Аутентификация системы (OAuth Client Credentials Flow)

Оглавление

Аутентификация системы используется в тех случаях, когда действие должно быть выполнено от имени приложения (системы), а не его пользователя.

Примеры

- создание нового пользователя по сигналу из ESB;
- вызов API противодействия мошенничеству (антифрода).

Описание механизма

1. Система выполняет вызов UIDM для получения системного токена, передает свой идентификатор и секрет.
2. UIDM выдает токен, сообщает время жизни токена.
3. Система сохраняет токен в локальном кеше для использования в вызовах к веб-приложениям.
4. Система выполняет вызовы требуемых веб-приложений, передавая системный токен в заголовке Authorization.
5. Когда время жизни токена истекло или он был отозван (веб-приложение возвратило код `401`), получить новый системный токен, начав сценарий заново.

Идентификатор системы и секрет назначаются заранее администратором UIDM.

Конечная точка

```
POST https://{sso_host}/sso/oauth2/access_token
```

Получение токена

Аутентификация выполняется по протоколу OAuth2.0 с передачей параметра `grant_type=client_credentials`.

В случае успеха в ответе будет сразу возвращен токен доступа. В случае ошибки ответ будет содержать данные, указывающие на произошедшую ошибку.

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

grant_type=client_credentials&realm=%2Fcustomer&client_id=<client_id>&client_secret=
<client_secret>
```

- sso_host — базовый адрес сервера WebSSO, например sso.rooxteam.com
- realm — группа пользователей WebSSO. Всегда используется значение %2Fcustomer, которое является uri-encoded значением /customer
- client_id — идентификатор клиента, соответствующий системе, например antifraud
- client_secret — пароль клиента
- grant_type — способ авторизации пользователя. Всегда используется значение client_credentials

Альтернативный способ аутентификации через заголовок Authorization

При необходимости вместо параметров client_id и client_secret можно передать заголовок Authorization:

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

grant_type=client_credentials&realm=%2Fcustomer
```

- sso_host — базовый адрес сервера WebSSO, например sso.rooxteam.com
- credentials — пара `идентификатор_клиента:пароль`, закодированная с помощью BASE64, например YW50aWZyYXVkOnBhc3N3b3Jk для `antifraud:password`
- realm — группа пользователей WebSSO, всегда используется значение %2Fcustomer, которое является uri-encoded значением /customer
- grant_type — способ авторизации пользователя, всегда используется значение client_credentials

Формат ответа при успешном получении токена

```
HTTP/1.1 200 OK
```

```
{
```

```
  "scope": "cid cn givenname sn telephoneNumber user_name",
  "token_type": "JWTToken",
  "expires_in": 1199,
  "access_token": "…"
}
```

- scope — список scope разрешенных для использования от имени пользователя, возможные значения задаются конфигурацией
- token_type — тип выданного токена.
- expires_in — время до истечения срока действия токена в секундах
- access_token — выданный access token

Формат ответа сервера после неуспешного автовхода

Если при попытке получения токена произошла ошибка, WebSSO отправляет её код вместе с описанием:

```
HTTP/1.1 401 Unauthorized
```

```
{
  "error_description": "Client authentication failed",
  "error": "invalid_client"
}
```

- error — строковый код ошибки
- error_description — описание ошибки

Проверка токена

Для получения информации о созданном токене можно использовать Token Introspection API (интроспекция токена):

```
GET /sso/oauth2/tokeninfo?access_token=<access_token> HTTP/1.1
Host: <sso_host>
Cache-Control: no-cache
```

- sso_host — базовый адрес сервера WebSSO, например sso.rooxteam.com
- access_token — токен, полученный в ответе на шаге Получение токена

Формат ответа для валидного токена

```
HTTP/1.1 200 OK
```

```
{
  "sub": "antifraud",
  "scope": [
    "user_name",
    "sn",
    "cn",
    "telephoneNumber",
    "givenname",
    "cid"
  ],
  "realm": "/customer",
  "roles": [
    "ROLE_SYSTEM"
  ],
  "token_type": "JWTToken",
  "expires_in": 1085,
  "client_id": "antifraud",
  "auth_level": "0",
  "access_token": "…"
}
```


- sub — идентификатор клиента;
- scope — список scope разрешенных для использования от имени пользователя, возможные значения задаются конфигурацией;
- realm — группа пользователей WebSSO, всегда используется значение /customer;
- roles — список ролей, присвоенных клиенту, для которого создавался токен;
- expires_in — время до истечения срока действия токена в секундах;
- client_id — идентификатор клиента;
- token_type — тип выданного токена;
- auth_level — выданный уровень авторизации пользователя. Всегда 0, игнорируется
- access_token — выданный токен доступа.

Формат ответа для невалидного токена

Если проверяемый токен указан неверно, либо его срок действия истёк, WebSSO возвращает ошибку:

```
HTTP/1.1 401 Unauthorized
```


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


- error — строковый код ошибки
- error_description — описание ошибки

Авторизация запросов к веб-приложениям с помощью выданного токена

Для выполнения запросов к веб-приложениям от имени системы с использованием полученного токена нужно добавить заголовок:

```
Authorization: Bearer sso_1.0_<access_token>
```

- access_token — токен полученный в ответе на шаге Получение токена

Примеры вызова

Чтение

```
GET /otp-settings-1.0/user/otp/settings/field HTTP/1.1
Host: api.rooxteam.com
Authorization: Bearer sso_1.0_eyAiYWxnIjog…
```

Модификация

```
POST /money/transfer/internal/card/tocard HTTP/1.1
Host: api.rooxteam.com
Authorization: Bearer sso_1.0_eyAiYWxnIjog…
Cache-Control: no-cache
Content-Type: application/json
```

Предлагаемый формат вызовов может отличаться от действительного; для получения подробностей необходимо обратиться к документации приложения (системы).

Примеры ответа

Ответ на успешный запрос

```
HTTP/1.1 200 OK
```

```
{
  "success" : true,
  "data" : {
    ...
  }
}
```

Ответ на запрос без токена, либо с токеном с истекшим сроком действия

```
HTTP/1.1 401 Unauthorized
```

```
{
  "error": {
    "code": 401,
    "message": "Unauthorized"
  }
}
```

Ответ на запрос с токеном к запрещённому ресурсу

```
HTTP/1.1 403 Forbidden
```

```
{
  "error": {
    "code": 403,
    "message": "Access is denied"
  }
}
```

Предлагаемый формат ответов может отличаться от действительного; для получения подробностей необходимо обратиться к документации приложения (системы).

Конфигурация

В настройках OAuth2.0 клиента установить свойство `clientClaims`. Параметр является lookup-таблицей, следует использовать особый синтаксис.

Пример файла конфигурации клиента

```
clientName=onlinebank_web

clientClaims[0]=propertykey=propertyvalue
```