Интеграция с RooX UIDM по OAuth 2.0

Оглавление

Общие положения

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

Пропуск шага автоматического входа

Для пропуска автоматического входа может быть использован параметр запроса:

- roox_skipAutoLogin
  - Доступные значения: true, false.
  - По умолчанию: false.
  - **Опционально**, **Регистрозависимо**

Включение одного из модулей автоматического входа

При необходимости сохранить механизм пропуска автоматического входа для всех модулей, обеспечивающих эту функциональность, но намеренно включить только один конкретный модуль, можно использовать параметр:

- roox_forceAutoLoginModule
- Значения: <module_name> - идентификатор модуля автоматического входа
  - **Опционально**, **Регистрозависимо**

Указанный модуль при правильно переданных требуемых для этого модуля параметрах будет применен без обычного механизма проверки очередности модулей; остальные модули применены не будут.
Если требуемые для модуля параметры не переданы или не корректны, модуль применен не будет.
Если параметр roox_skipAutoLogin не использован, то параметр roox_forceAutoLoginModule не имеет эффекта.

Доступные модули:

id модуля	описание работы модуля	параметр
login-by-jwt	автовход по JWT-токену в URL параметре	auto-login-jwt
autologin	автовход по токену в URL параметре	auto-login-jwt
login-by-postreg-token	автовход по PostReg-Token	postreg_token
agw	автовход через внешнюю систему аутентификации Authentication Gateway	
kerberos		
auto-login-by-ip		

Аутентификация

Для выполнения аутентификации нужно направить пользователя на соответствующий адрес RooX UIDM с указанием идентификатора клиента, пароля клиента и адреса обратного перехода. RooX UIDM выполнит аутентификацию пользователя по логину-паролю или другому способу, если это необходимо, после чего выполнит переход на указанный адрес и передаст при этом код, по которому можно получить токен.

Формат запроса

```
GET <sso_host>/sso/oauth2/authorize?
    login_hint=<login_hint>&
    prompt=<prompt>&
    realm=<realm>&
    response_type=<response_type>&
    client_id=<client_id>&
    service=external&
    redirect_uri=<redirect_url>&
    scope=<scope_value>&
    roox_skipAutoLogin=<roox_skipAutoLogin>&
    roox_forceAutoLoginModule=<roox_forceAutoLoginModule>&
    logout_reason=<logout_reason>&
    return_strategy=<return_strategy>&
    state=<state>&
    roox_pp=<roox_pp>&
    roox_postreg_token=<roox_postreg_token>
```

Параметры

- login_hint - параметр, значения которого будет предустановлено в поле логина в форме аутентификации. **опционально**
- prompt - запрос повторной аутентификации у пользователя. Возможные значения: login - пользователь должен всегда заново аутентифицироваться; none - пользователю не надо заново аутентифицироваться, если он уже был ранее аутентифицирован. **опционально**
- <sso_host> - базовый адрес сервера RooX UIDM, например sso.rooxteam.com. Может содержать порт, например sso.rooxteam.com:8080.
- realm - группа пользователей RooX UIDM. Возможны два значения: %2Fcustomer и %2b2b, которые является uri-encoded значениями /customer и /b2b.
- response_type - тип запрашиваемого разрешения на авторизацию (authorization grant). Возможные значения приведены в таблице возможных разрешений на авторизацию.
- client_id - идентификатор клиента, например selfcare, возможные значения зависят от конфигурации.
- service - механизм аутентификации в RooX UIDM, всегда используется значение external.
- redirect_uri - URL обратного редиректа, URL-encoded, например https%3A%2F%2Fsso.rooxteam.com%2Fsso%2Fsecure%2Flk.jsp для адреса `https://sso.rooxteam.com/sso/secure/lk.jsp`. Адрес должен быть в списке разрешенных в RooX UIDM, список разрешенных адресов конфигурируется администратором RooX UIDM.
- scope - список запрашиваемых scope через пробел в кодировке UTF-8. **опционально**, **регистрозависимо**
- roox_skipAutoLogin - флаг для пропуска шага автоматического входа с помощью HTTP header enrichment, при котором пользователь перенаправляется редиректом на HTTP (не HTTPS) endpoint, и в HTTP запрос подмешивается HTTP заголовок x-nokia-msisdn с доверенным идентификатором пользователя. Доступные значения: true, false. По умолчанию: false. **опционально**, **регистрозависимо**
- roox_forceAutoLoginModule - название модуля автовхода, который будет использован для автовхода даже при

активном параметре roox_skipAutoLogin

- logout_reason - код причины разрыва сессии. **опционально**
- return_strategy - код стратегии возврата при блокировке. **опционально**
- state - дополнительные данные, которые нужно передать при перенаправлении на redirect_uri после аутентификации. **опционально**
- roox_pp - флаг для включения режима аутентификации через внешнюю платформу. Доступные значения: True, False. По умолчанию: False. **опционально**, **регистрозависимо**
- roox_postreg_token - PostReg-Token выданный при создании учетной записи или привязки к партнерскому аккаунту для автовхода (пользователь не вводит данные учетной записи, а попадает автоматически в авторизованную зону)

Формат успешного ответа

Аутентифицированный пользователь будет перенаправлен на redirect_uri, указанный в запросе, с кодом авторизации в GET-параметре.

```
GET <redirect_url>?code=<auth_code>&state=<state>
```

Параметры

- <redirect_url> - URL обратного редиректа, например https://sso.rooxteam.com/sso/secure/lk.jsp.
- <auth_code> - код авторизации, например 1d601e9a-992d-44a7-be21-bca07fbc762c.
- <state> - дополнительные данные, которые были переданы в начале аутентификации в параметре state (если параметр state не указывался, то он будет отсутствовать и в конечном URL, на который происходит перенаправление).

Формат неуспешного ответа

Для пользователя может быть заблокирован конкретный ресурс по client_id. В этом случае вместо кода авторизации в GET-параметре будет описание ошибки.

```
GET <redirect_url>?
error=access_denied&error_description=The%20resource%20owner%20or%20authorization%20server
%20denied%20the%20request
```

Параметры

- <redirect_url> - URL обратного редиректа, например `https://sso.rooxteam.com/sso/secure/lk.jsp`.
- <error> - код ошибки согласно спецификации OAuth 2.0 RFC 6749 пункт 5.2, например access_denied.
- <error_description> - текстовое описание ошибки, например The%20resource%20owner%20or%20authorization%20server%20denied%20the%20request.

Формат ответа при блокировке пользователя

ВАЖНО

В текущей реализации процесса аутентификации не поддерживается.

Пользователь может быть заблокирован. В этом случае вместо кода авторизации в GET-параметре будет описание ошибки.

```
GET <redirect_url>?
error=access_denied&error_description=The%20resource%20owner%20or%20authorization%20server
%20denied%20the%20request.&error_subtype=user_blocked&expires_in=3600
```

Параметры

- <redirect_url> - URL обратного редиректа, например `https://sso.rooxteam.com/sso/secure/lk.jsp`
- <error> - код ошибки согласно спецификации OAuth 2.0 RFC 6749 пункт 5.2, например access_denied.
- <error_description> - текстовое описание ошибки, например The%20resource%20owner%20or%20authorization%20server%20denied%20the%20request.
- <error_subtype> - статус токена, список возможных статусов приведен ниже.
- <expires_in> - время до истечения срока блокировки (в секундах). Если параметр отсутствует, то учетная запись сможет быть разблокирована только администратором системы.

Формат ответа при блокировке OAuth 2.0 клиента

OAuth 2.0 клиент может быть заблокирован. В этом случае вместо кода аворизации в GET-параметре будет описание ошибки.

```
GET <redirect_url>?error=invalid_client&error_description=Client%20is%20blocked
```

Параметры

- <redirect_url> - URL обратного редиректа, например `https://sso.rooxteam.com/sso/secure/lk.jsp`
- <error> - код ошибки согласно спецификации OAuth 2.0 RFC 6749 пункт 5.2, например invalid_client.
- <error_description> - текстовое описание ошибки, например Client%20is%20blocked

Получение токена доступа

Для получения данных пользователя и возможности выполнять действия от его имени защищаемый клиент должен получить access token. Для этого нужно выполнить запрос на соответствующий URL RooX UIDM с указанием данных клиента и кода, полученного после аутентификации. Данный URL RooX UIDM можно также использовать для обновления access token с помощью refresh token. Результатом будет новый токен, либо сообщение об ошибке.

Формат запроса для получения access token по коду

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

```
realm=<realm>&
client_id=<client_id>&
client_secret=<client_secret>&
redirect_uri=<redirect_uri>&
grant_type=authorization_code&
code=<authorization_code>
```

Формат запроса для обновления access token по refresh token

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

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

Параметры

- <sso_host> - базовый адрес сервера RooX UIDM, например sso.rooxteam.com. Может содержать порт, например sso.rooxteam.com:8080.
- realm - группа пользователей RooX UIDM. Возможны два значения: %2Fcustomer и %2b2b, которые является uri-encoded значением /customer и /b2b.
- client_id - идентификатор клиента, например selfcare, возможные значения зависят от конфигурации.
- client_secret - пароль клиента, возможные значения зависят от конфигурации.
- redirect_uri - URL обратного редиректа, который был указан при аутентификации, URL encoded. Например https%3A%2F%2Fsso.rooxteam.com%2Fsso%2Findex.html для адреса https://sso.rooxteam.com/index.html.
- grant_type - способ получения access token. В сценарии получения access token по коду всегда используется значение authorization_code. В сценарии обновления access token по refresh token всегда используется значение refresh_token.
- code - код авторизации, полученный после аутентификации. Например 1d601e9a-992d-44a7-be21-bca07fbc762c.
- refresh_token - ранее выданный вместе с токеном refresh token. Например d5e0ccfe-501e-4740-8708-a45fa5d369ce.

Формат успешного ответа на запрос для получения access token по refresh token или коду, если для аутентификации использовался response_type=code

```
HTTP/1.1 200 OK
```

```
{
  "access_token": "ea6a2d0c-740d-4c90-8e43-5e2ce6853a76",
```

```
  "token_type": "Bearer",
  "expires_in": 1199,
  "refresh_token": "d5e0ccfe-501e-4740-8708-a45fa5d369ce",
  "refresh_expires_in": 11999,
  "scope": [
    "contactEmail",
    "displayName",
    "givenname",
    "companyMsisdn",
    "impersonator",
    "cn",
    "pp_username",
    "sn"
  ],
  "JWTToken": "eyAiY3R5IjogIkpXVCIsICJ0eXAiOiA"
}
```


Формат успешного ответа на запрос для получения access token по коду, если для аутентификации использовался response_type=code mpt

```
HTTP/1.1 200 OK
```


```
{
  "access_token": "ea6a2d0c-740d-4c90-8e43-5e2ce6853a76",
  "token_type": "Bearer",
  "expires_in": 1199,
  "refresh_token": "d5e0ccfe-501e-4740-8708-a45fa5d369ce",
  "refresh_expires_in": 11999,
  "mpt": "a4da560f-ffe0-4091-9f7f-cec964553c5b",
  "mpt_expires_in": 11999,
  "scope": [
    "contactEmail",
    "displayName",
    "givenname",
    "companyMsisdn",
    "impersonator",
    "cn",
    "pp_username",
    "sn"
  ],
  "JWTToken": "eyAiY3R5IjogIkpXVCIsICJ0eXAiOiA"
}
```


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 access token в секундах.
- token_type - тип выданного токена. Всегда Bearer.
- access_token - выданный access token.
- refresh_token - выданный refresh token. Может быть использован для получения нового access token без повторной аутентификации.

- refresh_expires_in - время до истечения срока действия refresh token в секундах.
- mpt - выданный мультиплатформенный токен, привязанный к выданному access token.
- mpt_expires_in - время до истечения срока действия mpt в секундах.
- JWTToken - JSON Web Token, содержащий информацию о токене и аутентифицированном пользователе. Содержимое определяется конфигурационным ключом com.rooxteam.sso.token.response.embedded.jwt.forward.claims. Данный параметр опционален и его наличие зависит от значения конфигурационного ключа com.rooxteam.sso.token.response.embedded.jwt.enabled.

Формат неуспешного ответа

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

```
{
  "error": <error>,
  "error_description": <error_description>
}
```

Параметры

- error - код ошибки согласно спецификации OAuth 2.0 RFC 6749 пункт 5.2
- error_description - текстовое описание ошибки, например, The provided access grant is invalid, expired, or revoked.

Возможные ошибки

Невалидный код или код с истекшим сроком действия

Аналогичная ошибка будет получена при блокировке учетной записи пользователя и при блокировке пользователю доступа к конкретному ресурсу по client_id

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

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

OAuth 2.0 клиент заблокирован

```
HTTP/1.1 403 Forbidden
```

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

Переданный redirect_uri не совпадает с указанным при аутентификации

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

```
{
  "error": "redirect_uri_mismatch",
  "error_description": "The redirection URI provided does not match a pre-registered value."
}
```

Не передан access token

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

```
{
  "error": "invalid_request",
  "error_description": "Missing access_token"
}
```

Указанный способ авторизации не поддерживается

Данная ошибка не должна возникать при использовании способа authorization_code

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

```
{
  "error_description": "Grant type is not supported: authorization_token",
  "error": "unsupported_grant_type"
}
```

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 GET запрос на соответствующий URL RooX 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 защищаемого ресурса. Если нет, ответ будет содержать информацию о требуемом уровне авторизации для получения доступа к запрашиваемому ресурсу.

Если требуется передать дополнительные параметры для аудита доступа к защищенным ресурсам, необходимо выполнить POST запрос и передать параметры в теле запроса в виде json-объекта.

Формат GET запроса на простую валидацию токена

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

Формат GET запроса на валидацию токена с указанием защищаемого ресурса

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

Формат POST запроса на валидацию токена с указанием защищаемого ресурса и дополнительными параметрами

```
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-Nokia-MSISDN": [
```

```
      "9876543210"
    ],
    "X-Forwarded-For": [
      "10.20.30.40",
      "10.10.35.46",
      "192.168.12.74"
    ]
  }
}
```

Параметры

- <sso_host> - базовый адрес сервера RooX UIDM, например sso.rooxteam.com, может содержать порт, например sso.rooxteam.com:8080.
- access_token - полученный 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, передаются "как есть" без дополнительной обработки или фильтрации. **опционально**

Формат успешного ответа

```
HTTP/1.1 200 OK
```

```
{
  "scope": [
    "sn",
    "companyMsisdn",
    "contactEmail",
    "cn",
    "givenname",
    "impersonator",
    "displayName"
  ],
  "roles": [
    "ROLE_CUSTOMER"
  ],
  "sn": "Петров",
  "givenname": "Пётр",
  "contactEmail": "example@roox.ru",
  "companyMsisdn": "9999999999",
  "cn": "9263752235",
  "realm": "/customer",
  "token_type": "Bearer",
  "authType": "login_password",
  "expires_in": 28,
  "sub": "bis_____199412412152222",
  "access_token": "7bdaeacc-3d80-415c-920f-a7c30ca5e743",
  "auth_level": "2",
  "networkAuthenticationType": "AUTO",
```

```
  "client_id": "selfcare"
}
```

Обязательные атрибуты

- scope - список scope (в формате JSON Array) разрешенных для использования от имени пользователя, возможные значения задаются конфигурацией.
- realm - группа пользователей RooX 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.
- client_id - идентификатор клиента, которому выдан токен, возможные значения зависят от конфигурации.
- sub - идентификатор аутентифицированного пользователя.

Опицональные атрибуты

- auth_level - выданный уровень авторизации.
- authType - способ, с помощью которого была произведена аутентификация пользователя.
- roles - роли пользователя.

Опциональные атрибуты scope

В ответе могут присутствовать опциональные атрибуты scope разрешенных для использования от имени пользователя. Например:

- cn - номер телефона пользователя;
- telephoneNumber - номер телефона пользователя;
- networkAuthenticationType - разрешен ли автоматический вход по устройству;
- pp_username - логин пользователя во внешней платформе (при аутентификации с roox_pp=True);
- companyMsisdn - корпоративный msisdn пользователя;
- contactEmail - email пользователя;
- givenname - имя пользователя;
- sn - фамилия пользователя.

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 пользователя и при блокировке пользовательского 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 RFC 6749 пункт 5.2
- error_description - текстовое описание ошибки

Формат неуспешного ответа при недостаточном уровне авторизации

Форма неуспешного ответа идентична форме успешного ответа за исключением кода состояния ответа, который равен 403, и наличия дополнительного параметра advices.

```
HTTP/1.1 403 Forbidden
```


```
{
  "scope": [
    "sn",
    "companyMsisdn",
    "contactEmail",
    "cn",
    "givenname",
    "impersonator",
    "displayName"
  ],
  "roles": [
    "ROLE_CUSTOMER"
  ],
  "sn": "Петров",
  "givenname": "Пётр",
  "contactEmail": "example@roox.ru",
  "companyMsisdn": "9999999999",
  "cn": "9263752235",
  "realm": "/customer",
  "token_type": "Bearer",
  "authType": "login_password",
  "expires_in": 28,
  "sub": "bis_____199412412152222",
  "access_token": "7bdaeacc-3d80-415c-920f-a7c30ca5e743",
  "auth_level": "2",
  "networkAuthenticationType": "AUTO",
  "client_id": "selfcare",
  "advices": {
    "required_auth_level": "9"
  }
}
```


Параметры

- advices - JSON с единственным полем required_auth_level, в котором содержится минимально необходимый уровень авторизации для получения доступа к запрошенному ресурсу.

Формат ответа при блокировке OAuth 2.0 клиента

```
HTTP/1.1 403 Forbidden
```

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

Формат ответа при предъявлении access token с истекшим сроком действия

```
HTTP/1.1 401 Unauthorized
```

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

Глобальный выход из RooX UIDM по ссылке

Глобальный выход из RooX UIDM может быть осуществлен переходом аутентифицированного пользователя на URL <sso_host>/sso/UI/Logout. При этом RooX UIDM завершит текущую сесиию пользователя и удалит все выданные в рамках данной сессии токены. Желательно добавить в URL параметр goto с адресом обратного перехода, на этот адрес пользователь будет перенаправлен сразу после выхода.

Формат запроса

```
GET <sso_host>/sso/UI/Logout?goto=<goto>
```

- <sso_host> - базовый адрес сервера RooX UIDM, например `sso.rooxteam.com`, может содержать порт, например `sso.rooxteam.com:8080`
- <goto> - URL обратного редиректа, URL-encoded, например `https%3A%2F%2Fsso.rooxteam.com%2Fsso%2Findex.html` для адреса `https://sso.rooxteam.com/index.html`

Формат успешного ответа

Пользователь будет перенаправлен на адрес, указанный в параметре goto

Формат неуспешного ответа

В случае ошибки выхода из RooX UIDM пользователь будет перенаправлен на адрес, указанный в параметре goto, при этом будут добавлены GET-параметры с кодом и описанием ошибки.

```
GET <goto>?error=<error>&error_description=<error_description>
```

Параметры

- ‹goto› - URL обратного редиректа, указанный в запросе
- ‹error› - код ошибки, например internal_error
- ‹error_description› - текстовое описание ошибки, например Cannot%access%20session%20store.

Выход через API

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

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

При инвалидации access token также инвалидируется связанный с этим токеном mpt.

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>&ip=<ip>&user_agent=<user_agent>&referer=
<referer>
```

Параметры

- ‹sso_host› - базовый адрес сервера RooX 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.
- ‹ip› - IP адрес пользователя в формате x.x.x.x. **опционально**
- ‹user_agent› - строка, идентифицирующая ПО клиента (браузера). **опционально**
- ‹referer› - Значение HTTP заголовка Referer из пользовательского запроса. **опционально**

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 RFC 7009 пункт 4.1.1
- error_description - текстовое описание ошибки

Кеширование токенов

Если защищаемый сервис находится под общим доменом второго уровня с RooX UIDM, то он может использовать кеширование результатов валидации токенов, чтобы не делать запрос на валидацию токена при каждом действии пользователя. В этом случае сервис должен отслеживать значение cookie sh, как только значение cookie изменится, необходимо сбросить кеш.

Если защищаемый сервис не находится под доменом второго уровня RooX UIDM, кешировать результаты валидации токенов запрещено, необходимо всегда делать запрос валидации в соответствии с пунктом Валидация токена доступа.

Вызов callback URL при инвалидации токенов

RooX 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 RooX UIDM будет отправлять оповещения. Список URL для оповещения настраивается администратором RooX UIDM.

Инвалидация токена пользователя

При инвалидации токена конкретного пользователя RooX UIDM отправляет соответствующее оповещение на callback URL. Защищаемый сервис, получив такое оповещение, должен немедленно инвалидировать сессию пользователя. Если защищаемый сервис использует кеширование 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 RooX 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 RooX 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 RooX 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 RooX 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 RooX UIDM.
- event - имя события.
- global - флаг полной блокировки сервиса, в этом оповещении всегда false.
- cn - номер телефона пользователя (msisdn).
- access_token - измененный access token.

Справочник возможных типов разрешений на авторизацию

Каждому типу разрешения на авторизацию соответствует своя схема авторизации. Значения (не uri-encoded) параметра response_type из пункта Аутентификация приведены в следующей таблице

значение	описание
code	В ответе содержится Authorization Code, который можно обменять на access token
code mpt	В ответе содержится Authorization Code, который можно обменять на access token и mpt

Справочник возможных причин разрыва сессии

Значения параметра logout_reason из пункта Аутентификация приведены в следующей таблице

код	описание
idle_timeout	Бездействие пользователя в течение определенного времени
session_timeout	Истечение времени сессии

Справочник возможных стратегий возврата при блокировке

Значения параметра return_strategy из пункта Аутентификация приведены в следующей таблице

код	описание
off	Отображение ошибки на виджете логина без перехода на redirect_uri. Используется по умолчанию

auto	Автоматический переход на redirect_uri при блокировке пользователя
manual	Отображение в виджете кнопки "Назад" при блокировке пользователя, переход переход на redirect_uri при нажатии этой кнопки

Справочник возможных статусов токена

код	описание
revoked	Пользователь аннулировал токен
user_blocked	Пользователь заблокирован

Использование параметра scope

Параметр scope (RFC 6749 пункт 3.3) может применяться в RooX UIDM для двух разных случаев.

- определение набора данных пользователя, к которым сервис получает доступ. По умолчанию любой ответ 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 запроса атрибуты настраиваются администратором RooX UIDM отдельно для каждого сервиса.

Атрибуты могут включать:

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, указаный 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 RooX UIDM.

scope - мнемоническое название ресурса защищаемого сервиса.

Диаграммы переходов для работы с RooX UIDM

Аутентификация и авторизация в RooX UIDM

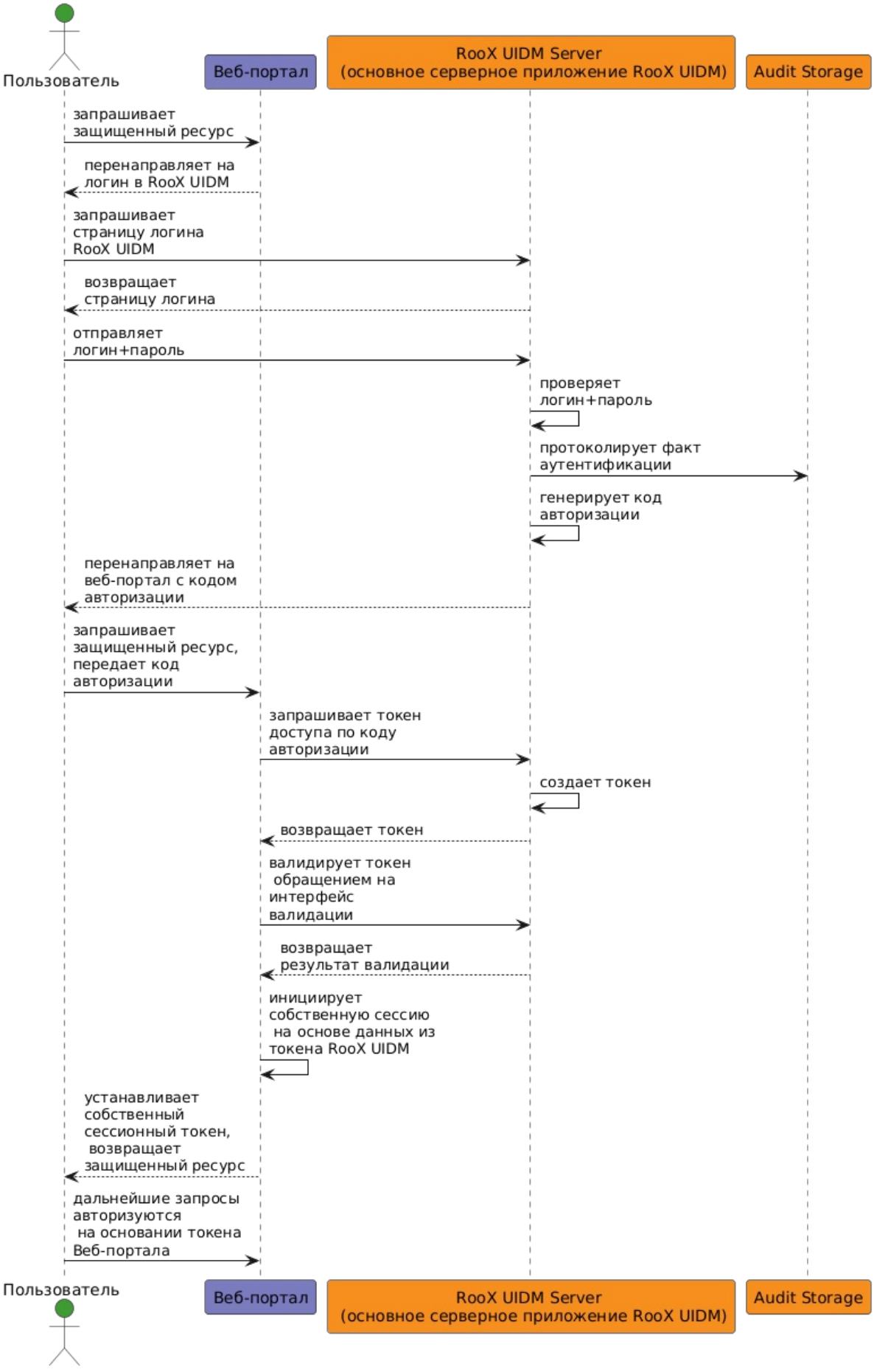

Рисунок 1. Аутентификация и авторизация в RooX UIDM

Повышение уровня авторизации

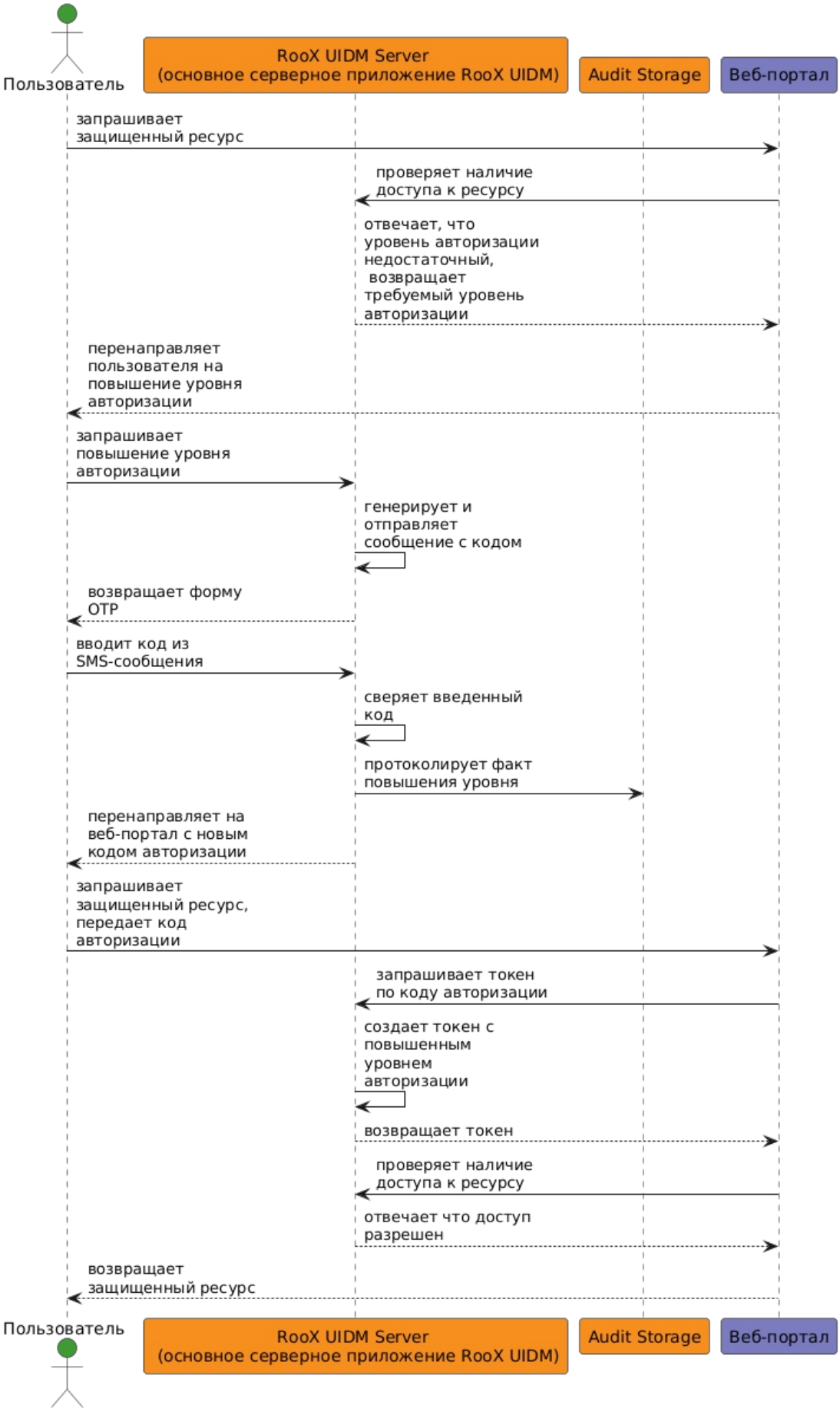

Рисунок 2. Повышение уровня авторизации

Отзыв токена

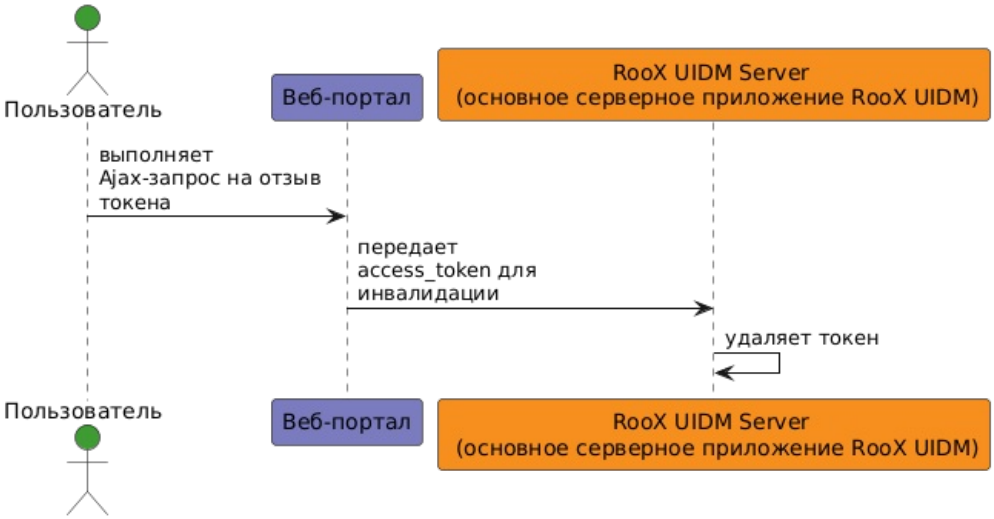

Рисунок 3. Отзыв токена через API

Logout через URL

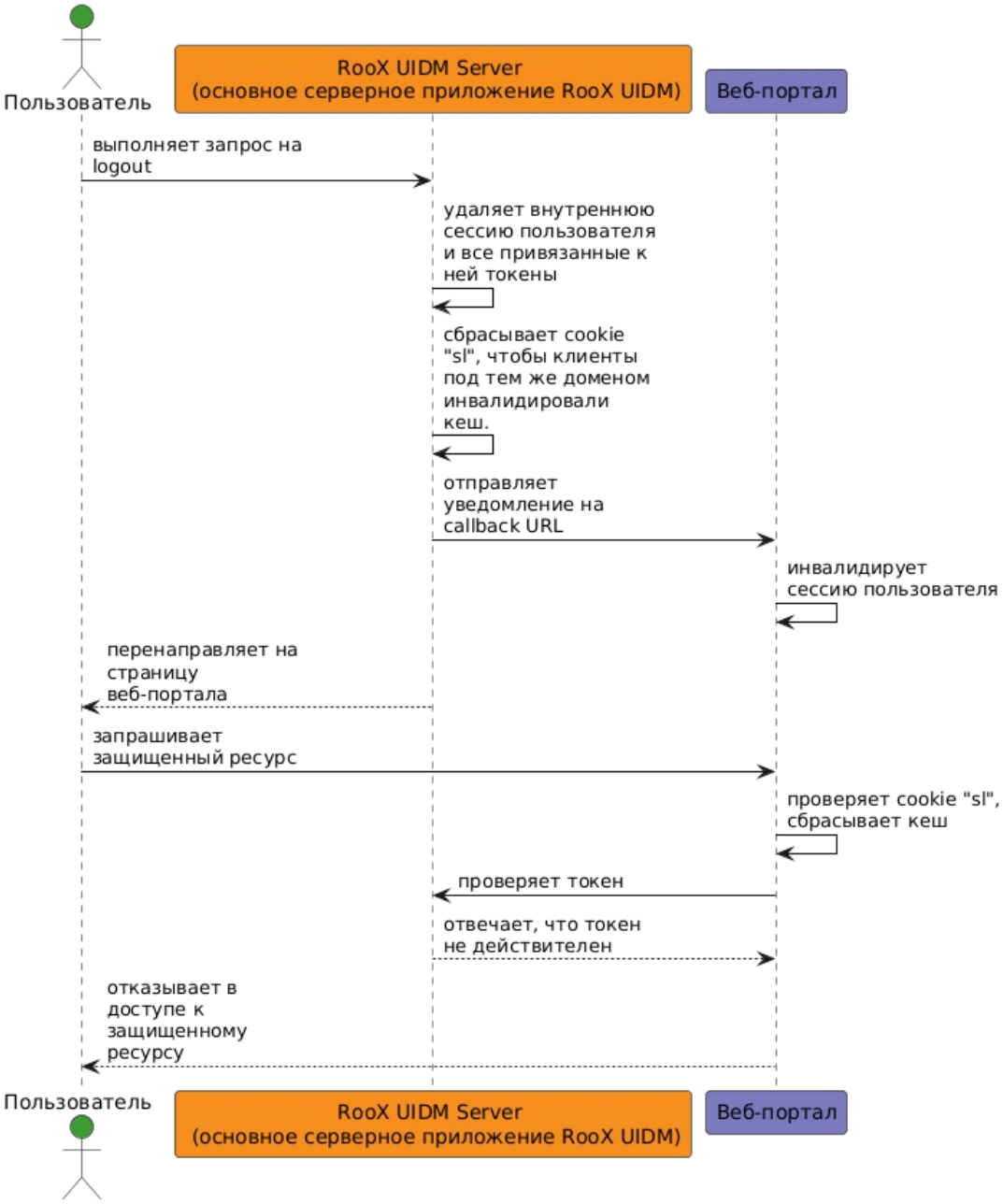

Рисунок 4. Logout через URL