API document overview

Getting started

Authentication API

The Authentication API allows merchants and payment gateways to integrate the 3D Secure 2 flow into their eCommerce site. All Authentication APIs start from `/api/v1/auth/...` and follow RESTFul naming conventions. There are 3 main authentication flows available:

- **App-based** - Authentication during a transaction on a Consumer Device that originates from an App provided by a registered agent (merchant, digital wallet, etc). For example, an eCommerce transaction originating during a checkout process within a merchant's App on the Consumer Device.
- **Browser-based** - Authentication during a transaction on a Consumer Device or Computer that originates from a website utilising a browser. For example, an eCommerce transaction originating during a checkout process within a website.
- **3DS Requestor Initiated (3RI)** - Confirmation of account information with no direct cardholder present. For example, a subscription-based eCommerce merchant confirming that an account is still valid.

Field Conditions

The following standards are used to identify the conditions of each field.

- **Required** - Sender shall include the data element in the identified message; ActiveServer checks for data element presence and validates the data element contents.
- **Conditional** - Sender shall include the data element in the identified message if the conditional inclusion requirements are met; ActiveServer checks for data element presence and validates data element contents. When no data is to be sent for a conditional data element, the data element should be **absent**.
- **Optional** - Sender may include the data element in the identified message; ActiveServer validates the data element contents when present. When no data is to be sent for an optional data element, the data element should be **absent**.

The latest Authentication APIs are available from here.

Auth API Authentication

All authentication API endpoints are secured by X.509 authentication which requires the client and server to be mutually authenticated. This is performed by using a client certificate. **ActiveServer** provides two types of client certificate: Merchant client certificate for each merchant, or Master Auth API client certificate for **Business admin** users. The steps to make an Auth API call are:

1. Make sure the instance is activated by following this guide.
2. Download the client certificate:
    - For Merchant client certificate, download from the merchant details page on the administration UI interface.
    - For Master Auth API client certificate, download from the user profile page on the administration UI interface.
3. Download the CA certificate chain from the user profile page. The CA certificate is in `.pem` format with the first certificate being the server CA, followed by GPayments Intermediate CA and finally the GPayments Root CA.

> **Note**
>
> If you are using GPayments 3DS Requestor Demo code, the CA certificate chain is already included.

4. You can follow the Integration guide to setup a 3DS requestor from the beginning to call the authentication API. For specific details on using the client certificate method chosen from step 2, see the Demo 3DS Requestor Configuration.

> **Warning**
>
> You will not see the `DOWNLOAD` button if your instance is not activated.

Auth API reference

Channel	Process	End point (API v1)	End point (API v2)
BRW	InitAuth	/api/v1/auth/brw/init/{messageCategory}	/api/v2/auth/brw/init

Channel	Process	End point (API v1)	End point (API v2)
	Auth	/api/v1/auth/brw	`authUrl` from initAuth response
	Result	/api/v1/auth/brw/result	/api/v2/auth/brw/result
	ChallengeStatus	/api/v1/auth/challenge/status	/api/v2/auth/challenge/status
APP	Auth	/api/v1/auth/app/{messageCategory}	/api/v2/auth/app
	Result	/api/v1/auth/app/result	/api/v2/auth/app/result
3RI		/api/v1/auth/3ri/{messageCategory}	/api/v2/auth/3ri

You can check the details of the **API message** for each endpoint in the API document.

Administration API

The Administration API is available to perform select administration tasks such as managing merchants. All Admin APIs start from `/api/v1/admin/...` and follow RESTFul naming conventions. For example, `creating merchant` which is described here, can be performed by the POST API endpoint **`/api/v1/admin/merchants`**.

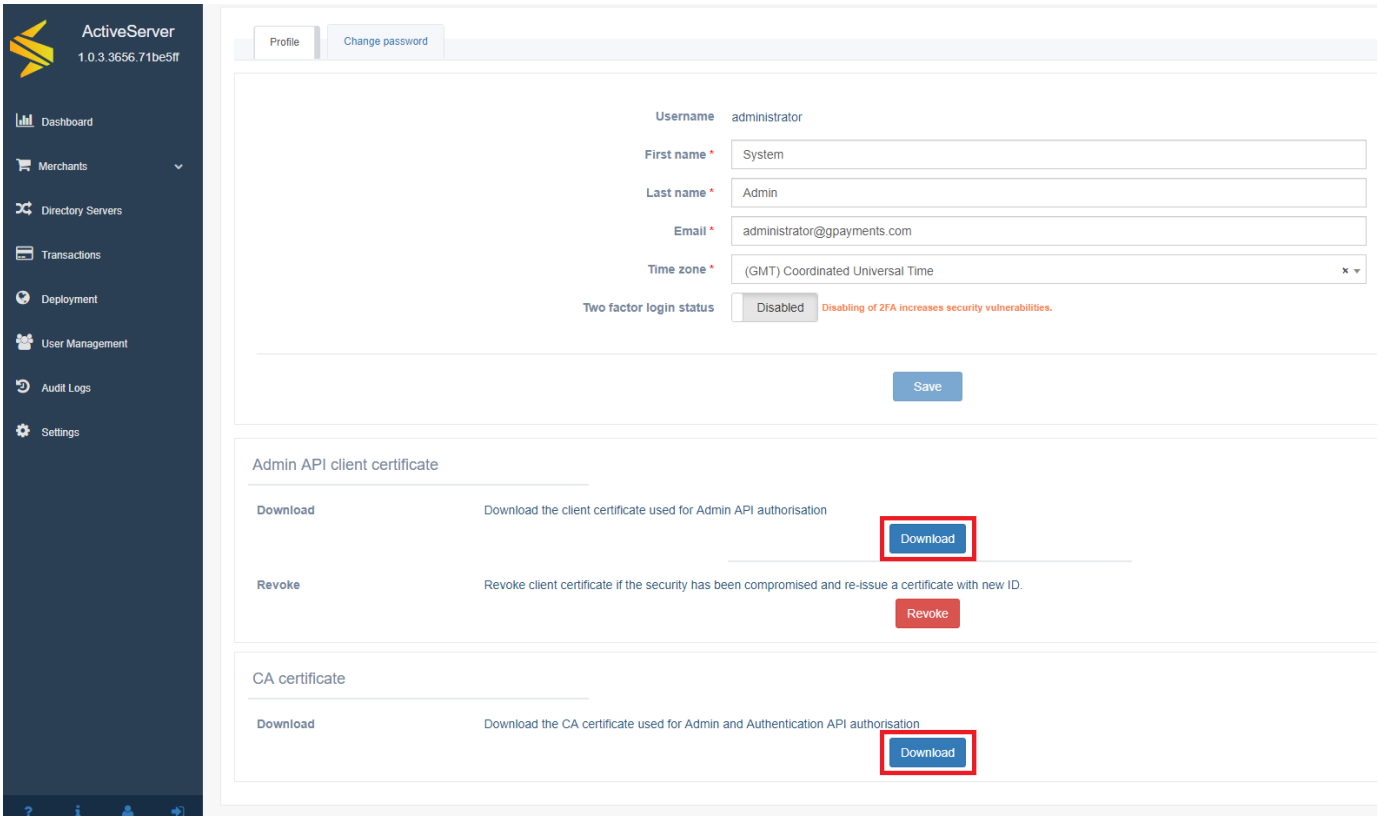
The latest Administration APIs are available from here.

Admin API Authentication

Admin API Authentication is performed the same as the Authentication API, using a client certificate. The steps to make an Admin API call are:

1. Make sure the instance is activated by following this guide.
2. Download the client certificate for the Administration API from the user profile page in the administration interface. This certificate is in `.p12` format.
3. Download the CA certificate chain certificate from the same page. The CA certificate is in `.pem` format with the first certificate being the server CA, followed by GPayments Intermediate CA and finally the GPayments Root CA.

4. Use the downloaded client certificate and CA bundle for API authentication by including it inside the HTTP client request. Refer to the HTTP client you are using on how to do this or you can follow the API quick start for testing.



Admin API Quick Start

To try out the API using cURL, you need to convert the X.509 certificate to PEM format by using **openssl** and using the following `curl` command.

1. Download and install openssl command line from https://www.openssl.org/.
2. Download the client certificate and CA certificate following the Admin API Authorisation section above.
3. Convert P12 to PEM format using OpenSSL
4. Perform the cURL request using your AS Admin API entry point. In this case we are getting a list of merchants from `/api/v1/admin/merchants` :

```
curl -k https://your.server.address:7443/api/v1/admin/merchants --cacert
cacerts.pem --cert crt.pem -v --key key_no_pass.pem
```

5. You should see the following cURL response:

```
< HTTP/1.1 200 OK
< Expires: 0
< Cache-Control: no-cache, no-store, max-age=0, must-revalidate
< X-XSS-Protection: 1; mode=block
< Pragma: no-cache
< X-Frame-Options: DENY
< Date: Mon, 27 May 2019 09:51:59 GMT
< X-Total-Count: 1
< Connection: keep-alive
< X-Content-Type-Options: nosniff
< Strict-Transport-Security: max-age=31536000 ; includeSubDomains
< Transfer-Encoding: chunked
< Content-Type: application/json;charset=UTF-8
< Link: </api/v1/admin/merchants?page=0&size=20>; rel="last",</api/v1/admin/
merchants?page=0&size=20>; rel="first"
<
* Connection #0 to host your.server.adress left intact
[{"bins":"40001 (Test
Acquirer)","merchantId":"123456789012345","name":"Test
Merchant","status":"ENABLED","merId":"48f3b030-ed1d-4f7f-
aa70-85cda288e0cb"}]%
```

Converting P12 to PEM using OpenSSL

Some HTTP client such as `curl` requires client certificate to be in `.pem` format. Following commands allow extracting the client certificate and its associated private key from `.p12` file into separate PEM formatted files.

1. Open up a terminal.
2. Extract the private key from the `.p12` file using the following command, entering the password when prompted:

```
openssl pkcs12 -in YOUR_P12_FILE_NAME.p12 -nocerts -out key.pem
```

3. Extract the certificate from the `.p12` file using the following command, entering the password when prompted:

```
openssl pkcs12 -in YOUR_P12_FILE_NAME.p12 -clcerts -nokeys -out crt.pem
```

4. Remove the passphrase from the private key

```
openssl rsa -in key.pem -out key_no_pass.pem
```