M2M API аутентификации через социальные сети и создания связок с ними

Оглавление

Создание привязки с социальной сетью в UIDM

После успешной аутентификации в соцсети пользователь может связать свой аккаунт в UIDM с аккаунтом из соцсети и в дальнейшем использовать соцсеть для аутентификации в UIDM. Все запросы должны быть выполнены в приведенной последовательности, так как параметр <execution> из каждого ответа используется как параметр в последующих запросах.

ВАЖНО

Возвращаемое значение <execution> в каждом из запросов к UIDM может обновляться. Необходимо использовать самое актуальное значение.

Список доступных для интеграции соцсетей.

Интеграция UIDM с социальными сетями.

Начало сценария создания привязки

Для начала сценария создания привязки необходимо отправить запрос в UIDM на /sso/oauth2/access_token. В ответе будет содержаться параметр <execution>, который необходимо включить в следующий запрос к UIDM.

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>&
realm=<realm>&
grant_type=urn%3Aroox%3Aparams%3Aoauth%3Agrant-type%3Am2m&
service=dispatcher&
response_type=token+cookie
```

Параметры

- ‹sso_host› - базовый адрес сервера UIDM, например sso.rooxteam.com
- ‹client_id› - идентификатор клиента, например selfcare
- ‹client_secret› - пароль клиента, например selfcare_password
- ‹realm› - url-encoded пользовательский realm, например %2Fcustomer
- ‹grant_type› - способ аутентификации пользователя, всегда используется значение urn:roox:params:oauth:grant-type:m2m
- ‹service› - используемый сервис. Всегда dispatcher.
- ‹response_type› - способ аутентификации пользователя
  - например, "token cookie" (в данных запроса URL-encoded представление: "token+cookie") - при таком значении параметра при успешной аутентификации токены возвращаются не только в теле, но и в Cookie
  - подробнее здесь: RFC 6749

Формат успешного ответа

В данном примере в ответе содержится JSON обьект, содержащий идентификаторы приложений соцсетей и параметр ‹execution› (в заголовке и в теле ответа), который необходимо включить в следующий запрос к UIDM.

```
HTTP/1.1 200 OK

Content-Type: application/json;charset=UTF-8
Set-Cookie: execution=<execution_value>;Version=0;Path=/;Secure; SameSite=Lax; HttpOnly
```

```
{
  "execution": "<execution_value>",
  "view": {
    "autologin": "skipped",
    "ssoUrl": "https://<sso_base_url>/sso",
    "isBlocked": false,
    "esiaRequestUri": "<esia_request_uri>",
    "esiaRedirectUri": "/esia_callback.jsp",
    "vkontakteAppId": "<vkontakte_app_id>",
    "yandexRedirectUri": "/yandex_callback.jsp",
    "googleRedirectUri": "/google_callback.jsp",
```

```
    "vkontakteRedirectUri": "/vk_callback.jsp",
    "vkontakteRequestScopesAsArray": [],
    "odnoklassnikiRedirectUri": "/ok_callback.jsp",
    "mailRuRedirectUri": "/mailru_callback.jsp"
  },
  "form": {
    "name": "loginForm",
    "fields": {},
    "errors": []
  },
  "serverUrl": "https://<sso_base_url>/sso/auth/login-widget-router",
  "step": "auth_form"
}
```

Параметры

- ‹execution_value› - значение параметра ‹execution› для следующего запроса к UIDM
- ‹vkontakte_app_id› - идентификатор приложения соцсети Vkontakte, например, 1234567

Аутентификации в социальной сети

Для привязки аккаунта соцсети к учетной записи UIDM необходимо сначала пройти аутентификацию в соцсети. После успешной аутентификации соцсеть отправляет некоторые данные ‹social_data›, которые в дальнейшем используются UIDM для аутентификации пользователя и создания связки, если она не была создана ранее.

Перед использованием ‹social_data› необходимо зашифровать алгоритмом, описанным в разделе ниже.

Детали формата запроса определяются соцсетью и описаны в документации подключаемой соцсети.

Примеры настройки приложений соцсетей: Интеграция UIDM с социальными сетями

Аутентификация в UIDM с помощью соцсети

Передача ранее полученных параметров соцсети ‹social_data›.

- Если у данного аккаунта соцсети нет привязки к аккаунту UIDM, то будет возвращена форма для ввода учетных данных UIDM (текущий ответ);
- если привязка существует, то в ответе будет результат аутентификации.

После данного запроса возможна отмена действия.

Формат запроса

```
POST /sso/oauth2/access_token

Host: <sso_host>
Accept: application/json
Content-Type: application/x-www-form-urlencoded
Cookie: execution=<execution_value>

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

```
_eventId=<_eventId>&
execution=<execution_value>&
response_type=token+cookie&
socialData=<social_data>
```

Параметры

- <sso_host> - базовый адрес сервера UIDM, например sso.rooxteam.com
- <client_id> - идентификатор клиента, например selfcare
- <client_secret> - пароль клиента, например selfcare_password
- <realm> - url-encoded пользовательский realm, например %2Fcustomer
- <grant_type> - способ аутентификации пользователя, всегда используется значение urn:roox:params:oauth:grant-type:m2m
- <service> - используемый сервис, например, vkontakte; список доступных идентификаторов соцсетей.
- <_eventId> - идентификатор действия, например, vkontakte; список доступных идентификаторов соцсетей.
- <execution> - идентификатор предсессии аутентификации
- <response_type> - способ аутентификации пользователя, всегда используется значение "token cookie"
- <socialData> - данные аутентификации в соцсети, преобразованные по алгоритму

Формат успешного ответа

```
HTTP/1.1 200 OK

set-cookie: execution=<execution_value>;Version=0;Path=/;Secure; SameSite=Lax; HttpOnly
```

```
{
  "execution": "<execution_value>",
  "view": {
    "avatarUrl": "<sn_profile_avatar_url>",
    "ssoUrl": "https://<<sso_base_url>>/sso",
    "isBlocked": false,
    "fullName": "<sn_profile_fullName>",
    "esiaRequestUri": "<esia_request_uri>",
    "esiaRedirectUri": "/esia_callback.jsp",
    "socialNetworkId": "<social_network_id>",
    "firstName": "<sn_profile_firstName>",
    "vkontakteAppId": "<vkontakte_app_id>",
    "yandexRedirectUri": "/yandex_callback.jsp",
    "googleRedirectUri": "/google_callback.jsp",
    "vkontakteRedirectUri": "/vk_callback.jsp",
    "vkontakteRequestScopesAsArray": [],
    "odnoklassnikiRedirectUri": "/ok_callback.jsp",
    "mailRuRedirectUri": "/mailru_callback.jsp"
  },
  "form": {
    "name": "loginForm",
    "fields": {},
    "errors": []
  },
```

```
  "serverUrl": "https://<sso_base_url>/sso/auth/login-widget-router",
  "step": "auth_form"
}
```

Параметры

- <socialNetworkId> - идентификатор соцсети, для которой прозведен вызов, например vkontakte; список доступных идентификаторов
- <avatarUrl> - данные учетной записи соцсети
- <fullName> - данные учетной записи соцсети
- <firstName> - данные учетной записи соцсети
- <form> - описание формы для аутентификации в UIDM

Формат ответа с ошибкой

Если при запросе обязательный параметр <socialData>:

- не предоставлен
- пустой
- содержит неверные данные, -

сервер возвращает ошибку:

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

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

Создание или обновление привязки аккаунта соцсети к учетной записи UIDM

Отправка логина и пароля от учетной записи UIDM для проверки наличия пользователя и подготовки создания привязки с аккаунтом соцсети.

- Если у учетной записи UIDM не было ранее привязок к выбранной соцсети
  - то в ответе будет параметр <step> = show_attach_form
- если существовала привязка, и она отлична от выбранной на предыдущих шагах
  - параметр <step> = show_reattach_form
  - на данном шаге привязка не удаляется и не обновляется
  - продолжение сценария удалит старую привязку и создаст новую с выбранной соцсетью
  - отмена действия остановит пересоздание привязки

После данного запроса возможна отмена действия.

Формат запроса

```
POST /sso/oauth2/access_token

Host: <sso_host>
Accept: application/json
Content-Type: application/x-www-form-urlencoded
Cookie: execution=<execution_value>

client_id=<client_id>&
client_secret=<client_secret>&
realm=%2Fcustomer&
grant_type=urn%3Aroox%3Aparams%3Aoauth%3Agrant-type%3Am2m&
service=dispatcher&
_eventId=next&
username=<username>&
password=<user_password>&
execution=<execution_value>&
response_type=token+cookie
```

Параметры

- <sso_host> - базовый адрес сервера UIDM, например sso.rooxteam.com
- <client_id> - идентификатор клиента, например selfcare
- <client_secret> - пароль клиента, например selfcare_password
- <realm> - url-encoded пользовательский realm, например %2Fcustomer
- <grant_type> - способ аутентификации пользователя, всегда используется значение urn:roox:params:oauth:grant-type:m2m
- <service> - используемый сервис, для этого запроса всегда dispatcher
- <_eventId> - идентификатор действия, для этого запроса всегда next
- <username> - логин учетной записи в UIDM
- <password> - пароль учетной записи в UIDM
- <execution> - идентификатор предсессии аутентификации
- +<response_type> - способ аутентификации пользователя, всегда используется значение "token cookie"

Формат успешного ответа

```
HTTP/1.1 200 OK

set-cookie: execution=<execution_value>;Version=0;Path=/;Secure; SameSite=Lax; HttpOnly
```

```
{
  "execution": "<execution_value>",
  "view": {
    "socialNetworkId": "<social_network_id>",
    "firstName": "<sn_profile_firstName>",
```

```
    "avatarUrl": "<sn_profile_avatar_url>",
    "fullName": "<sn_profile_fullName>",
    "step": "attach_form"
  },
  "form": {
    "name": "attachForm",
    "fields": {},
    "errors": []
  },
  "serverUrl": "https://<sso_base_url>/sso/auth/social-attach",
  "step": "<step>"
}
```


Параметры

- ‹socialNetworkId› - идентификатор соцсети, для которой прозведен вызов, например vkontakte; список доступных идентификаторов
- ‹avatarUrl› - данные учетной записи соцсети
- ‹fullName› - данные учетной записи соцсети
- ‹firstName› - данные учетной записи соцсети
- ‹step› - show_attach_form - при создании привязки, show_reattach_form - при необходимости обновления привязки

Формат ответа с ошибкой

```
HTTP/1.1 200 OK
set-cookie: execution=<execution_value>;Version=0;Path=/;Secure; SameSite=Lax; HttpOnly
```


```
{
  "execution": "<execution_value>",
  "view": {
    "avatarUrl": "<sn_profile_avatar_url>",
    "ssoUrl": "https://<<sso_base_url>>/sso",
    "isBlocked": false,
    "fullName": "<sn_profile_fullName>",
    "esiaRequestUri": "<esia_request_uri>",
    "esiaRedirectUri": "/esia_callback.jsp",
    "socialNetworkId": "<social_network_id>",
    "firstName": "<sn_profile_firstName>",
    "vkontakteAppId": "<vkontakte_app_id>",
    "yandexRedirectUri": "/yandex_callback.jsp",
    "googleRedirectUri": "/google_callback.jsp",
    "vkontakteRedirectUri": "/vk_callback.jsp",
    "vkontakteRequestScopesAsArray": [],
    "odnoklassnikiRedirectUri": "/ok_callback.jsp",
    "mailRuRedirectUri": "/mailru_callback.jsp"
  },
  "form": {
    "name": "loginForm",
    "fields": {},
    "errors": [
```

```
      {
        "message": "<error_type>"
      }
    ]
  },
  "serverUrl": "https://<sso_base_url>/sso/auth/login-widget-router",
  "step": "auth_form"
}
```

Типы ошибок

Поле JSON ответа form.errors содержит список возможных ошибок:

- social_mapping_disabled - обновление привязки для соцсети запрещено
- invalid_credentials - пара логин-пароль для учетной записи UIDM не верна
- прочие возможные ошибки

Подтверждение создания привязки и получение токена доступа

- Если предыдущий запрос завершился с параметром ‹step› = show_attach_form, то это действие подтверждает **создание привязки**
  - если необходимо отобразить информацию об аккаунте соцсети, с которым происходит привязка, можно использовать данные ответа предыдущего запроса:
    - ‹view.avatarUrl› - ссылка на изображение профиля
    - ‹view.fullName› - полное имя пользователя
    - ‹view.firstName› - имя пользователя
- если ‹step› = show_reattach_form то это действие подтверждает **обновление привязки**
  - если необходимо информацию отобразить об аккаунтах соцсети (новом и старом, с которым возник конфликт), можно использовать данные ответа предыдущего запроса:
    - ‹view.avatarUrl› - ссылка на изображение профиля; для новой привязки
    - ‹view.fullName› - полное имя пользователя; для новой привязки
    - ‹view.firstName› - имя пользователя; для новой привязки
    - ‹view.oldAvatarUrl› - ссылка на изображение профиля; для старой привязки
    - ‹view.oldFullName› - полное имя пользователя; для старой привязки

Отмена действия вместо текущего шага останавливает создание или обновление привязки.

Формат запроса

```
POST /sso/oauth2/access_token

Host: <sso_host>
Accept: application/json
Content-Type: application/x-www-form-urlencoded
Cookie: execution=<execution_value>

client_id=<client_id>&
client_secret=<client_secret>&
realm=<realm>&
```

```
grant_type=urn%3Aroox%3Aparams%3Aoauth%3Agrant-type%3Am2m&
service=dispatcher&
_eventId=next&
execution=<execution_value>&
response_type=token+cookie
```

Параметры

- <sso_host> - базовый адрес сервера UIDM, например sso.rooxteam.com
- <client_id> - идентификатор клиента, например selfcare
- <client_secret> - пароль клиента, например selfcare_password
- <realm> - url-encoded пользовательский realm, например %2Fcustomer
- <grant_type> - способ аутентификации пользователя, всегда используется значение urn:roox:params:oauth:grant-type:m2m
- <service> - используемый сервис, для этого запроса всегда dispatcher
- <_eventId> - идентификатор действия, для этого запроса всегда next
- <execution> - идентификатор предсессии аутентификации
- <response_type> - способ аутентификации пользователя, всегда используется значение "token cookie"

```
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":
"eyAiY3R5IjogRhlXVCIsICJ0eXZu5iAiand0IiwgImFsGuM6ICJIUzI1NiIgfQ.eyJhdWQiOlsieW90YV9sa19tMm
0iXSwed3ViIjoiYmlzX19fX18yRpg3NjU0MzIxIiwiABUoIjoxNjAxOTgzNTc4LCJhenAiOiJ5b3RhT5trX20ybSIs
ImFtciI6ImRldmljZV9zdG9yZWRfdG9rZW4iLCJ0b2tlbk5hbWUiOiFrCF90b2tlbiIsImlzcyI6InlvdSWwbGtfbT
JtIiwicmVhbG0i0uUvY3VzdG9tZXIiLCJ0b2tlblR5cGUiOiJCaVJlZGRlZEpXVFRva2VuIiwiZXhwIjoxNjAxOTg0
MTc4LCJpNOZiOjE2MDE5ODM1Nzh9._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

Отмена действия

После некоторых шагов последовательности возможна отмена действия (указано для шага, если возможность есть). В таком случае следующим необходимо выполнить запрос отмены действия.

Запрос отмены действия

Формат запроса

```
POST /sso/oauth2/access_token

Host: <sso_host>
Accept: application/json
Content-Type: application/x-www-form-urlencoded
Cookie: execution=<execution_value>

client_id=<client_id>&
client_secret=<client_secret>&
realm=<realm>&
grant_type=urn%3Aroox%3Aparams%3Aoauth%3Agrant-type%3Am2m&
service=dispatcher&
_eventId=cancel&
execution=<execution_value>
```

Параметры

- <service> - в данном запросе всегда dispatcher
- <_eventId> - в данном запросе всегда cancel

Формат успешного ответа

Описание ответа

```
HTTP/1.1 200 OK

set-cookie: execution=;Version=0;Path=/;Max-Age=0;Secure; SameSite=Lax; HttpOnly
set-cookie: execution=<execution_value>;Version=0;Path=/;Secure; SameSite=Lax; HttpOnly
```

```
{
  "execution": "<execution_value>",
  "view": {
    "avatarUrl": "<sn_profile_avatar_url>",
    "ssoUrl": "https://<<sso_base_url>>/sso",
```

```
    "isBlocked": false,
    "fullName": "<sn_profile_fullName>",
    "esiaRequestUri": "<esia_request_uri>",
    "esiaRedirectUri": "/esia_callback.jsp",
    "socialNetworkId": "<social_network_id>",
    "firstName": "<sn_profile_firstName>",
    "vkontakteAppId": "<vkontakte_app_id>",
    "yandexRedirectUri": "/yandex_callback.jsp",
    "googleRedirectUri": "/google_callback.jsp",
    "vkontakteRedirectUri": "/vk_callback.jsp",
    "vkontakteRequestScopesAsArray": [],
    "odnoklassnikiRedirectUri": "/ok_callback.jsp",
    "mailRuRedirectUri": "/mailru_callback.jsp"
  },
  "form": {
    "name": "loginForm",
    "fields": {},
    "errors": []
  },
  "serverUrl": "https://<sso_base_url>/sso/auth/login-widget-router",
  "step": "auth_form"
}
```


Общие для всех шагов сообщения об ошибках

Не указан или передан пустой параметр <execution>

Формат ответа с ошибкой

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


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


Не указан параметр <_eventId>

Формат ответа аналогичен ответу на первом шаге: Начало сценария создания привязки

Указан неверный параметр <_eventId>

Формат ответа аналогичен ответу на первом шаге: Начало сценария создания привязки, с заполненным списком ошибок в теле JSON ответа:


```
{
  "form": {
    "errors": [
      {
```

```
        "field": "username",
        "message": "may not be null"
      },
      {
        "field": "password",
        "message": "may not be null"
      }
    ]
  }
}
```

Алгоритм вычисления значения параметра ‹socialData›

Сторонний ресурс возвращает результат попытки авторизации пользователя в виде JSON. Данный JSON необходимо преобразовать и передать в UIDM в качестве параметра ‹socialData›. Для преобразования необходимо: представить JSON в формате x-www-form-urlencoded, закодировать полученную строку в UTF-8, взять битовое представление строки в кодировке UTF-8, закодировать битовое представление в Base64. Полученная строка из битового представления и есть значение параметра ‹socialData›.

Пример. Пусть JSON, который отправляет сторонний ресурс, есть

```
{
  "accessToken": "EAACo4Is07YsBAFygpkjSqxKEN8hOBZAWBlEZD",
  "data_access_expiration_time": 1574223509,
  "expiresIn": 6091,
  "signedRequest": "FmLQr-m3i9F9",
  "userID": "100007547412176"
}
```

Тогда x-www-form-urlencoded представление будет

```
accessToken=EAACo4Is07YsBAFygpkjSqxKEN8hOBZAWBlEZD&data_access_expiration_time=1574223509&
expiresIn=6091&signedRequest=FmLQr-m3i9F9&userID=100007547412176
```

Битовое представленние, закодированное в Base64, будет

```
YWNjZXNzVG9rZW49RUFBQ280SXMwN1lzQkFGeWdwa2pTcXhLRU44aE9CWkFXQmxFWkQmZGF0YV9hY2Nlc3NfZXhwaX
JhdGlvbl90aW1lPTE1NzQyMjM1MDkmZXhwaXJlc0luPTYwOTEmc2lnbmVkUmVxdWVzdD1GbUxRci1tM2k5RjkmdXNl
cklEPTEwMDAwNzU0NzQxMjE3Ng==
```

Поддерживаемые ресурсы и их идентификаторы

Каждый ресурс имеет уникальный идентификатор в SSO.

Таблица 1. Таблица поддерживаемых ресурсов и их идентификаторов

название ресурса	идентификатор

Вконтакте	vkontakte
Одноклассники	odnoklassniki
Twitter	twitter
Google	google
Яндекс	yandex
Mail.Ru	mailru