

# ***PAN-OS<sup>®</sup> and Panorama<sup>™</sup> API Guide***

## ***XML API Usage Guide 8.0***

---

## Contact Information

Corporate Headquarters:

Palo Alto Networks

3000 Tannery Way

Santa Clara, CA 95054

[www.paloaltonetworks.com/company/contact-support](http://www.paloaltonetworks.com/company/contact-support)

## About the Documentation

- For the most recent version of this guide or for access to related documentation, visit the Technical Documentation portal [www.paloaltonetworks.com/documentation](http://www.paloaltonetworks.com/documentation).
- To search for a specific topic, go to our search page [www.paloaltonetworks.com/documentation/document-search.html](http://www.paloaltonetworks.com/documentation/document-search.html).
- Have feedback or questions for us? Leave a comment on any page in the portal, or write to us at [documentation@paloaltonetworks.com](mailto:documentation@paloaltonetworks.com).

## Copyright

Palo Alto Networks, Inc.

[www.paloaltonetworks.com](http://www.paloaltonetworks.com)

© 2017-2018 Palo Alto Networks, Inc. Palo Alto Networks is a registered trademark of Palo Alto Networks. A list of our trademarks can be found at [www.paloaltonetworks.com/company/trademarks.html](http://www.paloaltonetworks.com/company/trademarks.html). All other marks mentioned herein may be trademarks of their respective companies.

## Last Revised

January 23, 2018

---

# Table of Contents

About the PAN-OS XML API.....	5
PAN-OS XML API Components.....	7
Structure of a PAN-OS XML API Request.....	8
API Authentication and Security.....	8
XML and XPath.....	9
XPath Node Selection.....	9
Get Started with the PAN-OS XML API.....	11
Enable API Access.....	13
Get Your API Key.....	14
Make Your First API Call.....	15
Explore the API.....	16
Use the API Browser.....	16
Use the CLI to Find XML API Syntax.....	18
Use the Web Interface to Find XML API Syntax.....	19
PAN-OS XML API Error Codes.....	21
PAN-OS XML API Use Cases.....	23
Upgrade a Firewall to the Latest PAN-OS Version (API).....	25
Show and Manage GlobalProtect Users (API).....	28
Query a Firewall from Panorama (API).....	30
Upgrade PAN-OS on Multiple HA Firewalls through Panorama (API).....	33
Automatically Check for and Install Content Updates (API).....	39
Enforce Policy using External Dynamic Lists and AutoFocus Artifacts (API).....	43
Configure SAML 2.0 Authentication (API).....	45
PAN-OS XML API Request Types.....	47
PAN-OS XML API Request Types and Actions.....	49
Request Types.....	49
Configuration Actions.....	49
Asynchronous and Synchronous Requests to the PAN-OS XML API.....	51
Configuration (API).....	52
Get Active Configuration.....	52
Get Candidate Configuration.....	54
Set Configuration.....	55
Edit Configuration.....	56
Delete Configuration.....	57
Rename Configuration.....	57
Clone Configuration.....	58
Move Configuration.....	58
Override Configuration.....	58
Multi-Move or Multi-Clone Configuration.....	59
View Configuration Node Values for XPath.....	59
Commit Configuration (API).....	61
Commit.....	61
Commit-All.....	62
Run Operational Mode Commands (API).....	64

---

Get Reports (API).....	68
Dynamic Reports.....	68
Predefined Reports.....	70
Custom Reports.....	70
Export Files (API).....	72
Export Packet Captures.....	72
Export Certificates and Keys.....	74
Export Technical Support Data.....	75
Import Files (API).....	77
Importing Basics.....	77
Import Files.....	77
Retrieve Logs (API).....	79
API Log Retrieval Parameters.....	79
Example: Use the API to Retrieve Traffic Logs.....	80
Apply User-ID Mapping and Populate Dynamic Address Groups (API).....	82
Get Version Info (API).....	86

# About the *PAN-OS XML API*

The PAN-OS® and Panorama™ XML API allows you to manage firewalls and Panorama through a programmatic XML-based API. Use this API to access and manage your firewall through a third-party service, application, or script.

The PAN-OS XML API uses a tree of XML nodes to map firewall or Panorama functionality. To make an API request, you must specify the XPath (XML Path Language) to the XML node that corresponds to a specific setting or action. XPath allows you to navigate through the hierarchical XML tree structure for firewalls and Panorama.

Because PAN-OS XML API functionality mirrors that of both the web interface and the CLI, you should familiarize yourself with both. Reading relevant portions of the PAN-OS Administrator's Guide will help you get a better understanding of firewall functionalities that you can access using the API. You should also be knowledgeable about web service APIs, HTTP, XML, and XPath.

- > [PAN-OS XML API Components](#)
- > [Structure of a PAN-OS XML API Request](#)



---

# PAN-OS XML API Components

Use the PAN-OS XML API when you want to automate tasks you need to perform, such as:

- Create, update, and modify firewall and Panorama configurations.
- Execute operational mode commands, such as restart the system or validate configurations.
- Retrieve reports.
- Manage users through User-ID.
- Update dynamic objects without having to modify or commit new configurations.

The PAN-OS XML API offers a number of components to automate access and configuration of Palo Alto Networks firewalls and Panorama.

Feature	Description
Full access to PAN-OS functionality	The PAN-OS XML API allows you to access almost all of the functionality normally provided through the firewall web interface and CLI.
Secure authentication and access using API key and admin roles	Use your administrative username and password to generate an API key to authenticate API calls. Granular roles allow you to grant API access to specific functionality including reports, logs, and operational mode commands.
Options to view XML syntax through API browser, CLI and web interface debug mode	To explore all various functions of the API, you can use the API browser through the firewall web interface. You can also enable debug mode through the CLI to see the API equivalent of CLI commands.

---

# Structure of a PAN-OS XML API Request

An API request typically comprises a number of parameters, as shown in the example below:

```
https://<firewall>/api/?  
type=<type>&action=<action>&xpath=<xpath>&key=<apikey>
```

- API key (**key=**): The API key allows you to authenticate yourself to the API when making requests. Learn about [API Authentication and Security](#) and how to [Get Your API Key](#).
- Request type (**type=**): Because the XML API allows you to perform a wide array of requests, you must first specify the type of request you want, ranging from configuration to operation, importing to exporting, and from reports to user ID. Learn more about [Request Types](#).
- Action (**action=**): When the request type is **config** (configuration) or **op** (operational mode command), you must also specify an associated action, such as **edit**, **delete**, or **move**. Learn more about [Configuration Actions](#).
- XML and XPath elements (**xpath=** or **cmd=**): When using configuration or operational mode commands on the firewall, you include only the XML or the XPath that specifies the XML node. Learn more about [XML and XPath](#) and [XPath Node Selection](#).

To make requests to the PAN-OS XML API, you can use the GET and POST methods.

Use a GET request when the query size is less than 2K and you want to pass strings in the Request URL. When using the GET method, append the query string to the request URL as a URL-encoded parameter string:

```
GET /api/?type=keygen&user=<username>&password=<password>
```

Use a POST request when you are sending large amounts of form data (the request size is between 2K to 5MB; limit the request size to 5MB) or when you are passing non-ASCII characters. Some API requests, such as importing files, require POST. When using the POST method, pass the parameters in the request body. In this example, the request body includes the login credentials:

```
POST /api/ HTTP/1.1  
Content-Type: application/x-www-form-urlencoded  
password=<password>&user=<username>&type=keygen
```

## API Authentication and Security

By default, all API requests must be made over HTTPS. Additionally, you must [Get Your API Key](#) and include it in the request to authenticate your API requests. Alternatively, you can use [Basic Authentication](#) with your admin credentials by passing the Base64 encoded **username:password** in an Authorization header field:

```
Authorization: Basic amJPbLxpbw9UaTpXb3JrKjIwMDA=
```



*You cannot use basic authentication when you [Get Your API Key](#).*



---

## XML and XPath

The PAN-OS XML API uses XML for both requests and responses. When making requests, construct an HTTPS GET or POST request with the correct type and action along with the correct XPath. Here is an example API request:

```
https://<firewall>/api/?type=config&action=show&key=<APIkey>&xpath=/config/  
devices/entry/vsys/entry/rulebase/security
```

Ensure you replace variables such as `<hostname>` and `<APIkey>` with the IP address or hostname of your firewall or Panorama and API key, respectively.

When making configuration requests (`type=config`), you can use XPath, a syntax for selecting nodes from within an XML document. Use the XPath to isolate and modify portions of your configuration. The XML configuration within PAN-OS uses four different types of nodes as shown here:

```
<users>  
:   <entry name="admin">  
:     <permissions>  
:       <role-based>  
:         <superuser>yes</superuser>  
:       </role-based>  
:     </permissions>  
:   </entry>  
:   <entry name="guest">  
:     <permissions>  
:       <role-based>  
:         <custom>  
:           <profile>NewUser</profile>  
:         </custom>  
:       </role-based>  
:     </permissions>  
:   </entry>  
</users>
```

- Root nodes are top-level nodes with no parent. Requesting the root node returns all child elements.
- Element nodes represent containers of information. Element nodes can contain other element nodes or simply act as a container of information. Example: `<permissions></permissions>`
- Attribute nodes are nodes that contain name/value pairs. Example: `<entry name="admin"></entry>`
- Text nodes contain plain text. Example: `<superuser>yes</superuser>`

[Explore the API](#) with the API browser, CLI, or debug console to learn how to construct XML requests.

## XPath Node Selection

There are various ways to specify the XPath for an XML node in an API request. The simplest is to use the location path of the resource. For example, to select all users within your management configuration, use the following path:

```
/config/mgt-config/users
```

The above path specifies the following XML node that includes all users:

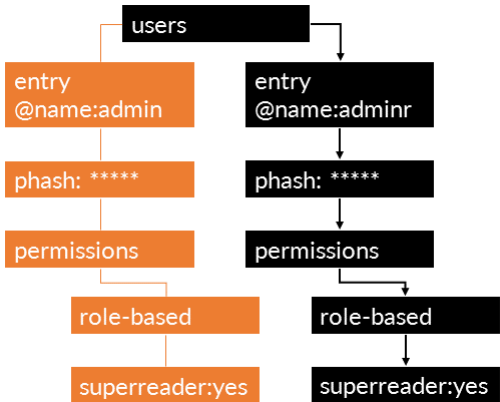
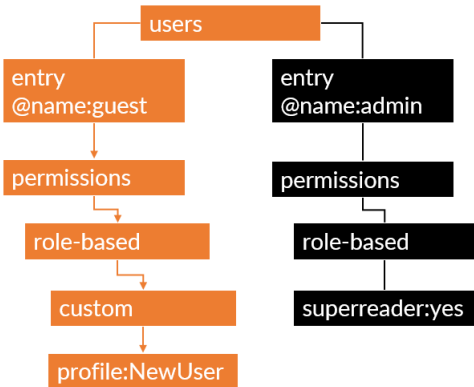
```
<users>  
  <entry name="admin">  
    <permissions>  
      <role-based>
```

```

        <superuser>yes</superuser>
      </role-based>
    </permissions>
  </entry>
  <entry name="guest">
    <permissions>
      <role-based>
        <custom>
          <profile>NewUser</profile>
        </custom>
      </role-based>
    </permissions>
  </entry>
</users>

```

Another method for selecting the XPath for an XML node is to select the specific node, such as the **superuser** or **NewUser** node within the node shown above. Use XPath syntax similar to the following to drill-down and select a specific node:

XML Node	XPath Syntax
 <pre> graph TD     users[users] --&gt; entry_adminr["entry @name:adminr"]     entry_adminr --&gt; phash_adminr["phash: ****"]     phash_adminr --&gt; permissions_adminr[permissions]     permissions_adminr --&gt; role_based_adminr[role-based]     role_based_adminr --&gt; superreader_adminr["superreader:yes"] </pre>	<pre> /config/mgt-config/users/entry/permissions/role-based/superuser[text()='yes'] </pre>
 <pre> graph TD     users[users] --&gt; entry_guest["entry @name:guest"]     entry_guest --&gt; permissions_guest[permissions]     permissions_guest --&gt; role_based_guest[role-based]     role_based_guest --&gt; custom_guest[custom]     custom_guest --&gt; profile_guest["profile:NewUser"] </pre>	<pre> /config/mgt-config/users/entry/permissions/role-based/custom/profile[text()='NewUser'] </pre>

# Get Started with the PAN-OS XML API

To use the PAN-OS XML API, first use your admin credentials to get an API key through the `keygen` command type. You can then use the API key to test a simple call.

- > [Enable API Access](#)
- > [Get Your API Key](#)
- > [Make Your First API Call](#)
- > [Explore the API](#)
- > [PAN-OS XML API Error Codes](#)

This guide tests API requests using cURL commands. However, you can use other API testing tools such as Postman and RESTClient to test API requests. By default, PAN-OS uses a self-signed certificate, so you will need to use `-k` parameter with cURL requests. Alternatively, you must replace the self-signed certificate with one from a known certificate authority. If you have an internal certificate authority, generate your own certificate and install it on the firewall.



---

# Enable API Access

The API supports the following types of Administrators and Admin roles:

- Dynamic roles: Superuser, Superuser (readonly), Device admin, Device admin (readonly), Vsys admin, Vsys admin (readonly)
- Role-based Admins: Device, Vsys, Panorama.

Admin Role profiles enable or disable features on the management interfaces of the firewall or Panorama, XML API, web interface, and CLI. For more details on Administrative Roles, see [Configure an Admin Role Profile](#).



*As a best practice, set up a separate admin account for XML API access.*

**STEP 1** | Select an Admin Role profile.

Go to **Device** > **Admin Roles** and select or create an admin role.

**STEP 2** | Select features available to the admin role.

1. Select the **XML API** tab.
2. Enable or disable XML API features from the list, such as **Report**, **Log**, and **Configuration**.
3. Select **OK** to confirm your change.

**STEP 3** | Assign the admin role to an administrator account.

See [Configure an Administrative Account](#).

---

# Get Your API Key

To use the API, generate the API key required for authenticating API calls. Request parameters should be URL encoded when used in HTTP requests.

**STEP 1 |** To generate an API key, make a GET or POST request to the firewall's hostname or IP addresses using the administrative credentials and `type=keygen`:

```
curl -k -X GET 'https://<firewall>/api/?  
type=keygen&user=<username>&password=<password>'
```

or

```
curl -k -X POST 'https://<firewall>/api/?  
type=keygen&user=<username>&password=<password>'
```

A successful API call returns `status="success"` along with the API key within the `key` element:

```
<response status="success">  
<result>  
<key>gJlQWE56987nBxIqyfa62sZeRtYuIo2BgzEA9UOnlZBhU</key>  
</result>  
</response>
```

**STEP 2 |** (Optional) Revoke an API key.

You can choose to revoke and then change an API key associated with an administrator account by [changing the password associated with the administrator account](#). Any API keys that were generated using the previous credentials would no longer be valid.



*If you want the firewall to generate a unique API key, [change the master key](#) on your firewall in order to generate a unique API key. If you have not changed the firewall master key from the default, all firewalls with the same username/password will return the same API key. Keep in mind, however, if you use Panorama to manage your firewalls, Panorama and all of the firewalls that it manages must have the same master key.*

---

# Make Your First API Call

Get Your [API Key](#) to make your first call to the PAN-OS XML API.

**STEP 1** | Make a cURL call to get system information, which returns the IP address, hostname, and model of your firewall. Be sure to include the API key:

```
curl -k 'https://<firewall>//api/?type=op&cmd=<show><system><info></info></system></show>&key=<apikey>'
```

**STEP 2** | Confirm that the response to the above request looks similar to this:

```
<response status="success">
:  <result>
:    <system>
:      <hostname>firewall</hostname>
:      <ip-address>10.27.0.8</ip-address>
:      <netmask>255.255.254.0</netmask>
:      <default-gateway>10.27.0.1</default-gateway>
:      <is-dhcp>no</is-dhcp>
:      <ipv6-address>unknown</ipv6-address>
:      <ipv6-link-local-address>fe80::21b:17dd:dedf:c04a/64</ipv6-link-
local-address>
:      <ipv6-default-gateway />
:      <mac-address>00:1b:17:ff:c0:4a</mac-address>
:      <time>Wed Feb 10 13:03:32 2016</time>
:      <uptime>1 days, 19:35:51</uptime>
:      <devicename>firewall</devicename>
:      <family>3000</family>
:      <model>PA-3020</model>
:      <serial>001901000114</serial>
:      <sw-version>7.1.</sw-version>
:      <global-protect-client-package-version>2.0.0</global-protect-client-
package-version>
:      <app-version>557-3138</app-version>
:      <app-release-date>2016/02/09 16:56:02</app-release-date>
:      <av-version>2261-2700</av-version>
:      <av-release-date>2016/02/09 15:26:53</av-release-date>
:      <threat-version>557-3138</threat-version>
:      <threat-release-date>2016/02/09 16:56:02</threat-release-date>
:      <wf-private-version>0</wf-private-version>
:      <wf-private-release-date>unknown</wf-private-release-date>
:      <url-db>paloaltonetworks</url-db>
:      <wildfire-version>27518-28208</wildfire-version>
:      <wildfire-release-date>2016/01/08 11:08:16</wildfire-release-date>
:      <url-filtering-version>2016.01.08.407</url-filtering-version>
:      <global-protect-datafile-version>1452328885</global-protect-
datafile-version>
:      <global-protect-datafile-release-date>2016/01/09 08:41:25</global-
protect-datafile-release-date>
:      <logdb-version>7.0.9</logdb-version>
:      <platform-family>3000</platform-family>
:      <vpn-disable-mode>off</vpn-disable-mode>
:      <multi-vsyz>on</multi-vsyz>
:      <operational-mode>normal</operational-mode>
:    </system>
:  </result>
</response>
```

---

# Explore the API

There are several ways you can explore the API and learn how to construct your XML requests:

- [Use the API Browser](#) on page 16
- [Use the CLI to Find XML API Syntax](#) on page 18
- [Use the Web Interface to Find XML API Syntax](#) on page 19

## Use the API Browser

Each firewall and Panorama provides an API browser that is accessible from your web browser. The API browser lets you navigate through and view the corresponding XPath and API URL.

### STEP 1 | Launch the web interface.

1. Use a web browser to navigate to the actual FQDN or IP address of your firewall:  
`https://<firewall>/`
2. Log in with your administrator credentials when prompted to log in to the web interface.

### STEP 2 | Launch the API Browser.

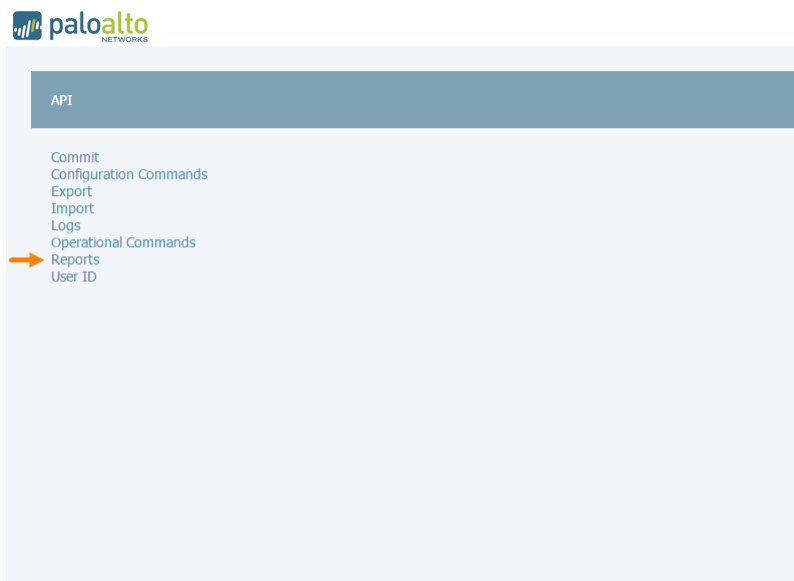
Go to the API browser URL on your firewall:

`https://<firewall>/api`

### STEP 3 | Drill-down to a request.

When you first open the API browser, the available [Request Types](#) display.

1. Select one of the request types to drill down to the next level of the XPath. Let's start with Configuration Commands, which equates to `type=report`:



2. Drill down further until you select a request that you want to test.

### STEP 4 | Test a request.

1. Select the URL to then test that request in the browser.



API > Reports > predefined > top-application-categories

**Rest API Url**

➔ /api/?type=report&async=yes&reporttype=predefined&reportname=top-application-categories

The browser shows the resulting XML response in the browser:

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<report reportname="top-application-categories" logtype="appstat">
  <result name="Top application categories" logtype="appstat" start="2016/01/26 00:00:00"
    start-epoch="1453795200" delta="86400" slabbed-start="2016/01/26 00:00:00" end="2016/01/26
    23:59:59" end-epoch="1453881599" slabbed-end="2016/01/26 23:59:59" generated-at="2016/01/27
    02:02:49" generated-at-epoch="1453888969" range="Tuesday, January 26, 2016">
    <entry>
      <category-of-name>networking</category-of-name>
      <nbytes>1578140</nbytes>
    </entry>
    <entry>
      <category-of-name>business-systems</category-of-name>
      <nbytes>236914</nbytes>
    </entry>
    <entry>
      <category-of-name>general-internet</category-of-name>
      <nbytes>690658</nbytes>
    </entry>
    <entry>
      <category-of-name>collaboration</category-of-name>
      <nbytes>1517594</nbytes>
    </entry>
    <entry>
      <category-of-name>unknown</category-of-name>
      <nbytes>480</nbytes>
    </entry>
  </result>
</report>
```

Along with the URL, the API browser also provides the XPath as necessary, as shown here for a description of a predefined application:

API > Configuration Commands > predefined > application > entry[@name='google-play'] > description

**XPath**

/config/predefined/application/entry[@name='google-play']/description

Submit

**Rest API Url**

/api/?type=config&action=get&xpath=/config/predefined/application/entry[@name='google-play']/description

## Use the CLI to Find XML API Syntax

Another method to determine the appropriate XML syntax and XPath for your API calls is through the [command-line interface \(CLI\)](#). This method works for **type=op** and **type=config** API calls.

Use the CLI to enable debug mode and then run the CLI command to receive the corresponding XML and XPath in the response.

### STEP 1 | Access the CLI.

Use an SSH client or terminal to access your [firewall](#) or [Panorama CLI](#).

### STEP 2 | Enable debug mode.

Enter the following command:

```
debug cli on
```

### STEP 3 | Run a CLI command.

Enter and run a CLI command. Example:

```
test url http://paloaltonetworks.com
<request cmd="op" cookie="7581536015878829"
uid="1206"><operations><test><url>http://paloaltonetworks.com</url></
test></operations></request>
```

### STEP 4 | Use the resulting response to create an API call.

Use the **cmd** value and the XML elements within the **operations** tag to form the API call:

```
https://<firewall>/api/?type=op&cmd=<test><url>http://
paloaltonetworks.com</url></test>&key=<apikey>
```



Depending on the CLI command, the XML tag values for **cmd** will vary. For example, here is a CLI command for showing firewall information: **run show system info**

The corresponding API call looks like this:

```
https://<firewall>/api/?type=op&cmd=<show><system><info></info></system></show>&key=<apikey>
```

## Use the Web Interface to Find XML API Syntax

You can use the web interface along with the available debug console to explore the XML and XPath necessary for your API calls.

First log into the web interface and then open a separate window where you can view the corresponding XML and XPath.

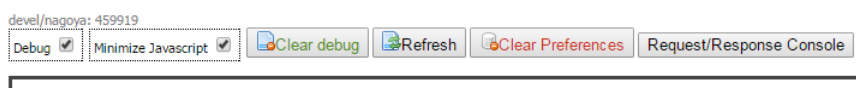
### STEP 1 | Launch the web interface.

Launch a web browser and enter the firewall's IP address or hostname. Enter your user credentials.

### STEP 2 | Launch the debug console.

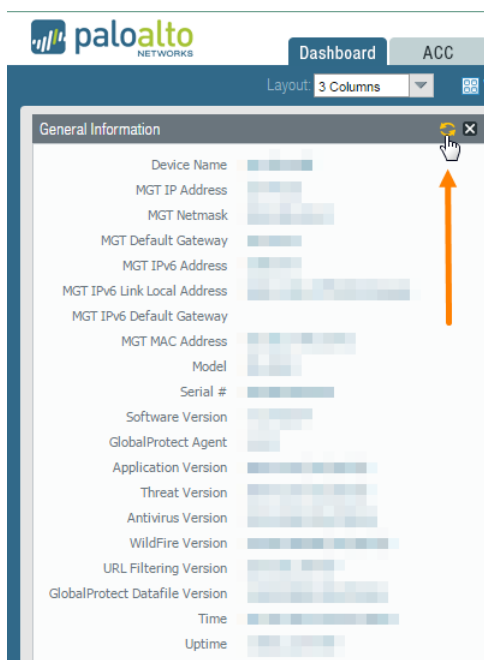
In a separate web browser window or tab, launch the debug console:

```
http://<firewall>/debug
```



### STEP 3 | Perform the action you want to replicate through the API.

In the web browser, navigate to the menu and item or action that you want to perform.



---

To aid in finding the relevant XML, select **Clear** in the debug console just before you select the final menu or action.

**STEP 4 |** View the resulting XML syntax in the debug console.

In the debug console, select **Refresh** and then navigate through the console to the syntax related to your choice or action:

```
<request cmd="op" cookie="3885378180190727">
  <operations xml="yes">
    <show>
      <system>
        <info/>
      </system>
    </show>
  </operations>
</request>
[2016/01/29 14:56:39] user=3885378180190727
<response status="success"><result><system><hostname>
```

Example XML within debug console:

```
<request cmd="op"
cookie="3885378180190727">
  <operations xml="yes">
    <show>
      <system>
        <info/>
      </system>
    </show>
  </operations>
</request>
```

The corresponding API call looks like this:

```
https://<firewall>/api/?type=op&cmd=<show><system><info></info></system></show>&key=<apikey>
```

# PAN-OS XML API Error Codes

The API response XML contains a status field and an error field. These are the available API error codes and names:

Error Code	Name	Description
400	Bad request	A required parameter is missing, an illegal parameter value is used.
403	Forbidden	Authentication or authorization errors including invalid key or insufficient admin access rights. Learn how to <a href="#">Get Your API Key</a> on page 14.
1	Unknown command	The specific config or operational command is not recognized.
2-5	Internal errors	Check with technical support when seeing these errors.
6	Bad Xpath	The xpath specified in one or more attributes of the command is invalid. Check the API browser for proper xpath values.
7	Object not present	Object specified by the xpath is not present. For example, entry[@name='value'] where no object with name 'value' is present.
8	Object not unique	For commands that operate on a single object, the specified object is not unique.
10	Reference count not zero	Object cannot be deleted as there are other objects that refer to it. For example, address object still in use in policy.
11	Internal error	Check with technical support when seeing these errors.
12	Invalid object	Xpath or element values provided are not complete.
14	Operation not possible	Operation is allowed but not possible in this case. For example, moving a rule up one position when it is already at the top.
15	Operation denied	Operation is allowed. For example, Admin not allowed to delete own account, Running a command that is not allowed on a passive device.
16	Unauthorized	The API role does not have access rights to run this query.
17	Invalid command	Invalid command or parameters.
18	Malformed command	The XML is malformed.
19-20	Success	Command completed successfully.

---

Error Code	Name	Description
21	Internal error	Check with technical support when seeing these errors.
22	Session timed out	The session for this query timed out.

# ***PAN-OS XML API Use Cases***

The following use cases highlight the use of the PAN-OS XML API, either to reduce repetitive steps or to automate tasks normally you perform through the web interface or CLI.

- > Upgrade a Firewall to the Latest PAN-OS Version (API) on page 25
- > Show and Manage GlobalProtect Users (API) on page 28
- > Query a Firewall from Panorama (API) on page 30
- > Upgrade PAN-OS on Multiple HA Firewalls through Panorama (API) on page 33
- > Automatically Check for and Install Content Updates (API) on page 39
- > Enforce Policy using External Dynamic Lists and AutoFocus Artifacts (API) on page 43
- > Configure SAML 2.0 Authentication (API) on page 45





---

# Upgrade a Firewall to the Latest PAN-OS Version (API)

You can use the PAN-OS XML API to update your firewall with the latest PAN-OS and Content Release versions.

## STEP 1 | Download the latest content update.

Use the following request to first download the latest content update:

```
curl -X GET 'https://<firewall>/api/?  
type=op&cmd=<request><content><upgrade><download><latest/>  
</download></upgrade></content></request>&key=<apikey>'
```

If successful, the response contains a `jobid` that you can use to check on the status of your request.

```
<response  
status="success" code="19">  
  <result>  
    <msg>  
      <line>Download job enqueued with jobid 2</line>  
    </msg>  
    <job>2</job>  
  </result>  
</response>
```

## STEP 2 | Check on the content download status.

Use the `jobid` to ensure that the content download completes successfully:

```
curl -X GET 'https://<firewall>/api/?type=op&action=get&job-  
id=2&key=<apikey>'
```

The response should include the following:

```
<response status="success">...
```

## STEP 3 | Install the latest content update.

Use the following request to install the newly downloaded content:

```
curl -X GET 'https://<firewall>/api/?  
type=op&cmd<request><content><upgrade><install> <version>latest</  
version></install></upgrade></content></request>key=<apikey>'
```

If successful, the response contains a `jobid` that you can use to check on the status of your request.

```
<response  
status="success" code="19">  
  <result>  
    <msg>  
      <line>Content install job enqueued with jobid  
3</line>  
    </msg>  
    <job>3</job>
```

```
</result>
</response>
```

#### STEP 4 | Check on the content installation status.

Use the jobid to ensure that the content installation completes successfully:

```
curl -X GET 'https://<firewall>/api/?type=op&action=get&job-id=3&key=<apikey>'
```

The response should include the following:

```
<response status="success">...
```

#### STEP 5 | Check for the latest PAN-OS software update.

After installing the latest Content Release update, check for the latest available PAN-OS software updates:

```
curl -X GET 'https://<firewall>/api/?
type=op&cmd=<request><system><software><check></check>
</software></system></request>&key=<apikey>'
```

In the response, the first entry is the latest version of PAN-OS:

```
<response
status="success">
<result>
<sw-updates last-updated-at="2015/10/20 14:16:30">
<msg />
<versions>
>
<version>7.1.0</version>
<filename>PanOS_3000-7.1.0-c65</filename>
<size>720</size>
<size-kb>737504</size-kb>
<released-on>2015/10/20 13:23:11</released-on>
...

```

#### STEP 6 | Download the latest PAN-OS software update.

1. In this case, the latest version is 7.1.0-c65, so download that version:

```
curl -X GET 'https://<firewall>/api/?
type=op&cmd=<request><system><software><download><version>7.1.0
-c65</version></download></software></system></request>&key=<apikey>'
```

2. Use the jobid in the response to ensure that the system update download completes successfully:

```
curl -X GET 'https://<firewall>/api/?type=op&action=get&job-id=318&key=<apikey>'
```

The response should include the following:

```
<response status="success">...
```

#### STEP 7 | Install the latest PAN-OS software update.

---

To install the latest system update, include the version in a software install request:

```
curl -X GET 'https://<firewall>/api/?  
type=op&cmd=<request><system><software><install><version>7.1.0-c65</  
version></install></software></system></request>&key=<apikey>'
```

#### STEP 8 | Check on the software installation status.

Use the `jobid` in the response to ensure that the system update installs successfully:

```
curl -X GET 'https://<firewall>/api/?type=op&action=get&job-  
id=320&key=<apikey>'
```

The response should include the following:

```
<response status="success">...
```

#### STEP 9 | Reboot the firewall.

After the system update installs successfully, trigger:

```
curl -X GET  
'https://<firewall>/api/?type=op&cmd=<request><restart><system></system></  
restart>  
</request>&key=<apikey>'
```

---

# Show and Manage GlobalProtect Users (API)

One common use of the PAN-OS XML API is to manage GlobalProtect users. You can use two API requests to view and then disconnect a Global Protect user who has been logged in for too long.

## STEP 1 | View all GlobalProtect users.

Make a request to view all GlobalProtect users:

```
curl -X GET 'https://<firewall>/api/?type=op&cmd=<show><global-protect-gateway><current-user>/></global-protect-gateway></show>&key=<apikey>'
```

The response contains a list of users along with related information including IP addresses, logins, and client information:

```
<response>
status="success">
<result>
<domain />
<islocal>yes</islocal>
<username>dward</username>
<computer>Dan's iPhone</computer>
<client>Apple iOS 8.1.2</client>
<vpn-type>Device Level VPN</vpn-type>
<virtual-ip>192.168.2.1</virtual-ip>
<public-ip>166.173.63.240</public-ip>
<tunnel-type>SSL</tunnel-type>
<login-time>Jan.22 01:50:36</login-time>
<login-time-utc>1421916636</login-time-utc>
<lifetime>2592000</lifetime>
</entry>
</result>
</response>
```

The `<login-time-utc>` field is the login date/time in UNIX time format (number of seconds elapsed since 00:00:00 1 Jan 1970). To find the list of users, filter the output for this field and compare the `<login-time-utc>` value to current date and time (or another date and time).

## STEP 2 | Disconnect a GlobalProtect user.

Upon identifying the user that you want to disconnect, send a request that includes the GlobalProtect gateway, username, computer, and a **force-logout** reason:

```
curl -X GET 'https://<firewall>/api/?type=op&cmd=<request><global-protect-gateway><client-logout><gateway>Home-N</gateway><user>dward</user><reason>force-logout</reason><computer>Dan's iPhone</computer></client-logout></global-protect-gateway></request>&key=<apikey>'
```

A successful response shows that the user has been successfully disconnected:

```
<response>
status="success">
<result>
<response status="success">
<gateway>Home-N</gateway>
```

---

```
<domain>(null)</domain>
<user>dward</user>
<computer>Dan's iPhone</computer>
</response>
</result>
</response>
```

---

# Query a Firewall from Panorama (API)

The **target** parameter on Panorama allows you to redirect queries to a managed firewall. Redirecting queries to firewalls helps to reduce time and the number of steps required to issue repetitive commands. Use the scripting language of your choice to store firewall serial numbers and use them to issue a query to several firewalls.



Currently, you can only use **type=op** queries when redirecting queries through Panorama.

## STEP 1 | Get a list of managed firewalls.

```
https://<panorama>/api/?type=op&cmd=<show><devices><all></all></devices></show>
```



If you want to get a list of connected firewalls only, use

```
https://<panorama>/api/?type=op&cmd=<show><devices><connected></connected></devices></show>
```

The response includes the serial number (serial) of each firewall.

```
<response
status="success">
  <result>
    <devices>
      name="007200002517">
        <serial>007200002342</serial>
        <connected>yes</connected>
        <unsupported-version>no</unsupported-version>
        <deactivated>no</deactivated>
        <hostname>PM-6-1-VM</hostname>
        <ip-address>10.3.4.137</ip-address>
        <mac-addr />
        <uptime>81 days, 20:39:41</uptime>
        <family>vm</family>
        <model>PA-VM</model>
        <sw-version>6.1.3</sw-version>
        <app-version>555-3129</app-version>
        <av-version>2254-2693</av-version>
        <wildfire-version>91873-101074</wildfire-version>
        <threat-version>555-3129</threat-version>
        <url-db>paloaltonetworks</url-db>
        <url-filtering-version>2016.02.02.416</url-filtering-version>
        <logdb-version>6.1.3</logdb-version>
        <vpnclient-package-version />
        <global-protect-client-package-version>0.0.0</global-protect-
client-package-version>
        <vpn-disable-mode>no</vpn-disable-mode>
        <operational-mode>normal</operational-mode>
        <multi-vsyz>no</multi-vsyz>
        <vsyz>
          name="vsyz1">
            <display-name>vsyz1</display-name>
            <shared-policy-status />
```

```

        <shared-policy-md5sum>4a0913667df83ff1098492e2e2ec1756</
shared-policy-md5sum>
    </entry>
</vsys>
</entry>

<!--truncated -->

</devices>
</result>
</response>

```

The response contains a `<serial>` XML element for each firewall.

## STEP 2 | Collect firewall serial numbers.

In your script or code, store the firewall serial numbers returned in the response to the previous request.

## STEP 3 | Query a firewall from Panorama.

A normal request to show system information on a firewall looks like this:

```

https://<firewall>/api/?type=op&cmd=<show><system><info></info></system></
show>

```

To directly target a firewall through Panorama, append the firewall serial number to the request:

```

https://<panorama>/api/?type=op&cmd=<show><system><info></info></system></
show>&target=<device-serial-number>

```

A successful response should look like this:

```

<response
status="success">
<result>
<system>
<hostname>firewall</hostname>
<ip-address>10.41.0.8</ip-address>
<netmask>255.255.224.0</netmask>
<default-gateway>10.41.0.1</default-gateway>
<is-dhcp>no</is-dhcp>
<ipv6-address>unknown</ipv6-address>
<ipv6-link-local-address>fe80::21c:17cf:feff:c04a/64</ipv6-link-local-
address>
<ipv6-default-gateway></ipv6-default-gateway>
<mac-address>00:1b:17:fc:c0:4a</mac-address>
<time>Tue Oct 27 13:39:09 2015</time>
<uptime>12 days, 0:05:26</uptime>
<devicename>pm-firewall</devicename>
<family>3000</family>
<model>PA-3020</model>
<serial>001802000104</serial>
<sw-version>7.1.0-c54</sw-version>
<global-protect-client-package-version>2.0.0</global-protect-client-
package-version>
<app-version>537-2965</app-version>
<app-release-date>2015/10/26 18:10:48</app-release-date>
<av-version>2149-2586</av-version>
<av-release-date>2015/10/26 15:31:55</av-release-date>
<threat-version>537-2965</threat-version>
<threat-release-date>2015/10/26 18:10:48</threat-release-date>

```

---

```
<wf-private-version>0</wf-private-version>
<wf-private-release-date>unknown</wf-private-release-date>
<url-db>paloaltonetworks</url-db>
<wildfire-version>80683-89773</wildfire-version>
<wildfire-release-date>unknown</wildfire-release-date>
<url-filtering-version>2015.10.27.226</url-filtering-version>
<global-protect-datafile-version>1445974904</global-protect-datafile-
version>
<global-protect-datafile-release-date>2015/10/27 19:41:44</global-protect-
datafile-release-date>
<logdb-version>7.0.9</logdb-version>
<platform-family>3000</platform-family>
<vpn-disable-mode>off</vpn-disable-mode>
<multi-vsyz>on</multi-vsyz>
<operational-mode>normal</operational-mode>
</system>
</result>
</response>
```

Repeat this request for each managed or connected firewall.



---

# Upgrade PAN-OS on Multiple HA Firewalls through Panorama (API)

This use case highlights the ability of the PAN-OS XML API to automate a more complex procedure, namely upgrading firewalls set up as active-passive high-availability (HA) pair. Normally, this procedure involves multiple, manual steps on individual firewalls.



*This is a high-level overview of the steps you must take in this procedure. Your script or application must incorporate error-checking and logic to implement this sequence of steps.*

## STEP 1 | Check for the latest PAN-OS software update through Panorama

Check for the latest available PAN-OS software updates. Include the firewall serial number in your request:

```
https://<panorama>/api/?type=op&cmd=<request><system><software><check></check></software></system></request>&target=007200002517&key=<apikey>
```

The response contains an array of results sorted to show the latest version first:

```
<response status="success">
  <result>
    <sw-updates last-updated-at="2016/02/03 08:29:09">
      <msg />
      <versions>
        >
          <version>7.1</version>
          <filename>PanOS_vm-7.1</filename>
          <size>540</size>
          <size-kb>553964</size-kb>
          <released-on>2016/02/02 10:57:20</released-on>
          <release-notes><![CDATA[https://10.44.2.19/updates/ReleaseNotes.aspx?type=sw&versionNumber=7.1.0-c158&product=panos&platform=vm]]></release-notes>
          <downloaded>no</downloaded>
          <current>no</current>
          <latest>yes</latest>
        </entry>
      <!-- truncated -->
    </versions>
  </sw-updates>
</result>
</response>
```

## STEP 2 | Download the latest PAN-OS software update.

1. In this case, the latest version is 7.1.0-c65, so download that version:

```
curl -X GET
'https://<firewall>/api/?
type=op&cmd=<request><system><software><download><version>7.1.0
-c65</version></download></software></system></request>&key=<apikey>'
```

2. Use the `jobid` in the response to ensure that the system update download completes successfully:

```
curl -X GET 'https://<firewall>/api/?type=op&action=get&job-id=318&key=<apikey>'
```

The response should include the following:

```
<response status="success">...
```

### STEP 3 | Install the latest PAN-OS software update.

To install the latest system update, include the version in a software install request:

```
curl -X GET 'https://<firewall>/api/?type=op&cmd=<request><system><software><install><version>7.1.0-c65</version></install></software></system></request>&key=<apikey>'
```

### STEP 4 | Check on the software installation status.

Use the `jobid` in the response to ensure that the system update installs successfully:

```
curl -X GET 'https://<firewall>/api/?type=op&action=get&job-id=<jobid>&key=<apikey>'
```

The response should include the following:

```
<response status="success">...
```

### STEP 5 | Get a list of connected firewalls.

Get a list of connected firewalls that Panorama manages:

```
https://<panorama>/api/?type=op&cmd=<show><devices><https://<panorama>/api/?type=op&cmd=<show><devices><connected></connected></devices></show>
```

The response includes the serial number (`serial`) of each firewall.

```
<response status="success">
: <result>
:   <devices>
:     name="007200002517">
:       <serial>007200002342</serial>
:       <connected>yes</connected>
:       <unsupported-version>no</unsupported-version>
:       <deactivated>no</deactivated>
:       <hostname>PM-6-1-VM</hostname>
:       <ip-address>10.3.4.137</ip-address>
:       <mac-addr />
:       <uptime>81 days, 20:39:41</uptime>
:       <family>vm</family>
:       <model>PA-VM</model>
:       <sw-version>6.1.3</sw-version>
:       <app-version>555-3129</app-version>
:       <av-version>2254-2693</av-version>
:       <wildfire-version>91873-101074</wildfire-version>
:       <threat-version>555-3129</threat-version>
:       <url-db>paloaltonetworks</url-db>
:       <url-filtering-version>2016.02.02.416</url-filtering-version>
:       <logdb-version>6.1.3</logdb-version>
```

```

        <vpnclient-package-version />
        <global-protect-client-package-version>0.0.0</global-protect-
client-package-version>
        <vpn-disable-mode>no</vpn-disable-mode>
        <operational-mode>normal</operational-mode>
        <multi-vsyz>no</multi-vsyz>
        <vsyz>
            name="vsyz1">
                <display-name>vsyz1</display-name>
                <shared-policy-status />
                <shared-policy-md5sum>4a0913667df83ff1098492e2e2ec1756</
shared-policy-md5sum>
            </entry>
        </vsyz>
    </entry>

    <!--truncated -->

</devices>
</result>
</response>

```

The response contains a <serial> XML element that contains each firewall serial number.

#### STEP 6 | Check for the latest PAN-OS software update.

Check to see if new software is available on your HA pair:

```

https://<panorama>/api/?type=op&cmd=<request><system><software><check></
check></software></system></request>&target=<serialnumber>&key=<apikey>

```

The response contains an array of results sorted to show the latest version first:

```

<response status="success">
<result>
<sw-updates last-updated-at="2016/02/03 08:29:09">
<msg />
<versions>
<version>7.1</version>
<filename>PanOS_vm-7.1</filename>
<size>540</size>
<size-kb>553964</size-kb>
<released-on>2016/02/02 10:57:20</released-on>
<release-notes><![CDATA[https://10.44.2.19/updates/ReleaseNotes.aspx?
type=sw&versionNumber=7.1.0-c158&product=panos&platform=vm]]></release-
notes>
<downloaded>no</downloaded>
<current>no</current>
<latest>yes</latest>
</entry>
<!-- truncated -->
</versions>
</sw-updates>
</result>
</response>

```

#### STEP 7 | Download the latest PAN-OS software update.

After determining the latest system update, download it to both firewalls in the HA pair:

```
https://<panorama>/api/?  
type=op&cmd=<request><system><software><download><version>7.1</version></  
download></software></system></request>&target=<serialnumber>&key=<apikey>
```

The response contains a job ID:

```
<response status="success" code="19">  
  <result>  
    <msg>  
      <line>Download job enqueued with jobid 3448</line>  
    </msg>  
    <job>3448</job>  
  </result>  
</response>
```

Use the job ID to check on the download status:

```
https://<panorama>/api/?type=op&cmd=<show><jobs><id>3448</id></jobs></  
show>&target=<serialnumber>&key=<apikey>
```

The response contains a job status of FIN when the download is complete:

```
<response status="success">  
  <result>  
    <job>  
      <tenq>2016/02/03 08:32:00</tenq>  
      <id>3448</id>  
      <user />  
      <type>Downld</type>  
      <status>FIN</status>  
      <stoppable>no</stoppable>  
      <result>OK</result>  
      <tfin>08:32:10</tfin>  
      <progress>08:32:10</progress>  
      <details>  
        <line>Successfully downloaded</line>  
        <line>Preloading into software manager</line>  
        <line>Successfully loaded into software manager</line>  
      </details>  
      <warnings />  
    </job>  
  </result>  
</response>
```

## STEP 8 | Suspend the active HA firewall.

Suspend the active firewall in your high-availability firewall pair:

```
https://<panorama>/api/?type=op&cmd=<request><high-  
availability><state><suspend></suspend></state></high-availability></  
request>&target=<serialnumber>&key=<apikey>
```

The response confirms the active firewall has been suspended:

```
<response status="success">  
  <result>Successfully changed HA state to suspended</result>  
</response>
```

---

## STEP 9 | Install the latest software update on the suspended HA pair.

After suspending the active HA firewall, install the system update on it:

```
https://<panorama>/api/?  
type=op&cmd=<request><system><software><install><version>version</  
version></install></software></system></  
request>&target=<serialnumber>&key=<apikey>
```

The response shows the system update is queued:

```
<response status="success" code="19">  
  <result>  
    <msg>  
      <line>Software install job enqueued with jobid 3453. Run 'show  
jobs id 3453' to monitor its status. Please reboot the device after the  
installation is done.</line>  
    </msg>  
    <job>3453</job>  
  </result>  
</response>
```

## STEP 10 | Check on the software installation status.

Use the jobid in the response to ensure that the system update installs successfully:

```
curl -X GET 'https://<panorama>/api/?type=op&action=get&job-  
id=jobid&target=<serialnumber>&key=<apikey>
```

The response should include the following:

```
<response status="success">...
```

## STEP 11 | Reboot the suspended HA peer.

After installing the latest system update, reboot the suspended HA peer:

```
https://<panorama>/api/?type=op&cmd=<request><restart><system></system></  
restart></request>&target=<serialnumber>&key=<apikey>
```

## STEP 12 | Verify that the upgrade is successful.

Show system information on your upgraded HA peer to ensure it has the latest system update and is operational:

```
https://<panorama>/api/?type=op&cmd=<show><system><info></info></system></  
show>&target=<serialnumber>&key=<apikey>
```

## STEP 13 | Makes the suspended HA peer active.

After you verify that the system update on the suspended HA peer is successful, make it active again:

```
https://<panorama>/api/?type=op&cmd=<request><high-  
availability><state><functional></functional></state></high-  
availability></request>&target=<serialnumber>&key=<apikey>
```

---

The response confirms the active firewall is now active:

```
<response status="success">  
<result>Successfully changed HA state to functional</result>  
</response>
```

**STEP 14** | Install the system update on the passive HA peer.

Once the suspended HA firewall is active, you can then repeat steps 5-8 on the now passive HA peer.

# Automatically Check for and Install Content Updates (API)

Using the XML API, you can programmatically check and install new content updates, including antivirus, WildFire, and GlobalProtect updates. Check for new updates available and download updates that have been released for at least one week.



*Download, upgrade, and installation requests are asynchronous. The API responds with a job ID while it processes your request. In your subsequent request, you use this job ID to check on the result of your original request:*

```
https://<firewall>/api/?type=op&cmd=<show><jobs><id></id></jobs></show>&key=<apikey>
```

**STEP 1** | Check for installed content on your firewall. Run the following request to view current system information:

```
https://<firewall>/api/?type=op&cmd=<show><system><info></info></system></show>&key=<apikey>
```

**STEP 2** | Confirm that the API response to the request in the previous step includes the currently installed updates on your firewall:

```
<response status="success">
  <result>
    <system>
      <hostname>pm-firewall</hostname>
      <ip-address>10.47.0.8</ip-address>
      <netmask>255.255.254.0</netmask>
      <default-gateway>10.47.0.1</default-gateway>
      <is-dhcp>no</is-dhcp>
      <ipv6-address>unknown</ipv6-address>
      <ipv6-link-local-address>fe80::21b:17ff:feff:c04a/64</ipv6-link-
local-address>
      <ipv6-default-gateway />
      <mac-address>00:1b:17:ff:c0:4a</mac-address>
      <time>Mon Jul 11 17:51:37 2016</time>
      <uptime>11 days, 7:38:34</uptime>
      <devicename>pm-firewall</devicename>
      <family>3000</family>
      <model>PA-3020</model>
      <serial>001801000104</serial>
      <sw-version>7.1.3</sw-version>
      <global-protect-client-package-version>2.0.0</global-protect-
client-package-version>
      <app-version>598-3427</app-version>
      <app-release-date>2016/07/09 22:30:55</app-release-date>
      <av-version>2416-2855</av-version>
      <av-release-date>2016/07/10 11:27:57</av-release-date>
      <threat-version>598-3427</threat-version>
      <threat-release-date>2016/07/09 22:30:55</threat-release-date>
      <wf-private-version>0</wf-private-version>
      <wf-private-release-date>unknown</wf-private-release-date>
      <url-db>paloaltonetworks</url-db>
```

```

    <wildfire-version>80426-81466</wildfire-version>
    <wildfire-release-date>2016/07/11 17:45:11</wildfire-release-
date>
    <url-filtering-version>2016.07.11.248</url-filtering-version>
    <global-protect-datafile-version>1468280405</global-protect-
datafile-version>
    <global-protect-datafile-release-date>2016/07/11 23:40:05</
global-protect-datafile-release-date>
    <logdb-version>7.0.9</logdb-version>
    <platform-family>3000</platform-family>
    <vpn-disable-mode>off</vpn-disable-mode>
    <multi-vsyst>on</multi-vsyst>
    <operational-mode>normal</operational-mode>
  </system>
</result>
</response>

```

**STEP 3** | Note the currently installed versions for the following updates, so that you can compare the values after you check for the latest updates:

- global-protect-client-package-version: GlobalProtect
- app-version: Application and threat signatures.
- av-version: Antivirus signatures
- wildfire-version: WildFire malware and antivirus signatures

**STEP 4** | Check for new, available updates with the following requests and store the version field in the response, which is the **version** field for GlobalProtect, and the **app-version** field for all others:

- GlobalProtect:

```

https://<firewall>/api/?type=op&cmd=<request><global-protect-
client><software><check></check></software></global-protect-client></
request>&key=<apikey>

```

- WildFire:

```

https://<firewall>/api/?type=op&cmd=<request><wildfire><upgrade><check></
check></upgrade></wildfire></request>&key=<apikey>

```

- Application & Threat:

```

https://<firewall>/api/?type=op&cmd=<request><content><upgrade><check></
check></upgrade></content></request>&key=<apikey>

```

- Antivirus:

```

https://<firewall>/api/?type=op&cmd=<request><anti-
virus><upgrade><check></check></upgrade></anti-virus></
request>&key=<apikey>

```

Example response:

```

<response status="success">
<result>
<sw-updates last-updated-at="2016/05/19 14:34:34">
<msg/>
<versions>

```



```

<entry>
<version>4.0.0-cl6</version>
<filename>PanGP-4.0.0-cl6</filename>
<size>44</size>
<size-kb>45321</size-kb>
<released-on>2016/07/08 15:41:18</released-on>
<release-notes>
<![CDATA[
https://firewall/updates/ReleaseNotes.aspx?type=sw&versionNumber=4.0.0-cl6&product=gpclient&platform=any
]]>
</release-notes>
<downloaded>no</downloaded>
<current>no</current>
<latest>no</latest>
<uploaded>no</uploaded>
</entry>
<!--TRUNCATED-->

```

Take note of the `released-on` XML field to verify that updates have been released for at least a week.

**STEP 5** | In your script or code, compare the version values for currently installed updates to new, available updates. It is recommended that you only install updates that have been available for at least a week.

**STEP 6** | Download the latest content updates with these requests:

- GlobalProtect:

```

https://<firewall>/api/?type=op&cmd=<request><global-protect-
client><software><download><version>versionnumber</version></download></
software></global-protect-client></request>&key=<apikey>

```

- WildFire:

```

https://<firewall>/api/?
type=op&cmd=<request><wildfire><upgrade><download><latest></latest></
download></upgrade></wildfire></request>&key=<apikey>

```

- Application & Threat:

```

https://<firewall>/api/?
type=op&cmd=<request><content><upgrade><download><latest></latest></
download></upgrade></content></request>

```

- Antivirus:

```

https://<firewall>/api/?type=op&cmd=<request><anti-
virus><upgrade><download><latest></latest></download></upgrade></anti-
virus></request>&key=<apikey>

```

The response contains a job ID that you can use to check on the status of the request. Example:

```

<response status="success" code="19">
<result>
<msg>
<line>Content install job enqueued with jobid 299</line>
</msg>
<job>299</job>

```

---

```
</result>
</response>
```

Learn more about [Asynchronous and Synchronous Requests to the PAN-OS XML API](#).

**STEP 7 |** Install the latest content updates with these requests:

- GlobalProtect:

```
https://<firewall>/api/?type=op&cmd=<request><global-protect-
client><software><install><version>versionnumber</version></install></
software></global-protect-client></request>&key=<apikey>
```

- WildFire:

```
https://<firewall>/api/?
type=op&cmd=<request><wildfire><upgrade><install><version>latest</
version></install></upgrade></wildfire></request>&key=<apikey>
```

- Application & Threat:

```
https://<firewall>/api/?
type=op&cmd=<request><content><upgrade><install>latest</latest></
install></upgrade></content></request>&key=<apikey>
```

- Antivirus:

```
https://<firewall>/api/?type=op&cmd=<request><anti-
virus><upgrade><install><version>latest</version></install></upgrade></
anti-virus></request>&key=<apikey>
```

The response contains a job ID that you can use to check on the status of the request.

---

# Enforce Policy using External Dynamic Lists and AutoFocus Artifacts (API)

This use case allows you to use data from AutoFocus threat intelligence to create an external dynamic list for your firewall.

Use the AutoFocus API to export AutoFocus artifacts (IP addresses, domains, or URLs) as an export list that you can host on a web server. Learn more about AutoFocus in [AutoFocus documentation](#). Then use the PAN-OS XML API to add this URL as an external dynamic list to enforce policy dynamically on the firewall. Learn more about how to [use an external dynamic list in policy](#).



To use AutoFocus, you must first [register and activate AutoFocus](#).

**STEP 1 | Build an AutoFocus export list.** For example, if you want to block potential attacks from the Sofacy group, search for Sofacy as the Tag, and then add the appropriate artifacts shown within the File Analysis tab, such as DNS Activity, HTTP Requests, and Connection Activity.

```
Use the AutoFocus API to export the AutoFocus artifacts. Include
you the AutoFocus API key, the label of the export list, and
specify that the list should be formatted for a PAN-OS block list.
("panosFormatted":true):
curl -X POST -H "Content-Type: application/json" -d '{
  "apiKey":"<apikey>",
  "panosFormatted":true,
  "label":"<export-list-name>"
}' "https://autofocus.paloaltonetworks.com/api/v1.0/export"
```

The response contains a list of IP addresses, domains, or URLs, depending on the artifacts you save:

```
{
  "export_list": [
    "176.31.112.10",
    "31.220.43.99",
    "40.76.58.209",
    "62.113.232.196",
    "95.215.47.207"
  ],
  "bucket_info": {
    "minute_points": 200,
    "daily_points": 100000
  }
}
```

**STEP 2 |** Host the export list as a text file on an external web server. To ensure that you have the latest list of artifacts, frequently refresh the hosted list.

**STEP 3 |** Add the URL for the export list to an external dynamic list. In this example the external dynamic list uses IP addresses:

```
https://<firewall>/api/?type=config&action=set&xpath=/config/devices/
entry[@name='localhost.localdomain']/vsys/entry[@name='vsys1']/external-
```

---

```
list/entry[@name='export-list-name']/type/ip&element=<url><edl-list-url></url><recurring><five-minute/></recurring>&key=<apikey>
```

**STEP 4** | Add the external dynamic list as match criteria in a security policy rule. In this example, the rule denies access to IP addresses on the external dynamic list for all users on your network:

```
https://<firewall>/api/?type=config&action=set&xpath=/config/devices/entry[@name='localhost.localdomain']/vsys/entry[@name='vsys1']/rulebase/security/rules/entry[@name='<security-policy-rulename>']@element=<to><member>any</member></to><from><member>any</member></from><source>any</source><destination><member><edl-list-name></member></destination><source-user><member>any</member></source-user><service><member>application-default</member></service><action>deny</action>&key=<apikey>
```

**STEP 5** | Commit the changes to the firewall:

```
https://<firewall>/api/?type=commit&cmd=<commit></commit>&key=<apikey>
```

You must commit only once when you add the reference to the EDL in a policy rule. Any changes to the external dynamic list do not require a commit.

---

# Configure SAML 2.0 Authentication (API)

Use the PAN-OS XML API to automate the configuration of SAML 2.0 single sign-on (SSO) and single logout (SLO). To configure SAML using the API, create scripts that import the SAML metadata file, create a SAML authentication profile, add users and user groups, and assign the authentication profile to firewall services. The following workflow provides an example of how to configure SAML using the XML API.

## STEP 1 | (Recommended) Import a metadata file from the IdP

The metadata file contains registration information and the certificate that the IdP uses to sign SAML messages. If you import a metadata file, you do not need to independently [Create a SAML Identity Provider \(IdP\) server profile](#). Include the metadata file path and SAML server profile name in your GET request:

- **key:** API key
- **file:** file path to SAML metadata file. The metadata file contains registration information, as well as the certificate that the IdP uses to sign SAML messages. Export the metadata file from the IdP to a client system that the firewall can access. The certificate specified in the file must meet [SAML](#) requirements. Refer also to your IdP documentation for instructions.
- **profile-name:** passphrase, up to 31 characters

```
curl -k -F file=@filename.txt -g 'https://<firewall>/api/?key=apikey&type=import&category=idp-metadata&profile-name=<profilename>'
```

If you perform this step, you can skip Step 2, [Create a SAML Identity Provider \(IdP\) server profile](#).

## STEP 2 | Create a SAML Identity Provider (IdP) server profile

If you do not import a metadata file, include IdP configuration parameters in your GET request to create a SAML IdP server profile:

- **key:** API key
- **vsys:** location, example values: shared, vsys1, vsys2
- **name:** server profile name
- **entity-id:** identity provider id
- **certificate:** ([Best Practice](#)) identity provider certificate
- **sso-url:** identity provider SSO URL
- **slo-url:** identity provider SLO URL
- **sso-binding:** SSO SAML HTTP binding, acceptable values: post, redirect
- **ssl-binding:** SSL SAML HTTP binding, acceptable values: post, redirect
- **max-clock-skew:** difference in system time as measured in seconds between firewall and IdP. The default value is 60 with a range of 1-900.
- **validate-idp-certificate:** ([Best Practice](#)) specify whether you want to validate the IdP certificate. The default value is yes.
- **want-auth-requests-signed:** specify whether the IdP expects a digital signature on authentication requests. The default value is no.

```
https://<firewall>/api/?key=<apikey>&type=config&action=set&xpath=/config/shared/server-profile/saml-idp/entry[@name='<server-profile-name>']&element=<certificate><cert-name></certificate><entity-id><https://example.com/sso></entity-id><sso-url><https://example.com/sso></sso-url><sso-bindings><post></sso-bindings><slo-url><https://example.com/slo></slo-url><slo-bindings>post</slo-bindings><max-clock-skew><max-
```

```
clock-skew></max-clock-skew><validate-idp-certificate><yes></validate-idp-certificate><want-auth-requests-signed><yes></want-auth-requests-signed>
```

### STEP 3 | Create a SAML authentication profile using the PAN-OS XML API

Include SAML authentication profile parameters in your GET request:

- **key:** API key
- **authentication-profile:** authentication profile name
- **enable-single-logout:** specify whether you want to enable SAML single logout. The default value is no.
- **request-signing-certificate:** request signing certificate name
- **server-profile:** SAML Identity Provider (IdP) server profile name
- **certificate-profile:** certificate profile name
- **attribute-name-username:** SAML username attribute
- **attribute-name-usergroup:** SAML user group attribute
- **attribute-name-access-domain:** SAML admin domain attribute
- **attribute-name-admin-role:** SAML admin role attribute

```
https://<firewall>/api/?key=<apikey>&type=config&action=set&xpath=/config/shared/authentication-profile/entry[@name='<authentication-profile-name>']/method/saml-idp&element=<enable-single-logout>no</enable-single-logout><request-signing-certificate><certificate-name></request-signing-certificate><server-profile><server-profile-name></server-profile><certificate-profile>profile-name</certificate-profile><attribute-name-username><username></attribute-name-username><attribute-name-usergroup><usergroup></attribute-name-usergroup><attribute-name-access-domain><access-domain></attribute-name-access-domain><attribute-name-admin-role><admin-role></attribute-name-admin-role>
```

### STEP 4 | Add users and user groups that are allowed to authenticate with this authentication profile

Include profile name and member list in your request:

- **key:** API key
- **authentication-profile:** authentication profile name
- **member:** users or user groups. To include specific users or group, include them in brackets: [member1, member 3]. To include all users, include **all**.

```
https://<firewall>/api/?key=<apikey>&type=config&action=set&xpath=/config/shared/authentication-profile/entry[@name='<authentication-profile-name>']/allow-list&element=<member><all></member>
```

### STEP 5 | Assign the authentication profile to firewall services that require authentication

For example, to assign the authentication profile to a superuser administrator account for web access, include these parameters in your GET request:

- **key:** API key
- **name:** admin username
- **authentication-profile:** name of the SAML authentication profile

```
https://<firewall>/api/?key=<apikey>&type=config&action=set&xpath=/config/mgt-config/users/entry[@name='<adminname>']/&element=<permissions><role-based><superuser>yes</superuser></role-based></permissions><authentication-profile><authprofilename></authentication-profile>
```

# ***PAN-OS XML API Request Types***

The following topics provide common request examples that you can use to better understand the PAN-OS XML API.

- > PAN-OS XML API Request Types and Actions on page 49
- > Asynchronous and Synchronous Requests to the PAN-OS XML API on page 51
- > Configuration (API) on page 52
- > Commit Configuration (API) on page 61
- > Run Operational Mode Commands (API) on page 64
- > Get Reports (API) on page 68
- > Export Files (API) on page 72
- > Import Files (API) on page 77
- > Retrieve Logs (API) on page 79
- > Apply User-ID Mapping and Populate Dynamic Address Groups (API) on page 82
- > Get Version Info (API) on page 86





---

# PAN-OS XML API Request Types and Actions

Use PAN-OS XML API to run various requests depending on the request type that you specify:

- [Request Types](#) on page 49
- [Configuration Actions](#) on page 49

## Request Types

You can currently use the following request types:

Syntax	Description
<code>type=keygen</code>	Generate API keys for authentication.
<code>type=config</code>	Modify the configuration.
<code>type=commit</code>	Commit firewall configuration, including partial commits.
<code>type=op</code>	Perform operational mode commands, including checking system status and validating configurations.
<code>type=report</code>	Get reports, including predefined, dynamic, and custom reports.
<code>type=log</code>	Get logs, including traffic, threat, and event logs.
<code>type=import</code>	Import files including configurations and certificates.
<code>type=export</code>	Export files including packet captures, certificates, and keys.
<code>type=user-id</code>	Update User-ID mappings.
<code>type=version</code>	Show the PAN-OS version, serial number, and model number.

## Configuration Actions

In addition to the request type that you specify, use available actions to modify or read configurations using `type=config`:

- [Actions for Modifying a Configuration](#) on page 49
- [Actions for Reading a Configuration](#) on page 50

### *Actions for Modifying a Configuration*

Configuration Action Type	Syntax
Set candidate configuration	<code>action=set</code>
Edit candidate configuration	<code>action=edit</code>

Configuration Action Type	Syntax
Delete candidate object	<b>action=delete</b>
Rename a configuration object	<b>action=rename</b>
Clone a configuration object	<b>action=clone</b>
Move a configuration object	<b>action=move</b>
Override a template setting	<b>action=override</b>
Move multiple objects in a device group or virtual system	<b>action=multi-move</b>
Clone multiple objects in a device group or virtual system	<b>action=multi-clone</b>
Show available subnode values and XPath's for a given XPath.	<b>action=complete</b>

Set and edit actions differ in two important ways:

- Set actions add, update, or merge configuration nodes, while edit actions replace configuration nodes.
- Set actions are non-destructive and are only additive, while edit actions can be destructive.

## *Actions for Reading a Configuration*

Configuration Action Type	Syntax
Get active configuration	<b>action=show</b>
Get candidate configuration	<b>action=get</b>

Show and get actions differ in three important ways:

- Show actions retrieve the active configuration, while get actions retrieve the candidate, uncommitted configuration.
- Show actions only work when the provided XPath specifies a single node. Get actions work with single and multiple nodes.
- Show actions can use relative XPath, while get actions require absolute XPath.

---

# Asynchronous and Synchronous Requests to the PAN-OS XML API

Most PAN-OS XML API requests are synchronous, meaning the response immediately provides the requested data. For example, when you [Make Your First API Call](#) and request system information, the API response is immediate and contains information such as the IP address, hostname, and model of your firewall.

However, there are some [Request Types](#) that require more time to process and are asynchronous, meaning they require more than one request to get final results. These API requests include the following:

- [Get Reports \(API\)](#)
- [Retrieve Logs \(API\)](#)
- [Export Technical Support Data](#)
- Some requests to [Run Operational Mode Commands \(API\)](#), including download, upgrade, and installation requests

With asynchronous requests, you first initiate a request. The API responds with a job ID while it processes your request. In your subsequent requests, you use this job ID to check on the results of your original request.

Examples (replace `jobid` with the actual job ID):

- Get reports:

```
https://firewall/api/?type=report&action=get&job-id=jobid
```

- Retrieve logs:

```
https://firewall/api/?type=log&action=get&job-id=jobid
```

- Export technical support data:

```
https://firewall/api/?type=export&category=tech-support&action=get&job-id=jobid
```

- Commit:

```
https://firewall/api/?type=op&cmd=<show><jobs><id>jobid</id></jobs></show>
```

- Operational commands:

```
https://firewall/api/?type=op&action=get&job-id=jobid
```

---

# Configuration (API)

The requests examples in these topics illustrate how you can use the PAN-OS XML API to configure your firewall.

- [Get Active Configuration](#) on page 52
- [Get Candidate Configuration](#) on page 54
- [Set Configuration](#) on page 55
- [Edit Configuration](#) on page 56
- [Delete Configuration](#) on page 57
- [Rename Configuration](#) on page 57
- [Clone Configuration](#) on page 58
- [Move Configuration](#) on page 58
- [Override Configuration](#) on page 58
- [Multi-Move or Multi-Clone Configuration](#) on page 59
- [View Configuration Node Values for XPath](#) on page 59

## Get Active Configuration

- [Use XPath to Get Active Configuration](#) on page 52
- [Use XPath to Get ARP Information](#) on page 53

### *Use XPath to Get Active Configuration*

Use **action=show** with no additional parameters to retrieve the entire active configuration.

**STEP 1 |** Use the **xpath** parameter to target a specific portion of the configuration. For example, to retrieve just the security rulebase: **xpath=/config/devices/entry/vsys/entry/rulebase/security**:

```
https://<firewall>/api/?type=config&action=show&key=apikey&xpath=/config/  
devices/entry/vsys/entry/rulebase/security
```

There is no trailing backslash character at the end of the XPath.

**STEP 2 |** Confirm that the XML response for the query looks similar to the following (truncated):

```
<response status="success">  
  <result>  
    <security>  
      <rules>  
        <entry name="IT DNS Services">  
          <profile-setting>  
            <group>  
              <member>best-practice</member>  
            </group>  
          </profile-setting>  
          <to>  
            <member>untrust</member>  
          </to>  
          <from>  
            <member>trust</member>  
          </from>  
          <source>
```

```

        <member>any</member>
      </source>
      <destination>
        <member>Data Center</member>
      </destination>
      <source-user>
        <member>any</member>
      </source-user>
      <category>
        <member>any</member>
      </category>
      <application>
        <member>dns</member>
      </application>
      <service>
        <member>application-default</member>
      </service>
      <hip-profiles>
        <member>any</member>
      </hip-profiles>
      <action>allow</action>
      <tag>
        <member>Best Practice</member>
      </tag>
      <log-start>no</log-start>
      <log-setting>default</log-setting>
    </entry>
  ...
</rules>
</security>
</result>
</response>

```

## Use XPath to Get ARP Information

Follow this procedure to use XPath to Get ARP Information.

**STEP 1** | Use the following request to retrieve ARP information:

```
https://<firewall>//api/?type=op&command=<show><arp><entry name='all'></arp></show>
```

**STEP 2** | Confirm that the XML response for the query looks like the following (truncated):

```

<response status="success">
  <result>
    <max>3000</max>
    <total>16</total>
    <timeout>1800</timeout>
    <dp>dp0</dp>
    <entries>
      <entry>
        <status>c</status>
        <ip>10.47.0.1</ip>
        <mac>00:1b:17:00:2f:13</mac>
        <ttl>1743</ttl>
        <interface>ethernet1/1</interface>
        <port>ethernet1/1</port>
      </entry>
    <entry>

```

```

<status>c</status>
<ip>10.47.0.10</ip>
<mac>00:50:56:93:68:6f</mac>
<ttl>386</ttl>
<interface>ethernet1/1</interface>
<port>ethernet1/1</port>
</entry>
<!-- truncated -->
</result>
</response>

```

## Get Candidate Configuration

Get the candidate configuration from a firewall by specifying the portion of the configuration to get. Use the following request, including the **xpath** parameter to specify the portion of the configuration to get.

```
https://<firewall>/api/?type=config&action=get&xpath=<path-to-config-node>
```

Configuration Node	API Request
Firewall candidate configuration	<pre>https://&lt;firewall&gt;/api/? type=config&amp;action=get&amp;xpath=/config/devices/ entry/vsys/entry[@name='vsys1']&amp;key=&lt;api_key&gt;</pre>
Firewall candidate configuration through Panorama	<pre>https://&lt;panorama&gt;/api/? type=config&amp;action=get&amp;xpath=/ config/devices/entry/vsys/ entry[@name='vsys1']&amp;target=&lt;serial&gt;&amp;key=&lt;panorama_api_key&gt;</pre>
Firewall candidate configuration through Panorama without specifying a firewall	<pre>https://&lt;panorama&gt;/api/? type=config&amp;action=get&amp;xpath=/config/devices/ entry/*[name()!='vsys'] /config/devices/entry/ vsys/entry[@name='vsys1']&amp;key=&lt;panorama_api_key&gt;</pre>
Address objects in a virtual system (vsys).	<pre>https://&lt;firewall&gt;/api/? type=config&amp;action=get&amp;xpath=/config/devices/ entry/vsys/entry[@name='vsys1']/address</pre> <p>The response looks similar to the following:</p> <pre> &lt;response status="success" code="19"&gt;   &lt;result total-count="1" count="1"&gt;     &lt;address admin="name" dirtyId="8" time="2015/10/20 15:32:36"&gt;       &lt;entry name="testobject"&gt;         &lt;ip-netmask&gt;2.2.2.2&lt;/ip-netmask&gt;       &lt;/entry&gt;       &lt;entry name="test1"&gt;         &lt;ip-netmask&gt;1.1.1.1&lt;/ip-netmask&gt;       &lt;/entry&gt;       ...     &lt;/address&gt;   &lt;/result&gt; &lt;/response&gt; </pre>

Configuration Node	API Request
	<pre>&lt;/result&gt; &lt;/response&gt;</pre>
Pre-rules pushed from Panorama.	<pre>https://&lt;firewall&gt;/api/? type=config&amp;action=get&amp;xpath=/config/panorama/ vsys/entry[@name='vsys']/pre-rulebase/security</pre>
Full list of all applications.	<pre>https://&lt;firewall&gt;/api/? type=config&amp;action=get&amp;xpath=/config/predefined/ application</pre>
Details on the specific application.	<pre>https://&lt;firewall&gt;/api/? type=config&amp;action=get&amp;xpath=/config/predefined/ application/entry[@name='hotmail']</pre>

## Set Configuration

Use **action=set** to add or create a new object at a specified location in the PAN-OS configuration. Use the **xpath** parameter to specify the location of the object in the configuration. For example, if you are adding a new rule to the security rulebase, the xpath-value would be:

```
/config/devices/entry[@name='localhost.localdomain']/vsys/
entry[@name='vsys1']/rulebase/security
```

Use the **element** parameter to specify a value for the object you are adding or creating using XML.

Configuration Node	API Request
Create a new rule called rule1 in security policy	<pre>https://&lt;firewall&gt;/api/? key=apikey&amp;type=config&amp;action=set&amp;key=keyvalue&amp;xpath=xpath- value&amp;element=element-value</pre> <p>where the xpath-value is:</p> <pre>/config/devices/entry/vsys/entry/rulebase/ security/rules/entry[@name='rule1']</pre> <p>and the element-value is:</p> <pre>&lt;source&gt;&lt;member&gt;src&lt;/member&gt;&lt;/ source&gt;&lt;destination&gt;&lt;member&gt;dst&lt;/member&gt;&lt;/ destination&gt;&lt;service&gt;&lt;member&gt;service&lt;/member&gt;&lt;/ service&gt;&lt;application&gt;&lt;member&gt;application&lt;/ member&gt;&lt;/application&gt;&lt;action&gt;action&lt;/ action&gt;&lt;source-user&gt;&lt;member&gt;src-user&lt;/member&gt;&lt;/ source-user&gt;&lt;option&gt;&lt;disable-server-response- inspection&gt;yes-or-no&lt;/disable-server-response- inspection&gt;&lt;/option&gt;&lt;negate-source&gt;yes-or- no&lt;/negate-source&gt;&lt;negate-destination&gt;yes-or- no&lt;/negate-destination&gt;&lt;disabled&gt;yes-or-no&lt;/</pre>

Configuration Node	API Request
	<pre>disabled&gt;&lt;log-start&gt;yes-or-no&lt;/log-start&gt;&lt;log-end&gt;yes-or-no&lt;/log-end&gt;&lt;description&gt;description&lt;/description&gt;&lt;from&gt;&lt;member&gt;src-zone&lt;/member&gt;&lt;/from&gt;&lt;to&gt;&lt;member&gt;dst-zone&lt;/member&gt;&lt;/to&gt;</pre>
Add an additional member to an address group or list	<p>Include the 'list' node in the xpath using the <b>member[ text() = 'name' ]</b> syntax and include the members in the element parameter. For example, to add an additional static address object named <b>abc</b> to an address group named <b>test</b>, use:</p> <pre>https://&lt;firewall&gt;/api/?type=config&amp;action=set&amp;xpath=/config/devices/entry/vsys/entry[@name='vsys1']/address-group/entry[@name='test']&amp;element=&lt;static&gt;&lt;member&gt;abc&lt;/member&gt;&lt;/static&gt;</pre>
Create a new IP address on a specific interface	<p>Specify the interface and IP address in the request:</p> <pre>https://&lt;firewall&gt;/api?key=&lt;apikey&gt;&amp;type=config&amp;action=set&amp;xpath=/config/devices/entry[@name='localhost.localdomain']/network/interface/ethernet/entry[@name='ethernet1/1']/layer3/ip&amp;element=&lt;entry name='5.5.5.5/24'&gt;</pre>
Enable or disable a security rule	<pre>https://&lt;firewall&gt;/api?key=&lt;apikey&gt;&amp;type=config&amp;action=set&amp;xpath=/config/devices/entry[@name='localhost.localdomain']/vsys/entry[@name='&lt;vsys1&gt;']/rulebase/security/rules/entry[@name='&lt;rule-name&gt;']&amp;element=&lt;disabled&gt;yes&lt;/disabled&gt;</pre> <p>Alternatively, use <b>&lt;disabled&gt;no&lt;/disabled&gt;</b> to enable a rule.</p>

## Edit Configuration

Use **action=edit** to replace an existing object hierarchy at a specified location in the configuration with a new value. Use the **xpath** parameter to specify the location of the object, including the node to be replaced. Use the **element** parameter to specify a new value for the object using its XML object hierarchy (as seen in the output of **action=show**).

**STEP 1** | Replace the application(s) currently used in a rule rule1 with a new application:

```
https://<firewall>/api/?type=config&action=edit&key=apikey&xpath=xpath-value&element=element-value
```

where

```
xpath=/config/devices/entry/vsys/entry/rulebase/security/rules/entry[@name='rule1']/application&element=<application><member>app-name</member></application>
```



**STEP 2** | Use the response from the config show API request to create the XML body for the element.

```
https://<firewall>/api/?type=config&action=show
```

**STEP 3** | Optionally replace all members in a node with a new set of members using the entry tag in both the xpath and element parameters. For example, to replace all the address objects in the address group named test with two new static members named abc and xyz, use:

```
https://<firewall>/api/?type=config&action=edit&xpath=/  
config/devices/entry/vsys/entry[@name='vsys1']/address-group/  
entry[@name='test']&element=<static><entry name='test'><member>abc</  
member><member>xyz</member></entry></static>
```

## Delete Configuration

Use **action=delete** to delete an object at a specified location in the configuration. Use the **xpath** parameter to specify the location of the object to be deleted.

- Delete a rule named rule1 in the security policy:

```
https://<firewall>/api/?type=config&action=delete&xpath=/config/devices/  
entry/vsys/entry/rulebase/security/rules/entry[@name='rule1']
```

- Delete a single member object in a group, use the object name in the xpath as **member[text()='name']**. For example, to delete a static address object named abc in an address group named test, use the following xpath:

```
https://<firewall>/api/?type=config&action=delete&xpath=/config/devices/  
entry/vsys/entry[@name='vsys1']/address-group/entry[@name='test']/static/  
member[text()='abc']
```

## Rename Configuration

Use **action=rename** to rename an object at a specified location in the configuration. Use the **xpath** parameter to specify the location of the object to be renamed. Use the **newname** parameter to provide a new name for the object.

**STEP 1** | Use the following API query to rename an address object called **old\_address** to **new\_address**:

```
https://<firewall>/api/?type=config&action=rename&xpath=/  
config/devices/entry/vsys/entry[@name='vsys1']/address/  
entry[@name='old_address']&newname=new_address
```

**STEP 2** | Confirm that the XML response for the request looks like the following:

```
<response status="success" code="20"><msg>command succeeded</msg></  
response>
```

---

## Clone Configuration

Use **action=clone** to clone an existing configuration object. Use the **xpath** parameter to specify the location of the object to be cloned. Use the **from** parameter to specify the source object, and the **newname** parameter to provide a name for the cloned object.

**STEP 1** | Use the following API query to clone a security policy called rule1 to rule2:

```
https://<firewall>/api/?type=config&action=clone&xpath=/config/devices/entry/vsys/entry[@name='vsys1']/rulebase/security/rules&from=/config/devices/entry/vsys/entry[@name='vsys1']/rulebase/security/rules/entry[@name='rule1']&newname=rule2
```

**STEP 2** | Confirm that the XML response for the request looks like the following:

```
<response status="success" name="rule2"/>
```

A corresponding success log is recorded in the Configuration log:

```
1,2014/03/19 19:07:45,0009C100708,CONFIG,0,0,2014/03/19
19:07:45,10.66.18.1,,clone,admin,Web,Succeeded, config devices entry vsys
vsys1 rulebase security rules,384,0x8000000000000000
```

## Move Configuration

Use **action=move** to move the location of an existing configuration object. Use the **xpath** parameter to specify the location of the object to be moved, the **where** parameter to specify type of move, and **dst** parameter to specify the destination path.

- **where=after&dst=xpath**
- **where=before&dst=xpath**
- **where=top**
- **where=bottom**

**STEP 1** | Use the following API query to move a security policy called rule1 to come after rule2:

```
https://<firewall>/api/?type=config&action=move&xpath=/config/devices/entry/vsys/entry[@name='vsys1']/rulebase/security/rules/entry[@name='rule1']&where=after&dst=rule2
```

**STEP 2** | Confirm that the XML response for the request looks like the following:

```
<response status="success" code="20"><msg>command succeeded</msg></response>
```

## Override Configuration

Use **action=override** to override a setting that was pushed to a firewall from a template. Use the **xpath** parameter to specify the location of the object to override.

**STEP 1** | Override the SNMP Trap profile configuration settings that were pushed to the firewall using a template:

```
https://<firewall>/api/?type=config&action=override&xpath=/config/shared/log-settings/snmptrap&element=<entry name="snmp" src="tpl"><version src="tpl"><v2c src="tpl"><server src="tpl"><entry name="test" src="tpl"><manager src="tpl">2.2.2.2</manager><community src="tpl">test</community></entry></server></v2c></version></entry>
```

**STEP 2** | Confirm that the XML response for the request looks like the following:

```
<response status="success" code="20"><msg>command succeeded</msg></response>
```

## Multi-Move or Multi-Clone Configuration

Use the **action=multi-move** and **action=multi-clone** actions to move and clone addresses across device groups and virtual systems. Templates do not support the multi-move and multi-clone capability.

The syntax for multi-move and multi-clone specifies the xpath for the destination where the addresses will be moved to, the xpath for the source and the list of objects within the specified source. It also includes a flag for displaying the errors when the firewall performs a referential integrity check on the multi-move or multi-clone action.

- Move addresses **addr1**, **addr2**, to device group **norcal** from device group **socal**:

```
https://<firewall>/api/?type=config&action=multimove&xpath=/config/devices/entry[@name='localhost.localdomain']/devicegroup/entry[@name='norcal']/address&element=<selected-list><source xpath="/config/devices/entry[@name='localhost.localdomain']/devicegroup/entry[@name='socal']/address"><member>addr1</member><member>addr2</member></source></selected-list><all-errors>no</all-errors>
```

- Clone addresses **addr1**, **addr2**, to device group **norcal** from device group **socal**:

```
https://<firewall>/api/?type=config&action=multiclon&xpath=/config/devices/entry[@name='localhost.localdomain']/devicegroup/entry[@name='norcal']/address&element=<selected-list><source xpath="/config/devices/entry[@name='localhost.localdomain']/devicegroup/entry[@name='socal']/address"><member>addr1</member><member>addr2</member></source></selected-list><all-errors>no</all-errors>
```

## View Configuration Node Values for XPath

Use **action=complete** action along with an XPath to see possible values that are available with the XPath node.

**STEP 1** | View the possible values, such as network interfaces, for multi-vsyz firewalls, use the following command:

```
https://<firewall>/api/?type=config&action=complete&xpath=/config/devices/entry[@name='localhost.localdomain']/vsyz&key=apikey
```

---

**STEP 2** | Confirm that the XML response for the request looks like the following:

```
<response status="success" code="19">
  <completions>
    <completion value="vsys1" vxpath="/config/devices/
entry[@name='localhost.localdomain']/vsys/entry[@name='vsys1']"
current="yes" help-string="vsys1"/>
  </completions>
</response>
```

---

# Commit Configuration (API)

You can use the commit API request to commit a candidate configuration to a firewall.



You can validate or revert a candidate configuration before committing it using [Run Operational Mode Commands \(API\)](#) on page 64.

- [Commit](#) on page 61
- [Commit-All](#) on page 62

## Commit

Replace the **body** element in the **cmd** parameter with the XML element for the corresponding commit operation.



Use the [API Browser](#) to find different options available for use with force and partial commits.

**STEP 1** | Use one of the following requests to commit a configuration:

- **Commit**— Commit candidate changes to the firewall.

```
https://<firewall>/api/?type=commit&cmd=<commit></commit>
```

- **Force Commit**—

```
https://<firewall>/api/?type=commit&cmd=<commit><force></force></commit>
```

- **Partial commit while excluding shared objects and device and network configuration**—

```
https://<firewall>/api/?  
type=commit&action=partial&cmd=<commit><partial><device-and-  
network>excluded</device-and-network><shared-object>excluded</shared-  
object></partial>
```

- **Partial commit admin-level changes**— To commit admin-level changes on a firewall, include the administrator name in the request.

```
https://<firewall>/api/?  
&type=commit&action=partial&cmd=<commit><partial><admin><member><admin-  
name></member></admin></partial>
```

- **Partial commit admin-level changes on a firewall or Panorama while excluding shared objects**— Include the administrator name in the request.

```
https://<firewall>/api/?  
&type=commit&action=partial&cmd=<commit><partial><device-and-  
network>excluded</device-and-network><shared-object>excluded</shared-  
object><admin><member><admin-name></member></admin></partial>
```

**STEP 2** | Confirm that the XML response indicates that there were no changes to commit or that the changes are queued for commit:

- No pending changes to commit:

```
<response status="success" code="19">
  <msg>There are no changes to commit.</msg>
</response>
```

- Pending changes:

```
<response status="success" code="19">
  <result>
    <msg>
      <line>Commit job enqueued with jobid 4</line>
    </msg>
    <job>4</job>
  </result>
</response>
```

**STEP 3** | Query the status of the job using the job ID:

```
https://<firewall>/api/?type=op&cmd=<show><jobs><id>4</id></jobs></show>
```

**STEP 4** | Confirm that the XML response details state the Configuration was committed successfully:

```
<response status="success">
  <result>
    <job>
      <tenq>2011/10/20 20:41:44</tenq>
      <id>4</id>
      <type>Commit</type>
      <status>FIN</status>
      <stoppable>no</stoppable>
      <result>OK</result>
      <tfin>20:42:22</tfin>
      <progress>20:42:22</progress>
      <details>
        <line>Configuration committed successfully</line>
      </details>
      <warnings />
    </job>
  </result>
</response>
```

## Commit-All

To centrally manage firewalls from Panorama, use the commit-all API request type to push and validate shared policy to the firewalls using device groups and configuration to Log Collectors and firewalls using templates or template stacks.

Commit Type	API Request
Pre-commit policy validation.	<pre>https://&lt;panorama&gt;/api/? type=commit&amp;action=all&amp;cmd= &lt;commit-all&gt;&lt;shared-</pre>

Commit Type	API Request
	<pre>policy&gt;&lt;validate-only&gt;&lt;/validate-only&gt;&lt;/shared-policy&gt;&lt;/commit-all&gt;</pre>
Specific Device group commit.	<pre>https://&lt;panorama&gt;/api/? type=commit&amp;action=all&amp;cmd=&lt;commit-all&gt;&lt;shared-policy&gt;&lt;device-group&gt;&lt;entry name="&lt;device-group-name&gt;"&gt;&lt;/device-group&gt;&lt;/shared-policy&gt;&lt;/commit-all&gt;</pre>
Virtual system (vsys) commit.	<pre>https://&lt;panorama&gt;/api/? type=commit&amp;action=all&amp;cmd=&lt;commit-all&gt;&lt;shared-policy&gt;&lt;device-group&gt;&lt;entry name="&lt;device-group-name&gt;"&gt;&lt;devices&gt;&lt;entry name="&lt;serial_number&gt;"&gt;&lt;vsys&gt;&lt;member&gt;vsys-name&lt;/member&gt;&lt;/vsys&gt;&lt;/entry&gt;&lt;/devices&gt;&lt;/device-group&gt;&lt;/shared-policy&gt;&lt;/commit-all&gt;</pre>
Specific firewall commit.	<pre>https://&lt;panorama&gt;/api/? type=commit&amp;action=all&amp;cmd=&lt;commit-all&gt;&lt;shared-policy&gt;&lt;device-group&gt;&lt;entry name="&lt;device-group-name&gt;"&gt;&lt;devices&gt;&lt;entry name="&lt;serial_number&gt;"&gt;&lt;/devices&gt;&lt;entry/&gt;&lt;/device-group&gt;&lt;/shared-policy&gt;&lt;/commit-all&gt;</pre>

Use the [API Browser](#) to find other options available for granular commit operations on Panorama. In the **cmd** parameter, you must replace the XML element for the corresponding **commit-all** operation.

# Run Operational Mode Commands (API)


Use any of the operational mode commands available on the command line interface with the following API request:

```
https://<firewall>/api/?type=op&cmd=<xml-body>
```

Use the [API Browser](#) to explore operational mode commands and a complete listing of all the options available for the `xml-body` and their corresponding operation.



Some requests operational mode commands, including download, upgrade, and installation requests, are asynchronous, meaning they require more than one request to get final results. Learn more about [Asynchronous and Synchronous Requests to the PAN-OS XML API](#).

Operational Command	API Request
System restart.	<pre>https://&lt;firewall&gt;/api/? type=op&amp;cmd=&lt;request&gt;&lt;restart&gt;&lt;system&gt;&lt;/system&gt;&lt;/ restart&gt;&lt;/request&gt;</pre>
System software version installation.	<pre>https://&lt;firewall&gt;/api/? type=op&amp;cmd=&lt;request&gt;&lt;system&gt;&lt;software&gt;&lt;install&gt;&lt;version&gt;versi version&gt;&lt;/install&gt;&lt;/software&gt;&lt;/system&gt;&lt;/request&gt;</pre>
Multi-vsyst mode.	<pre>https://&lt;firewall&gt;/api/? type=op&amp;cmd=&lt;set&gt;&lt;system&gt;&lt;setting&gt;&lt;multi-vsyst&gt;&lt;/ multi-vsyst&gt;&lt;/setting&gt;&lt;/system&gt;&lt;/set&gt;</pre>
User Activity Report scheduling.	<pre>https://&lt;firewall&gt;/api/? type=op&amp;cmd=&lt;schedule&gt;&lt;uar-report&gt;&lt;user&gt;username&lt;/ user&gt;&lt;title&gt;titlename&lt;/title&gt;&lt;/uar-report&gt;&lt;/ schedule&gt;</pre>
Detailed information on applications and threats from the firewall.	<pre>https://&lt;firewall&gt;/api/? type=op&amp;cmd=&lt;show&gt;&lt;predefined&gt;&lt;xpath&gt;/predefined/ threats/vulnerability/entry[@name='30003']&lt;/ xpath&gt;&lt;/predefined&gt;&lt;/show&gt;</pre> <p> Only PAN-OS 8.0.4 and later releases support this API call.</p>
Full configuration validation.	<pre>https://&lt;firewall&gt;/api/? type=op&amp;cmd=&lt;validate&gt;&lt;full&gt;&lt;/full&gt;&lt;/validate&gt;</pre>
Partial configuration validation.	<pre>https://&lt;firewall&gt;/api/? type=op&amp;cmd=&lt;validate&gt;&lt;partial&gt;&lt;device-and-</pre>



Operational Command	API Request
	<code>network&gt;excluded&lt;/device-and-network&gt;&lt;/partial&gt;&lt;/validate&gt;</code>
Configuration saving.	<code>https://&lt;firewall&gt;/api/? type=op&amp;cmd=&lt;save&gt;&lt;config&gt;&lt;to&gt;filename&lt;/to&gt;&lt;/config&gt;&lt;/save&gt;</code>
Configuration loading.	<code>https://&lt;firewall&gt;/api/? type=op&amp;cmd=&lt;load&gt;&lt;config&gt;&lt;from&gt;filename&lt;/from&gt;&lt;/config&gt;&lt;/load&gt;</code>
Partial revert of admin-level changes for a candidate configuration on a firewall.	<code>https://&lt;firewall&gt;/api/? type=op&amp;cmd=&lt;revert&gt;&lt;config&gt;&lt;partial&gt;&lt;admin&gt;&lt;member&gt;admin-name&lt;/member&gt;&lt;/admin&gt;&lt;/partial&gt;&lt;/config&gt;&lt;/revert&gt;</code>
Partial revert of admin-level changes to Panorama by a specific administrator within a specific device group	<code>https://&lt;panorama&gt;/api/? type=op&amp;cmd=&lt;revert&gt;&lt;config&gt;&lt;partial&gt;&lt;admin&gt;&lt;member&gt;&lt;admin-name&gt;&lt;/member&gt;&lt;/admin&gt;&lt;device-group&gt;&lt;member&gt;&lt;device-group-name&gt;&lt;/member&gt;&lt;/device-group&gt;&lt;no-template/&gt;&lt;no-template-stack/&gt;&lt;no-log-collector-group/&gt;&lt;no-log-collector/&gt;&lt;device-and-network&gt;excluded&lt;/device-and-network&gt;&lt;/partial&gt;&lt;/config&gt;&lt;/revert&gt;</code>
Base64-encoded metadata of a SAML authentication profile.	<code>https://&lt;firewall&gt;/api/?type=op&amp;cmd=&lt;show&gt;&lt;sp-metadata&gt;&lt;management&gt;&lt;authprofile&gt;&lt;SAML-auth-profile-name&gt;&lt;/authprofile&gt;&lt;/management&gt;&lt;/sp-metadata&gt;&lt;/show&gt;</code>
Summary of changes between the active and candidate configuration.	<code>https://&lt;firewall&gt;/api/? type=op&amp;cmd=&lt;show&gt;&lt;config&gt;&lt;list&gt;&lt;change-summary/&gt;&lt;/list&gt;&lt;/config&gt;&lt;/show&gt;</code>
Commit locks	<code>https://&lt;firewall&gt;/api/? key=&lt;apikey&gt;&amp;type=op&amp;cmd=&lt;show&gt;&lt;commit-locks/&gt;&lt;/show&gt;</code>
Show WildFire appliances connected to Panorama.	<code>https://&lt;panorama&gt;/api/? key=&lt;apikey&gt;&amp;&amp;type=op&amp;cmd=&lt;show&gt;&lt;wildfire-appliance&gt;&lt;connected&gt;&lt;/connected&gt;&lt;/wildfire-appliance&gt;&lt;/show&gt;</code>
System summary about WildFire appliances or WildFire clusters.	<ul style="list-style-type: none"> <li>• <b>WildFire Appliance:</b> <pre>https://&lt;panorama&gt;/api/? key=&lt;apikey&gt;&amp;&amp;type=op&amp;cmd=&lt;show&gt;&lt;wildfire-appliance&gt;&lt;all&gt;&lt;/all&gt;&lt;/wildfire-appliance&gt;&lt;/show&gt;</pre> </li> </ul>

Operational Command	API Request
	<ul style="list-style-type: none"> <li>• <b>WildFire Cluster:</b> <pre data-bbox="672 275 1455 401">https://&lt;panorama&gt;/api/? key=&lt;apikey&gt;&amp;&amp;type=op&amp;cmd=&lt;show&gt;&lt;wildfire- appliance-cluster&gt;&lt;all&gt;&lt;/all&gt;&lt;/wildfire- appliance-cluster&gt;&lt;/show&gt;</pre> </li> </ul>
<p>Generate a list of Firewalls connected and sending data to a WildFire appliance or WildFire cluster.</p>	<ul style="list-style-type: none"> <li>• <b>WildFire Appliance:</b> <pre data-bbox="672 495 1455 674">https://&lt;panorama&gt;/api/? key=&lt;apikey&gt;&amp;&amp;type=op&amp;cmd=&lt;show&gt;&lt;wildfire- appliance&gt;&lt;devices-reporting-data&gt;&lt;serial- number&gt;&lt;serial_number&gt;&lt;/serial-number&gt;&lt;/ devices-reporting-data&gt;&lt;/wildfire-appliance&gt;&lt;/ show&gt;</pre> </li> <li>• <b>WildFire Cluster:</b> <pre data-bbox="672 747 1455 926">https://&lt;panorama&gt;/api/? key=&lt;apikey&gt;&amp;&amp;type=op&amp;cmd=&lt;show&gt;&lt;wildfire- appliance-cluster&gt;&lt;devices-reporting- data&gt;&lt;name&gt;&lt;cluster_name&gt;&lt;/name&gt;&lt;/devices- reporting-data&gt;&lt;/wildfire-appliance-cluster&gt;&lt;/ show&gt;</pre> </li> </ul>
<p>Display configuration details about a specified WildFire appliance or WildFire cluster.</p>	<ul style="list-style-type: none"> <li>• <b>WildFire Appliance:</b> <pre data-bbox="672 1031 1455 1178">https://&lt;panorama&gt;/api/? key=&lt;apikey&gt;&amp;&amp;type=op&amp;cmd=&lt;show&gt;&lt;wildfire- appliance&gt;&lt;info&gt;&lt;serial- number&gt;&lt;serial_number&gt;&lt;/serial-number&gt;&lt;/info&gt;&lt;/ wildfire-appliance&gt;&lt;/show&gt;</pre> </li> <li>• <b>WildFire Cluster:</b> <pre data-bbox="672 1251 1455 1398">https://&lt;panorama&gt;/api/? key=&lt;apikey&gt;&amp;&amp;type=op&amp;cmd=&lt;show&gt;&lt;wildfire- appliance-cluster&gt;&lt;info&gt;&lt;name&gt;&lt;cluster_name&gt;&lt;/ name&gt;&lt;/info&gt;&lt;/wildfire-appliance-cluster&gt;&lt;/ show&gt;</pre> </li> </ul>
<p>Display registration activity for a specified WildFire appliance or WildFire cluster.</p>	<ul style="list-style-type: none"> <li>• <b>WildFire Appliance:</b> <pre data-bbox="672 1503 1455 1713">https://&lt;panorama&gt;/api/? key=&lt;apikey&gt;&amp;&amp;type=op&amp;cmd=&lt;show&gt;&lt;wildfire- appliance&gt;&lt;last-device- registration&gt;&lt;all&gt;&lt;serial- number&gt;&lt;serial_number&gt;&lt;/serial-number&gt;&lt;/ all&gt;&lt;/last-device-registration&gt;&lt;/wildfire- appliance&gt;&lt;/show&gt;</pre> </li> <li>• <b>WildFire Cluster:</b> <pre data-bbox="672 1787 1455 1902">https://&lt;panorama&gt;/api/? key=&lt;apikey&gt;&amp;&amp;type=op&amp;cmd=&lt;show&gt;&lt;wildfire- appliance-cluster&gt;&lt;last-device- registration&gt;&lt;all&gt;&lt;name&gt;&lt;cluster_name&gt;&lt;/name&gt;&lt;/</pre> </li> </ul>

Operational Command	API Request
	<pre>all&gt;&lt;/last-device-registration&gt;&lt;/wildfire- appliance-cluster&gt;&lt;/show&gt;</pre>
Display statistics for a specified WildFire appliance or WildFire cluster.	<ul style="list-style-type: none"> <li> <b>WildFire Appliance:</b> <pre>https://&lt;panorama&gt;/api/? key=&lt;apikey&gt;&amp;&amp;type=op&amp;cmd=&lt;show&gt;&lt;wildfire- appliance&gt;&lt;statistics&gt;&lt;days&gt;&lt;days_up_to_31&gt;&lt;/ days&gt;&lt;type&gt;&lt;all_or_file_or_general&gt;&lt;/ type&gt;&lt;serial-number&gt;&lt;serial_number&gt;&lt;/name&gt;&lt;/ statistics&gt;&lt;/wildfire-appliance&gt;&lt;/show&gt;</pre> </li> <li> <b>WildFire Cluster:</b> <pre>https://&lt;panorama&gt;/api/? key=&lt;apikey&gt;&amp;&amp;type=op&amp;cmd=&lt;show&gt;&lt;wildfire- appliance- cluster&gt;&lt;statistics&gt;&lt;hours&gt;&lt;hours_up_to_24&gt;&lt;/ minutes&gt;&lt;type&gt;&lt;all_or_file_or_general&gt;&lt;/ type&gt;&lt;name&gt;&lt;cluster_name&gt;&lt;/name&gt;&lt;/statistics&gt;&lt;/ wildfire-appliance-cluster&gt;&lt;/show&gt;</pre> </li> </ul>
Display a list of supported VM images on the specified WildFire appliance.	<pre>https://&lt;panorama&gt;/api/? key=&lt;apikey&gt;&amp;&amp;type=op&amp;cmd=&lt;show&gt;&lt;wildfire- appliance&gt;&lt;vm-images&gt;&lt;serial- number&gt;&lt;serial_number&gt;&lt;/serial-number&gt;&lt;/vm- images&gt;&lt;/wildfire-appliance&gt;&lt;/show&gt;</pre>

---

# Get Reports (API)

The XML API provides a way to quickly pull the results of any report defined in the system using the **type=report** parameter.

You can access three kinds of reports:

- Dynamic Reports (ACC reports)—**reporttype=dynamic**
- Predefined Reports—**reporttype=predefined**
- Custom Reports—**reporttype=custom**

To retrieve a specific report by name, use the **reportname** parameter:

```
https://<firewall>/api/?type=report&reporttype=dynamic|predefined|  
custom&reportname=<name>
```



*When you request a report, the API responds asynchronously with a job ID, which you can use to retrieve the reports. Learn more about [Asynchronous and Synchronous Requests to the PAN-OS XML API](#).*

- [Dynamic Reports](#)
- [Predefined Reports](#)
- [Custom Reports](#)

## Dynamic Reports

You can use the API to view a number of dynamic reports, such as **top-applications-summary**, **top-blocked-url-summary**, and **top-spyware-threats-summary**. For dynamic reports, provide either a

specific period using the **period** or a time frame using **starttime** and **endtime** options (use a + instead of a space between the date and timestamp). Use **topn** to determine the number of rows.

Dynamic Report Type	API Request
Full dynamic report list.	<code>https://&lt;firewall&gt;/api/? type=report&amp;reporttype=dynamic</code>
Last 60 seconds.	<code>https://&lt;firewall&gt;/api/? type=report&amp;reporttype=dynamic&amp;reportname=top-app- summary&amp;period=last-60-seconds&amp;topn=5</code>
Last 15 minutes.	<code>https://&lt;firewall&gt;/api/? type=report&amp;reporttype=dynamic&amp;reportname=top-app- summary&amp;period=last-15-minutes&amp;topn=5</code>
Last hour.	<code>https://&lt;firewall&gt;/api/? type=report&amp;reporttype=dynamic&amp;reportname=top-app- summary&amp;period=last-hour&amp;topn=5</code>
Last 12 hours.	<code>https://&lt;firewall&gt;/api/? type=report&amp;reporttype=dynamic&amp;reportname=top-app- summary&amp;period=last-12-hrs&amp;topn=5</code>
Last calendar day.	<code>https://&lt;firewall&gt;/api/? type=report&amp;reporttype=dynamic&amp;reportname=top-app- summary&amp;period=last-calendar-day&amp;topn=5</code>
Last 7 days	<code>https://&lt;firewall&gt;/api/? type=report&amp;reporttype=dynamic&amp;reportname=top-app- summary&amp;period=last-7-days&amp;topn=5</code>
Last 7 calendar days	<code>https://&lt;firewall&gt;/api/? type=report&amp;reporttype=dynamic&amp;reportname=top-app- summary&amp;period=last-hour&amp;topn=5</code>
Last calendar week.	<code>https://&lt;firewall&gt;/api/? type=report&amp;reporttype=dynamic&amp;reportname=top-app- summary&amp;period=last-calendar-week&amp;topn=5</code>
Last 30 days	<code>https://&lt;firewall&gt;/api/? type=report&amp;reporttype=dynamic&amp;reportname=top-app- summary&amp;period=last-30-days&amp;topn=5</code>

## Predefined Reports

Predefined reports always return data for the last 24-hour period. You can also get this list by following the link for predefined reports, such as **top-applications**, **top-attackers**, and **bandwidth-trend** on the API browser.

Dynamic Report Type	API Request
Full predefined report list.	<code>https://&lt;firewall&gt;/api/? type=report&amp;reporttype=predefined</code>
Top applications.	<code>https://&lt;firewall&gt;/api/? type=report&amp;async=yes&amp;reporttype=predefined&amp;reportname=top- application-categories</code>
Top attackers.	<code>https://&lt;firewall&gt;/api/? type=report&amp;async=yes&amp;reporttype=predefined&amp;reportname=top- attackers</code>
Top victims.	<code>https://&lt;firewall&gt;/api/? type=report&amp;async=yes&amp;reporttype=predefined&amp;reportname=top- victims</code>

## Custom Reports

For custom reports, the selection criteria, such as time frame, group-by, and sort-by are part of the report definition. The API returns any shared custom reports. Note that quotes are not required around the report name and any spaces in the report name must be URL encoded to %20.

For custom reports created in a specific VSYS, you can retrieve them directly by specifying the **vsys** parameters.

**STEP 1** | Retrieve the report definition from the configuration:

```
https://<firewall>/api/?type=config&action=get&xpath=/config/devices/  
entry/vsys/entry[@name='vsys1']/reports/entry[@name='report-abc']
```

**STEP 2** | Create a job to retrieve a dynamic report using **reporttype=dynamic**, **reportname=custom-dynamic-report**, and **cmd=report-definition** where **report-definition** is the XML definition retrieved in the previous query:

```
https://<firewall>/api/?type=report&reporttype=dynamic&reportname=custom-  
dynamic-report&cmd=<type><appstat><aggregate-by><member>category-  
of-name</member><member>technology-of-name</member></aggregate-by></  
appstat></type><period>last-24-hrs</period><topn>10</topn><topm>10</  
topm><query>(name neq '') AND (vsys eq 'vsys1')</query>
```

The response includes the job ID you can use to view the results:

```
<response  
status="success">
```

---

```
<result>
  <msg>
    <line>Report job enqueued with jobid 6</line>
  </msg>
  <job>6</job>
</result>
</response>
```

**STEP 3** | View the dynamic report:

```
https://<firewall>/api/?type=report&action=get&job-id=jobid
```

---

# Export Files (API)

You can export certain types of files from the firewall using the **type=export** parameter in the API request.

Use the category parameter to specify the type of file that you want to export.

- Configuration—**category=configuration**
- Certificates/Keys—**category=certificate | high-availability-key | key-pair**
- Response pages—**category= application-block-page | captive-portal-text | file-block-continue-page | file-block-page | global-protect-portal-custom-help-page | global-protect-portal-custom-login-page | global-protect-portal-custom-welcome-page | ssl-cert-status-page | ssl-optout-text | url-block-page | url-coach-text | virus-block-page**
- Technical support data—**category=tech-support**
- Device State—**category=device-state**

Use cURL tools to export the file from the firewall and save locally with a local file name:

```
curl -o <filename> "https://<firewall>/api/?<query-parameters>"
```

When using the API query from a web browser, you can specify **to=filename** as an optional parameter if you would like to provide a different name when saving the file locally.

- [Export Packet Captures](#)
- [Export Certificates and Keys](#)
- [Export Technical Support Data](#)

## Export Packet Captures

You can export packet captures from the firewall by specifying the PCAP type using the **category** parameter:

- [Export Application PCAPS](#) on page 73
- [Export Threat, Filter, and Data Filtering PCAPs](#) on page 74



## Export Application PCAPS

Application PCAPs are organized by a directory/filename structure where the directory is a date in **yyyymmdd** format. Filenames for application pcaps use a **SourceIP-SourcePort-DestinationIP-DestinationPort-SessionID.pcap** format.

Application PCAP Type	API Request
Application PCAP directory list.	<pre>https://&lt;firewall&gt;/api/? type=export&amp;category=application-pcap</pre>
List of files under a directory using the <b>from</b> parameter to indicate date.	<pre>https://&lt;firewall&gt;/api/? type=export&amp;category=application- pcap&amp;from=&lt;yyyymmdd&gt;</pre>
Application PCAP file by name using the <b>from</b> parameter.	<pre>https://&lt;firewall&gt;/api/? type=export&amp;category=application- pcap&amp;from=&lt;yyyymmdd&gt;/&lt;filename&gt;</pre> <p>The file will be retrieved and saved locally using the name yyyymmdd-filename.</p>
Application PCAP file saved locally with a custom name using the <b>to</b> parameter.	<pre>https://&lt;firewall&gt;/api/? type=export&amp;category=application- pcap&amp;from=&lt;yyyymmdd&gt;/&lt;filename&gt;&amp;to=&lt;localfile&gt;</pre>

## Export Threat, Filter, and Data Filtering PCAPs

To export threat PCAPs, you need to provide the PCAP ID from the threat log and the search time, which is the time that the PCAP was received on the firewall. Threat PCAP filenames use a `pcapID.pcap` format.

PCAP Type	API Request
Threat PCAP using PCAP ID and search	<pre>https://&lt;firewall&gt;/api/? type=export&amp;category=threat-pcap&amp;pcap- id=&lt;id&gt;&amp;search-time=&lt;yyyy/mm/dd hr:min:sec&gt;</pre>
List of filtered PCAPs	<pre>https://&lt;firewall&gt;/api/? type=export&amp;category=filters-pcap</pre>
Specific filtered PCAP file	<pre>https://&lt;firewall&gt;/api/? type=export&amp;category=filters-pcap&amp;from=&lt;filename&gt;</pre>
List of data filtering PCAP file names	<pre>https://&lt;firewall&gt;/api/?type=export&amp;category=dlp- pcap&amp;dlp-password=&lt;password&gt;</pre>
Specific data filtering PCAP file	<pre>https://&lt;firewall&gt;/api/? type=export&amp;category=dlp-pcap&amp;dlp- password=&lt;password&gt;&amp;from=&lt;filename&gt;&amp;to=&lt;localfile&gt;</pre>

## Export Certificates and Keys

Use the following procedure to export certificates and keys.

**STEP 1** | To export certificates and keys, specify query parameters `certificate-name`, `format`, and `passphrase`:

```
https://<firewall>/api/?type=export&category=<certificate>&certificate-  
name=<certificate_name>&passphrase=<passphrase>&format=<pkcs12>  
| <pem>&include-key=<yes>  
| <no>&vsys=<vsys>  
| <omit this parameter to import it into a shared location>
```

- **certificate-name**—name of the certificate object on the firewall
- **passphrase**—required when including the certificate key
- **format**—certificate format, **pkcs12** or **pem**
- **include-key**—yes or no parameter to include or exclude the key
- **vsys**—virtual system where the certificate object is used. Ignore this parameter if the certificate is a shared object.

**STEP 2** | Confirm that the XML response includes the certificate:

```
-----BEGIN CERTIFICATE-----  
MIIDXTCCAkWgAwIBAgIJAJC1HiIAZAIIMA0GCSqGSIb3Df  
BAYTAKFVMRMwEQYDVQIDApTb211LVN0YXRlMSEwHwYDVx  
aWRnaXRzIFB0eSBMdGQwHhcNMTEzMjMxMDg1OTQ0WhcNMT
```

```
<!-- TRUNCATED -->
-----END CERTIFICATE-----
```

## Export Technical Support Data

Debug log data sizes are large, so the API uses an asynchronous job scheduling approach to retrieve technical support data. Learn more about [Asynchronous and Synchronous Requests to the PAN-OS XML API](#). The values for the action parameter are:

- **action=null**—When an action parameter is not specified, the system creates a new job to retrieve tech support data. The initial query creates a job ID that you can then use to check on the status of the job, retrieve results, or delete the job.
- **action=status**—Check the status of the job. This returns an XML response with a status element; when the status text data is `FIN` the job is completed and the tech support file can be retrieved.

Example:

```
https://<firewall>/api/?type=export&category=tech-support&action=status&job-id=299
```

- **action=get**—Retrieve the tech support file as an attachment. The response contains a `application/octet-stream` content-type and a content-disposition header with a suggested filename; for example:

```
Content-Type: application/octet-stream
Content-Length: 19658186
Content-Description: File Transfer
Content-Transfer-Encoding: binary
Content-Disposition: attachment; filename=techsupport-8469.tgz
```

- **action=finish**—Stop an active job.

### STEP 1 | Create a job to retrieve technical support data.

Use the following request:

```
https://<firewall>/api/?type=export&category=tech-support
```

The response includes a job ID:

```
<response status="success" code="19"> <result> <msg> <line>Exec job enqueued
with jobid 2</line> </msg> <job>2</job> </result> </response>
```

### STEP 2 | Check on the status of the job.

Use the job ID returned in the previous response as the job-id parameter:

```
https://<firewall>/api/?type=export&category=tech-support&action=get&job-id=id
```

A status value of `FIN` indicates the data is ready to be retrieved.

```
<response status="success">
  <result>
    <job>
      <tenq>2012/06/14 10:11:09</tenq>
      <id>2</id>
      <user />
      <type>Exec</type>
      <status>FIN</status>
```

```
<stoppable>no</stoppable>
<result>OK</result>
<tfin>10:12:39</tfin>
<progress>10:12:39</progress>
<details />
<warnings />
<resultfile>//tmp/techsupport.tgz</resultfile>
</job>
</result>
</response>
```

### STEP 3 | Retrieve the tech support data.

```
https://<firewall>/api/?type=export&category=tech-support&action=get&job-id=id
```

When using cURL, you can specify the output file name as an option to cURL (-o). After a successful retrieval of the job data, the job is automatically deleted by the system.

### STEP 4 | (Optional) Stop the active job in case of error.

If there is an error or issue with the export job, it may not complete. In cases like this, stop the active job:

```
https://<firewall>/api/?type=export&category=tech-support&action=finish&job-id=id
```

The response includes a success message:

```
<response status="success">
  <msg>Job 2 removed.</msg>
</response>
```

---

# Import Files (API)

You can import certain types of files, including as software, content, licenses, and configurations into the firewall using the `type=import` parameter in the API request.

Use `type=import` and specify the category to import these types of files:

- `Software—category=software`
- `Content—category=<anti-virus | content | url-database | signed-url-database>`
- `Licenses—category=license`
- `Configuration—category=configuration`
- `Certificates/key—category=<certificate | high-availability-key | key-pair>`
- `Response pages—category=<application-block-page | captive-portal-text | file-block-continue-page | file-block-page | global-protect-portal-custom-help-page | global-protect-portal-custom-login-page | global-protect-portal-custom-welcome-page | ssl-cert-status-page | ssl-optout-text | url-block-page | url-coach-text | virus-block-page>`
- `Clients—category=global-protect-client`
- `Custom logo—category=custom-logo`
- [Importing Basics](#)
- [Import Files](#)

## Importing Basics

Use cURL to import files to the firewall.

- Import files to a firewall:

```
curl --form file=@<filename> "https://firewall/api/?<query-parameters>"
```

- Import files to a firewall via Panorama. First import the file to Panorama, then run a request batch upload-install op command:

```
http://<panorama>/api/?type=op&cmd=<request><batch><anti-virus><upload-install><uploaded-file><your-file-name-here></uploaded-file><devices><serialnumber></devices></upload-install></anti-virus></batch></request>
```

## Import Files

Use the [API Browser](#) to see a full list of import categories.

- Import a certificate or key by specifying the type of the certificate or key file using the `category` parameter:
  - `category=certificate`
  - `category=keypair`
  - `category=high-availability-key`
- (`category=certificate` or `category=keypair` only) Specify these additional parameters for the certificate file and keypair imports:

- 
- **certificate-name**—name of the certificate object on the firewall
  - **format**—certificate format, **pkcs12** or **pem**
  - **passphrase**—required when including the certificate key
  - **vsys**—virtual system where the certificate object is used. Ignore this parameter if the certificate is a shared object.

```
https://<firewall>/api/?type=import&category=certificate&certificate-  
name=<certificate_name>&format=pkcs12 | pem&passphrase=text&vsys=<vsys>
```

- Import a GlobalProtect response pages using an additional parameter for the security profile in which the page should be imported:

```
profile=profilename
```

- Import custom logos to different locations based on the **where** parameter:

```
where=<login-screen | main-ui | pdf-report-footer | pdf-report-header>
```

# Retrieve Logs (API)


Retrieve logs from a firewall using the API.

- [API Log Retrieval Parameters](#) on page 79
- [Example: Use the API to Retrieve Traffic Logs](#) on page 80

## API Log Retrieval Parameters

Specify the log type with additional optional parameters to retrieve logs from a firewall.

Parameter	Description
log-type	<p>The type of logs to retrieve:</p> <ul style="list-style-type: none"><li>• <code>log-type=traffic</code>—Traffic logs</li><li>• <code>log-type=threat</code>—Threat logs</li><li>• <code>log-type=config</code>—Config logs</li><li>• <code>log-type=system</code>—System logs</li><li>• <code>log-type=hipmatch</code>—GlobalProtect Host Information Profile (HIP) matching logs</li><li>• <code>log-type=wildfire</code>—WildFire logs</li><li>• <code>log-type=url</code>—URL filtering logs</li><li>• <code>log-type=data</code>—Data filtering logs</li><li>• <code>log-type=corr</code>—Correlated event logs as seen in the user interface within <b>Monitor &gt; Automated Correlated Engine &gt; Correlated Events</b>.</li><li>• <code>log-type=corr-detail</code>—Correlated event details as seen in the user interface when you select an event within <b>Monitor &gt; Automated Correlated Engine &gt; Correlated Events</b>.</li><li>• <code>log-type=corr-categ</code>—Correlated events by category, currently compromised hosts seen within <b>ACC &gt; Threat Activity &gt; Compromised Hosts</b>.</li><li>• <code>log-type=userid</code>—User-ID logs</li><li>• <code>log-type=auth</code>—Authentication logs</li><li>• <code>log-type=gtp</code>—GPRS Tunneling Protocol (GTP) logs</li><li>• <code>log-type=external</code>—External logs</li><li>• <code>log-type=iptag</code>—IP tag logs</li></ul>
query	<p>(Optional) Specify the match criteria for the logs. This is similar to the query provided in the web interface under the Monitor tab when viewing the logs. The query must be URL encoded.</p>
nlogs	<p>(Optional) Specify the number of logs to retrieve. The default is 20 when the parameter is not specified. The maximum is 5000.</p>
skip	<p>(Optional) Specify the number of logs to skip when doing a log retrieval. The default is 0. This is useful when retrieving logs in batches where you can skip the previously retrieved logs.</p>

Parameter	Description
dir	(Optional) Specify whether logs are shown oldest first ( <b>forward</b> ) or newest first ( <b>backward</b> ). Default is <b>backward</b> .
action	<p>(Optional) Log data sizes can be large so the API uses an asynchronous job scheduling approach to retrieve log data. The initial query returns a Job ID (<b>job-id</b>) that you can then use for future queries with the <b>action</b> parameter:</p> <ul style="list-style-type: none"> <li>• <b>action=get</b>—Check status of an active job or retrieve the log data when the status is <b>FIN</b> (finished). This is slightly different than the asynchronous approach to retrieve tech support data where a separate status action is available.</li> <li>• <b>action=finish</b>—Stop an active job.</li> <li>• <b>Not specified</b>—When not specified, such as during an initial query, the system creates a new job to retrieve log data.</li> </ul> <p> <a href="#">Learn more about Asynchronous and Synchronous Requests to the PAN-OS XML API.</a></p>

## Example: Use the API to Retrieve Traffic Logs

Follow these steps to use the API retrieve traffic logs.

**STEP 1** | Create a job to retrieve all traffic logs that occurred after a certain time:

```
https://<firewall>/api/?type=log&log-type=traffic&query=(receive_time geq '2012/06/22 08:00:00')
```



A web-browser will automatically URL encode the parameters, but when using cURL, the query parameter must be URL encoded.

Response:

```
<response
status="success" code="19">
  <result>
    <msg>
      <line>query job enqueued with jobid 18</line>
    </msg>
    <job>18</job>
  </result>
</response>
```

**STEP 2** | Retrieve traffic log data using the following request using the job ID as the value returned in the previous response:

```
https://<firewall>/api/?type=log&action=get&job-id=<id>
```

**STEP 3** | Confirm that the XML response looks similar to the following:

```
<response status="success" ">
```



```
<result>
<job>...</job>
<log>
<logs count="20" progress="100">
<entry logid="5753304543500710425"> <domain>1</domain>
  <receive_time>2012/06/13 15:43:17</receive_time> <serial>001606000117</
serial> <segno>6784588</segno>
<actionflags>0x0</actionflags> <type>TRAFFIC</type> <subtype>start</
subtype> <config_ver>1</config_ver> <time_generated>2012/06/13 15:43:17</
time_generated>
<src>172.16.1.2</src> <dst>10.0.0.246</dst> <natsrc>10.16.0.96</natsrc>
<natdst>10.0.0.246</natdst> <rule>default allow</rule>
```

When the job status is FIN (finished), the response automatically includes all the logs in the XML data response. The `<log>` node in XML is not present when the job status is still pending. After successful log data retrieval, the system automatically deletes the job.

**STEP 4 | (Optional)** Delete and active log retrieval job. To delete an active log retrieval job, run the following query:

```
https://<firewall>/api/?type=log&action=finish&job-id=<id>
```

A successful completion returns a job ID.

# Apply User-ID Mapping and Populate Dynamic Address Groups (API)

Use the **type=user-id** parameter to apply User-ID mapping information directly to the firewall. If you are using a third-party VPN solution or have users who are connecting to an 802.1x enabled wireless network, the User-ID API enables you to map users to groups so that you can capture log-in events and send them to the User-ID agent or directly to the firewall. Additionally, you can use the API to register the IP-to-user mapping information from the input file to populate the members of a Dynamic Address Group on the firewall.

```
curl -F key=<apikey> --form file=@<filename> "https://<firewall>/api/?type=user-id"
```

or

```
curl --data-urlencode key=<apikey> -d type=user-id --data-urlencode "cmd=xml-document" https://<firewall>/api/
```


With your User-ID API requests, you can use the following optional parameters:

- **vsys=vsys\_id**—Specify the vsys where you want to apply User-ID mapping.
- **target=serialnumber**—Specify the firewall by serial number when redirecting through Panorama.



- Use a GET request if the URL query size is less than 2K and a POST request if the request size is between 2K to 5MB. Limit the query size to 5MB.
- When multiple login or logout events are generated at the same time, make sure to follow these guidelines to ensure optimal firewall performance:
  - Design your application to queue events and perform batch API updates instead of sending single event or mapping updates.
  - Limit the number of concurrent API calls to five. This limit ensures that there is no performance impact to the firewall web interface as the management plane web server handles requests from both the API and the web interface.

Use the information in the following table to apply User-ID mapping information to a firewall:

Mapping or Registration Action	API Request
User-ID mapping for a login, logout, or groups.	Use this input file format when providing a User-ID mapping for a login event, logout event, or for groups:
 When multiple login or logout events are generated at the same time, make sure to follow these guidelines to ensure optimal firewall performance:	<pre>&lt;uid-message&gt;   &lt;version&gt;1.0&lt;/version&gt;   &lt;type&gt;update&lt;/type&gt;   &lt;payload&gt;     &lt;login&gt;       &lt;entry name="domain\uid1" ip="10.1.1.1" timeout="20"&gt;     &lt;/entry&gt;     &lt;/login&gt;     &lt;groups&gt;       &lt;entry name="group1"&gt;         &lt;members&gt;</pre>

Mapping or Registration Action	API Request
<ul style="list-style-type: none"> <li>• <i>Design your application to queue events and perform batch API updates instead of sending single event or mapping updates.</i></li> <li>• <i>Limit the number of concurrent API calls to five. This limit ensures that there is no performance impact to the firewall web interface as the management plane web server handles requests from both the API and the web interface.</i></li> </ul>	<pre data-bbox="625 216 1455 569">         &lt;entry name="user1"/&gt;         &lt;entry name="user2"/&gt;       &lt;/members&gt;     &lt;/entry&gt;     &lt;entry name="group2"&gt;       &lt;members&gt;         &lt;entry name="user3"/&gt;       &lt;/members&gt;     &lt;/entry&gt;   &lt;/groups&gt; &lt;/payload&gt; &lt;/uid-message&gt;&lt;/uid-message&gt; </pre> <p data-bbox="625 583 1455 646">You can include a HIP report by including a <b>&lt;hip-report&gt;&lt;/hip-report&gt;</b> XML container within an <b>&lt;entry&gt;</b> parent element.</p>
Multi-User System Entry	<p data-bbox="625 1522 1455 1717">Use the following input file format to set up a terminal server entry on the firewall and to specify the port range and block size of ports that will be assigned per user. If you are using the default port range (1025 to 65534) and block size (200) you do not need to send a multiusersystem setup message; the firewall will automatically create the terminal server object when it receives the first login message.</p> <pre data-bbox="625 1732 1455 1913"> &lt;uid-message&gt; &lt;payload&gt;   &lt;multiusersystem&gt;     &lt;entry ip="10.1.1.2"       startport="xxxxx" endport="xxxxx"       blocksize="xxx"&gt; </pre>

Mapping or Registration Action	API Request
	<pre>         &lt;/multiusersystem&gt;       &lt;/payload&gt;     &lt;type&gt;update&lt;/type&gt;     &lt;version&gt;1.0&lt;/version&gt;   &lt;/uid-message&gt; </pre>
User-ID XML multiuser system login event	<p>When the terminal servers sends a login event payload to the firewall, it can contain multiple login events. The firewall uses the information in the information in the login message to populate its user mapping table. For example, if the firewall received a packet with a source address and port of 10.1.1.23:20101, it would map the request to user jparker for policy enforcement.</p> <pre> &lt;uid-message&gt; &lt;payload&gt;   &lt;login&gt;     &lt;entry name="acme\jparker" ip="10.1.1.23" blockstart="20100"&gt;   &lt;/login&gt; &lt;/payload&gt; &lt;type&gt;update&lt;/type&gt; &lt;version&gt;1.0&lt;/version&gt; &lt;/uid-message&gt; </pre>
User-ID XML multiuser system logout	<p>Upon receipt of a logout event message with a blockstart parameter, the firewall removes the corresponding IP address-port-user mapping. If the logout message contains a username and IP address, but no blockstart parameter, the firewall removes all mappings for the user. If the logout message contains an IP address only, the firewall removes the multi-user system and all associated mappings.</p> <pre> &lt;uid-message&gt; &lt;payload&gt;   &lt;logout&gt;     &lt;entry user="domain\uid2" ip="10.1.1.2" blockstart="xxxxxx"&gt;   &lt;/logout&gt; &lt;/payload&gt; &lt;type&gt;update&lt;/type&gt; &lt;version&gt;1.0&lt;/version&gt; &lt;/uid-message&gt; </pre>
Dynamic Address Group IP address registration	<pre> &lt;uid-message&gt;   &lt;version&gt;1.0&lt;/version&gt;   &lt;type&gt;update&lt;/type&gt;   &lt;payload&gt;     &lt;register&gt;       &lt;entry ip="10.1.1.1"&gt;         &lt;tag&gt;           &lt;member&gt;CBB09C3D-3416-4734-BE90-0395B7598DE3&lt;/ member&gt;         &lt;/tag&gt;       &lt;/entry&gt;     &lt;/register&gt;     &lt;unregister&gt; </pre>

Mapping or Registration Action	API Request
	<pre data-bbox="623 212 1451 470">       &lt;entry ip="10.1.1.3"/&gt;     &lt;tag&gt;       &lt;member&gt;CBB09C3D-3416-4734-BE90-0395B7598DE5&lt;/member&gt;     &lt;/tag&gt;   &lt;/entry&gt; &lt;/unregister&gt; &lt;/payload&gt; &lt;/uid-message&gt; </pre>

---

# Get Version Info (API)

Use the **type=version** request type to show the PAN-OS version for a firewall or Panorama. In addition to the PAN-OS version, this request provides a direct way to obtain the serial number and model number.

**STEP 1** | Make a request to the PAN-OS XML API and with **type=version** along with your API key:

```
https://<firewall>/api/?type=version&key=<apikey>
```

**STEP 2** | Confirm that the XML response contains the software version, model, serial number, and whether multi-vsyz mode is on:

```
<response status="success">
  <result>
    <sw-version>7.1.0</sw-version>
    <multi-vsyz>off</multi-vsyz>
    <model>pa-vm</model>
    <serial>007000001222</serial>
  </result>
</response>
```