Admin API

The Frame Admin API service is based on standard REST HTTPS calls. All API endpoints require signed authentication via HTTP headers and role-based access control configured within Frame Console.

The typical workflow for an API call is as follows:

1. Authenticate with the Frame REST API to sign the HTTPS API request. You'll need to create a signature with the `client_id`, the `client_secret`, and a timestamp. The `client_id` and `client_secret` is obtained by a Frame administrator in Frame Console.
2. Make the HTTPS REST call to the Admin API endpoint using the `client_id`, timestamp, and the signature in the HTTP header.
3. The Frame REST API verifies the signature of the HTTPS request, verifies the requestor has sufficient authorization to make the request, and will then send the requested response.

Frame Admin API is served from the following URL:

```
https://api.console.nutanix.com/v1
```

In the next section, we will review how to properly configure Frame to authenticate and authorize Frame Admin API requests.

Authentication and Authorization

All REST API calls require signed requests for authentication of each 3rd party service request. To sign an API request call, a **client_id** and **client_secret** (obtained from Frame Console by an administrator) are required for signature creation and verification.

How to Provision API Credentials

1. Navigate to your Frame Admin Console as an Administrator.
2. Navigate to the **Customer** or **Organization** page and select **Users** from the left-hand menu.
3. Enable the **API** option and click **Save**. The **API** tab will appear on the Users Page. This tab lists all of the API providers that have been created at the Customer or Organization

level.

4. Go to the newly created **API** tab and click **Add API User**.

5. Enter a name for the API provider and choose the role and scope (customer, organization, or account) you would like to grant to the API key. You may assign more than one role and scope to the API provider by clicking on **Add a role**. Click **Add** at the bottom of the window once you have defined the name, role(s), and scope(s).

6. After successfully creating a new API Use, its credentials will be displayed. Record the `client_id` and `client_secret` to be used later by your API calls.

Making API Calls

In the last section, you generated a `client_id` and a `client_secret`.

The `client_id` is a unique string that Frame uses to identify and authorize the proper entity and API permissions.

The `client_secret` is an HMAC-SHA256 key used to generate the signature and is required by the signature verification process.

A signature is created by applying the HMAC-SHA256 algorithm to a string that is constructed from the `client_id` and a timestamp. The timestamp is the current time in seconds since epoch.

The proper steps for creating a signature are as follows:

1. Get the current timestamp.
2. Create a string by concatenating the timestamp with the `client_id` (timestamp + client_id).
3. Apply HMAC-SHA256 on the newly created string using the `client_secret`.

In order to verify the signature, our API also requires the `client_id`, and the timestamp in your request's HTTP headers. The `client_secret` should **NOT** be included.

Python 3 Example

Below is a simple example of a **Python 3** script that can be used to generate the signature needed for the API call and then the HTTPS request.

```
#!/bin/env python

import hashlib
import hmac
import time
import requests

# Client credentials
client_id = "<Your API Client ID goes here>"
client_secret = b"<Your API Client Secret goes here>"

# Create signature
timestamp = int(time.time())
to_sign = "%s%s" % (timestamp, client_id)
signature = hmac.new(client_secret, to_sign.encode('utf-8'), hashlib.sha256).hexdigest()

# Prepare http request headers
headers = { "X-Frame-ClientId": client_id, "X-Frame-Timestamp": str(timestamp), "X-Frame-
Signature": signature }

# Make request
r = requests.get("https://api.console.nutanix.com/v1/accounts", headers=headers)
accounts = r.json()

# Print accounts
for account in accounts:
  print(account)
```

API Endpoints

Lastly, each Frame entity (Customer, Organization, Account, Infrastructure) has its own set of endpoints.

Revision #7
Created 1 October 2025 04:55:16
Updated 19 January 2026 06:42:11 by Nikola Savic