

Google Apps Connector Configuration for TWS 4.2

Document revisions

Version	Date	Comment
1.1	05/07/2015	Initial document
1.2	03/09/2015	New Google API
1.3	12/01/2016	New Client ID creation
1.4	12/08/2016	New Interface

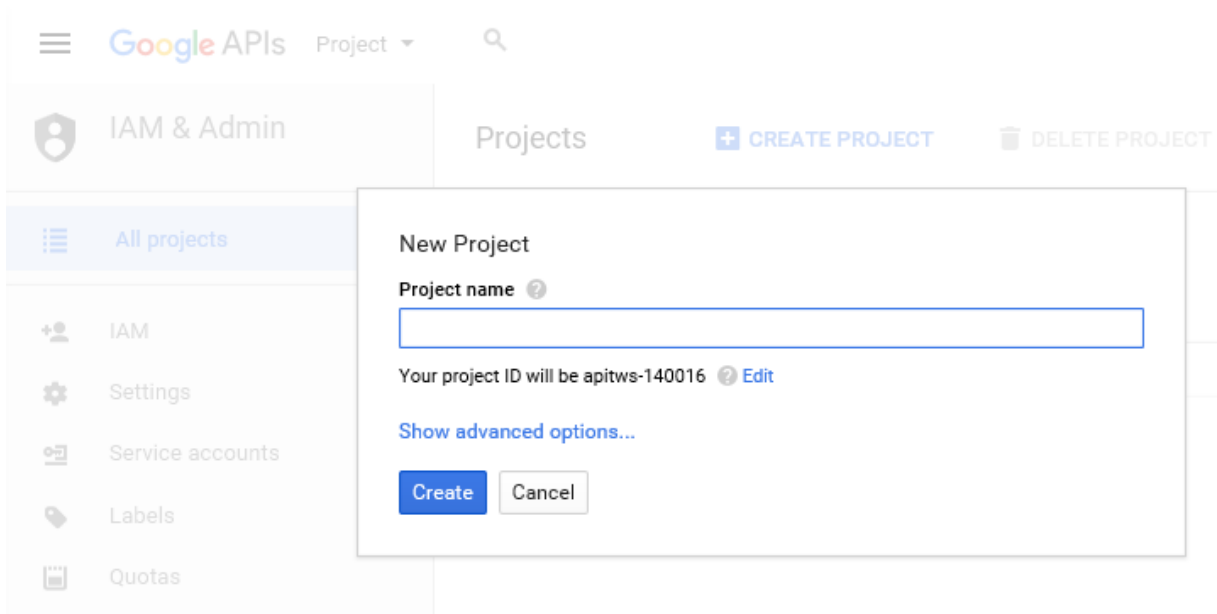
Summary

- Google Apps Account Setup 4
 - Creating a Google Apps new project 4
 - Creating a Service account key 4
 - Creating a Client ID associated with the service account key 6
 - Activate your APIs 8
 - Authorize APIs 9
- Configuring connectors in TWS 12
 - Prerequisites 12
 - Configuration 12
 - Creating a private directory 13
 - Creating a calendar connector 13

Google Apps Account Setup

Creating a Google Apps new project

Go to: <https://console.developers.google.com/project> and create a new project. Open this project and then go to “Credentials”.



Creating a Service account key

In “Credentials” and then “Credentials” tab, create the new credentials. Choose “Service account key”.

Credentials

[Credentials](#) OAuth consent screen Domain verification

API key
Identifies your project using a simple API key to check quota and access.
For APIs like Google Translate.

OAuth client ID
Requests user consent so your app can access the user's data.
For APIs like Google Calendar.

Service account key
Enables server-to-server, app-level authentication using robot accounts.
For use with Google Cloud APIs.

Help me choose
Asks a few questions to help you decide which type of credential to use.

[Create credentials](#)

Then choose "New Service Account" and enter a name for this account. Here is "apitws". Give the "Viewer" role for the "Project" type. Then select "P12" and do "Create".

Credentials

[←](#)

Create service account key

Service account
New service account

Service account name ?
apitws

Service account ID
apitws @apitwss.iam.g...

Key type
Downloads a file that contains the private key. Store the file securely as it can't be recovered if lost.

JSON
Recommended

P12
For backward compatibility with code using the P12 format

[Create](#) [Cancel](#)

Role ?
Viewer

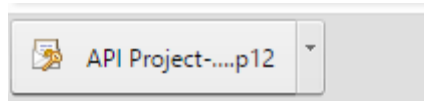
Selected Roles
✓ Viewer

Roles

- Project
- App Engine
- Billing
- Logging
- Other

Owner
Editor
✓ Viewer
Browser
Service Account Actor

A new private key is created. The browser will offer to download the *P12 file*. Save this file on your computer.



The service account key is created.

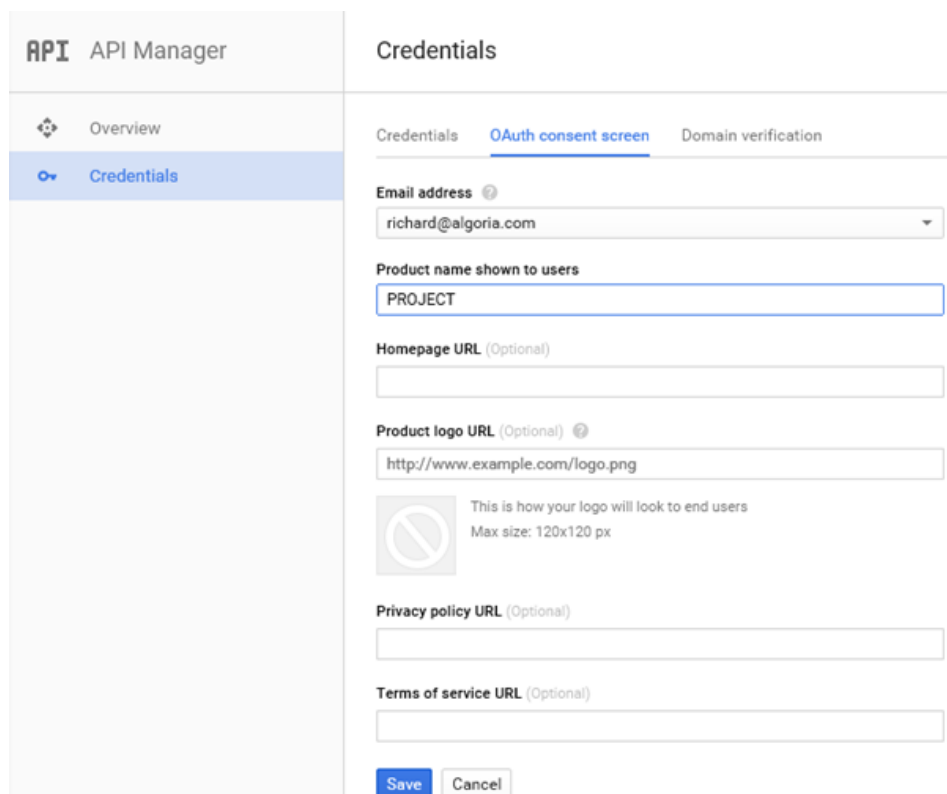
Service account keys

[Manage service accounts](#)

<input type="checkbox"/> ID	Creation date ▾	Service account
<input type="checkbox"/> 8a8d3022a02c36fd1f78c44e086a500f3253e593	Jan 11, 2016	apitws

Creating a Client ID associated with the service account key

Before creating the client ID, you must enter a product name. In the menu "Credentials" and the "OAuth consent Screen" tab, enter a product name as below:

A screenshot of the Google API Manager interface. On the left is a sidebar with 'API Manager' at the top and 'Overview' and 'Credentials' as menu items. The 'Credentials' menu item is selected. The main content area is titled 'Credentials' and has three tabs: 'Credentials', 'OAuth consent screen' (which is active), and 'Domain verification'. Below the tabs are several form fields: 'Email address' with a dropdown menu showing 'richard@algoria.com'; 'Product name shown to users' with a text input field containing 'PROJECT'; 'Homepage URL (Optional)' with an empty text input field; 'Product logo URL (Optional)' with a text input field containing 'http://www.example.com/logo.png'; a logo preview area showing a placeholder icon and the text 'This is how your logo will look to end users' and 'Max size: 120x120 px'; 'Privacy policy URL (Optional)' with an empty text input field; and 'Terms of service URL (Optional)' with an empty text input field. At the bottom of the form are 'Save' and 'Cancel' buttons.

Then in the "Credentials" menu and the "Credentials" tab, click "Manage Service Accounts" under "Service Account Keys". Select the account created and at the right the button allows you to edit the account.

Check the "Enable Google Apps Domain-wide Delegation" box and save.

Service Accounts + CREATE SERVICE ACCOUNT 🗑 DELETE + PERMISSIONS

Service accounts for project "apitwss"

A service account represents a Google Cloud service identity, such as code running on Compute Engine VMs, App Engine apps, or systems running outside Google. [Learn more](#)

Find a service account

<input type="checkbox"/>	Service account name ^	Service account ID	Key ID
<input type="checkbox"/>	apitws	apitws@apitwss.iam.gserviceaccount.com	a4b645a848f1b6aa41c5e580bc5e43785f1d694d

Edit service account

Service account name ?

Enable Google Apps Domain-wide Delegation
Grants a client access to all users' data on a Google Apps domain without manual authorization on their part. [Learn more](#)

Thus, a new client ID was created. This will be used to access the data and APIs. Remember that "Client ID".

Credentials

Credentials OAuth consent screen Domain verification

Create credentials to access your enabled APIs. [Refer to the API documentation](#) for details.

OAuth 2.0 client IDs

<input type="checkbox"/>	Name	Creation date	Type	Client ID
<input type="checkbox"/>	Client du compte de service apitws	Jan 11, 2016	Service account client	104373606319905538924

Activate your APIs

Open the "Overview" menu, "API Library" tab and in the "Google Apps APIs" section you will find the 2 APIs to enable.

API Library Enabled APIs (6)

Search all 100+ APIs

Popular APIs

- Google Cloud APIs**
 - Compute Engine API
 - BigQuery API
 - Cloud Storage API
 - Cloud Datastore API
 - Cloud Deployment Manager API
 - Cloud DNS API
 - More
- Google Maps APIs**
 - Google Maps Android API
 - Google Maps SDK for iOS
 - Google Maps JavaScript API
 - Google Maps Embed API
 - Google Places API for Android
 - Geocoding API
 - More
- Google Apps APIs**
 - Drive API
 - Drive SDK
 - Calendar API
 - Gmail API
 - Google Apps Marketplace SDK
 - Admin SDK
 - More
- Mobile APIs**
 - Cloud Messaging for Android
 - Google Play Game Services
 - Google Play Developer API
 - Google Places API for Android
- Social APIs**
 - Google+ API
 - Blogger API
 - Google+ Pages API
 - Google+ Domains API
- YouTube APIs**
 - YouTube Data API
 - YouTube Analytics API
- Advertising APIs**
 - AdSense Management API
 - DCM/DFA Reporting And Trafficking API
 - Ad Exchange Seller API
 - Ad Exchange Buyer API
 - DoubleClick Search API
 - Analytics API
- Other popular APIs**
 - Translate API
 - Custom Search API
 - URL Shortener API
 - PageSpeed Insights API
 - Fusion Tables API
 - Web Fonts Developer API

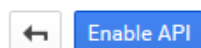
In the search box, you can search for "Contacts".

API Library Enabled APIs (6)

contacts Back to popular APIs

Name	Description
Google Contacts CardDAV API	An API to synchronize contacts.
Contacts API	The Google Contacts API lets you manage your contacts.

Select "Contacts API" and Enable API.



Contacts API

The Google Contacts API lets you manage your contacts.



[Learn more](#)

Do the same again for the « *Calendars* » API.

When finished, select the “*Enabled APIs*” menu to check that your APIs are active.

API Library [Enabled APIs \(6\)](#)

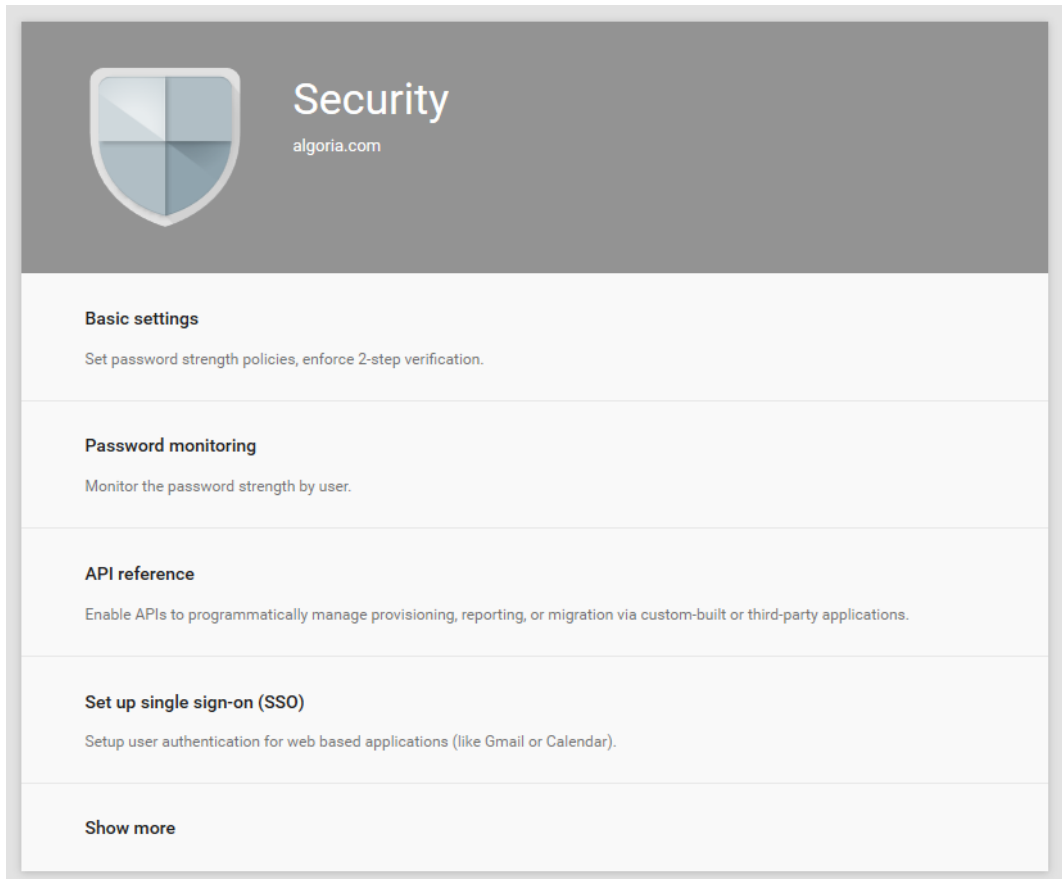
Some APIs are enabled automatically. You can disable them if you're not using their services.

API ^	Quota		
Books API	 0%	Disable	
Calendar API	 0%	Disable	
Contacts API	 0%	Disable	
Custom Search API	 0%	Disable	
Google Apps Reseller API	 0%	Disable	
Google Cloud Pub/Sub		Disable	

Authorize APIs

Go to: <https://www.google.fr/intx/fr/work/apps/business/> . Connect you to your domain Google Apps and select Administration console.

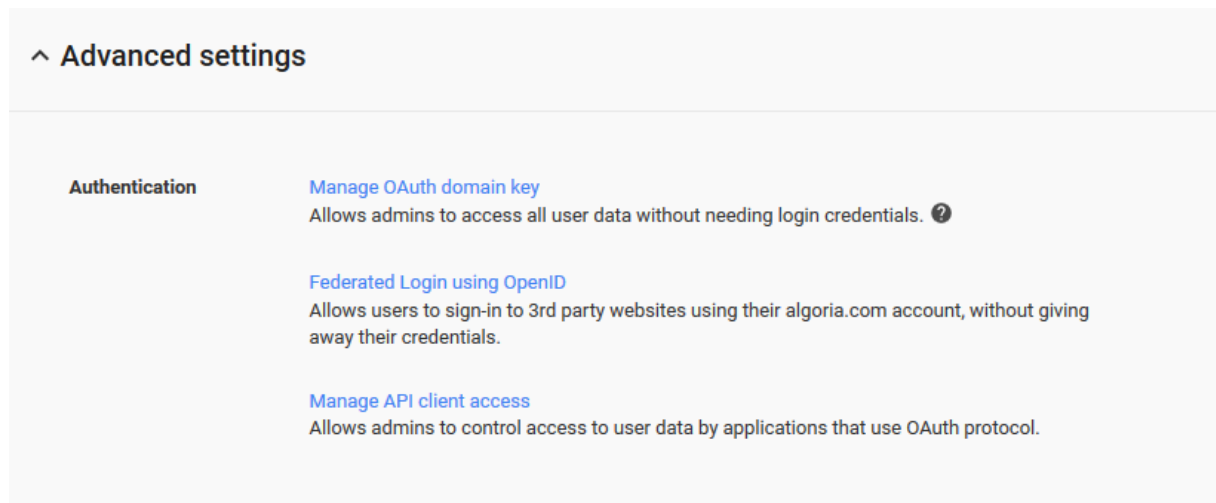
Open the “*Security*” menu and click on “*Show more*”.




The image shows a 'Security' settings page for 'algoria.com'. At the top left is a shield icon. The page is divided into several sections:

- Basic settings**: Set password strength policies, enforce 2-step verification.
- Password monitoring**: Monitor the password strength by user.
- API reference**: Enable APIs to programmatically manage provisioning, reporting, or migration via custom-built or third-party applications.
- Set up single sign-on (SSO)**: Setup user authentication for web based applications (like Gmail or Calendar).
- Show more**: A link to expand the settings.

Open “Advanced settings” and click “Manage API client access”.



The image shows the 'Advanced settings' page. It features a section titled 'Authentication' with the following options:

- Manage OAuth domain key**: Allows admins to access all user data without needing login credentials. 
- Federated Login using OpenID**: Allows users to sign-in to 3rd party websites using their algoria.com account, without giving away their credentials.
- Manage API client access**: Allows admins to control access to user data by applications that use OAuth protocol.

Now you must save the Web applications to access data services.

Find your "*CLIENT ID*" created previously and fill "*Customer Name*" (see chap. *Creating a customer ID associated with the service account key*). Then in the "*One or more API scopes*" complete the following URL separated by commas, as shown below:

`https://www.google.com/calendar/feeds/,https://www.google.com/m8/feeds/,https://www.googleapis.com/auth/calendar`

Manage API client access
Developers can register their web applications and other API clients with Google to enable access to data in Google services like Calendar. You can having to individually give consent or their passwords. [Learn more](#)

Authorized API clients The following API client domains are registered with Google and authorized

Client Name	One or More API Scopes
<input type="text" value="10437360631990553892"/> Example: www.example.com	<input type="text" value="https://www.googleapis.com/auth/calendar"/> <input type="button" value="Authorize"/> Example: http://www.google.com/calendar/feeds/ (comma-delimited)

Then click on "*Authorize*".

10437360631990553892	Calendar (Read/Write) https://www.google.com/calendar/feeds/
10437360631990553892	Contacts (Read/Write) https://www.google.com/m8/feeds/
10437360631990553892	Calendar (Read-Write) https://www.googleapis.com/auth/calendar

Your Google Apps account is configured correctly.

Configuring connectors in TWS

Prerequisites

TWS can communicate with Google Apps with:

- The email address previously configured in the Google service account
- The *P12* file
- The domain name you use in Google Apps

Configuration

Rename *P12* file this way: « *api-google-[Username].p12* »

The [Username] matches the name of the user present in the e-mail address of the Google Apps developer account.

<input type="checkbox"/>	Service account ^	Email address
<u>Ex:</u> <input type="checkbox"/>	apitws	apitws@tw-google-apps.com.iam.gserviceaccount.com

Email: apitws@tw-google-apps.com.iam.gserviceaccount.com

Name of the user: *apitws*

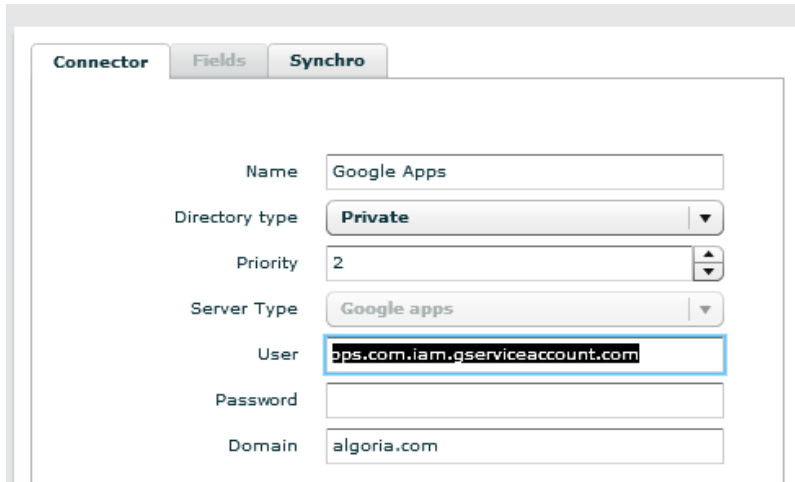
File name: ***api-google-apitws.p12***

Then copy the file in the directory [InstallTWS]\TWS4\TWS_Web\TWS_Config.

TWS can synchronize the directory and Google Apps calendars. To do that, the email address of the TWS user should match the email address of the Google Apps user.

Creating a private directory

In the TWS administration, create a new directory with *Google Apps* as the server type and *private* as the directory type and name it as you want.



The screenshot shows the 'Connector' configuration page in TWS administration. It has three tabs: 'Connector', 'Fields', and 'Synchro'. The 'Connector' tab is active. The form contains the following fields:

Name	Google Apps
Directory type	Private
Priority	2
Server Type	Google apps
User	ops.com.iam.gserviceaccount.com
Password	
Domain	algoria.com

In the *User* field, enter the full email address.

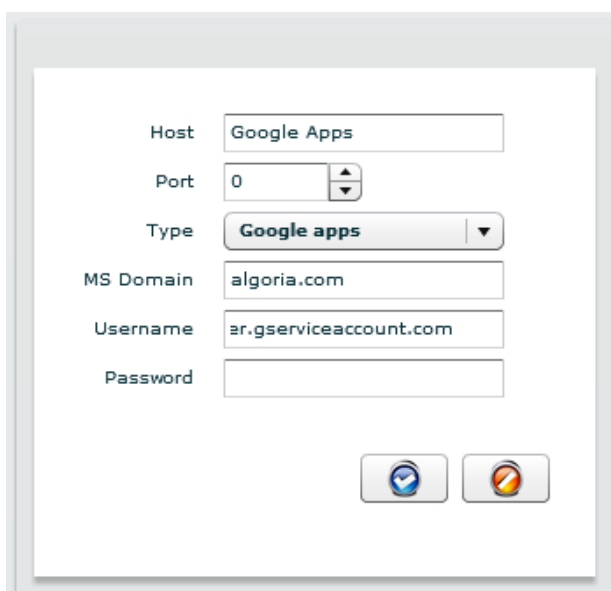
In the *Domain* field, enter the name of your domain.

The password field is not used.

Save and you can start synchronizing.

Creating a calendar connector

In the TWS administration, create a new collaboration connector of Google Apps.



The screenshot shows the configuration page for a collaboration connector. The form contains the following fields:

Host	Google Apps
Port	0
Type	Google apps
MS Domain	algoria.com
Username	ar.gserviceaccount.com
Password	

At the bottom of the form, there are two buttons: a blue checkmark button and a red and yellow circular button.

Choose the type Google Apps.

In the *User* field, enter the full email address.

In the *Domain* field, enter the name of your domain.

The password field is not used.

The Host field is not used. You can use this field to name the connector.