XIQ-C API – Postman Runner Starters Guide

30 May 2023
Version 1.0

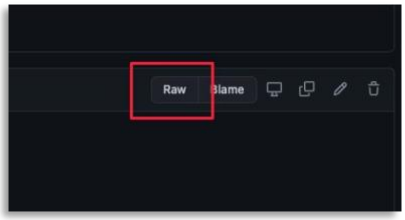
Table of Contents

Introduction

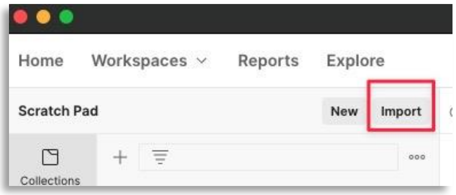
This document describes creating an environment in Postman to use the ExtremeCloudIQ Controller (XIQ-C) Postman API collection.

Postman Preparation

Download the Postman collection and environment variables files from the Extreme Networks GitHub page. In GitHub, you can select the file to open it in GitHub and then select the **raw** button. This will allow you to download the file right from the browser. For Mac use cmd+s, for PC use ctrl+s.



To import the files, in Postman click the import button and select your files. Postman will automatically recognize Postman data, and confirm the name, format, and what the file will import as. Click Import to bring your data into Postman.

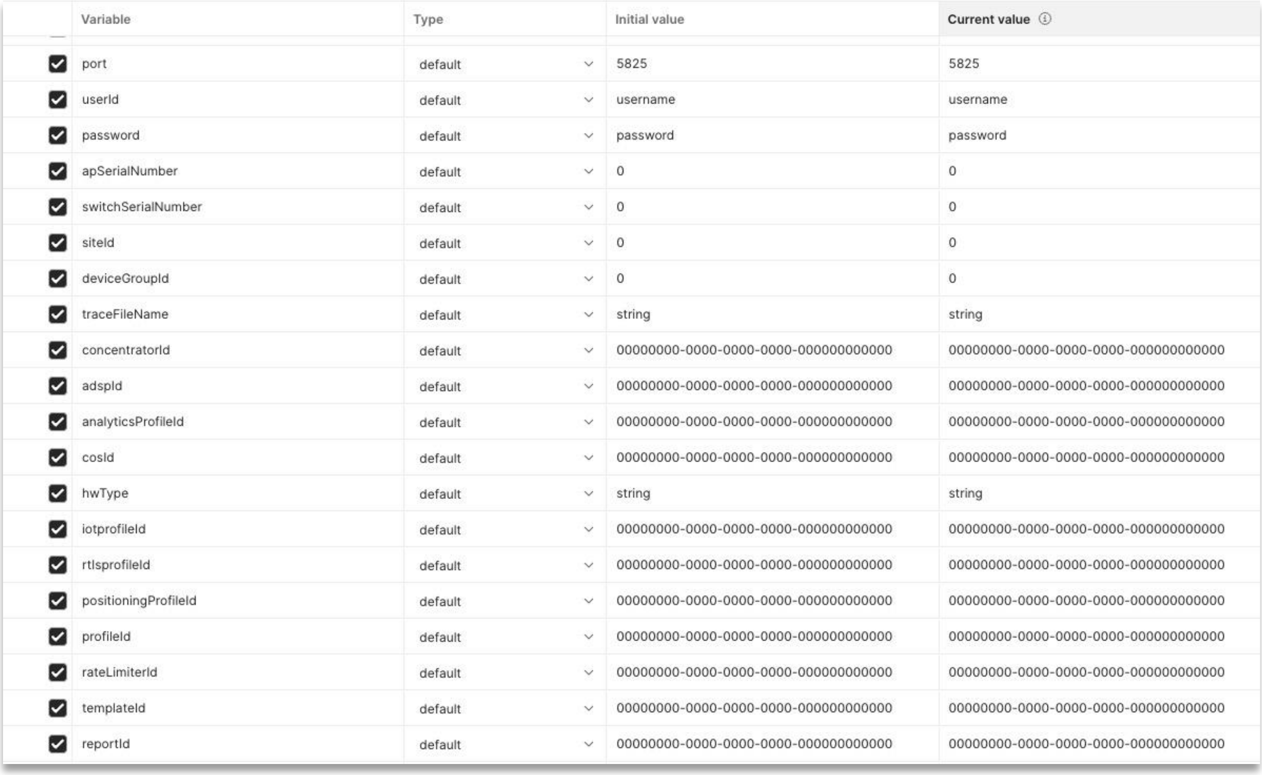


Setup Environment Variables

This Collection leverages Postman Environment variables. In the upper right corner of the Postman app, click the dropdown arrow and select the XIQ Environment that you imported. With that selected, click the "**eye**" icon to the right of the selection box, and click the **edit** button.

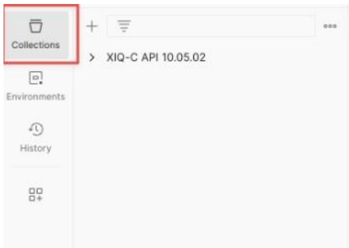
You can then change the "**Current Values**" for any of the variables here. Enter your ExtremeCloudIQ Controller IP address or FQDN, the port used for the controller, your username, and password in the current value field. You can also enter the IDs and serial numbers you want to use.

There are multiple IDs available to use. This makes the collection easier to use as you will not have to modify every API call with device IDs, topology IDs, etc. Instead, you can leverage the environment variables.



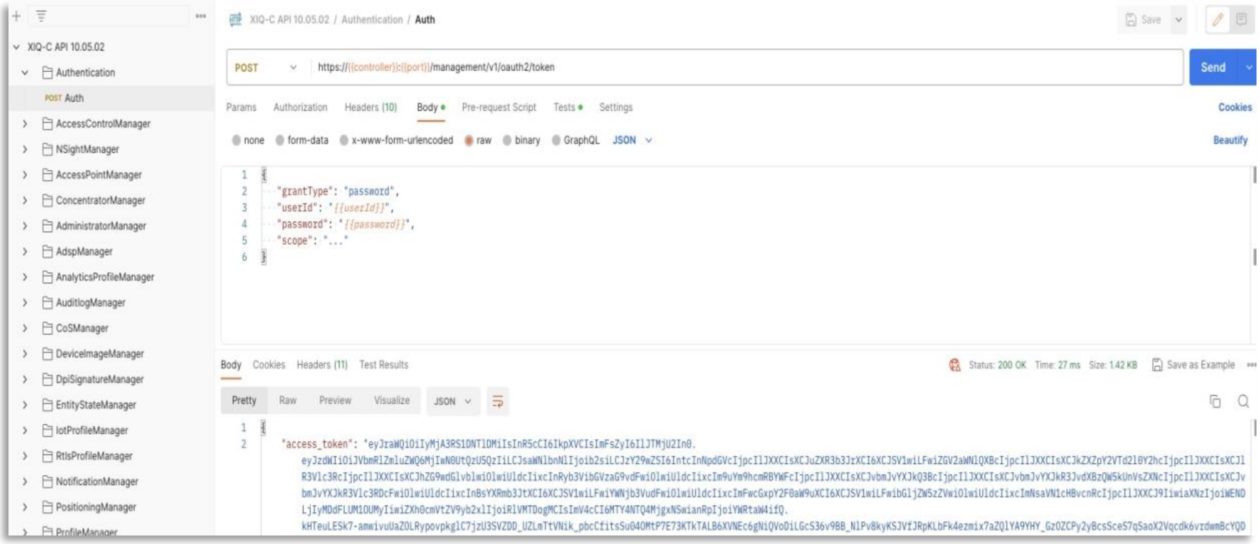
	Variable	Type	Initial value	Current value
☑	port	default	5825	5825
☑	userId	default	username	username
☑	password	default	password	password
☑	apSerialNumber	default	0	0
☑	switchSerialNumber	default	0	0
☑	siteId	default	0	0
☑	deviceGroupId	default	0	0
☑	traceFileName	default	string	string
☑	concentratorId	default	00000000-0000-0000-0000-000000000000	00000000-0000-0000-0000-000000000000
☑	adspId	default	00000000-0000-0000-0000-000000000000	00000000-0000-0000-0000-000000000000
☑	analyticsProfileId	default	00000000-0000-0000-0000-000000000000	00000000-0000-0000-0000-000000000000
☑	cosId	default	00000000-0000-0000-0000-000000000000	00000000-0000-0000-0000-000000000000
☑	hwType	default	string	string
☑	iotprofileId	default	00000000-0000-0000-0000-000000000000	00000000-0000-0000-0000-000000000000
☑	rtlsprofileId	default	00000000-0000-0000-0000-000000000000	00000000-0000-0000-0000-000000000000
☑	positioningProfileId	default	00000000-0000-0000-0000-000000000000	00000000-0000-0000-0000-000000000000
☑	profileId	default	00000000-0000-0000-0000-000000000000	00000000-0000-0000-0000-000000000000
☑	rateLimiterId	default	00000000-0000-0000-0000-000000000000	00000000-0000-0000-0000-000000000000
☑	templateId	default	00000000-0000-0000-0000-000000000000	00000000-0000-0000-0000-000000000000
☑	reportId	default	00000000-0000-0000-0000-000000000000	00000000-0000-0000-0000-000000000000

Verify Environment



On the left, make sure Collections in selected. You should see the name of the collection that you imported. Click the little arrow next to the name to drop down the XIQ API map.

In the *Authentication* category, select the *Auth* request. You can see the **username** and **password** variables set in the **Body**. Variables will be surrounded by {{ }}. After clicking the **send** button, you will receive an access token from XIQ-C in the response body.



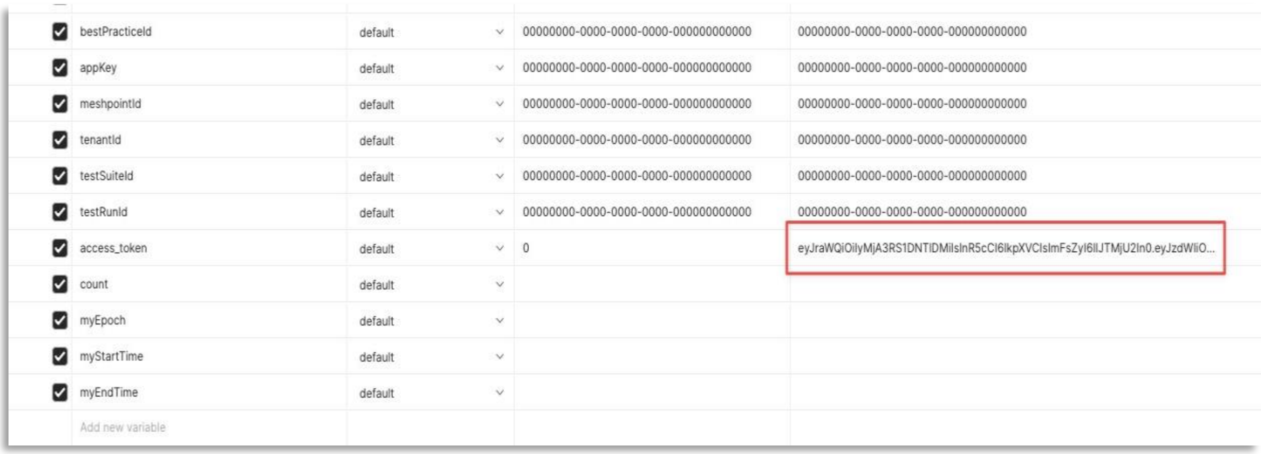
POST https://{{controller}}:{{port}}/management/v1/oauth2/token

```
{
  "grantType": "password",
  "userId": "{{userId}}",
  "password": "{{password}}",
  "scope": "..."
}
```

Status: 200 OK Time: 27 ms Size: 1.42 KB

```
{
    "access_token": "eyJraWQiOiIyMjA3RS1DNTlDMiIsInR5cCI6IkpXVCIsImFsZyI6IlJTMjU2In0.
        eyJzdWIiOiJVbmRlZmluZWQ6MjIwN0UtQzU5QzIiLCJsaWNlbnNlIjoib2siLCJzY29wZSI6IntcInNpdGVcIjpcIlJXXCIsXCJuZXR3b3JrXCI6XCJSV1wiLFwiZGV2aWNlQXBcIjpcIlJXXCIsXCJkZXZpY2VTd2l0Y2hcIjpcIlJXXCIsXCJl
        R3Vlc3RcIjpcIlJXXCIsXCJhZG9wdGlvblwiOlwiUldcIixcInRyb3VibGVzaG9vdFwiOlwiUldcIixcIm9uYm9hcmRBYWFcIjpcIlJXXCIsXCJvbmJvYXJkQ3BcIjpcIlJXXCIsXCJvbmJvYXJkR3JvdXBzQW5kUnVsZXNcIjpcIlJXXCIsXCJv
        bmJvYXJkR3Vlc3RDcFwiOlwiUldcIixcInBsYXRmb3JtXCI6XCJSV1wiLFwiYWNjb3VudFwiOlwiUldcIixcImFwcGxpY2F0aW9uXCI6XCJSV1wiLFwibGljZW5zZVwiOlwiUldcIixcImNsaVN1cHBvcnRcIjpcIlJXXCJ9IiwiaXNzIjoiWEND
        LjIyMDdFLUM1OUMyIiwiZXhwcmVtZV9yb2xlIjoiRlVMTDogMCIsImV4cCI6MTY4NTQ4MjgxNSwianRpIjoiYWRtaW4ifQ.
        kHTeuLESk7-amwivuUaZOLRypovpkglC7jzU3SVZDD_UZLmTtVNik_pbcCfitsSu04OMtP7E73KTkTALB6XVNEc6gNiQVoDiLGcS36v9BB_NlPv8kyKSJVfJRpKLbFk4ezmix7aZQlYA9YHY_GzOZCPy2yBcsSceS7qSaoX2Vqcdk6vrdwmBcYQD
```

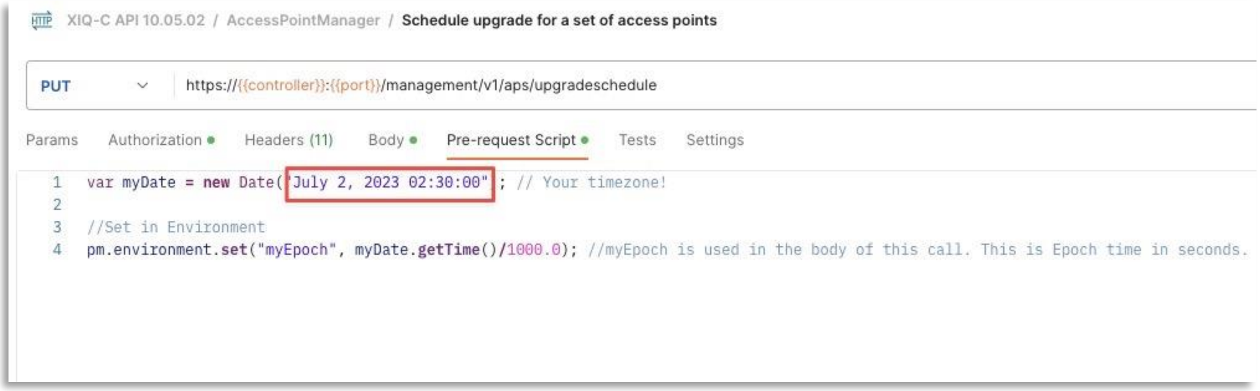
The access token is automatically saved to the environment variables as *access_token* and will be used by all the other API requests in the whole collection.



☑	bestPracticeId	default	00000000-0000-0000-0000-000000000000	00000000-0000-0000-0000-000000000000
☑	appKey	default	00000000-0000-0000-0000-000000000000	00000000-0000-0000-0000-000000000000
☑	meshpointId	default	00000000-0000-0000-0000-000000000000	00000000-0000-0000-0000-000000000000
☑	tenantId	default	00000000-0000-0000-0000-000000000000	00000000-0000-0000-0000-000000000000
☑	testSuiteId	default	00000000-0000-0000-0000-000000000000	00000000-0000-0000-0000-000000000000
☑	testRunId	default	00000000-0000-0000-0000-000000000000	00000000-0000-0000-0000-000000000000
☑	access_token	default	0	eyJraWQiOilyMjA3RS1DNTlDMilsInR5cCI6IkpXVCIsImFsZyI6IlJTMjU2In0.eyJzdWliO...
☑	count	default		
☑	myEpoch	default		
☑	myStartTime	default		
☑	myEndTime	default		
	Add new variable			

The myEpoch, myStartTime, and myEndTime will all be set in the Postman Collection by **Pre-request scripts** for the API calls that require Epoch Time.

For example, in the *AccessPointManager* category the *PUT Schedule upgrade for a set of access points* call you can enter a readable date/time format in the **Pre-request Script** and when you run the script will convert that date/time to Epoch time and update the environment variables.



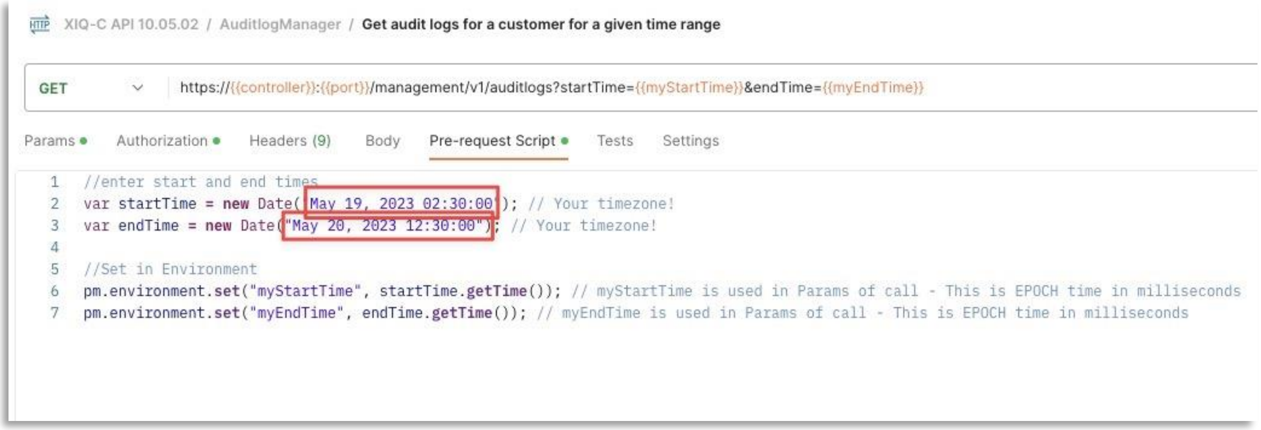
XIQ-C API 10.05.02 / AccessPointManager / **Schedule upgrade for a set of access points**

PUT https://{{controller}}:{{port}}/management/v1/aps/upgradeschedule

```
var myDate = new Date("July 2, 2023 02:30:00"); // Your timezone!

//Set in Environment
pm.environment.set("myEpoch", myDate.getTime()/1000.0); //myEpoch is used in the body of this call. This is Epoch time in seconds.
```

Another example would be the *Get audit log for a customer for a given time range* request in the *AuditlogManager* category.



XIQ-C API 10.05.02 / AuditlogManager / **Get audit logs for a customer for a given time range**

GET https://{{controller}}:{{port}}/management/v1/auditlogs?startTime={{myStartTime}}&endTime={{myEndTime}}

```
//enter start and end times
var startTime = new Date("May 19, 2023 02:30:00"); // Your timezone!
var endTime = new Date("May 20, 2023 12:30:00"); // Your timezone!

//Set in Environment
pm.environment.set("myStartTime", startTime.getTime()); // myStartTime is used in Params of call - This is EPOCH time in milliseconds
pm.environment.set("myEndTime", endTime.getTime()); // myEndTime is used in Params of call - This is EPOCH time in milliseconds
```