InComm

TIBCO API Exchange OAuth 2.0

**User Guide**

Version 1.0

****

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Prepared By / Last Updated By** | **Reviewed By** | **Approved By** |
| **Name** | Dwiz Kumar |  |  |
| **Role** |  |  |  |
| **Signature** |  |  |  |
| **Date** | 30 June 2015 |  |  |

Table of Contents

[1.0 Document Overview 3](#_Toc423442732)

[1.1 Document Purpose 3](#_Toc423442733)

[1.2 References 3](#_Toc423442734)

[1.3 Definitions, Acronyms and Abbreviations 3](#_Toc423442735)

[2.0 Introduction (OAuth 2.0) 3](#_Toc423442736)

[3.0 Concept of key players and example of OAuth 3](#_Toc423442737)

[3.1.1 Key players in OAuth’s transaction: 3](#_Toc423442738)

[3.1.2 Examples of OAuth 4](#_Toc423442739)

[3.1.3 Example (YouTube) scenario revisited 4](#_Toc423442740)

[4.0 Configuration 4](#_Toc423442741)

[4.1 Joomla config to enable OAuth 4](#_Toc423442742)

[4.2 API Manager config for OAuth client 5](#_Toc423442743)

[4.3 API Gateway config for OAuth server 10](#_Toc423442744)

[4.4 Gateway config for Project 11](#_Toc423442745)

[5.0 OAuth flows supported in TIBCO 12](#_Toc423442746)

[5.1 Authorization Code 12](#_Toc423442747)

[5.1.1 Authorization Code Implementation 12](#_Toc423442748)

[5.2 Password Credential 14](#_Toc423442749)

[5.2.1 Password Credential Implementation 14](#_Toc423442750)

[5.3 Client Credential 15](#_Toc423442751)

[5.3.1 Client Credential Implementation 15](#_Toc423442752)

[6.0 Appendix A – Change Log 16](#_Toc423442753)

# Document Overview

## Document Purpose

This document is intended to provide guideline on setting up TIBCO API for OAuth 2.0. As a pre-requisite,

It is required to have TIBCO API Gateway, Joomla & TIBCO API Manager installed and configured successfully as per the product installation document.

It is required to have API Gateway project to be configured in TIBCO Gateway ConfigUI. (Please refer XXXXXXXX document).

## References

Use this section to fill-in reference documents used

|  |  |
| --- | --- |
| ***Document#*** | ***Document Title*** |
|  |  |
|  |  |

## Definitions, Acronyms and Abbreviations

Use this section to fill-in definitions of acronyms & abbreviations used in this document

|  |  |
| --- | --- |
| ***Term, Acronym*** | ***Definition*** |
|  |  |
|  |  |

# Introduction (OAuth 2.0)

OAuth is an authentication protocol that allows an owner to approve one application interacting with another on owner’s behalf without giving away owner’s password.

The OAuth 2.0 authorization framework enables a third-party application to obtain limited access to an HTTP service, either on behalf of a resource owner by orchestrating an approval interaction between the resource owner and the HTTP service, or by allowing the third-party application to obtain access on its own behalf.

# Concept of key players and example of OAuth

### Key players in OAuth’s transaction:

There are 3 main key players in oauth transaction: the user, the consumer, and the service provider.

1. **Resource Owner** - The one holds the account and credentials. Typically a User can be considered as a Resource Owner.
2. **Client** - The third party app/service who want to use resources of Resource Owner / User.
3. **Server** - The party with which Resource Owner holds an Account.

### Examples of OAuth

**Posting a YouTube video on Facebook wall from YouTube:**

1. **Resource Owner** – the user (eg: us), holding the account with Facebook.
2. **Client** - Youtube.com is the client from which we want to utilize the service.
3. **Server** - Facebook with which we are holding the account.

**Correlating YouTube example with TIBCO API Exchange**

1. **Resource Owner** – the user (eg: present in the owners.property file).
2. **Client** – TIBCO API Manager acts as a client.
3. **Server** – TIBCO API Gateway acts as an authorization server.

### Example (YouTube) scenario revisited

When we try to share the video from YouTube, it will redirect us to Facebook. we will authenticate to Facebook server and Facebook being OAuth enabled, its API will supply a temporary security token with limited scope of access which is valid for a short period of time. Using this security token, YouTube will publish the Video on Facebook wall on our behalf.

# Configuration

## Joomla config to enable OAuth

1. In the browser run joomla as:

http://<hostname>:<port>/administrator

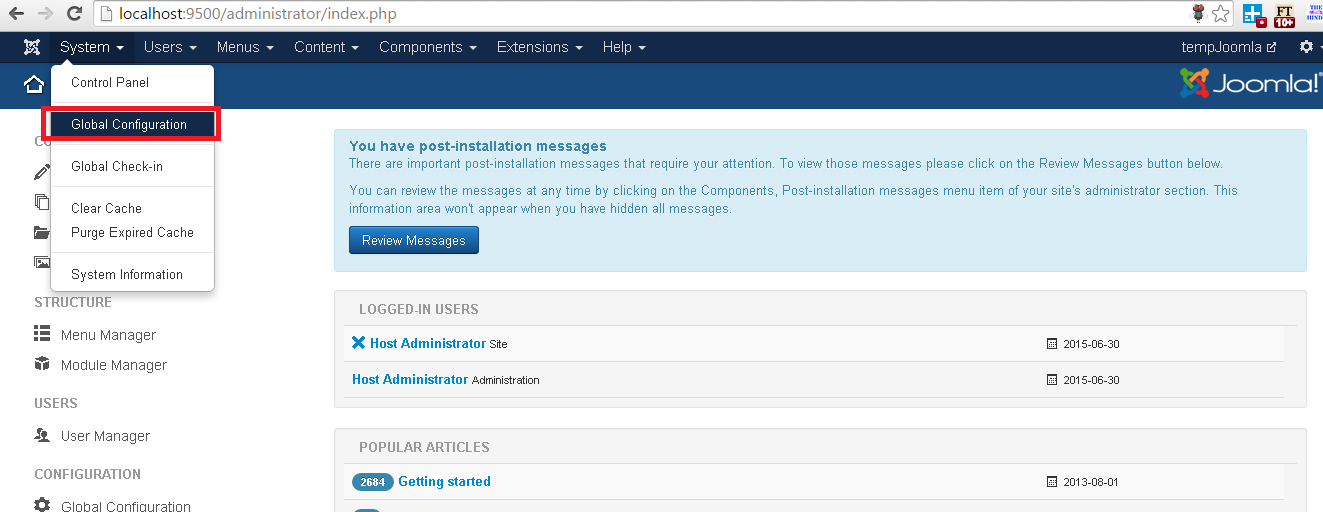
(eg: http://localhost:9500/administrator)

1. Enter the creds as :

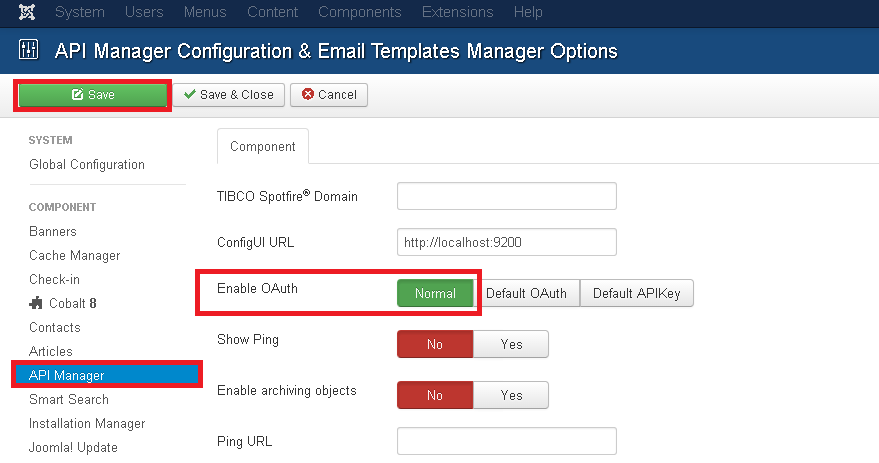
Username: admin

Password: admin

1. Select Global Configuration from the system drop down list.

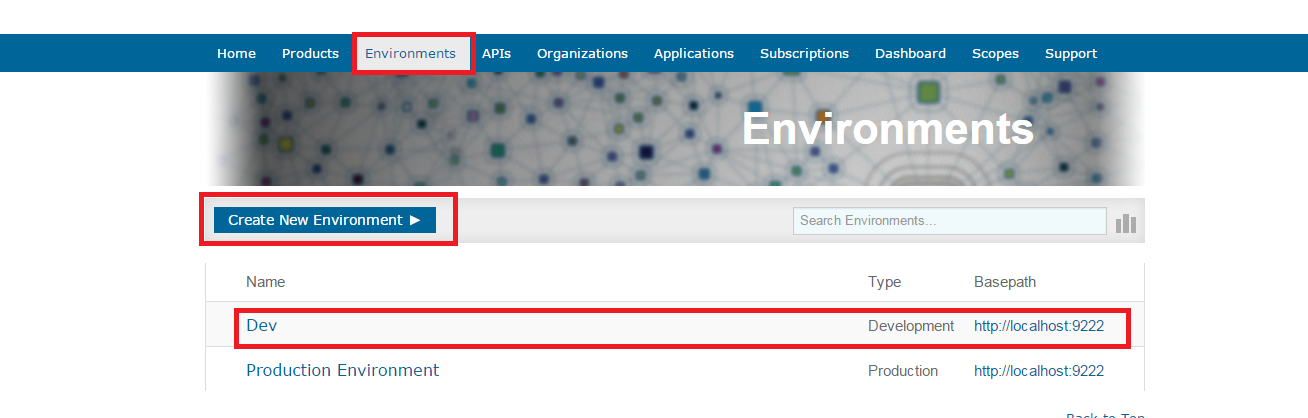


1. Select API Manager and enable oauth to ‘normal’. Save the configuration.

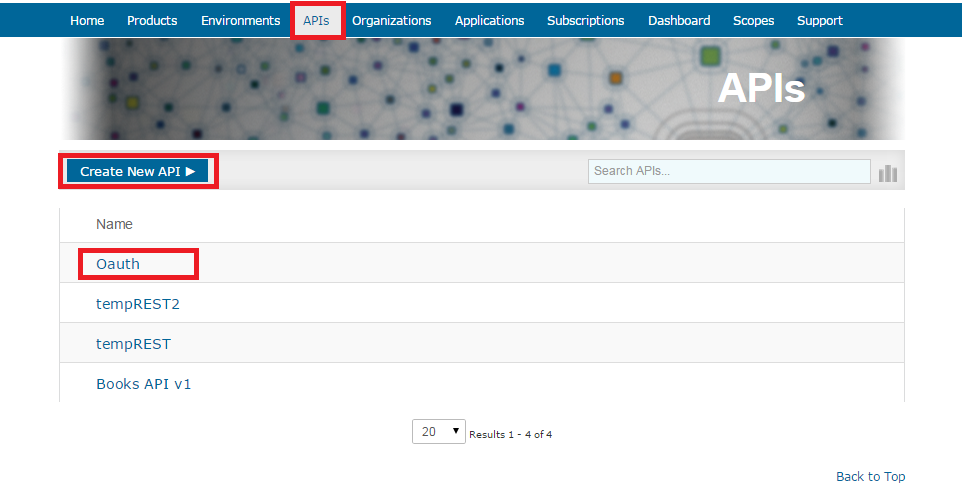


## API Manager config for OAuth client

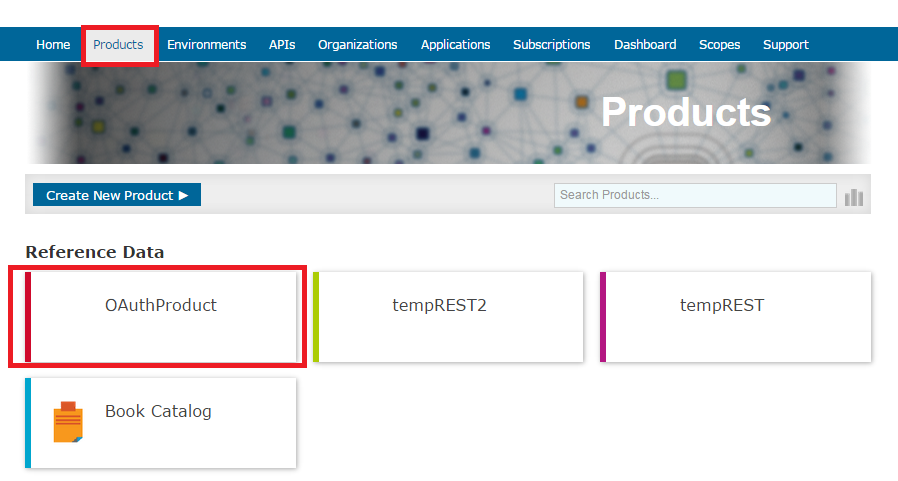
1. Login to the manager portal. (Please refer TIBCO APIX Manager user guide for manager configuration related help).
2. Create an environment.



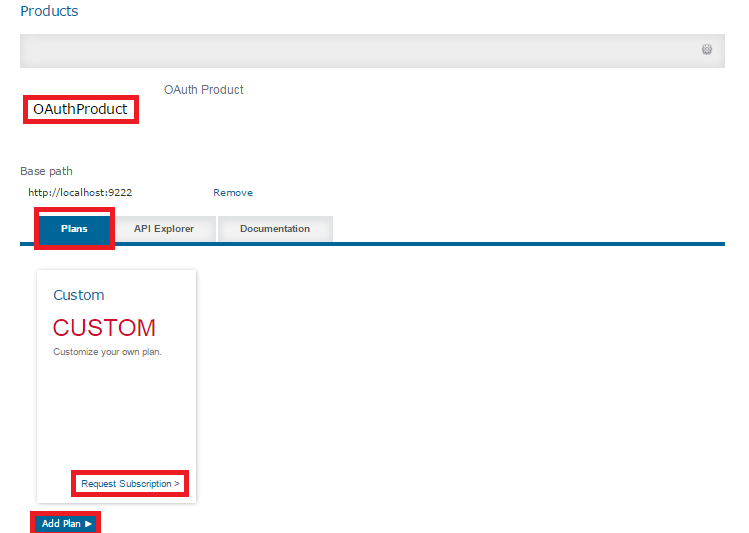
1. Create new API for oauth.



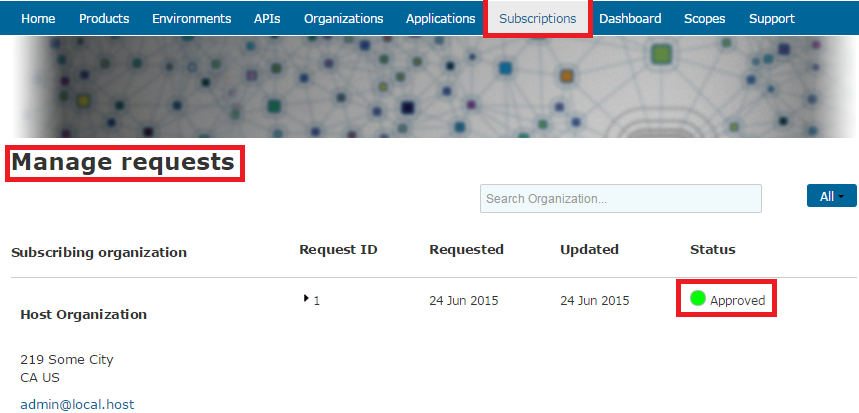
1. Configure a new product with the newly created API.



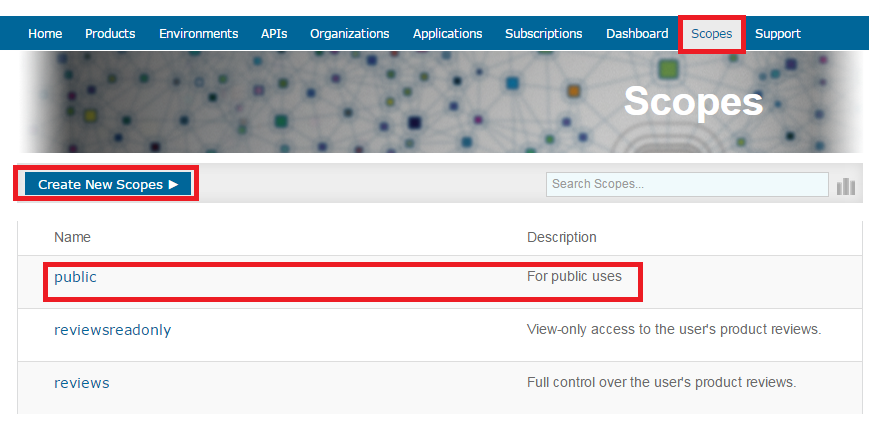
1. Click on the OAuthProduct and create a plan. Click on Request Subscription for that plan.



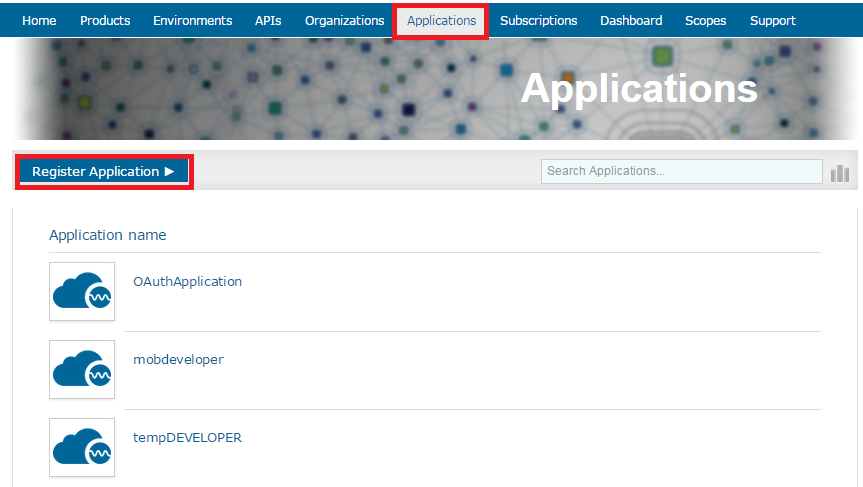
1. Select on Requests from the subscriptions dropdown list and approve the pending request by clicking ‘pending’.



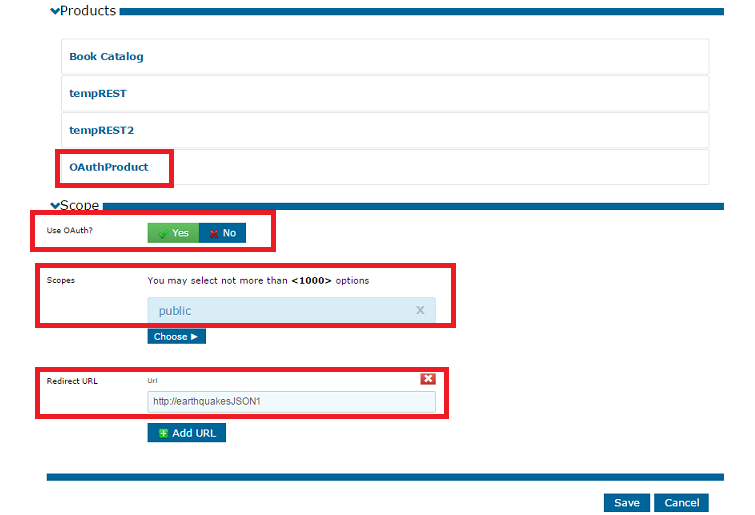
1. Create a public scope under scope tab.



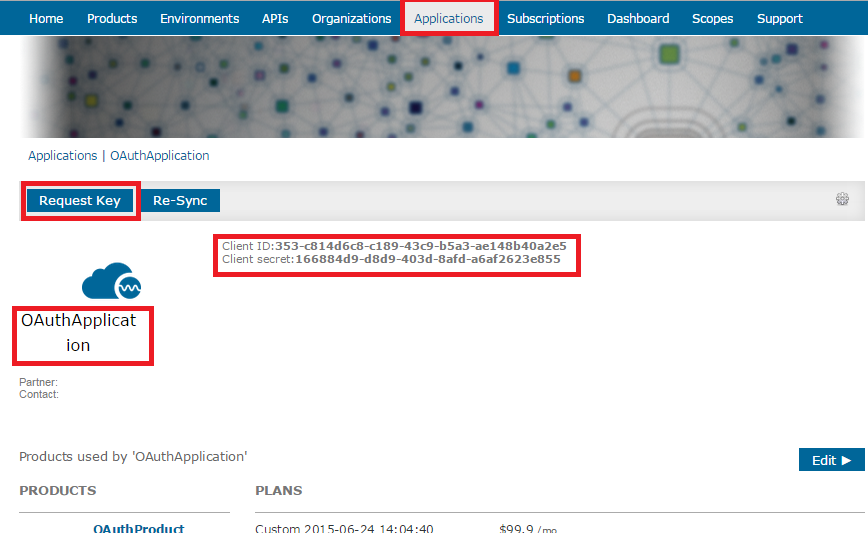
1. Register an application by selecting Register Application under the application tab.



1. OAuthProduct will be appeared after subscription for the product is approved. Select OAuthProduct and enable the plan by clicking on the enable check box. Select Yes for use OAuth, map scopes and provide the redirect url. Save the configuration.



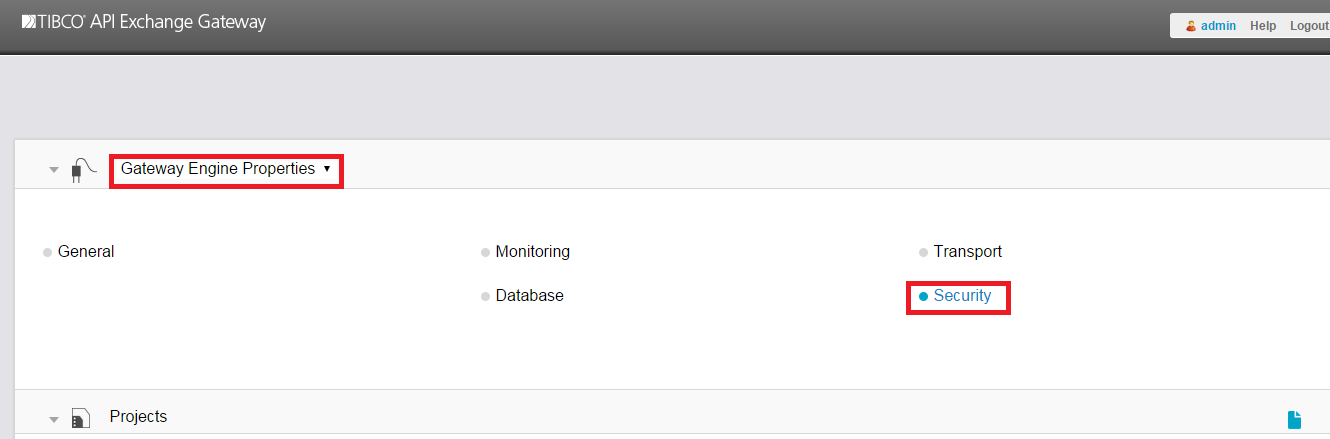
1. Select the OAuthApplication and request for the client key by clicking on Request key.



1. The generated client key is used to configure api key in TIBCO API gateway.

## API Gateway config for OAuth server

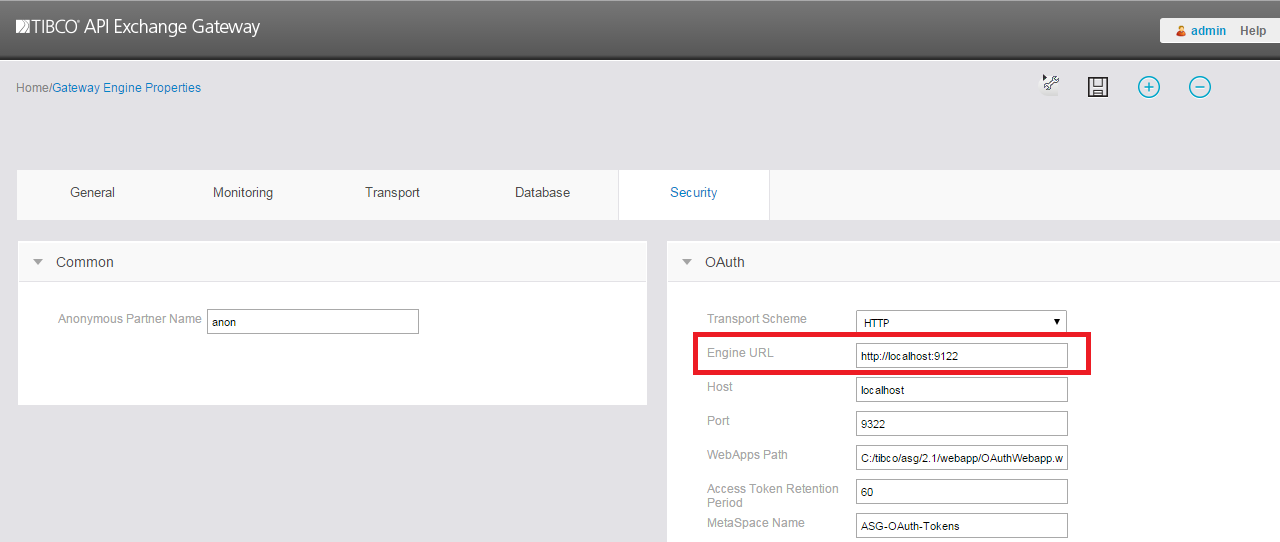
1. Select ‘Gateway engine properties’ in the home page of configUI and click on security tab.

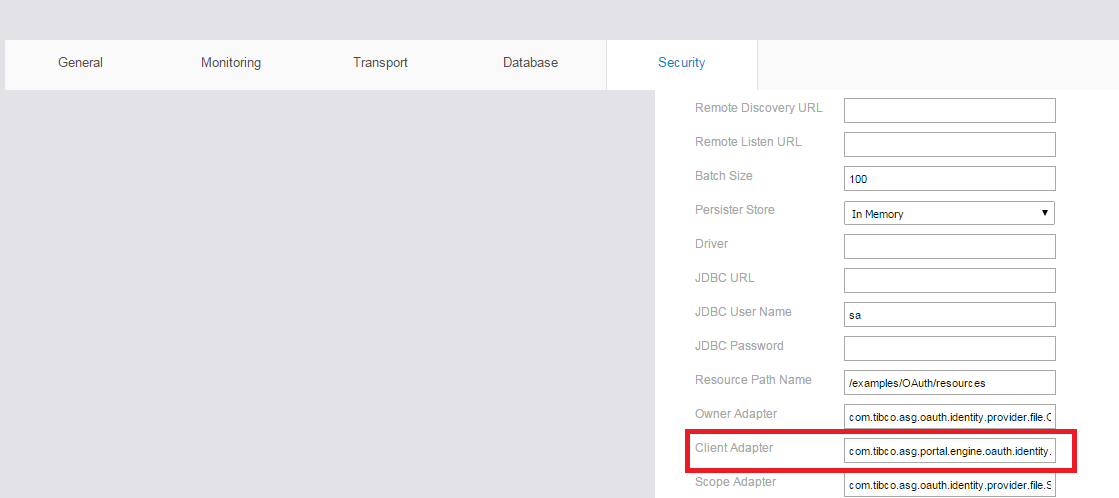


1. Expand the OAuth node to see the OAuth server properties. Update the engine url and client adapter as shown below:

Engine URL: <http://localhost:9122>

Client adapter: com.tibco.asg.portal.engine.oauth.identity.provider.ClientAdapterImpl

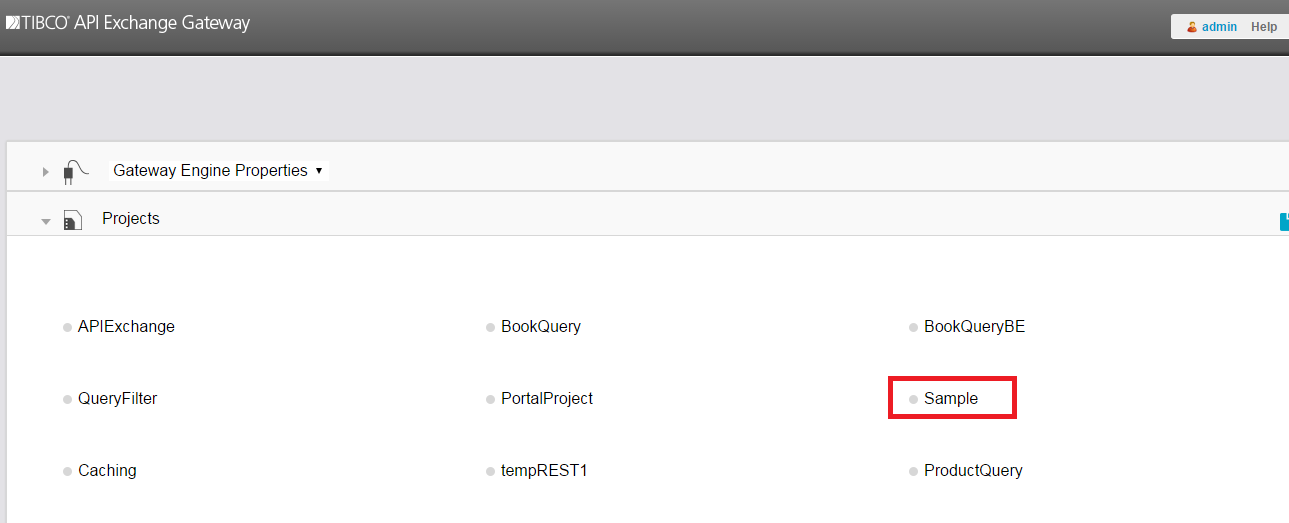




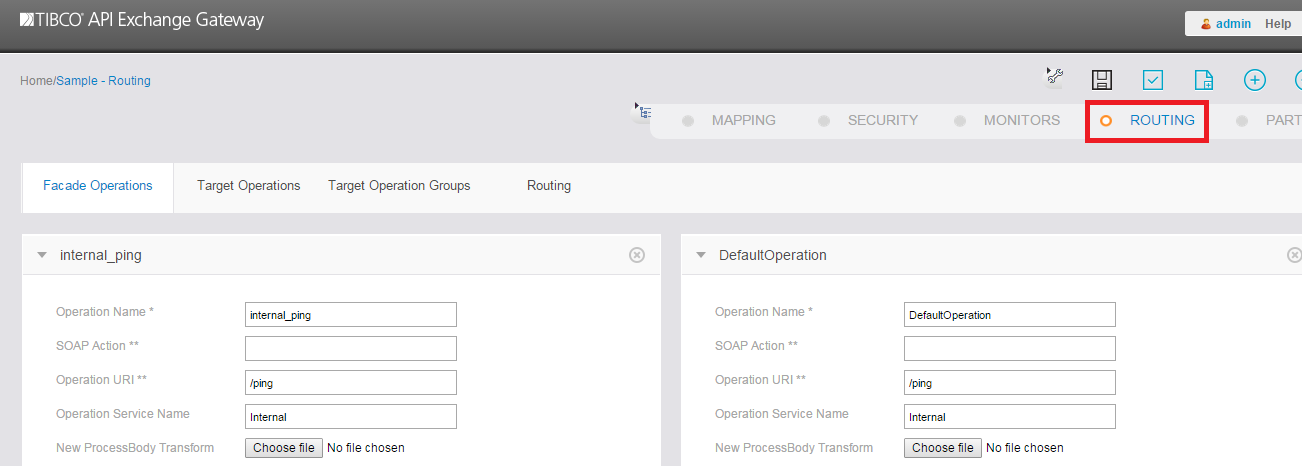
1. Click on save and return to the configUI home.

## Gateway config for Project

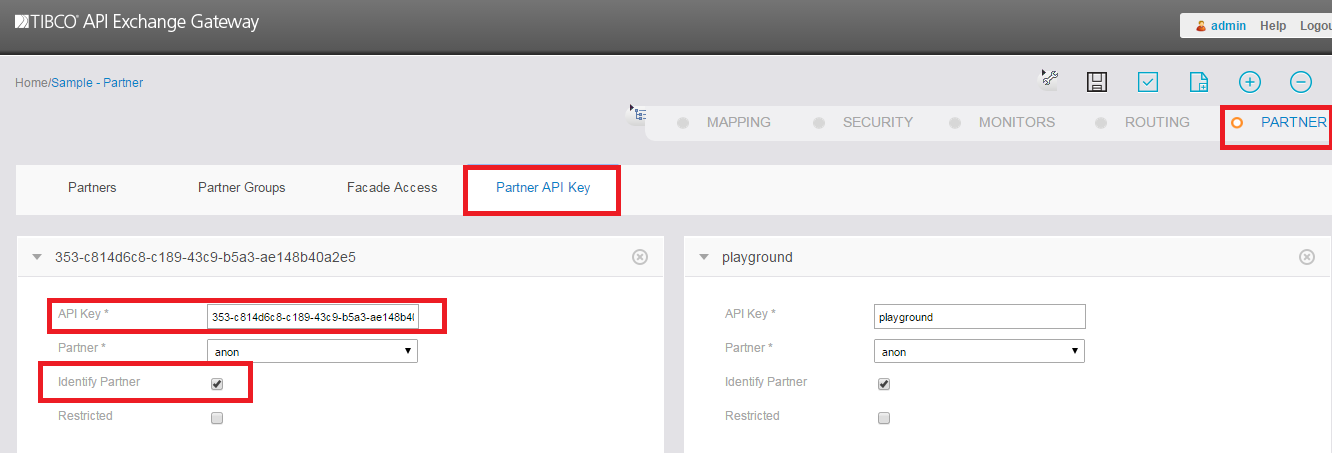
1. Create a sample project and click on the sample project.



1. Click on the routing tab and configure façade operation, target operation and routing. (Please refer TIBCO\_APIX\_User Guide for Routing configuration)



1. Click on the partner tab and configure patner and façade access.
2. Configure api key with the client Id obtained from TIBCO apix manager and map it to façade access.(please refer TIBCO\_APIX user guide for partner related configuration)



# OAuth flows supported in TIBCO

TIBCO supports below three OAuth flows:-

1. Authorization Code.
2. Password Credential.
3. Client Credential.

## Authorization Code

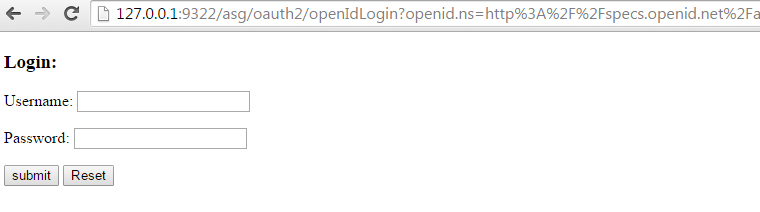
The authorization code is obtained by using an authorization server as an intermediary between the client and resource owner. Instead of requesting authorization directly from the resource owner, the client directs the resource owner to an authorization server, which in turn directs the resource owner back to the client with the authorization code.

Before directing the resource owner back to the client with the authorization code, the authorization server authenticates the resource owner and obtains authorization. Because the resource owner only authenticates with the authorization server, the resource owner's credentials are never shared with the client.

### Authorization Code Implementation

1. Get Authcode (Browser): Use below url. Client will redirect the owner to the auth server login page.

http://localhost:9322/asg/oauth2/authorize?response\_type=code&client\_id=353-c814d6c8-c189-43c9-b5a3-ae148b40a2e5&client\_secret=166884d9-d8d9-403d-8afd-a6af2623e855



1. Use the creds.

Username: john

Password: password

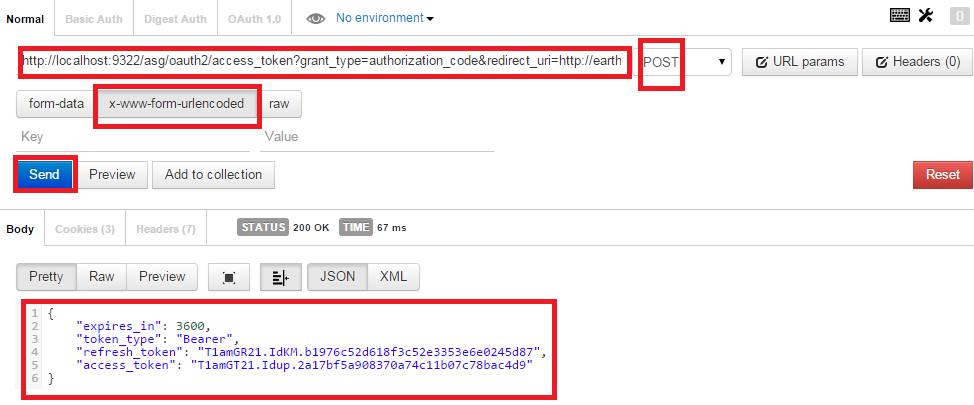
1. Auth server redirects back the resource owner to the client page with auth code.





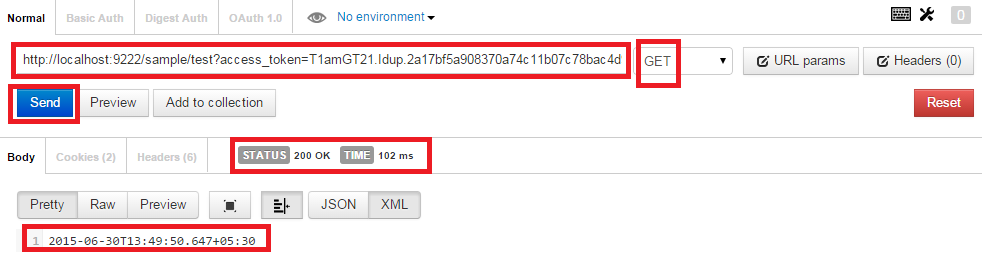
1. Post for the access token. (use postman/any other client tool)

http://localhost:9322/asg/oauth2/access\_token?grant\_type=authorization\_code&redirect\_uri=http://earthquakesJSON1&client\_id=353-c814d6c8-c189-43c9-b5a3-ae148b40a2e5&client\_secret=166884d9-d8d9-403d-8afd-a6af2623e855&code=6ef2edf76efee118b28512c176d9e869



1. ESB request (use Postman/any other client tool)

http://localhost:9222/sample/test?access\_token=T1amGT21.Idup.2a17bf5a908370a74c11b07c78bac4d9



## Password Credential

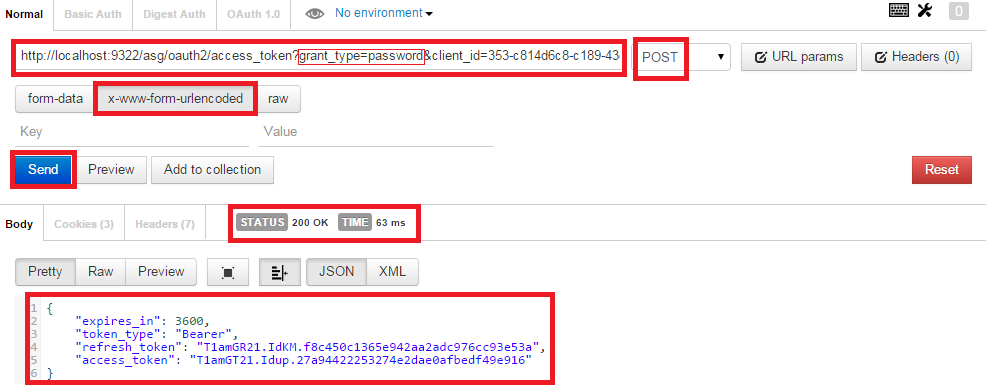
The resource owner password credentials (i.e., username and password) can be used directly as an authorization grant to obtain an access token. The credentials should only be used when there is a high degree of trust between the resource owner and the client (e.g., the client is part of the device operating system or a highly privileged application), and when other authorization grant types are not available (such as an authorization code).

Even though this grant type requires direct client access to the resource owner credentials, the resource owner credentials are used for a single request and are exchanged for an access token. This grant type can eliminate the need for the client to store the resource owner credentials for future use, by exchanging the credentials with a long-lived access token or refresh token.

### Password Credential Implementation

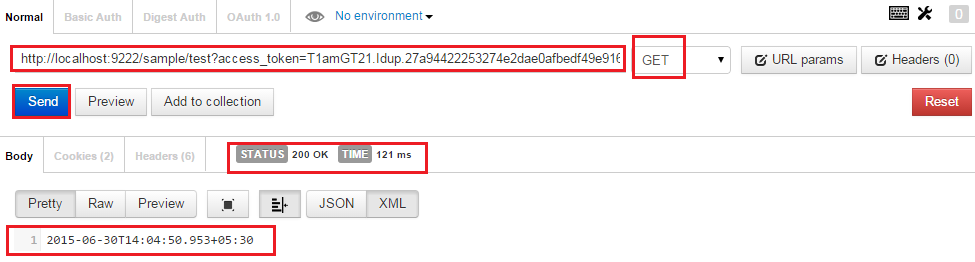
1. Post for access token (use postman/ any client tool)

<http://localhost:9322/asg/oauth2/access_token?grant_type=password&client_id=353-c814d6c8-c189-43c9-b5a3-ae148b40a2e5&client_secret=166884d9-d8d9-403d-8afd-a6af2623e855&username=john&password=password>



1. ESB request (use Postman/any other client tool)

http://localhost:9222/sample/test?access\_token=T1amGT21.Idup.27a94422253274e2dae0afbedf49e916



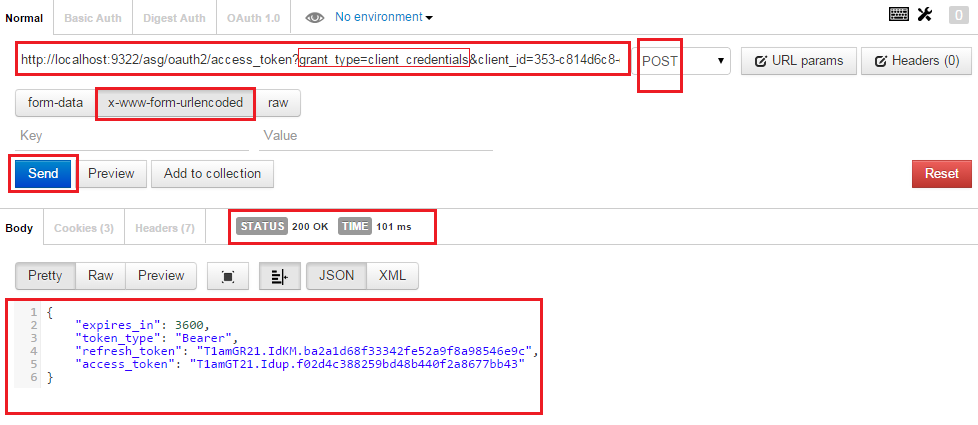
## Client Credential

The client credentials (or other forms of client authentication) can be used as an authorization grant when the authorization scope is limited to the protected resources under the control of the client, or to protected resources previously arranged with the authorization server. Client credentials are used as an authorization grant typically when the client is acting on its own behalf (the client is also the resource owner) or is requesting access to protected resources based on an authorization previously arranged with the authorization server.

### Client Credential Implementation

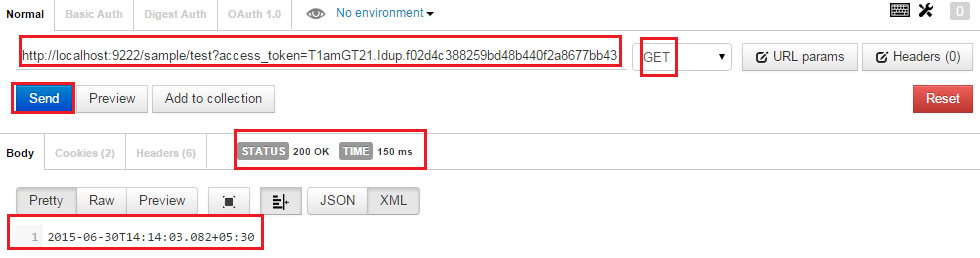
1. Post for access token (use postman/ any client tool).

<http://localhost:9322/asg/oauth2/access_token?grant_type=client_credentials&client_id=353-c814d6c8-c189-43c9-b5a3-ae148b40a2e5&client_secret=166884d9-d8d9-403d-8afd-a6af2623e855>



1. ESB request (use Postman/any other client tool).

http://localhost:9222/sample/test?access\_token=T1amGT21.Idup.f02d4c388259bd48b440f2a8677bb43



# Appendix A – Change Log

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Version Number*** | ***Changes Made*** | | | |
| Vx.y | Initial baseline created on 30-June-2015 | | | |
| Vx.y |  | | | |
| **Section No.** | **Changed By** | **Effective Date** | **Changes Effected** |
|  |  |  |  |
| Vx.y |  | | | |
| **Section No.** | **Changed By** | **Effective Date** | **Changes Effected** |
|  |  |  |  |
| Vx.y |  | | | |
| **Section No.** | **Changed By** | **Effective Date** | **Changes Effected** |
|  |  |  |  |