API v1

API v1 (Deprecated)

> ProVision's APIv1 system has been replaced by APIv2, and is now considered deprecated.

It is highly recommended that any customer utilizing ProVision's API for custom scripting refer to APIv2 documentation instead, and consider upgrading existing APIv1 scripts to use APIv2.

However, APIv1 documentation will remain accessible from the links below in order to support legacy uses.

APIv1 - Overview

The 6Connect API is a RESTful API to access your data in the 6Connect tools. ReST relies on stateless, client-server communication, and is usually always implemented using the HTTP protocol (the 6Connect API uses HTTPS). It is a simple and lightweight alternative to Web Services and can implemented in nearly any language. The 6Connect API operates similarly to other popular ReST APIs you may have worked with, such as Facebook or Twitter. You simply create an HTTP GET or POST request according to our standard, send it to the server, and receive data back.

To learn more about request formatting, making requests, and the tools available, see "Making APIv1 Requests", below.
You can also get the PHP SDK for PHP libraries and sample code.

Here are some important details about our ReST implementation:

- The API only comes with the full 6Connect IPAM product. If you would like to upgrade to the full version, contact sales@6Connect.com.
- All transactions are over HTTPS (SSL - port 443) only. Any transaction not using SSL will be rejected, and you will have potentially exposed sensitive data.
- All API results are formatted in JSON. XML support is coming soon.
- All requests are either HTTP GET or POST requests. We suggest using POST if the length of data in the request is over 8KB.
- You can use any language you would like to query the API. We currently have an SDK for PHP. Looking at the sample code would probably help you implement it in any language though.

Making APIv1 Requests

API requests can be generated within the web UI by the API Request Generator, or generated programmatically in any language.

An API request looks like this: https://[your instance]/ex/api/v1/api.php?target=ipam&action=get&type=IP&mask=24

An API response is a JSON-encoded text string, and looks like this:


```
{"success":1, "message":"1 blocks found", "data":[{"id":"7539","oct1":"1","oct2":"2","oct3":"3","oct4":"0","
mask":"24","child1":null,"child2":null,"is_assigned":"0","is_swipped":"0", "is_aggregate":"1","custid":"
holding","last_updated_time":"2012-03-20 09:49:00","description":null,"parent":null,"rir":"ARIN","notes":"
2012-03-20 09:49:00","generic_code":null,"region":null,"vlan":null,"arin_net_id":null,"arin_cust_id":null,"
arin_swip_time":"0000-00-00 00:00:00","assigned_time":"2012-03-20 09:45:12"}]}
```


Instructions on decoding this return data can be found in the API endpoint documentation pages.

Using API Keys:
When using the API without pre-established authentication to ProVision, you must include both your API Key and a specially-prepared query hash parameter, like so:

> https://[your instance]/ex/api/v1/api.php?target=ipam&action=get&type=IP&mask=24&apiKey=00-TMHQV8CV2XZYABCD&hash=8jxj4IApYmgb5IZ0wBY4tFv+WilXb5JuVpjrwupyXQo=

API Keys can be generated from your ProVision instance by navigating to the Admin panel by using the gear icon in the upper right hand corner, then navigating to the API tab. The API tab will present the API authentication information in the following format:

> API Key: 00-TMHQV8CV2XZYABCD

Secret Key: 6e04e5822ce10fecc8947dedxc46878

The secret key serves as an API password and is used in the creation of the API Authentication hash. The formula for creating a API query hash from an API query and a Secret Key is the following:

Hash = Base64Encode(Sha256HMACHash (QueryString, SecretKey))

In PHP, this would be performed with the following line of code:

$hash = *base64_encode(hash_hmac('sha256', $_SERVER['QUERY_STRING'], $secretKey, TRUE));*

Because the hash function is computed based on the query string, you must calculate a unique hash for every API request!

Example

Lets say you wanted to create a hash for the following API request:

https://[your instance]/api/v1/api.php?target=ipam&action=get&type=IP&mask=24

And that your API Key and Secret Key are as follows:

API Key: 00-TMHQV8CV2XZYABCD

Secret Key: 6e04e5822ce10fecc8947dedxc46878

The first step is to append your API Key to the URL. The API Key indicates which user is executing the API query.

https://[your instance]/api/v1/api.php?target=ipam&action=get&type=IP&mask=24&apiKey=00-TMHQV8CV2XZYABCD

The next step is to isolate the Query String from the request URL. The Query String is everything which follows the question mark. So,

Query String: target=ipam&action=get&type=IP&mask=24&apiKey=00-TMHQV8CV2XZYABCD

The next step is to calculate the SHA256 hash of this string with your Secret Key. In PHP, this would be:

$sha256 = hash_hmac('sha256', "target=ipam&action=get&type=IP&mask=24&apiKey=00-TMHQV8CV2XZYABCD", "48b278ec873bda4738923dbc467f8669", TRUE);

As this value has been 256-bit hashed, it will contain many unprintable characters. The solution to this is to encode it in base 64 for transport. Again, in PHP:

$hash = *base64_encode($sha256);*

Calculating it out yields the completed hash:

$hash = yneSFMyxPPe+3W4IOkVp50K3VStatBcRRak+2ygDUWQ=

The calculated hash can then be appended to the full API Query URL to form a completed request:

https://[your instance]/api/v1/api.php?target=ipam&action=get&type=IP&mask=24&apiKey=00-TMHQV8CV2XZYABCD&hash=yneSFMyxPPe+3W4IOkVp50K3VStatBcRRak+2ygDUWQ=

A Note on False Positives

ProVision utilizes several possible authentication schemes of which key-based API authentication is only one. Session-based, username/password authentication is used for the majority of user interaction with the ProVision front end. Because session information is stored in browsers cookies, a browser can be authenticated to execute API commands as long as the session is active.

Unfortunately, this can lead to confusion when using a machine-based API as the user might use an authenticated browser session to test API-Key based API queries. These queries will always succeed regardless of whether the API Query Hash was calculated correctly as the system defaults to Session-based authentication when it is available.

To ensure that session-based authentication is not polluting your API-Key based testing, always use a separate browser which is not logged in to your ProVision instance to test API queries.

Other Languages

The 6Connect API can be used in just about any scripting or programming language. We have a PHP SDK that provides example code, and several useful functions for interacting with the API. Even if you don't want to use PHP, the samples will help you create code in other languages

APIv1 - Getting Started with the SDK

The 6connect API allows you to access to data and functions of the 6connect web tools. The SDK for PHP or Python will help you get this setup quickly by outlining the requirements, prerequisites and provide sample code.

SDK for PHP

Prerequisites

The API only comes with a licensed 6connect ProVision application. If you would like access to a ProVision license please contact sales@6connect.com.

Create Your API Credentials

To use the 6connect SDK for PHP, you will need a 6connect API Key and Secret Key.

To create your API Key and Secret Key:

- Log into your 6connect instance (hosted or local)
- Click on the Admin icon, and go into the Administration section.
- Click on the "API" tab.
- Select the user from the drop down you want to enable API access for, and click "Generate Keys".
- The API Key and the Secret Key will now appear directly below that.

*Note that generating a new API will automatically revoke an older API Key.

> 6connect recommends that each user accessing the API have their own API key configured. However, you can alternatively setup API users by functionality or roles. While the platform is flexible, you should follow your organizations security policies.

Important!

> Your Secret Key is a secret! Only you and 6connect should ever know this information. It is important to keep it confidential to protect the privacy of your data. Store it securely and never share this key with other users or place it on other systems. Never include the secret key in requests to 6connect, support requests to 6connect, and never e-mail it to anyone. Do not share it outside your organization. No one who legitimately represents 6connect will ever ask you for your Secret Key.

Requirements

Aside from following the prerequisites, you will need a basic understanding of object oriented programming in PHP and the right tools installed on your system to use the API.

Minimum Requirements

- PHP 5.5 or newer.
- PHP JSON and PCRE extensions (XML will be coming soon).
- Curl PHP extension compiled with OpenSSL libraries. Click here for more information on curl.

If you aren't sure what is running on your system, you can create a php page on your system and call phpinfo() and view this page in a browser, or run php -i on the command line.

Install the SDK

Download the file "6connect_ProVision_PHP_SDK_5_1_4.tar" from the APIv1 SDK page.

Configure the SDK Security Credentials

- Extract the zipped tar file to a directory.

- Open the api-config.php located in the downloaded SDK files.
- Read through the file and place in your instance name (or path for local installs), API Key and Secret Key information as specified.
- Make sure all files are in the same directory (the core class looks for a config file in the same directory by default).
- Run the sample code api-examples.php!

Important!

You must setup user API access before running the sample. See the previous section "Create Your API Credentials" for more information.

SDK for Python

Install the SDK

Download the file "6c-api-examples-python.zip" from the APIv1 SDK page.

Configure the SDK Security Credentials

- Extract the zipped tar file to a directory.
- Open the apiclient.py located in the downloaded SDK files.
- Read through the file and place in your instance name (or path for local installs), API Key and Secret Key information as specified.
- Make sure all files are in the same directory (the core class looks for a config file in the same directory by default).
- Run the sample code api-examples.php!

Important!

You must setup user API access before running the sample. See the previous section "Create Your API Credentials" for more information.

APIv1 Documentation (Deprecated)

- API Module - Admin and Audit
- API Module - DHCP
- API Module - DNS
- API Module - IPAM
- API Module - LIR
- API Module - Peering
- API Module - Resource
- API Module - VLAN
- APIv1 SDK
- How Do I...