API управления мультиккаунтом

Версия продукта: uidm-documentation-web_develop-3.6.0

1. История изменений

Версия документа	Дата изменения	Комментарий
v1.0.0	2016-12-12	Создание документа

Unresolved directive in common-multiaccount-m2m-api.adoc - include::_common-agreements.adoc[]

Unresolved directive in common-multiaccount-m2m-api.adoc - include::_multiaccount-intro.adoc[]

2. Сценарий создания привязки

2.1. Назначение

Сценарий используется master-аккаунтом для создания привязки со slave-аккаунтом.

2.2. Предусловия

1. Master-аккаунт должен быть аутентифицирован в WebSSO и иметь не просроченный access token.
2. Известен номер телефона slave-аккаунта
3. Slave-аккаунт может получить СМС-сообщение

2.3. Результат работы сценария

1. Создана привязка между master-аккаунтом и slave-аккаунтом

2.4. Общие замечания

NOTE	В каждый запрос кроме 1 нужно подставлять текущий execution сеанса - идентификатор предсессии аутентификации, полученный от WebSSO в предыдущем запросе.

2.5. Шаги сценария

1. Клиент отправляет запрос 1, передавая access token master-аккаунта
2. Сервер проверяет переданный access token, не заблокирован ли master-аккаунт и имеет ли он возможность создавать привязки
3. Сервер возвращает "форму" ввода номер телефона slave-аккаунта и имя привязки либо ошибку, если создание привязки для текущего master-аккаунта невозможно либо токен не валиден
4. Клиент запрашивает номер телефона у конечного пользователя (способ получения на усмотрение клиента). Клиент нормализует номер телефона к формату E.164. Опционально передается имя привязки, если пользователь ввел его
5. Пользователь вводит номер телефона slave-аккаунта
6. Клиент отправляет запрос 2, передавая номер телефона slave-аккаунта и опционально имя создаваемой привязки
7. Сервер ищет аккаунт по номеру телефона, проверяет блокировки на аккаунте
8. Сервер генерирует OTP-код и отправляет СМС-сообщение с OTP-кодом на номер телефона slave-аккаунта
9. Сервер возвращает "форму" ввода OTP-кода или сообщение об ошибке, если аккаунт не

найден или заблокирован

10. Клиент запрашивает OTP-код у конечного пользователя (способ получения на усмотрение клиента)
11. Пользователь вводит OTP-код
12. Клиент отправляет запрос 3, передавая введенный пользователем OTP-код
13. Сервер проверяет OTP-код
14. Сервер отдает "форму" подтверждения создания привязки либо ошибку, если код введен неверно. С формой передается информация о создаваемой привязке (номера телефонов, прочее)
15. Клиент запрашивает у пользователя подтверждение (способ получения на усмотрение клиента)
16. Пользователь подтверждает создание привязки
17. Клиент отправляет запрос 4
18. Сервер создает привязку
19. Сервер возвращает ответ, сигнализирующий об успешном окончании сценария и передает access token slave-аккаунта
20. Клиент отображает пользователю информацию о созданной привязке

В случае неверно введенного OTP-кода шаги 10 - 13 повторяются вплоть до исчерпания попыток ввода.

Пользователь может запросить новый OTP-код, если он не получил предыдущий, в таком случае шаги 6 - 9 повторяются.

Сценарий может быть прерван пользователем на любом этапе. Клиент отправляет запрос "Отмена", если имеет техническую возможность определить прерывание сценария пользователем.

2.6. Формат запроса 1 - старт сценария

Запрос начинает выполнение нового сценария.

```
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=multiaccount_create&
accessToken=<accessToken>
```

2.6.1. Параметры запроса

- 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 сценария, для данного сценария всегда multiaccount_create
- realm - группа пользователей WebSSO, всегда используется значение %2Fcustomer, которое является uri-encoded значением /customer
- accessToken - access token master-аккаунта

Результатом выполнения запроса будет валидация существующего токена и отображение "формы" ввода номера телефона slave-аккаунта.

2.6.2. Формат успешного ответа

После получения результата надо отобразить пользователю форму для ввода номера телефона slave-аккаунта и опционального имени привязки.

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

```
{
  "step": "choose_slave",
  "execution": "bedd093e-1ee6-410a-bd9b-
075aded0f096_AAABFQAAAQBW4FwlLivr4jJunJzBnhjuj59nRdWBlk6+N68EOI5EcNZTMw9SVWUHSe5EylXt1
t/qq6gjyBCtuCSknqdBunoe255w+PHOf2h7bT5pLjTNX7Gd/BvubY2s9xp4x1qEucpHK47H24bEv2xzh8CbO+A
qkXysVhz9jOXpzOpiiOkxTmg0XYc072sAQ8x/vUCrAaF5uwOpOMpPQxng38VouNYpNiAXxqgrQKj0xpn9Dz5Gs
5neIBh8areZxBi0xjtIxP5P6y4uV2vZ8eorvnYsY1aFxebo3omEFAbtciVXGx+6AjQ5V0dZ9BX0Hs5DcC3f94M
HLNOjvZvLlcrNsddkgqx1AAAACWV4ZWN1dGlvbjoIbhXoS3iibP/tHiQWPEmlbZPQfaXDHBhRFjdAdUXuuqtlX
Ig0kziwuAMwoBa+43VWHNHoNPL962J+d4kscmZgjL9T2GQtASNJr2GWsEPPvNydIC54JtN/HFWqNeJlqEGFw4I
QwlapQ9aTPzUpYgrok4KxGbVVPPOWlShpSmb5K3D21fh2VFJQQMtB40nZrYsxiaR2Bzbq3jyeLYaPjm8cof1tG
B34Hx3G3F2FPbhEiB5UQpLhKWNIafrM20YzNEstyyaDMw+yV5RV3AdjodxFnE20Q6MOy8clG8Cqe7OOEOy1B4p
A4gWNNuJA9HE3rZyicT6LZhE6GXZJ0Sb4uzrhgmq3Uo8Xe5N8/ezDa6GC8DzNS/IcjLFA2coQIgFXaydJ5DS/m
s0lMRxuLowBRQvo5G8HKPxyCsSGivMe83o0cBdmi0+UOfYxZ7urzx4JOq2+meZg6eFxYFOCqxCyLiinQiYQaZ4
EPLGvYIM8tI2L/ItXRD0UMOC0TJcsGlYHgnFnJQNgc/BZH9naL24I6A+duiX5ZSmCBJ3l2s2O6Qk1ozYgHzsHh
oKyykB+wH+zd8aZhe1viMpmuO0zmnNWQ1kL/S+afqJCDhNx7F5vnp5OxP+V34HTxg6cB+dnCOVdpucNDkZRcr6
3EHeDWhuHWqTAFH1qK7zZJ6n61K9Y2xMMv/rNthimX7oiv0at7kybDLuRwRP5xei0J4MszlsTbWC+3KF4OOY/w
eeDYDaLk8Hv4+yAo4UqO+Oy1Re3ncYDSqdaFl2fYBVrHeVJIKPByo68Hj32Sbo0CRHPa1wNbWX3T4XeoqUTNtw
C6U8i0sqYZO31oeghIN+HvIRsDRx8fGKQYVhE6m1uRjyg2VZPUB6MhTEFzncs43bH4GHY9ruCpqJQvxSsy/Fb2
voDPqC6C10FHtZZl2MUiOz6Yi6XaAZ/Cp3SI1mlo7/AGyfLUHgYnvnKuBGveztxFTlaUpqmkI6OwSSFgQe012x
Yj2BGsD9fzJL83S5P7S5TyWFCr6SvvSRiRKmuhLqRympO3U4H60N96PV6TWQ1fXBvxg9Xa1lKu/+KtQsByCdfm
fddTtYtExsMP7Nyr410dixXVFC03iWwz70vV3tqk55QrqE3Yg5qj/iJsq/4jHLlzPWzyQtm6h36WJtX1jVPBbi
cLIOOd59IwhuXl1bB/aEKHzD5bn4BIZnJfOme3hhb6d80Rd3pd7/sQpU/GgA7jQo3YaZ96EdyRiuzPytqdv1qI
UnSqtrzrNh8YyLhio4d9bf5vE/W7V0DbFmFhSCrSffQHFpmXCIVTF+2eZNYbxXwv+f9E2JlgCgryo0h4m/sQL2
u+cWqpebwRLDLaHEXF3FPt6dPct08FKprkL01OtU46TiFlsV87PbWnSg2BHGdRzyiZuN+VaxqMk8RtN4foeobS
JdOfmZdinxDP2H/VYaMP/oe/PORl+NquMktSy71+yt+s4ZcZlg=",
  "serverUrl": "http://sso.mgf.hosted:8080/sso/auth/_multiaccount-authenticate-slave-
by-otp",
  "form": {
    "fields": {
      "slaveLogin": {
        "constraints": [
          {
            "name": "NotEmpty"
          }
        ]
      },
      "displayName": {
        "constraints": [
          {
            "attributes": {
              "max": 2000,
              "min": 0
            },
            "name": "Size"
          }
        ]
      }
    },
    "errors": [],
    "name": "multiaccountChooseSlaveForm"
  },
```

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

- step - имя формы, для данного шага всегда choose_slave, что означает "форма ввода номера телефона slave-аккаунта"
- execution - идентификатор предсессии аутентификации, передать в следующем запросе к WebSSO
- serverUrl - игнорировать
- form - описание полей "формы", которую клиент должен отрисовать пользователю
- slaveLogin - описание поля ввода номера телефона slave-аккаунта. Поле представлено как более общее - логин, поскольку в общем случае WebSSO может поддерживать создание привязки и по логину.
- displayName - описание поля ввода имени привязки

2.7. Формат запроса 2 - ввод номера телефона slave-аккаунта

```
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=multiaccount_create&
execution=<execution>&
_eventId=next&
slaveLogin=<slaveLogin>&
displayName=<displayName>
```

2.7.1. Параметры запроса

- 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 сценария, для данного сценария всегда multiaccount_create
- execution - идентификатор предсессии аутентификации, полученный от WebSSO в предыдущем запросе
- realm - группа пользователей WebSSO, всегда используется значение %2Fcustomer, которое является uri-encoded значением /customer
- _eventId - идентификатор следующего действия, всегда next для этого запроса
- slaveLogin - номер телефона slave-аккаунта в формате E.164, например +79210000000
- displayName - опциональное имя привязки

Результатом выполнения запроса будет отправка OTP-кода на номер телефона slave-аккаунта

2.7.2. Формат успешного ответа

После получения результата надо отобразить пользователю форму для ввода OTP-кода.

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


```
{
  "step": "enter_otp_form",
  "execution": "dec7dacf-as2d-4345-aae5-
d08410cb8f95_H4sIAAAAAAAAAN1WTWwcRRYu27GdxE5wEnYF",
  "serverUrl": "http://sso.rooxteam.com/sso/auth/otp-sms",
  "form": {
    "fields": {
      "otpCode": {
        "constraints": [
          {
            "name": "NotNull"
          }
        ]
      }
    },
    "errors": [],
    "name": "otpForm"
  },
  "view": {
    "otpCodeAvailableAttempts": 2,
    "msisdn": "+79876543210",
    "nextOtpPeriod": 120,
    "blockedFor": 0,
    "isBlocked": false
  }
}
```


- step - имя формы, для данного шага всегда enter_otp_form, что означает "форма ввода OTP-кода"
- execution - идентификатор предсессии аутентификации, передать в следующем запросе к WebSSO
- serverUrl - игнорировать
- form - описание полей "формы", которую клиент должен отрисовать пользователю
- otpCode - описание поля ввода OTP-кода
- view - клиент на усмотрение может отобразить пользователю эти данные
- view.otpCodeAvailableAttempts - кол-во оставшихся попыток ввода OTP кода
- view.msisdn - номер телефона, на который отправлен OTP SMS

- view.nextOtpPeriod - время в секундах до наступления возможности отправки нового OTP кода
- view.blockedFor - время до разблокировки в секундах, имеет смысл только если isBlocked == true
- view.isBlocked - признак временной блокировки пользователя

2.8. Формат запроса 3 - ввод OTP-кода пользователем и валидация его на сервере

Данный запрос валидирует введенный OTP код пользователем. Результатом выполнения данного запроса будет отображение формы подтверждения создания привязки.

```
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=multiaccount_create&
execution=<execution>&
_eventId=validate&
displayName=<displayName>
```

2.8.1. Параметры запроса

- 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 сценария, для данного сценария всегда multiaccount_create
- execution - идентификатор предсессии аутентификации, полученный от WebSSO в предыдущем запросе
- realm - группа пользователей WebSSO, всегда используется значение %2Fcustomer, которое является uri-encoded значением /customer

- _eventId - идентификатор следующего действия: validate - для запроса на валидацию либо send для заказа повторного OTP-кода
- otpCode - введенный пользователем OTP код, передается для _eventId=validate

2.8.2. Формат успешного ответа

После получения результата надо отобразить пользователю форму для подтверждения создания привязки.

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

```
{
  "step": "enter_otp_form",
  "execution": "dec7dacf-as2d-4345-aae5-
d08410cb8f95_H4sIAAAAAAAAAN1WTWwcRRYu27GdxE5wEnYF",
  "serverUrl": "http://sso.rooxteam.com/sso/auth/_multiaccount-attach",
  "form": {
    "fields": {},
    "errors": [],
    "name": "attachForm"
  },
  "view": {
    "displayName": "My mapping",
    "slaveMsisdn": "+79210000000",
    "masterMsisdn": "+79310000000"
  }
}
```

- step - имя формы, для данного шага всегда enter_otp_form, что означает "форма ввода OTP-кода"
- execution - идентификатор предсессии аутентификации, передать в следующем запросе к WebSSO
- serverUrl - игнорировать
- form - описание полей "формы", которую клиент должен отрисовать пользователю
- view - клиент на усмотрение может отобразить пользователю эти данные
- view.displayName - имя привязки, если пользователь указал его
- view.slaveMsisdn - номер телефона slave-аккаунта
- view.masterMsisdn - номер телефона master-аккаунта

2.9. Формат запроса □4 - подтверждение создания привязки

Данный запрос создает привязку. Результатом выполнения данного запроса будет создание привязки и окончание сценария.

```
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=multiaccount_create&
execution=<execution>&
_eventId=next
```

2.9.1. Параметры запроса

- 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 сценария, для данного сценария всегда multiaccount_create
- execution - идентификатор предсессии аутентификации, полученный от WebSSO в предыдущем запросе
- realm - группа пользователей WebSSO, всегда используется значение %2Fcustomer, которое является uri-encoded значением /customer
- _eventId - идентификатор следующего действия: всегда next для данного запроса

2.9.2. Формат успешного ответа

Полученный ответ содержит в себе новый access token для входа под slave-аккаунтом. Клиент может проигнорировать этот токен, а может использовать сразу же для переключения пользователя в slave-аккаунт.

NOTE	Следует заметить, что старый access token master-аккаунта продолжает действовать

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

```
{
  "token_type": "Bearer",
  "scope": "cn",
  "access_token": "0c7a0d61-43a0-4e2e-9a4a-7749a91f126a",
  "expires_in": 59
}
```

- access_token - выданный access token slave-аккаунта
- expires_in - время до истечения срока действия токена в секундах

2.10. Возможные ограничения на поля формы

название	описание	атрибуты	описание атрибута
NotNull	Обязательность поля		
Size	Длина строкового параметра	min	Минимальная длина
		max	Максимальная длина
Pattern	Regexp для строкового параметра	regexp	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.

2.11. 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

3. Сценарий перехода в slave-аккаунт

3.1. Назначение

Сценарий используется master-аккаунтом для перехода в slave-аккаунт.

3.2. Предусловия

1. Master-аккаунт должен быть аутентифицирован в WebSSO и иметь не просроченный access token
2. Установлена привязка master-аккаунта и некоторого slave-аккаунта
3. Известен идентификатор привязки, по которой хочется произвести переключение. Идентификатор можно получить запросом списка привязок для текущего аккаунта в WebAPI

3.3. Результат работы сценария

1. На WebSSO установлена новая сессия под slave-аккаунтом. В сессии хранится информация о том, что она была установлена путем переключения и сохранен идентификатор master-аккаунта.

3.4. Шаги сценария

1. Клиент отправляет запрос 1, передавая access token master-аккаунта и идентификатор привязки
2. Сервер проверяет переданный access token, не заблокирован ли master-аккаунт
3. Сервер ищет привязку по идентификатору, проверяет не заблокирована ли она и что текущий аккаунт действительно является master-аккаунтом в данной привязке
4. Сервер возвращает ответ, сигнализирующий об успешном окончании сценария и передает access token slave-аккаунта
5. Клиент устаналивает новую сессию в своей системе на основе полученного access token’а

3.5. Формат запроса №1 - старт сценария и переключение в slave-аккаунт

```
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=multiaccount_impersonate_slave&
accessToken=<accessToken>&
multiaccountMappingId=<multiaccountMappingId>
```

3.5.1. Параметры запроса

- 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 сценария, для данного сценария всегда multiaccount_impersonate_slave
- realm - группа пользователей WebSSO, всегда используется значение %2Fcustomer, которое является uri-encoded значением /customer
- accessToken - access token master-аккаунта
- multiaccountMappingId - идентификатор привязки, по которой произвести переход в slave-аккаунт

3.5.2. Формат успешного ответа

Полученный ответ содержит в себе новый access token для входа под slave-аккаунтом. Клиент может проигнорировать этот токен, а может использовать сразу же для переключения пользователя в slave-аккаунт.

NOTE	Следует заметить, что старый access token master-аккаунта продолжает действовать

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

```
{
  "token_type": "Bearer",
  "scope": "cn",
  "access_token": "0c7a0d61-43a0-4e2e-9a4a-7749a91f126a",
  "expires_in": 59
}
```

① access_token - выданный access token slave-аккаунта

② expires_in - время до истечения срока действия токена в секундах

3.6. 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

4. Сценарий перехода обратно в master-аккаунт

4.1. Назначение

Сценарий используется master-аккаунтом для перехода из slave-аккаунта обратно в master-аккаунт.

4.2. Предусловия

1. Master-аккаунт должен аутентифицироваться в WebSSO, выполнить переход в slave-аккаунт и иметь не просроченный access token slave-аккаунта
2. Установлена привязка master-аккаунта и некоторого slave-аккаунта

4.3. Результат работы сценария

1. На WebSSO установлена новая сессия под master-аккаунтом.

4.4. Шаги сценария

1. Клиент отправляет запрос 1, передавая access token slave-аккаунта
2. Сервер проверяет переданный access token, не заблокирован ли slave-аккаунт
3. Сервер проверяет по access token, что данная сессия была установлена путем перехода в slave-аккаунт
4. Сервер возвращает ответ, сигнализирующий об успешном окончании сценария и передает access token master-аккаунта
5. Клиент устаналивает новую сессию в своей системе на основе полученного access token’а

4.5. Формат запроса □1 - старт сценария и переключение в master-аккаунт

```
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=multiaccount_impersonate_master&
accessToken=<accessToken>
```

4.5.1. Параметры запроса

- 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 сценария, для данного сценария всегда multiaccount_impersonate_master
- realm - группа пользователей WebSSO, всегда используется значение %2Fcustomer, которое является uri-encoded значением /customer
- accessToken - access token slave-аккаунта

4.5.2. Формат успешного ответа

Полученный ответ содержит в себе новый access token для входа под master-аккаунтом.

NOTE	Следует заметить, что старый access token slave-аккаунта продолжает действовать

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

```
{
  "token_type": "Bearer",
  "scope": "cn",
  "access_token": "0c7a0d61-43a0-4e2e-9a4a-7749a91f126a",
  "expires_in": 59
}
```


① access_token - выданный access token master-аккаунта

② expires_in - время до истечения срока действия токена в секундах

5. Требования к реализации ПО

5.1. Общие требования

НЕОБХОДИМО синхронизировать системные часы ресурс сервера со службой точного времени NTP.

НЕДОПУСТИМО передавать секретный ключ ресурс сервера в сторонние IT-системы кроме WebSSO в запросах на /token, /tokeninfo и /policyEvaluation.

НЕДОПУСТИМО протоколировать секретный ключ ресурс сервиса.

НЕДОПУСТИМО передавать токен доступа в любые другие IT-системы кроме WebSSO и другие Resource Server.

НЕДОПУСТИМО протоколировать токен в немаскированном виде.

НЕДОПУСТИМО отключать проверку SSL-сертификатов при любых обращениях к WebSSO.

Система МОЖЕТ поддерживать другие способы аутентификации кроме входа через WebSSO. В таких случаях ТРЕБУЕТСЯ указание этого факта в спецификации интеграции. Система проверяет доступ к своим ресурсам самостоятельно. При аутентификации через WebSSO НЕДОПУСТИМО запрашивать персональные данные пользователей (в том числе логин, пароль) на страницах ресурс сервера кроме сценарией использования M2M API при получении согласования со службой безопасности компании.

5.2. Рекомендации по мониторингу

Рекомендуется реализовать мониторинг доступности WebSSO непосредственно с серверов приложений. Для этого можно регулярно выполнять обращения на

<базовый адрес WebSSO>/sso/isAlive.jsp , например https://id.test.com:7443/sso/isAlive.jsp

Проверка должна выполняться не чаще, чем 1 раз в 10 секунд. В случае, если запрос возвращает HTTP код, отличный от 200, система мониторинга должна уведомлять о проблемах с коннективностью до WebSSO.

Технически проверка доступности WebSSO может быть реализована любым удобным способом, важно, чтобы проверка выполнялась с того же сервера, на котором расположеное приложение-клиент WeSSO.

Unresolved directive in common-multiaccount-m2m-api.adoc - include::_common-links.adoc[]

Unresolved directive in common-multiaccount-m2m-api.adoc - include::_endpoints-mgf.adoc[]