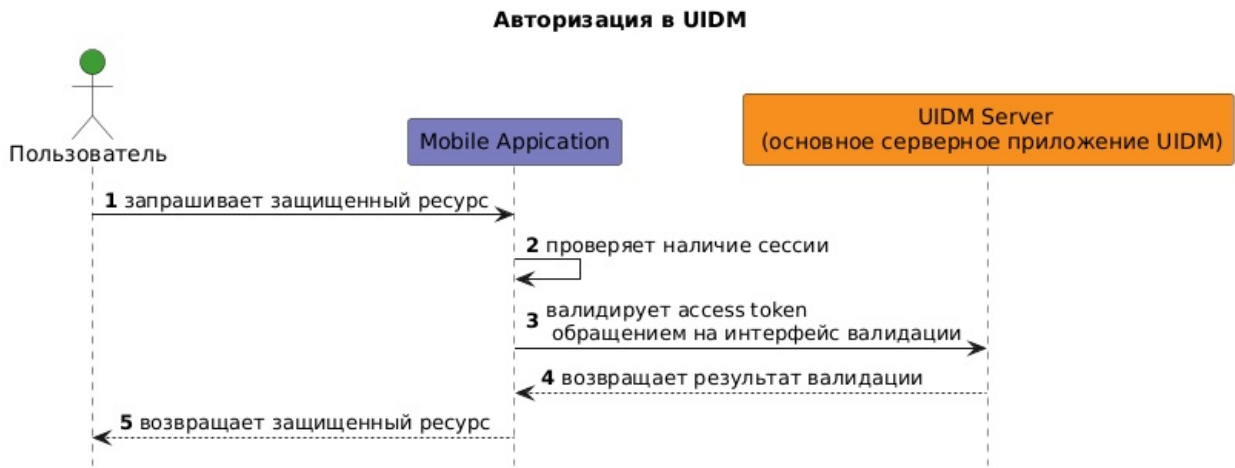
- определение списка идентификаторов ресурсов защищаемых сервисов, которые могут быть использованы сервисом для данного пользователя. Часть методов может быть включена в список по умолчанию, остальные должны быть запрошены при аутентификации. Идентификатор метода может содержать ограничение на минимальный уровень авторизации пользователя. В этом случае соответствующий scope будет выдан только при наличии требуемого уровня авторизации.

Перечень доступных сервису scope и ограничения на минимальный уровень авторизации настраиваются администратором UIDM.

scope - мнемоническое название ресурса защищаемого сервиса.

Диаграммы переходов для работы с UIDM

Авторизация в UIDM при существующей сессии

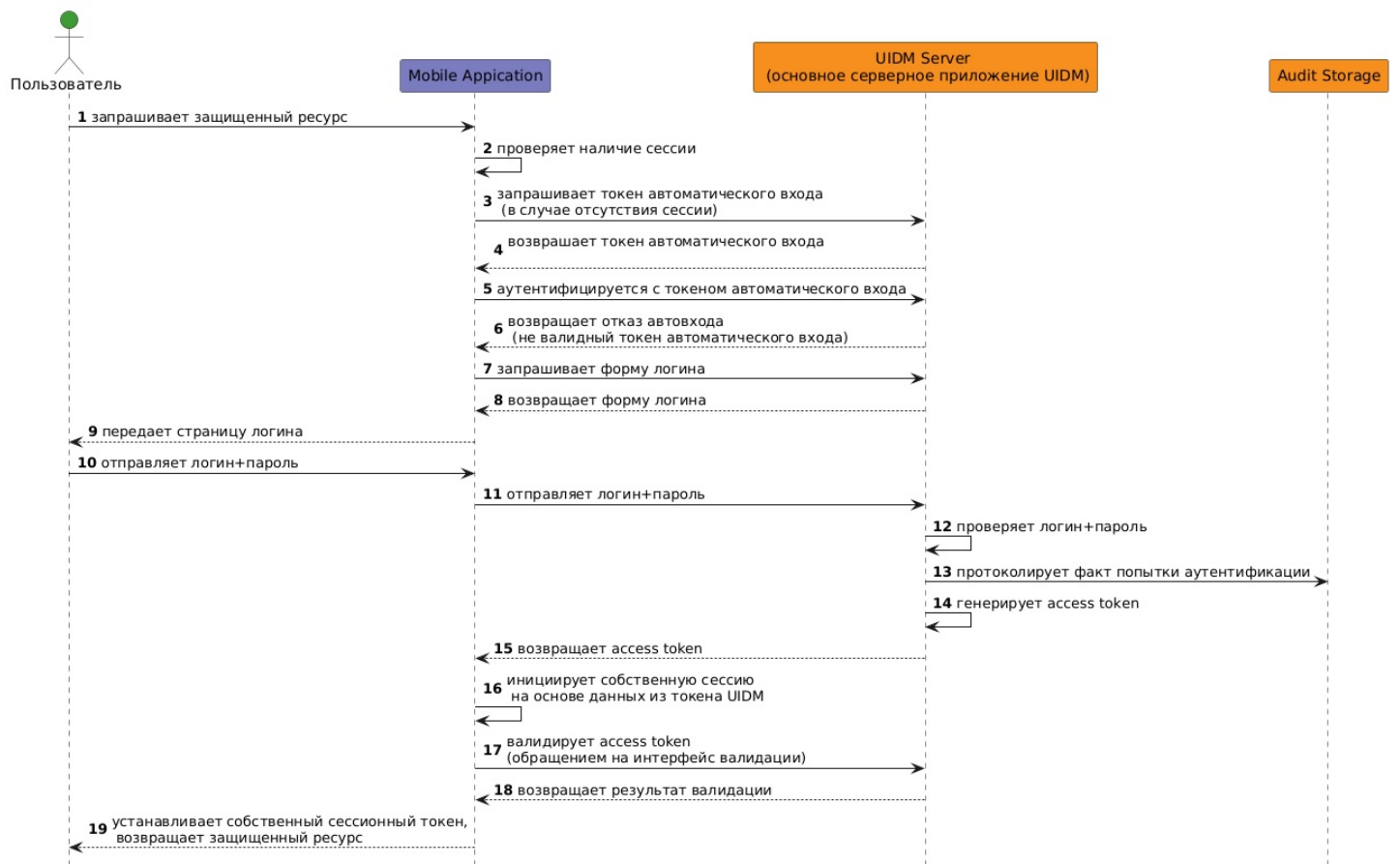


Аутентификация в UIDM, отсутствие сессии, успешный автовход



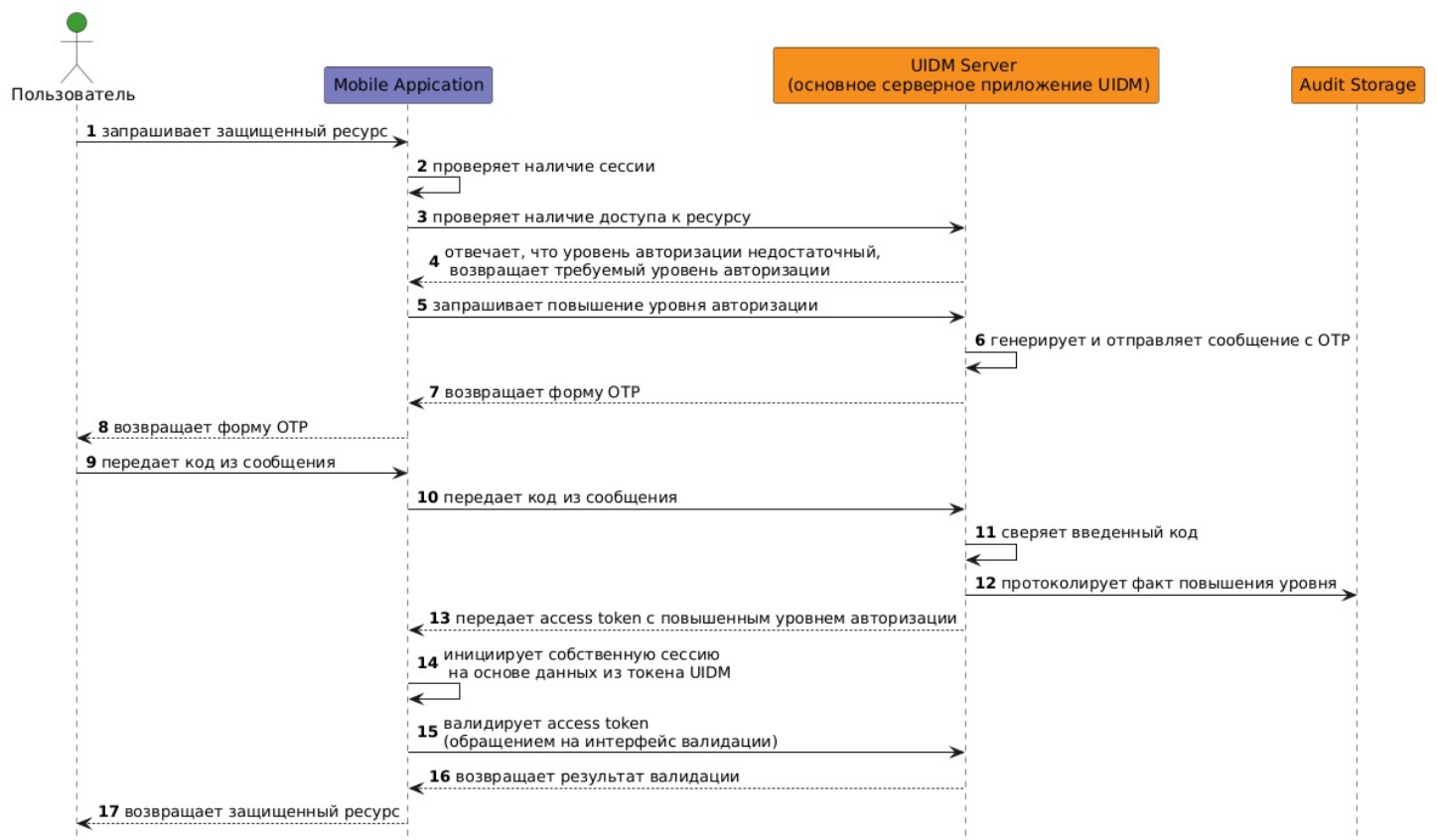
Аутентификация и авторизация в UIDM

Аутентификация и авторизация в UIDM



Повышение уровня авторизации

Повышение уровня авторизации



Отзыв токена

Отзыв токена

