

# Secure Firewall REST API Introduction

## Contents

---

[Introduction](#)

[Additional Information](#)

[Configuration](#)

[API Explorer Walk-Through](#)

[Using API Explorer](#)

[Test FMC API Explorer GET Method](#)

---

## Introduction

This document describes the REST API configuration introduction for Cisco Secure Firewall using Firewall Management Center API explorer.

## Additional Information

REST API is an Application Programming Interface that can communicate based on RESTful principles. REST APIs communicate via HTTP requests and perform Create, Read, Update, and Delete (CRUD) operations within a resource. Configuration through REST API enables a great deal of possibilities to automate and streamline the way you configure Secure Firewall devices.

The main advantages of using REST API are:

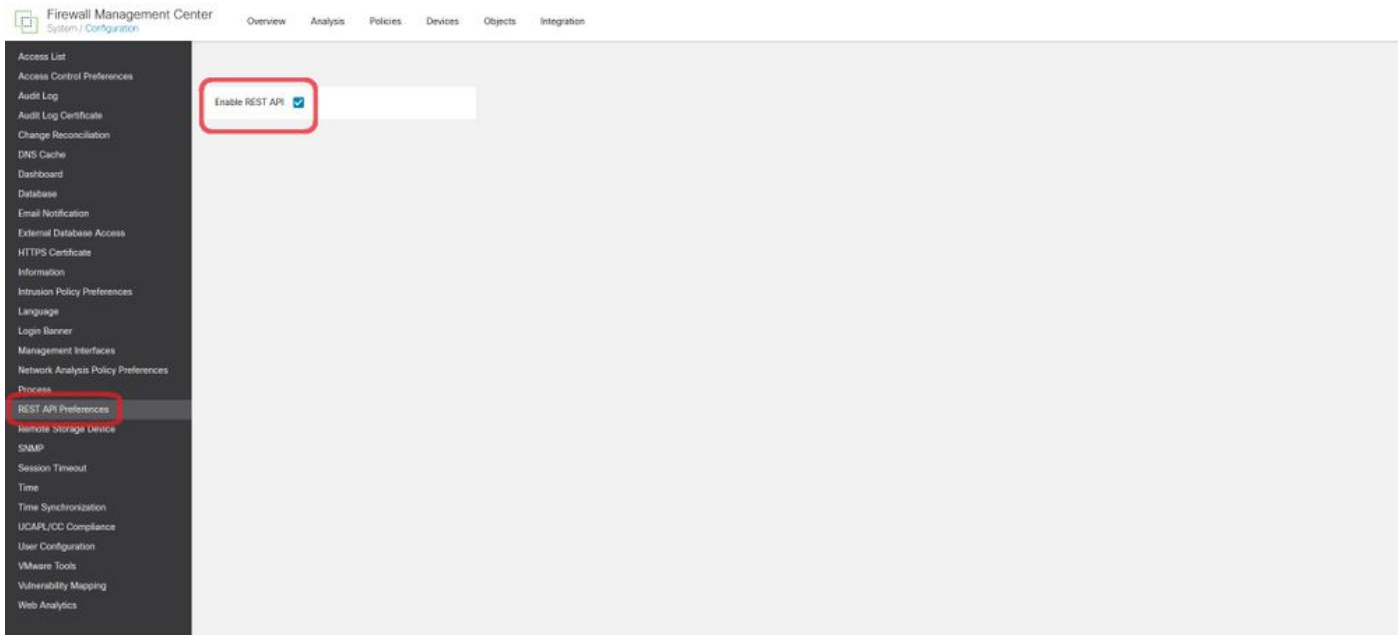
- Scalability - Since Operations can be extended to several resources.
- Flexibility - Easy to implement in different software development environments; like most APIs, it uses XML, JSON, and HTTP.
- Automation - You can streamline configuration processes for several devices at a time by performing configuration changes in bulk, reducing time-consuming repetitive configuration tasks.

REST API relies on the same authentication as the FMC/FDM and uses OAUTH2.0. Each function in the REST API maps to the same permissions in FMC and FDM.

## Configuration

### API Explorer Walk-Through

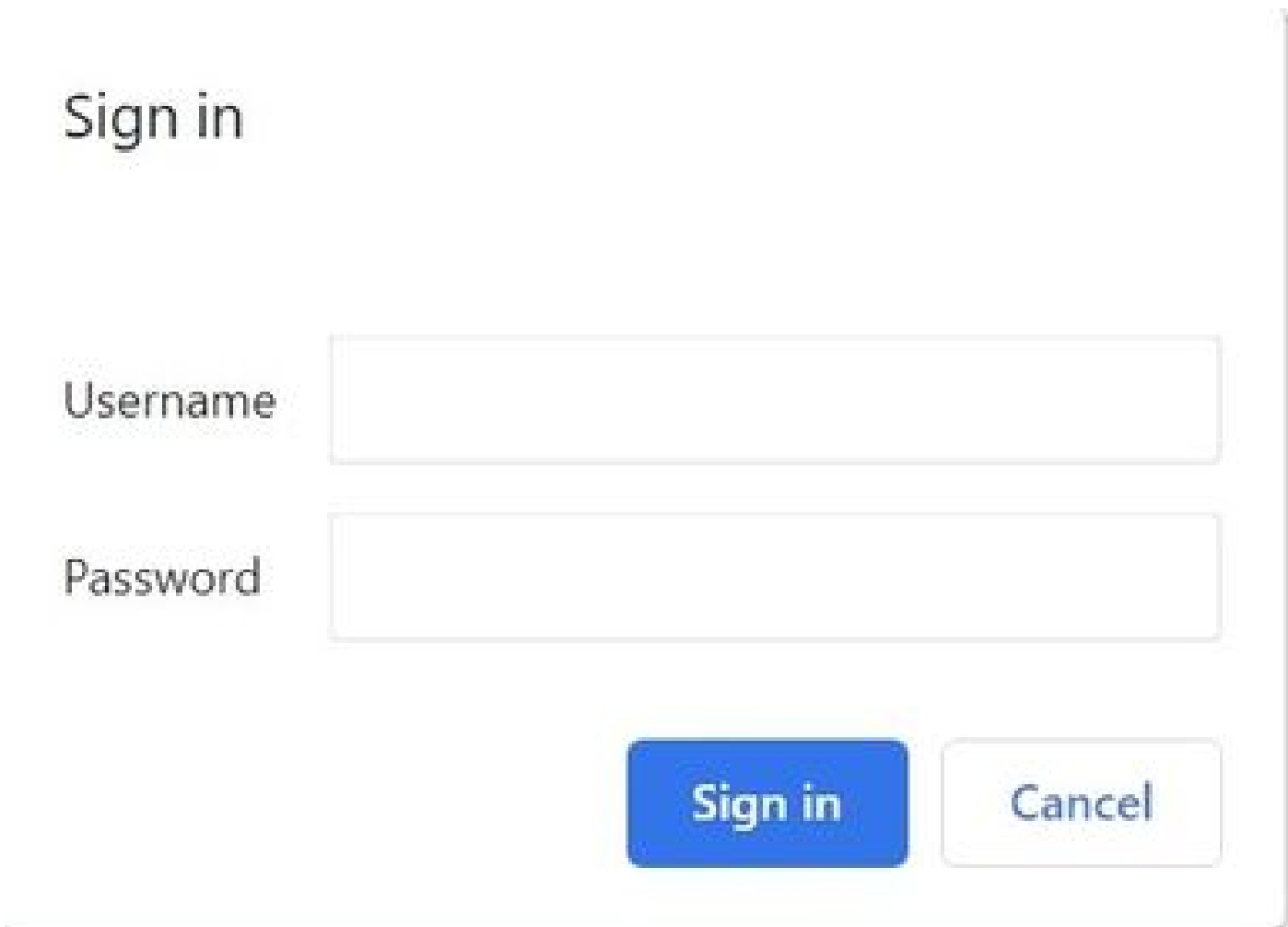
REST API is enabled by default within FMC. You can confirm it is enabled by navigating to System > Configuration > REST API Preferences.



*Enable Rest API*

FMC and FDM have a built-in interface called API Explorer, which is a helpful tool for reviewing the capabilities and functions of REST API. For FMC, API Explorer can be accessed with this URL; [https://<management\\_center\\_IP\\_address>/api/api-explorer](https://<management_center_IP_address>/api/api-explorer).

Login using FMC GUI credentials:



Sign in using your FMC GUI credentials

Once accessing the API explorer, the homepage is displayed. Here you can find the top ribbon, domains, and configuration sections. In the top right corner, you can find the version information as well as helpful resources:

Download OAS 2.0 Spec    Download OAS 3.0 Spec    Logout

# Cisco Firewall Management Center Open API Specification 1.0.0 OAS3

/fmc\_oas3.json

Specifies the REST URLs and methods supported in the Cisco Firewall Management Center API. Refer to the version specific [REST API Quick Start Guide](#) for additional information.

[Cisco Technical Assistance Center \(TAC\) - Website](#)  
[Send email to Cisco Technical Assistance Center \(TAC\)](#)  
[Cisco Firewall Management Center Licensing](#)

Domains  
Global

- Troubleshoot >
- Backup >
- Network Map >
- Devices >
- Policy Assignments >
- Device HA Pairs >

Top Ribbon

Next, find all the configuration sections, starting with the Domains. Choosing this dropdown displays all existing FMC Domains.

Download OAS 2.0 Spec    Download OAS 3.0 Spec    Logout

# Cisco Firewall Management Center Open API Specification 1.0.0 OAS3

/fmc\_oas3.json

Specifies the REST URLs and methods supported in the Cisco Firewall Management Center API. Refer to the version specific [REST API Quick Start Guide](#) for additional information.

[Cisco Technical Assistance Center \(TAC\) - Website](#)  
[Send email to Cisco Technical Assistance Center \(TAC\)](#)  
[Cisco Firewall Management Center Licensing](#)

Domains  
Global

Domains

Configuration sections and capabilities are shown next, including features that are supported by FMC:

Troubleshoot	>
Backup	>
Network Map	>
Devices	>
Policy Assignments	>
Device HA Pairs	>
Health	>
Chassis	>
Updates	>
Users	>
Intelligence	>
License	>
Search	>
Audit	>
Integration	>
Device Groups	>
Status	>
Device Clusters	>
System Information	>
Object	>
Policy	>
Deployment	>

### *Configuration Sections*

Finally, at the bottom of the page, you can find the **Schemas** section. Here you can have a look at some of the configurations in JSON for additional supported features that you can use as a reference to build your HTTP requests for these features:

Schemas	>
---------	---

### *Schemas*

## Using API Explorer

Now, going back to the configuration sections, navigate to **Devices**:

Network Map	
Devices	
GET	/api/fmc_config/v1/domain/{domainUUID}/devices/devicerecords/{objectId}
PUT	/api/fmc_config/v1/domain/{domainUUID}/devices/devicerecords/{objectId}
DELETE	/api/fmc_config/v1/domain/{domainUUID}/devices/devicerecords/{objectId}
GET	/api/fmc_config/v1/domain/{domainUUID}/devices/devicerecords
POST	/api/fmc_config/v1/domain/{domainUUID}/devices/devicerecords
DELETE	/api/fmc_config/v1/domain/{domainUUID}/devices/devicerecords
GET	/api/fmc_config/v1/domain/{domainUUID}/devices/devicerecords/{containerUUID}/fpphysicalinterfaces/{objectId}
PUT	/api/fmc_config/v1/domain/{domainUUID}/devices/devicerecords/{containerUUID}/fpphysicalinterfaces/{objectId}
GET	/api/fmc_config/v1/domain/{domainUUID}/devices/devicerecords/{containerUUID}/fplogicalinterfaces/{objectId}
PUT	/api/fmc_config/v1/domain/{domainUUID}/devices/devicerecords/{containerUUID}/fplogicalinterfaces/{objectId}
DELETE	/api/fmc_config/v1/domain/{domainUUID}/devices/devicerecords/{containerUUID}/fplogicalinterfaces/{objectId}
GET	/api/fmc_config/v1/domain/{domainUUID}/devices/devicerecords/{containerUUID}/fplogicalinterfaces
POST	/api/fmc_config/v1/domain/{domainUUID}/devices/devicerecords/{containerUUID}/fplogicalinterfaces
GET	/api/fmc_config/v1/domain/{domainUUID}/devices/devicerecords/{containerUUID}/inlinesets/{objectId}
PUT	/api/fmc_config/v1/domain/{domainUUID}/devices/devicerecords/{containerUUID}/inlinesets/{objectId}
DELETE	/api/fmc_config/v1/domain/{domainUUID}/devices/devicerecords/{containerUUID}/inlinesets/{objectId}
GET	/api/fmc_config/v1/domain/{domainUUID}/devices/devicerecords/{containerUUID}/inlinesets
POST	/api/fmc_config/v1/domain/{domainUUID}/devices/devicerecords/{containerUUID}/inlinesets
GET	/api/fmc_config/v1/domain/{domainUUID}/devices/devicerecords/{containerUUID}/virtualswitches/{objectId}

### Devices configuration

REST API for FMC supports the next HTTP methods. Note that each one of them performs a CRUD operation:

- GET – Read
- POST – Create
- PUT – Update/Replace
- DELETE – Delete

The Unified Resource Identifier (URI) accompanies each of these methods with the corresponding path to each object:

GET	/api/fmc_config/v1/domain/{domainUUID}/devices/devicerecords
-----	--

*/api/fmc\_config/v1/domain/{domainUUID}/devices/devicerecords*

By choosing one of these methods, you can expand and see the parameters included in your GET HTTP request:

- Filter
- Offset
- Limit
- Expanded
- Domain Universally Unique Identifier (UUID)

GET /api/fmc\_config/v1/domain/{domainUUID}/devices/devicerecords

Retrieves or modifies the device record associated with the specified ID. Registers or unregisters a device. If no ID is specified for a GET, retrieves list of all device records.

Parameters
Try it out

Name	Description
<b>filter</b> string <small>(query)</small>	Filter to retrieve or delete device records based upon filter parameters specified.  For bulk deletion, we need the filter="ids:" with <code>bulk=true</code> flag, Value is of format : <code>"ids:id1,id2,..."</code> . <code>ids:id1,id2,...</code> is a comma-separated list of device uuids to be deleted.  For fetching device records, Filter criteria should be <code>name:{name};hostName:{hostName};serialNumber:{ABCXXXXX};containerType:{value};version:{x.x.x};clusterBootstrapSupported:{true false};analyticsOnly:{true false};includeOtherAssociatedPolicies:{true false}</code>  <code>containerType</code> -- Allowed values are <code>{DeviceCluster DeviceHAPair DeviceStack}</code>  <code>clusterBootstrapSupported</code> -- Allowed values are <code>{true false}</code>  <code>analyticsOnly</code> -- Allowed values are <code>{true false}</code>  <code>includeOtherAssociatedPolicies</code> -- Allowed values are <code>{true false}</code> . When set to <code>true</code> , will give following policies if assigned to device: <code>[ RAVpn , FTDS25Vpn , PlatformSettingsPolicy , QosPolicy , NatPolicy , FlexConfigPolicy ]</code>  <input style="width: 100%; border: 1px solid #add8e6;" type="text" value="filter - Filter to retrieve or delete device record"/>
<b>offset</b> integer(\$int32) <small>(query)</small>	Index of first item to return.  <input style="width: 100%; border: 1px solid #add8e6;" type="text" value="offset - Index of first item to return."/>
<b>limit</b> integer(\$int32) <small>(query)</small>	Number of items to return.  <input style="width: 100%; border: 1px solid #add8e6;" type="text" value="limit - Number of items to return."/>
<b>expanded</b> boolean <small>(query)</small>	If set to true, the GET response displays a list of objects with additional attributes.  <input style="width: 100%; border: 1px solid #add8e6;" type="text" value="--"/>
<b>domainUUID</b> * required string <small>(path)</small>	Domain UUID  <input style="width: 100%; border: 1px solid #add8e6;" type="text" value="e276abec-e0f2-11e3-8169-6d9ed49b625f"/>

Responses
Link

Code	Description
------	-------------

GET devices/devicerecords



**Note:** Domain UUID is crucial when generating the HTTP requests since each object has a unique identifier assigned, and such is required to perform operations.

<b>domainUUID</b> * required	Domain UUID
string (path)	<input type="text" value="e276abec-e0f2-11e3-8169-6d9ed49b625f"/>

*Device Records Domain UUID*

Copy the **Domain UUID**:

e276abec-e0f2-11e3-8169-6d9ed49b625f

Next, you can see the Responses section, where you can find the Curl and Request URL along with the default Server Response to this method and some server response examples.

Responses

Code	Description	Links
200	OK	No links

Media type:

Examples:

Controls Accept Header

Example Value | Schema

```
{
  "links": {
    "parent": "string",
    "self": "string"
  },
  "paging": {
    "pages": 0,
    "offset": 0,
    "limit": 0,
    "count": 0
  },
  "items": {
    "hostname": "string",
    "metadata": {
      "lastUser": {
        "name": "string",
        "links": {
          "parent": "string",
          "self": "string"
        }
      },
      "id": "string",
      "type": "string"
    },
    "isPartOfContainer": true,
    "ispVersion": "string",
    "inventoryDetails": {
      "cpoType": "string",
      "conCore": "string"
    }
  }
}
```

default

Error

Media type:

Example Value | Schema

Responses Section.

## Test FMC API Explorer GET Method

Now you are ready to test API explorer functionality by clicking Try it out:

GET /api/fmc\_config/v1/domain/{domainUUID}/devices/devicerecords

Retrieves or modifies the device record associated with the specified ID. Registers or unregisters a device. If no ID is specified for a GET, retrieves list of all device records.

Parameters

Name	Description
------	-------------

Select Try it out

For this particular HTTP GET request (of devices, device records), you are not required to include any other UUID or additional parameters and you can choose **Execute**:



filter - Filter to retrieve or delete device record

offset  
integer(\$int32)  
(query)  
Index of first item to return.  
offset - Index of first item to return.

limit  
integer(\$int32)  
(query)  
Number of items to return.  
limit - Number of items to return.

expanded  
boolean  
(query)  
If set to true, the GET response displays a list of objects with additional attributes.  
--

domainUUID \* required  
string  
(path)  
Domain UUID  
e276abec-e0f2-11e3-8169-6d9ed49b625f

**Execute**

Responses

Code	Description	Links
200	OK	No links

Select Execute

FMC returns a Server Response 200 if the HTTP GET request was successful and the Response body contains Device information for all registered devices in your FMC.

**Execute** Clear

Responses

Curl

```
curl -X 'GET' \
  'https://10.88.240.53/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/devices/devicerecords' \
  -H 'accept: application/json' \
  -H 'X-auth-access-token: f7da489d-f20e-4948-ac71-9cdf84e86b5'
```

Request URL

```
https://10.88.240.53/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/devices/devicerecords
```

Server response

Code	Details
200	Response body

```
{
  "links": {
    "self": "https://10.88.240.53/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/devices/devicerecords?offset=0&limit=25"
  },
  "items": [
    {
      "id": "6bad6bbc-0b05-11e3-9a47-84ecf73b3ccf",
      "type": "Device",
      "links": {
        "self": "https://10.88.240.53/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/devices/devicerecords/6bad6bbc-0b05-11e3-9a47-84ecf73b3ccf"
      },
      "name": "FTDv-703"
    }
  ],
  "paging": {
    "offset": 0,
    "limit": 25,
    "count": 1,
    "pages": 1
  }
}
```

Response headers

```
accept-ranges: bytes
cache-control: no-store
connection: Keep-Alive
content-encoding: gzip
content-security-policy: base-uri 'self'
content-type: application/json
date: Fri, 29 Sep 2023 13:43:29 GMT
keep-alive: timeout=5,max=100
referrer-policy: same-origin
server: Apache
strict-transport-security: max-age=31536000; includeSubDomains
transfer-encoding: chunked
vary: Accept-Charset,Accept-Encoding,Accept-Language,Accept
```

200 GET Response Output.

From this output, notice that there is one FTD managed by this FMC, named FTDv-703.

**domainUUID** \* required

string  
(path)

Domain UUID

e276abec-e0f2-11e3-8169-6d9ed49b625f

*GET Device Records Domain UUID*

You can write down the ID value as it is used in order to access the API requests targeted to this FTD in particular. Copy the **ID**:

```
<#root>
```

```
"name": "FTDv-703"
```

```
"id": "6bad6bbc-0b05-11ee-9a47-84ecf73b3ccf"
```

As a final example, you can retrieve all Interface configurations of a particular Managed device (FTDv-703) by using the UUID of a device (obtained from the earlier response) in this method:

```
<#root>
```

```
"id": "6bad6bbc-0b05-11ee-9a47-84ecf73b3ccf"
```

Navigate to GET - Devices > Device records > physicalinterfaces.

```
<#root>
```

```
/api/fmc_config/v1/domain/{domainUUID}/devices/devicerecords/{containerUUID}/physicalinterfaces
```

FMC replies (with the Server Response output) and you can see that this device (FTD) has two data Interfaces and a diagnostic interface configured with their corresponding UUID and configurations.

Execute Clear

Responses

Curl

```
curl -X 'GET' \
  https://10.88.240.53/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/devices/devicerecords/6bad6bbc-0b05-11ee-9a47-84ecf73b3ccf/physicalinterfaces' \
  -H 'accept: application/json' \
  -H 'X-auth-access-token: 449403c9-5d1b-4a89-85f6-67858df3709b'
```

Request URL

```
https://10.88.240.53/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/devices/devicerecords/6bad6bbc-0b05-11ee-9a47-84ecf73b3ccf/physicalinterfaces
```

Server response

Code	Details
200	<p>Response body</p> <pre>{   "items": [     {       "links": {         "self": "https://10.88.240.53/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/devices/devicerecords/6bad6bbc-0b05-11ee-9a47-84ecf73b3ccf/physicalinterfaces/00505683-9582-0ed3-0000-004294967553"       },       "type": "PhysicalInterface",       "id": "00505683-9582-0ed3-0000-004294967553",       "name": "GigabitEthernet0/0"     },     {       "links": {         "self": "https://10.88.240.53/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/devices/devicerecords/6bad6bbc-0b05-11ee-9a47-84ecf73b3ccf/physicalinterfaces/00505683-9582-0ed3-0000-004294967554"       },       "type": "PhysicalInterface",       "id": "00505683-9582-0ed3-0000-004294967554",       "name": "GigabitEthernet0/1"     },     {       "links": {         "self": "https://10.88.240.53/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/devices/devicerecords/6bad6bbc-0b05-11ee-9a47-84ecf73b3ccf/physicalinterfaces/00505683-9582-0ed3-0000-004294967555"       },       "type": "PhysicalInterface",       "id": "00505683-9582-0ed3-0000-004294967555",       "name": "Diagnostic0/0"     }   ],   "paging": { </pre> <p>Response headers</p> <pre>accept-ranges: bytes cache-control: no-store </pre>

GET Device Records Physical Interfaces Response.

<#root>

From Response body:

"type": "PhysicalInterface",

"id": "005056B3-9582-0ed3-0000-004294967553",

"name": "GigabitEthernet0/0"

"type": "PhysicalInterface",

"id": "005056B3-9582-0ed3-0000-004294967554",

"name": "GigabitEthernet0/1"

"type": "PhysicalInterface",

```
"id": "005056B3-9582-0ed3-0000-004294967555",
```

```
"name": "Diagnostic0/0"
```

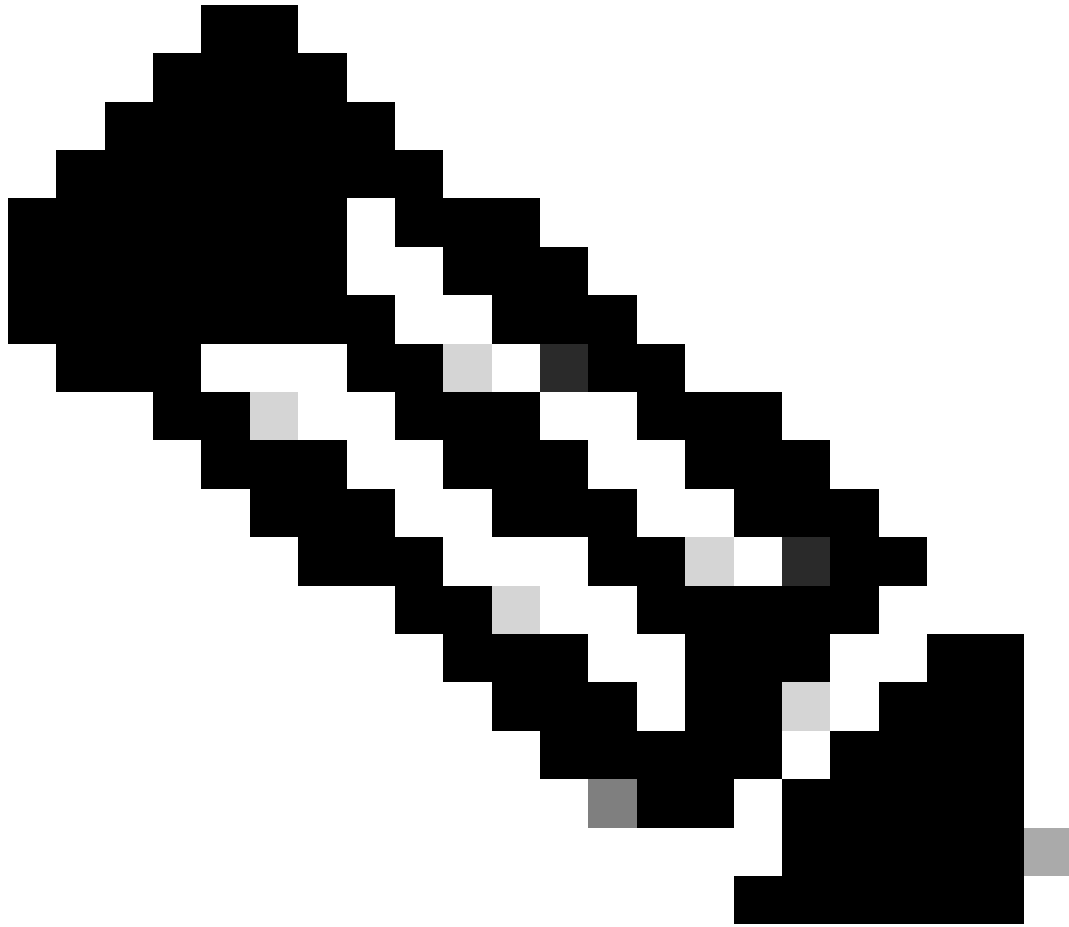
The previous tree-like structure and the logic of accessing the HTTP methods are applicable to all objects. Proceeding from general to specific UUID, you can read, modify, or add configuration changes to the FMC and specific managed devices.



*URI Structure.*

The FMC API explorer can be of great use as a guide or reference in order to view the supported features and configuration methods, so you can design and customize your code for configuration deployments.

You can also interact with FMC API using multiple API platforms like Postman or from a local host through Python or Perl script.



**Note:** You can visit the [Secure-Firewall Repository](#) in Github in order to view a great deal of Templates and Automation Resources.

---