Единая авторизация ТПУ

Введение

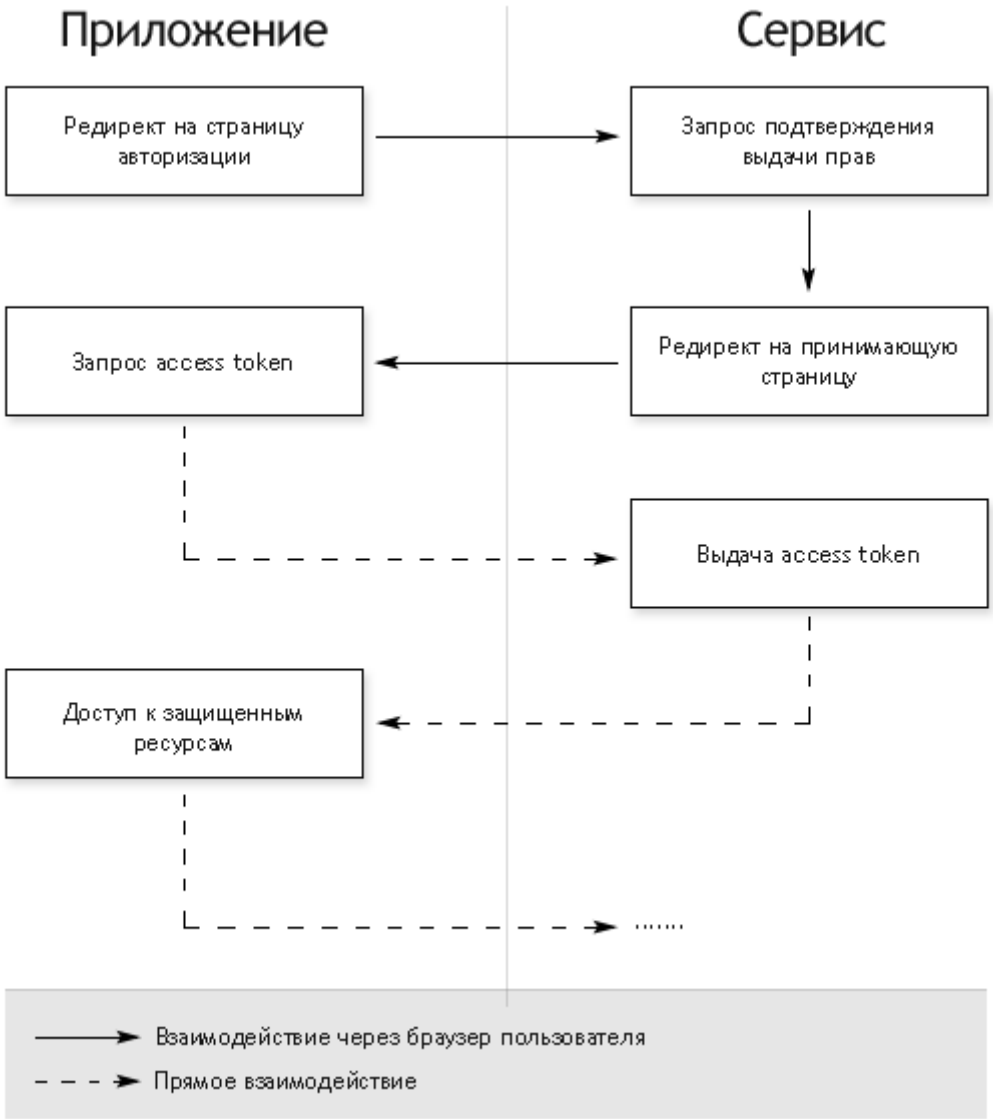
Данный сервис позволяет провести авторизацию пользователей ТПУ в приложении, сохранив при этом учетные данные пользователей.

Адрес сервиса в сети интернет: https://oauth.tpu.ru

Описание сервиса

Данный сервис авторизации, позволяющий выдать одному сервису (приложению) права на доступ к ресурсам пользователя на другом сервисе. Протокол избавляет от необходимости доверять приложению логин и пароль, а также позволяет выдавать ограниченный набор прав, а не все сразу.

Авторизация для приложений, имеющих серверную часть

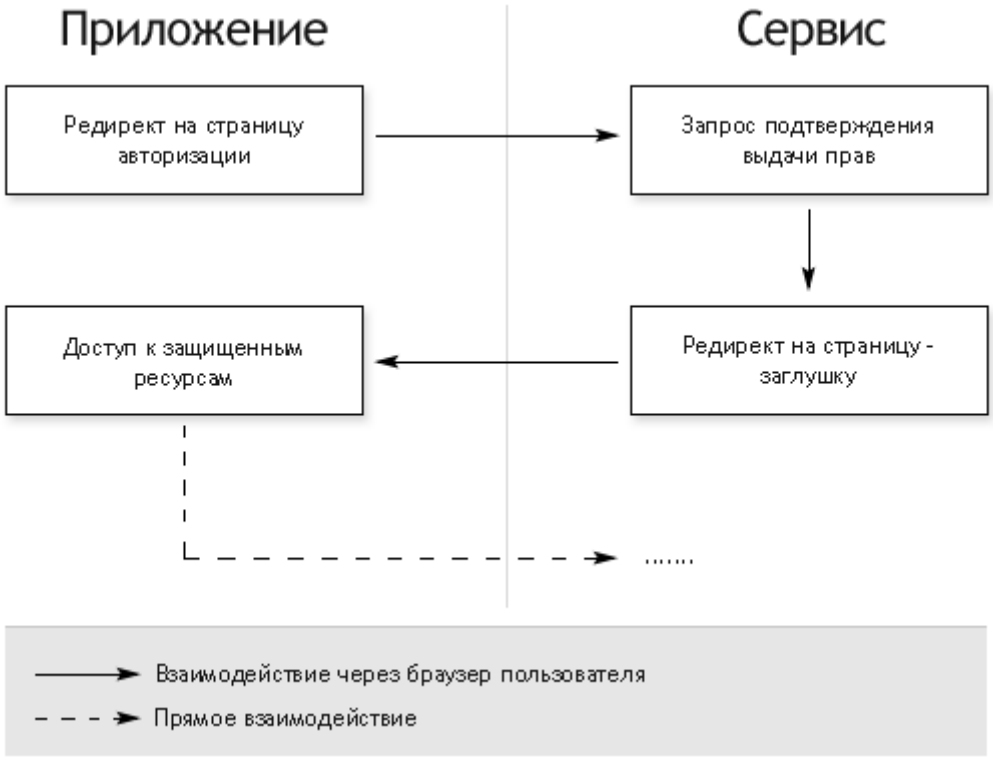


1. Редирект на страницу авторизации

2. На странице авторизации у пользователя запрашивается подтверждение выдачи прав
3. В случае согласия пользователя, браузер редиректится на URL, указанный при открытии страницы авторизации, с добавлением в GET-параметры специального ключа — *authorization code*
4. Сервер приложения выполняет POST-запрос с полученным *authorization code* в качестве параметра. В результате этого запроса возвращается *access token*

Это самый сложный вариант авторизации, но только он позволяет сервису однозначно установить приложение, обращающееся за авторизацией (это происходит при коммуникации между серверами на последнем шаге). Во всех остальных вариантах авторизация происходит полностью на клиенте и по понятным причинам возможна маскировка одного приложения под другое. Это стоит учитывать при внедрении OAuth-аутентификации в API сервисов.

Авторизация полностью клиентских приложений



1. Открытие встроенного браузера со страницей авторизации
2. У пользователя запрашивается подтверждение выдачи прав
3. В случае согласия пользователя, браузер редиректится на страницу-заглушку во фрагменте (после #) URL которой добавляется *access token*
4. Приложение перехватывает редирект и получает *access token* из адреса страницы

Этот вариант требует поднятия в приложении окна браузера, но не требует серверной части и дополнительного вызова сервер-сервер для обмена authorization code на access token.

Восстановление предыдущей авторизации

Access token имеет ограниченный срок годности (30 минут), так как он передается по открытым каналам. Чтобы не заставлять пользователя проходить авторизацию после истечения срока действия ***access token***'а, в дополнение к ***access token***'у возвращается еще ***refresh token***. По нему можно получить ***access token*** с помощью HTTP-запроса.

Описание процесса **восстановление авторизации**

OpenID Connect

OpenID Connect (*OIDC*) — это безопасный механизм, позволяющий приложению связаться со службой идентификации, чтобы получить необходимые данные о пользователе и вернуть их обратно в приложение, обеспечив полную защиту данных.

Термины и понятия

Используемые сокращения при работе с сервером авторизации

Параметр	Описание
client_id	Идентификатор приложения
client_secret	Защищенный ключ приложения
redirect_uri	Адрес, на который будет передан code (домен указанного адреса должен соответствовать основному домену в настройках приложения и перечисленным значениям в списке доверенных redirect uri — адреса сравниваются до корневого домена).
response_type	Тип ответа, по умолчанию значение ***code***.
state	Произвольная строка, которая будет возвращена вместе с результатом авторизации.
grant_type	Способ авторизации

Рабочий процесс

Процесс авторизации

Для авторизации пользователя через единую авторизацию пользователей ТПУ необходимо выполнить следующие действия:

1. Создать приложение (перейти по ссылке https://api.tpu.ru/dashboard).
2. Создать платформу и указать версию приложения (получить **api_key**).
3. Настроить авторизацию через Oauth (получить **client_id** и **client_secret**) - указать доверительные домены (каждый на отдельной строке)
4. Дождаться проверки приложения со стороны ТПУ и начать пользоваться.
5. Прописать все IP адреса серверов (и локальных машин разработчиков при необходимости) в доверенную зону на которую будет отправлен ответ от сервера авторизации (https://api.tpu.ru/dashboard МЕНЮ->Авторизация->Oauth2 Oauyh2 Сервер)

Процесс авторизации

1. Необходимо перенаправить браузер пользователя по адресу используя метод GET

> **https://oauth.tpu.ru/authorize**

передав следующие параметры

Ключ	Значение
client_id (обязательно)	Наименование клиента на страничке вашего приложения https://api.tpu.ru/dashboard МЕНЮ->Авторизация->Oauth2 Oauyh2 Сервер
redirect_uri (обязательно)	Ссылка, на которую после успешной авторизации сервер делает редирект пользователя. > “ домен указанного адреса должен соответствовать основному домену в настройках приложения и перечисленным значениям в списке доверенных redirect uri — адреса сравниваются до корневого домена

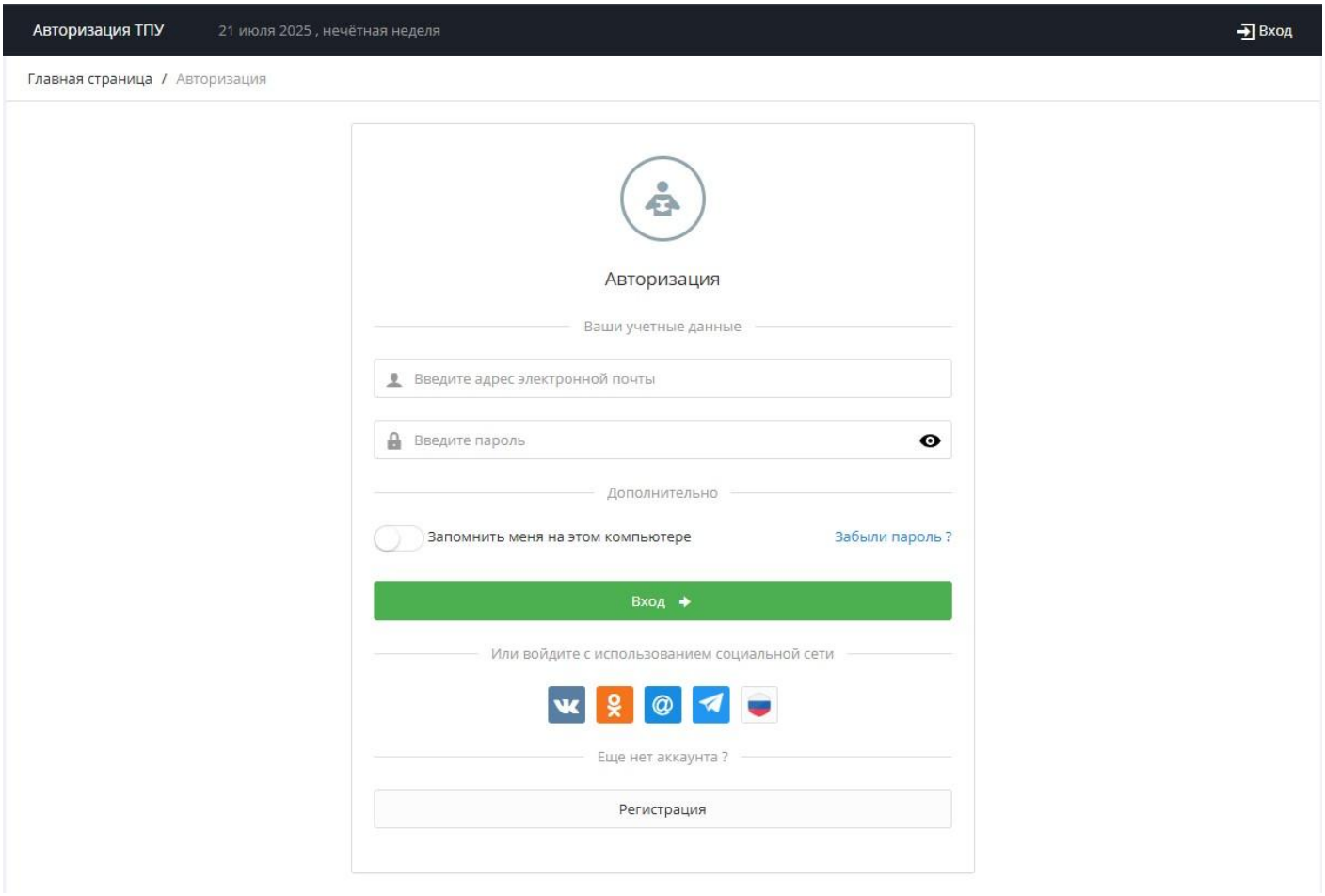
Ключ	Значение
response_type (обязательно)	Тип ответа, который Вы хотите получить. В большинстве случаев достаточно указать **code**.
state (обязательно)	**Параметр состояния** в протоколах авторизации позволяет восстановить предыдущее состояние приложения. > “ Его используют, чтобы защитить приложение от атак, связанных с подделкой запросов (CSRF). Для этого с каждым запросом аутентификации связывают уникальное и не поддающееся угадыванию значение

Пример запроса

```
https://oauth.tpu.ru/authorize?client_id=1&redirect_uri=http://example.com/callback&response_type=code
&state=bdc1c79ecb83c00122d24a77e06aa5dc16c8280f7541e89a32108659c353f5
```

Ввод корпоративных данных

2. Если пользователь уже авторизован, то браузер будет перенаправлен по адресу **redirect_uri**, в противном случае пользователю будет предложено ввести свой корпоративный логин и пароль в системе.



3. Получение токена авторизации **access_token** для пользователя и для приложения. методом GET передаем параметры на следующий адрес:

> **https://oauth.tpu.ru/access_token**

Ключ	**Значение**
client_id (обязательно)	Идентификатор приложения
client_secret (обязательно)	Секретный ключ приложения
timestamp	Текущая метка времени в формате Unix timestamp Данный параметр обязательный, если требуется произвести обезличенную авторизацию приложения.
code (обязательно, для авторизации пользователей)	Одноразовый короткоживущий код, который клиент должен использовать для обмена на токен доступа, полученный на первом этапе авторизации. Данное поле обязательно - если авторизуется пользователь.

Ключ	Значение
state (обязательно)	**Параметр состояния** в протоколах авторизации позволяет восстановить предыдущее состояние приложения. > “ Его используют, чтобы защитить приложение от атак, связанных с подделкой запросов (CSRF). Для этого с каждым запросом аутентификации связывают уникальное и не поддающееся угадыванию значение
grant_type (обязательно)	Тип авторизации > “ разрешённые типы можно посмотреть в настройках авторизации вашего приложения
redirect_uri *(обязательно)*	должно совпадать с предыдущим запросом
sig (обязательно, для авторизации приложений)	Контрольная сумма GET-параметров из HTTP-запроса. Данный параметр обязательный если был пропущен первый шаг получения **code.** > “ Ключом для расчета контрольной суммы указывается секретный ключ доверенного приложения, а **не** ключ версии приложения.

Пример запроса

```
https://oauth.tpu.ru/access_token?client_id=1&client_secret=H2PkHm&redirect_uri=http://mysite.ru&code=7a6fa4dff77a228eeda56603b8f53806c883f011c40b72630bb50df056f6479&grant_type=authorization_code
```

В результате выполнения данного запроса Ваш сервер получит вновь созданный **access_token**. Вместе с **access_token** серверу возвращается время жизни ключа **expires_in** в секундах. **expires_in** по умолчанию 24 часа. Процедуру авторизации приложения необходимо повторять в случае истечения срока действия **access_token**, смены пользователем своего логина или пароля, или удаления приложения из настроек. Также в ответе присутствует **refresh_token** - срок годности 1 неделя, с помощью его можно получить новый **access_token** не проводя процедуру авторизации повторно.

4. Проверка токена авторизации, данный метод вернёт сведения об указанном токене авторизации

> **https://oauth.tpu.ru/check-token**

В HTTP заголовке-*Authorization*

Требуется указать строку следующего формата

> “ Bearer <Токен авторизации>

В случае успешной проверки токена, придёт ответ в формате JSON, в котором будет полезная информация, идентификатор клиента, авторизовавшего пользователя (**client_id**), идентификатор пользователя (**user_id**), идентификатор личности (**lichnost_id**), контекст пользователя (**username**), время создания токена, а так же тип токена (**type**). Токены бывают двух типов - личные (**personal**) и системные (**system**).

Персональные токены получены - путём прямой авторизации пользователя в приложении (ввод логина и пароля), системные токены формируются в доверительных приложениях и микро-сервисах с использованием механизма отложенной авторизации.

Пример ответа


```
{
   "message": "Valid",
   "body": {
      "created": "2025-07-21 09:32:26",
      "expired": "2025-07-22 09:32:26",
      "client_id": "my-app-id-123456",
      "type": "personal",
      "user_id": 123,
      "lichnost_id": 123,
      "username": "portal_login"
   }
}
```


5. Получение данных об авторизированном пользователе

> **https://oauth.tpu.ru/user**

В HTTP заголовке-Authorization
Требуется указать строку следующего формата > “ Bearer <Токен авторизации>

Пример ответа


```
{
   "user_id": 59568,
   "lichnost_id": 745454,
   "elements": {
      "familiya": "Иванов",
      "imya": "Иван",
      "otchestvo": "Иванович"
   },
   "full_name": "Иванов Иван Иванович",
   "last_name": "Иванов",
   "first_name": "Иван",
   "patronymic": "Иванович",
   "email": "login@tpu.ru",

   "login": "login",
   "message": "OK"
}
```


6. Завершение сессии пользователя. Для успешной завершении сессии пользователя нужно сделать редирект на следующий URL.

> ***https://oauth.tpu.ru/auth/logout?redirect=<redirect_uri>***

Ключ	**Значение**

redirect_uri	Ссылка, на которую, после успешного завершения сессии, сервер делает редирект пользователя.

> На этой странице вы должны локально завершить сессию пользователя и сделать редирект на главную страницу вашего приложения.

Вместо заключения

В вашем приложении вы должны хранить в сессии **идентификатор авторизованного пользователя** и **access_token**. Для получения других данных о пользователе вы можете воспользоваться доступными методами RESTful API.

Секретные ключи вашего приложения вы не должны хранить на клиенте и обязаны обеспечить надлежащую их сохранность.

Восстановление авторизации

После авторизации токен обычно существует ограниченное количество времени, для того чтобы продолжить использовать авторизацию пользователя - реализуйте обновление токена для поддержания аутентификации без взаимодействия с пользователем следующим образом:

Для получения нового **access_token** необходимо выполнить GET-запрос на следующий эндпоинт, передав обязательные параметры:

> **https://oauth.tpu.ru/access_token**

Ключ	Значение
client_id (обязательно)	Идентификатор приложения
client_secret (обязательно)	Секретный ключ приложения
grant_type *(обязательно)*	Значение должно **refresh_token**
refresh_token (обязательно)	Требуется указать ключ обновления, полученный при авторизации

Пример запроса

```
https://oauth.tpu.ru/access_token?client_id=1&client_secret=12345678&grant_type=refresh_token&refresh_token=def502000c9e8750e3a5a39be466d664732e21303a491ef50cea65d2da1e4533ade84dfc1b503ddd282849a14e05996d176f211a8a979d2f47f65565b94da1e9be750e6e78053df0d00168d5b937a54e74e5e7adfa769d8377fd054fd9bee9f49f46dfb4afaefebc5cb582210f225cec21ee5e0fcfcd6808f365df72026f2424c52af750cac6987e5738c14a172094241499bb1fd1131ea20c2dbd72a09e5522940f601895beb9443700f4561766da0344876f821ae9de08832859b35540e679cedad51ad52aa1d0d4c7e5e7b3dcbecab278e3755dfdb94303c4b41b32c7800c9724cf081f71427afc9611b75ae726d1ba5dbb6237bc21c74326212adc83b8d50c440ffd6516847ea6ae643d77b3bd4109113ac250d1ced80a
```

В результате выполнения данного запроса Ваш сервер получит вновь созданный **access_token**. Вместе с **access_token** серверу возвращается время жизни ключа **expires_in** в секундах. Процедуру авторизации приложения необходимо повторять в случае истечения срока действия **access_token**, смены пользователем своего логина или пароля, или удаления приложения из настроек. Также в ответе присутствует **refresh_token** - срок годности 1 неделя, с помощью его можно получить новый **access_token** не проводя процедуру авторизации повторно.

Вместо заключения

Полезная информация

Ссылки

- Текущая версия драфта стандарта OAuth 2.0
- Официальный сайт OAuth
- Рабочая группа по выработке стандарта (архивы)