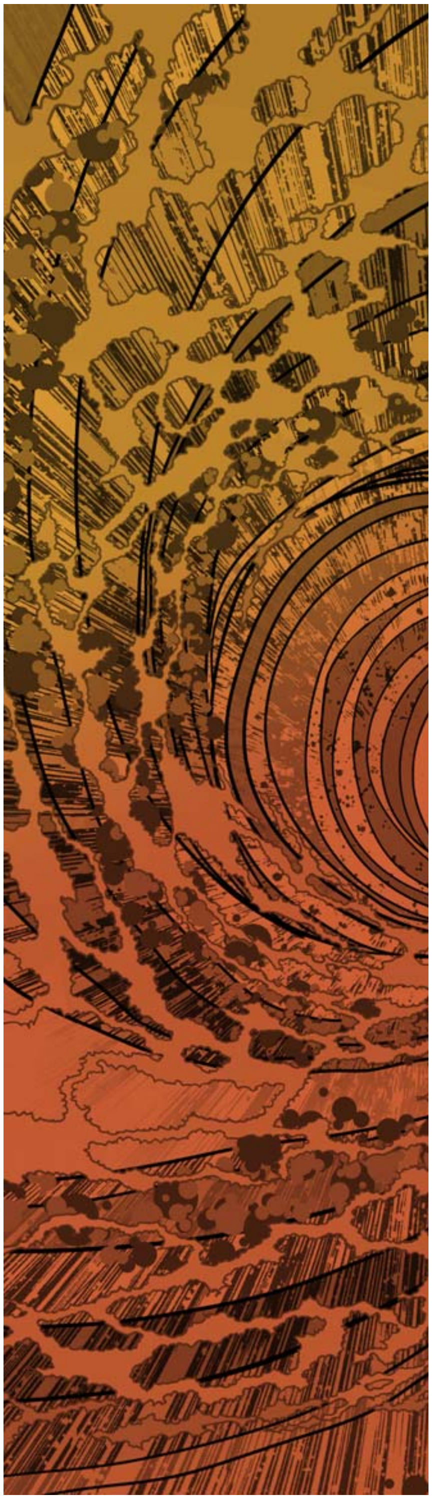


PAN-OS® and Panorama™ XML API Usage Guide

Version 7.1

Contact Information

Corporate Headquarters:
Palo Alto Networks
4401 Great America Parkway
Santa Clara, CA 95054
https://www.paloaltonetworks.com/company/contact-support

About this Guide

This API reference guide covers the features and usage of the PAN-OS XML API. For additional information, refer to the following resources:

- For information on how to configure other components in the Palo Alto Networks Next-Generation Security Platform, go to the Technical Documentation portal: https://www.paloaltonetworks.com/documentation or search the documentation.
- For access to the knowledge base, discussion forums, and videos, refer to https://live.paloaltonetworks.com.
- For contacting support, for information on support programs, to manage your account or devices, or to open a support case, refer to https://www.paloaltonetworks.com/support/tabs/overview.html.
- For the most current PAN-OS and Panorama 7.1 release notes, go to https://www.paloaltonetworks.com/documentation/71/pan-os/pan-os-release-notes.html.

To provide feedback on the documentation, please write to us at: documentation@paloaltonetworks.com.

Palo Alto Networks, Inc.

www.paloaltonetworks.com

Revision Date: September 21, 2016

Table of Contents

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.

Use the PAN-OS XML API to automate tasks 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

Because PAN-OS XML API functionality mirrors that of the web interface and CLI, 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 the API can access. You should also be knowledgeable about web service APIs, HTTP, XML, and XPath.

PAN-OS XML API Components

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-0S 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 of 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 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 just either include the XML or XPath specifying the specific XML node. Learn more about XML and XPath and XPath Node Selection.

You can make requests to the PAN-OS XML API using the GET or POST method. Use a POST request when you are sending large amounts of form data, or when you are passing non-ASCII characters. Some API requests, such as importing files, require POST. Use a GET request when passing 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
```

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 a 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 node: Nodes that contain name/value pairs such as: `<entry name="admin"></entry>`
- Text nodes contain plain text such as: `<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 select the XPath for API requests.

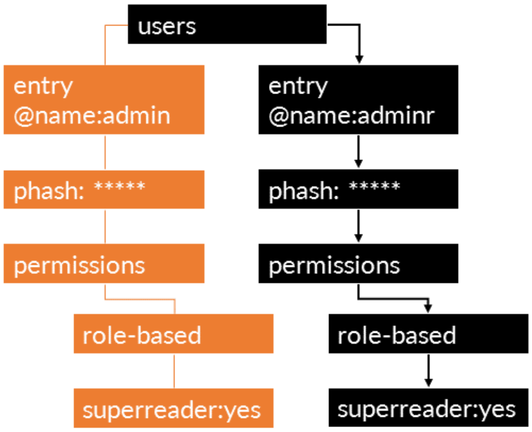
The simplest is to use the location path of the resource. For example, to select users within your management configuration, use the following path:

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

This path selects the following XML node that includes a list of 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>
```

Perhaps you want to select a specific node, such as the `superuser` text node as shown in this diagram:



To select based on the text value of an element you can search, use syntax similar to the following example:

```
/config/mgt-config/users/entry/permissions/role-based/superuser[text()='yes']
```

This path shows only the specific node that contains the `superuser` with a text value of `yes`:

```
<superuser>yes</superuser>
```

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.

> 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.

- ▲ Enable API Access
- ▲ Get Your API Key
- ▲ Make Your First API Call
- ▲ Explore the API

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 the PAN-OS Adminstrator's Guide.

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

Enable 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.

Get Your API Key

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

```
curl -X GET '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.

Generating an API key using the same administrator account credentials returns unique API keys every time, and all of the keys are valid.

Change the master password 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.

Make Your First API Call

Get Your API Key to make your first call to the PAN-OS XML API.

Make Your First API Call

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

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-vsys>on</multi-vsys>
      <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
- ▲ Use the CLI to Find XML API Syntax
- ▲ Use the Web Interface to Find XML API Syntax

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.

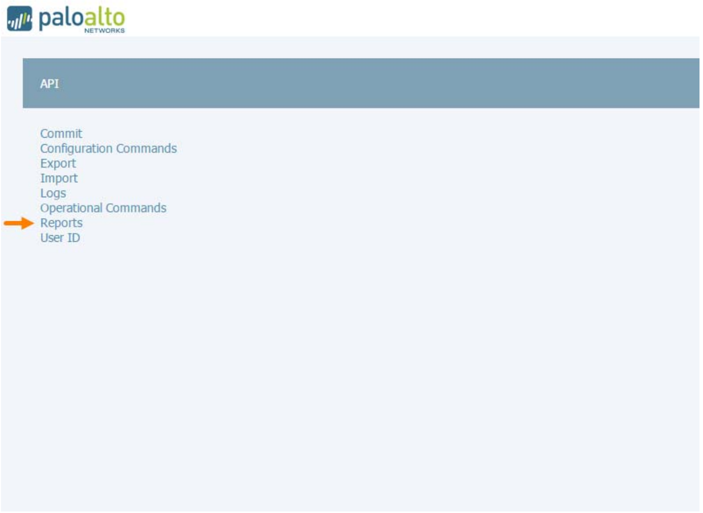
Use the API Browser to Explore the API		
Step 1	Launch the web interface.	1. Use a web browser to navigate to the actual FQDN or IP address of your firewall: `https://firewall/` 2. Log in with your administrator credentials when prompted to log in to the web interface.
Step 2	Launch the API Browser.	Go to the API browser URL on your firewall: `https://firewall/api`

Use the API Browser to Explore the API (Continued)

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`:

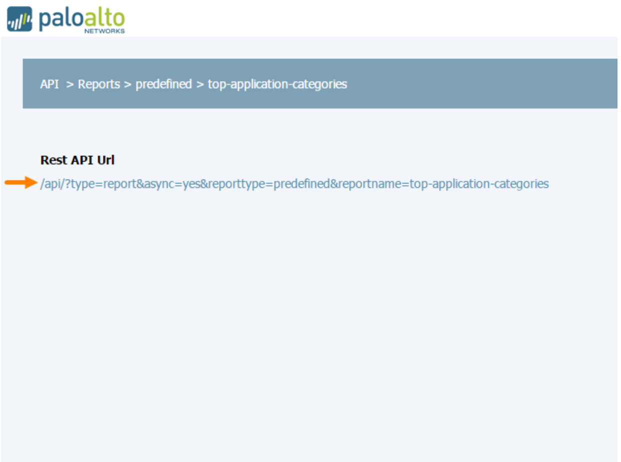


2. Drill down further until you select a request that you want to test.

Use the API Browser to Explore the API (Continued)

Step 4 Test a request.

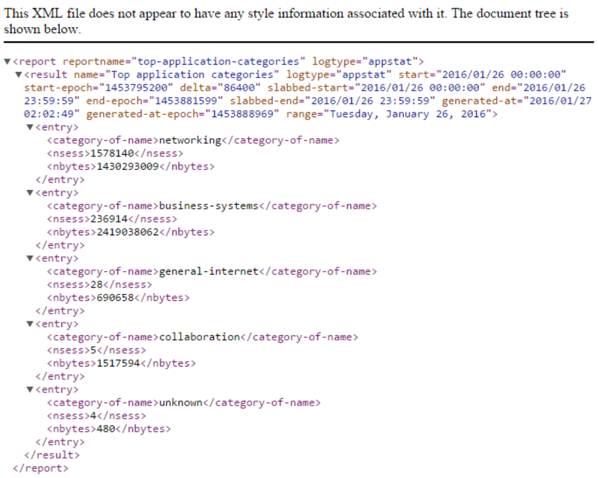
3. Select the URL to then test that request in the browser.



Rest API Url

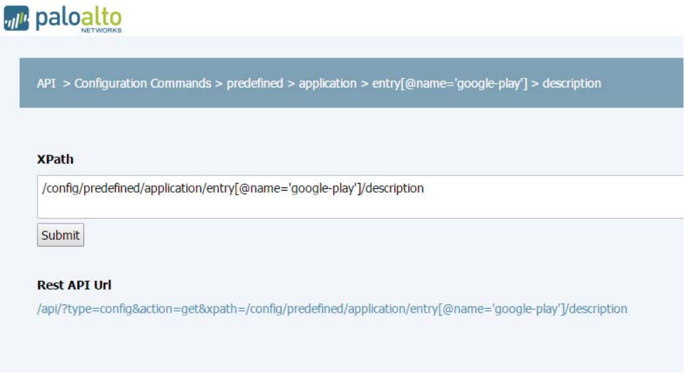
/api/?type=report&async=yes&reporttype=predefined&reportname=top-application-categories

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.

```
▼<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>
       <nsess>1578140</nsess>
       <nbytes>1430293009</nbytes>
     </entry>
    ▼<entry>
       <category-of-name>business-systems</category-of-name>
       <nsess>236914</nsess>
       <nbytes>2419038062</nbytes>
     </entry>
    ▼<entry>
       <category-of-name>general-internet</category-of-name>
       <nsess>28</nsess>
       <nbytes>690658</nbytes>
     </entry>
    ▼<entry>
       <category-of-name>collaboration</category-of-name>
       <nsess>5</nsess>
       <nbytes>1517594</nbytes>
     </entry>
    ▼<entry>
       <category-of-name>unknown</category-of-name>
       <nsess>4</nsess>
       <nbytes>480</nbytes>
     </entry>
   </result>
 </report>
```

Use the API Browser to Explore the API (Continued)	
	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.

Use the CLI to Find XML API Syntax and XPath		
Step 1	Access the CLI.	Use an SSH client or terminal to access your firewall or Panorama CLI. For more information, learn how to access the CLI on your firewall or Panorama.
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://paloaltonetw`
		`orks.com</url></test></operations></request>`

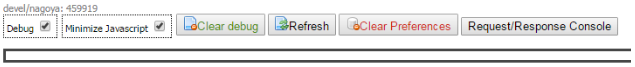
Use the CLI to Find XML API Syntax and XPath (Continued)		

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.

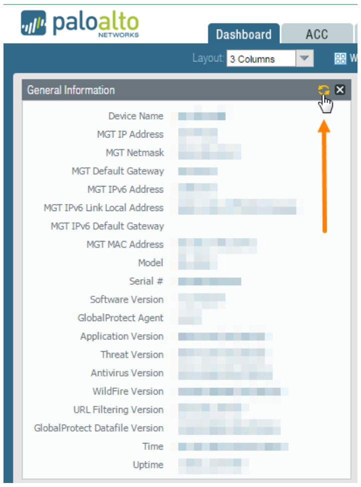
First log into the web interface and then open a separate window where you can view the corresponding XML and XPath.

Use the Web Interface and Debug Console to Find XML API Syntax 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`  devel/nagoya: 459919 Debug ☑ Minimize Javascript ☑ Clear debug Refresh Clear Preferences Request/Response Console

Use the Web Interface and Debug Console to Find XML API Syntax and XPath (Continued)

Step 3 Perform the action you want to replicate through the API.

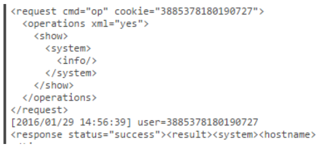
In the web browser, navigate to the menu and item or action that you want to perform.



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 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)

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

<table>
<tr><th colspan="3">Upgrade a Firewall to the Latest PAN-OS Version</th></tr>
<tr><td>Step 1</td><td>Download the latest content update.</td><td>Use the following request to first download the latest content update:
<pre>curl -X GET
'https://firewall/api/?type=op&cmd=<request><content><upgrade><download><latest/>
</download></upgrade></content></request>&key=apikey'</pre>
If successful, the response contains a <code>jobid</code> that you can use to check on the status of your request.
<pre><response status="success" code="19">
  <result>
    <msg>
      <line>Download job enqueued with jobid 2</line>
    </msg>
    <job>2</job>
  </result>
</response></pre></td></tr>
<tr><td>Step 2</td><td>Check on the content download status.</td><td>Use the <code>jobid</code> to ensure that the content download completes successfully:
<pre>curl -X GET
'https://firewall/api/?type=op&action=get&job-id=2&key=apikey'</pre>
The response should include the following:
<pre><response status="success">...</pre></td></tr>
<tr><td>Step 3</td><td>Install the latest content update.</td><td>Use the following request to install the newly downloaded content:
<pre>curl -X GET
'https://firewall/api/?type=op&cmd<request><content><upgrade><install>
<version>latest</version></install></upgrade></content></request>key=apikey'</pre>
If successful, the response contains a <code>jobid</code> that you can use to check on the status of your request.
<pre><response status="success" code="19">
  <result>
    <msg>
      <line>Content install job enqueued with jobid 3</line>
    </msg>
    <job>3</job>
  </result>
</response></pre></td></tr>
</table>

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

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'`

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

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. Using two API requests, you can view and then disconnect a Global Protect user who has been logged in for too long.

Show and Manage GlobalProtect Users	

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).

Show and Manage GlobalProtect Users (Continued)

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><glo
bal-protect-gateway><client-logout>
  <gateway>Home-N</gateway><user>dward</user><re
ason>force-logout</reason>
  <computer>Dan's%20iPhone</computer></client-lo
gout></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. Using the scripting language or your choice, you can 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.

<table>
<tr><th colspan="2">Query a Firewall from Panorama</th></tr>
<tr><td>Step 1 Get a list of connected firewalls.</td><td>Get a list of connected firewalls that Panorama manages:
<pre>
https://panorama/api/?type=op&cmd=<show><devices
><
https://panorama/api/?type=op&cmd=<show><devices
><connected></connected></devices></show>
</pre>
The response includes the serial number (serial)of each firewall.
<pre>
<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-c
lient-package-version>
        <vpn-disable-mode>no</vpn-disable-mode>
        <operational-mode>normal</operational-mode>
        <multi-vsys>no</multi-vsys>
        <vsys>
           name="vsys1">
            <display-name>vsys1</display-name>
            <shared-policy-status />

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

  <!--truncated -->

    </devices>
  </result>
</response>
</pre>
The response contains a `<serial>` XML element for each firewall.</td></tr>
<tr><td>Step 2 Collect firewall serial numbers.</td><td>In your script or code, store the firewall serial numbers returned in the response to the previous request.</td></tr>
</table>

Query a Firewall from Panorama (Continued)

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-vsys>on</multi-vsys>
<operational-mode>normal</operational-mode>
</system>
</result>
</response>
```

Repeat this request for each 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. You script or application must incorporate error-checking and logic to implement this sequence of steps.

Upgrade PAN-OS on Multiple Firewalls through Panorama

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><syst
em><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/ReleaseNote
s.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><sys
tem><software><download><version>7.1.0
-c65</version></download></software></system></r
equest>&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">`...

Upgrade PAN-OS on Multiple Firewalls through Panorama (Continued)

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-vsys>no</multi-vsys>
        <vsys>
           name="vsys1">
            <display-name>vsys1</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 that contains each firewall serial number.

Upgrade PAN-OS on Multiple Firewalls through Panorama (Continued)

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><syst
em><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/updat
es/ReleaseNotes.aspx?type=sw&versionNumber=7.1.0
-c158&product=p
anos&platform=vm]]></release-notes>
<downloaded>no</downloaded>
<current>no</current>
<latest>yes</latest>
</entry>
<!-- truncated -->
</versions>
</sw-updates>
</result>
</response>
```

Upgrade PAN-OS on Multiple Firewalls through Panorama (Continued)

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>
```

Upgrade PAN-OS on Multiple Firewalls through Panorama (Continued)

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>`

Upgrade PAN-OS on Multiple Firewalls through Panorama (Continued)		
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.

PAN-OS XML API Request Types

This 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

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

- Request Types
- Configuration Actions

Request Types

You can currently use the following request types:

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

Configuration Actions

In addition to the request type that you specify, these are the available actions when modifying or reading configurations using `type=config`:

- Actions for Modifying a Configuration
- Actions for Reading a Configuration

Actions for Modifying a Configuration

Configuration Action Type	Syntax
Set candidate configuration	`action=set`
Edit candidate configuration	`action=edit`
Delete candidate object	`action=delete`
Rename a configuration object	`action=rename`
Clone a configuration object	`action=clone`
Move a configuration object	`action=move`
Override a template setting	`action=override`
Move multiple objects in a device group or virtual system	`action=multi-move`
Clone multiple objects in a device group or virtual system	`action=multi-clone`
Show available subnode values and XPaths for a given XPath.	`action=complete`

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	`action=show`
Get candidate configuration	`action=get`

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.

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
- ▲ Get Candidate Configuration
- ▲ Set Configuration
- ▲ Edit Configuration
- ▲ Delete Configuration
- ▲ Rename Configuration
- ▲ Clone Configuration
- ▲ Move Configuration
- ▲ Override Configuration
- ▲ Multi-Move or Multi-Clone Configuration
- ▲ View Configuration Node Values for XPath

Get Active Configuration

Using `action=show` with no additional parameters returns the entire active configuration.

Get 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>
```

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>
```

Get ARP Information

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

Address objects in a VSYS.	`https://firewall//api/?type=config&action=get&xpath=/config/devices/entry/vsys/entry[@name='vsys1']/address`
	The response looks similar to the following:
	`<response status="success" code="19">`
	`    <result total-count="1" count="1">`
	`        <address admin="name" dirtyId="8" time="2015/10/20 15:32:36">`
	`            <entry name="testobject">`
	`                <ip-netmask>2.2.2.2</ip-netmask>`
	`            </entry>`
	`            <entry name="test1">`
	`                <ip-netmask>1.1.1.1</ip-netmask>`
	`            </entry>`
	`...`
	`        </address>`
	`    </result>`
	`</response>`

Pre-rules pushed from Panorama.	`https://firewall//api/?type=config&action=get&xpath=/config/panorama/vsys/entry[@name='vsys']/pre-rulebase/security`
Detailed information on Applications and Threats from the firewall.	`https://firewall/api/?type=config&action=get&xpath=/config/predefined/threats/vulnerability/entry[@name='30003']`
Full list of all applications.	`https://firewall/api/?type=config&action=get&xpath=/config/predefined/application`
Details on the specific application.	`https://firewall/api/?type=config&action=get&xpath=/config/predefined/application/entry[@name='hotmail']`

Set Configuration

Using `action=set`, you can add or create a new object at a specified location in the configuration hierarchy. 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 its XML representation (as seen in the output of `action=show`).

Set Configuration

Step 1 Create a new rule called rule1 in the security policy:

```
https://firewall/api/?type=config&action=set&key=keyvalue&xpath=xpath-value&element=element-value
```

where the xpath-value is:

```
/config/devices/entry/vsys/entry/rulebase/security/rules/entry[@name='rule1']
```

and the element-value is:

```
<source><member>src</member></source><destination><member>dst</member></destination><service><member>service</member></service><application><member>application</member></application><action>action</action><source-user><member>src-user</member></source-user><option><disable-server-response-inspection>yes-or-no</disable-server-response-inspection></option><negate-source>yes-or-no</negate-source><negate-destination>yes-or-no</negate-destination><disabled>yes-or-no</disabled><log-start>yes-or-no</log-start><log-end>yes-or-no</log-end><description>description</description><from><member>src-zone</member></from><to><member>dst-zone</member></to>
```

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 To add an additional member to a group/list, include the 'list' node in the xpath using the `member[text()='name']` syntax and include the members in the element parameter. For example, to add an additional static address object named `abc` to an address group named test, use:

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

Edit Configuration

Using action=edit, you can 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).

Edit Configuration

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

Using `action=delete`, you can 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 Configuration

- 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

Using `action=rename`, you can 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.

Rename Configuration

Step 1 Rename an address object called `old_address` to `new_address` using the following API query:

```
https://firewall/api/?type=config&action=rename&xpath=/config/devices/entry/vsys/entr
y[@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

Using `action=clone`, you can 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.

Clone Configuration

Step 1 Clone a security policy called rule1 into rule2 using the following API query:

```
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
```

Clone Configuration

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

Using `action=move`, you can 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`

Move Configuration

Step 1 Move a security policy called `rule1` after `rule2`, use the following API query:

```
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

Using `action=override`, you can 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.

Override Configuration

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/s
nmptrap&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

The `action=multi-move` and `action=multi-clone` actions allow you 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.

Multi-Move or Multi-Clone Configuration

- 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-li
st><source
xpath="/config/devices/entry[@name='localhost.localdomain']/devicegroup/entry[@name='s
ocal']/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=multiclone&xpath=/config/devices/entry[@name=
'localhost.localdomain']/devicegroup/entry[@name='norcal']/address&element=<selected-l
ist><source
xpath="/config/devices/entry[@name='localhost.localdomain']/devicegroup/entry[@name='s
ocal']/address"><member>addr1</member><member>addr2</member></source></selected-list><
all-errors>no</all-errors>
```

View Configuration Node Values for XPath

The `action=complete` action allows you to provide an XPath and see the possible values that are available under the XPath node.

View Configuration Node Values for XPath

Step 1 View the possible values, such as network interfaces, for multi-vsys firewalls, use the following command:

```
https://firewall/api/?type=config&action=complete&xpath=/api/?type=config&action=get&
xpath=/config/devices/entry[@name='localhost.localdomain']/vsys&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 commit candidate configuration to a firewall using the commit API request.

> You can validate a candidate configuration before committing it using Run Operational Mode Commands (API).

- ▲ Commit
- ▲ Commit-all

Commit

Use the API Browser to find different options available for use with force and partial commits. Replace the `body` element in the `cmd` parameter with the XML element for the corresponding commit operation.

Commit

Step 1 Use one of the following requests to commit a configuration:

- Commit—

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

- Force Commit—

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

- Partial commit—

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

Step 2 Confirm that the XML response for the request looks like one of the following:

- 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>
```

Commit

Step 4 Confirm that the XML response for the request looks like the following:

```
<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, you can use the commit-all API request type to push and validate shared policy to the firewalls using device groups and configuration to the firewalls using templates or template stacks.

Commit Type	API Request
Pre-commit policy validation.	https://*panorama*/api/?type=commit&action=all&cmd=<commit-all><shared-policy><validate-only></validate-only></shared-policy></commit-all>
Device group commit.	https://*panorama*/api/?type=commit&action=all&cmd=<commit-all><shared-policy><device-group><entry%20name="device-group-name"/></device-group></shared-policy></commit-all>
VSYS commit.	https://*panorama*/api/?type=commit&action=all&cmd=<commit-all><shared-policy><device-group><entry%20name="device-group-name"/><devices><entry%20name="serial_number"><vsys><member>vsys-name</member></vsys></entry></devices></device-group></shared-policy></commit-all>
Specific firewall commit.	https://*panorama*/api/?type=commit&action=all&cmd=<commit-all><shared-policy><device-group><entry%20name="device-group-name"><devices><entry%20name="serial_number"></devices><entry/></device-group></shared-policy></commit-all>

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 using 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.

Operational Command	API Request
System restart.	https://*firewall*/api/?type=op&cmd=<request><restart><system></system></restart></request>
System software version installation.	https://*firewall*/api/?type=op&cmd=<request><system><software><install><version>version_number</version></install></software></system></request>
Multi-vsys mode.	https://*firewall*/api/?type=op&cmd=<set><system><setting><multi-vsys></multi-vsys></setting></system></set>
User Activity Report scheduling.	https://*firewall*/api/?type=op&cmd=<schedule><uar-report><user>username</user><title>titlename</title></uar-report></schedule>
Full configuration validation.	https://*firewall*/api/?type=op&cmd=<validate><full></full></validate>
Partial configuration validation.	https://*firewall*/api/?type=op&cmd=<validate><partial><device-and-network>excluded</device-and-network></partial></validate>
Configuration saving.	https://*firewall*/api/?type=op&cmd=<save><config><to>filename</to></config></save>
Configuration loading.	https://*firewall*/api/?type=op&cmd=<load><config><from>filename</from></config></load>

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.

Get Reports (API)

The XML API provides a way to quickly pull the results of any report defined in the system using the `ype=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 view a number of dynamic reports using the API such as `top-applications-summary`, `top-blocked-url-summary`, and `top-spyware-threats-summary`. For dynamic reports, you can provide the 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.	https://*firewall*/api/?type=report&reporttype=dynamic
Last 60 seconds.	https://*firewall*/api/?type=report&reporttype=dynamic&reportname=top-app-summary&period=last-60-seconds&topn=5
Last 15 minutes.	https://*firewall*/api/?type=report&reporttype=dynamic&reportname=top-app-summary&period=last-15-minutes&topn=5
Last hour.	https://*firewall*/api/?type=report&reporttype=dynamic&reportname=top-app-summary&period=last-hour&topn=5
Last 12 hours.	https://*firewall*/api/?type=report&reporttype=dynamic&reportname=top-app-summary&period=last-12-hrs&topn=5
Last calendar day.	https://*firewall*/api/?type=report&reporttype=dynamic&reportname=top-app-summary&period=last-calendar-day&topn=5
Last 7 days	https://*firewall*/api/?type=report&reporttype=dynamic&reportname=top-app-summary&period=last-7-days&topn=5

Dynamic Report Type	API Request
Last 7 calendar days	https://*firewall*/api/?type=report&reporttype=dynamic&reportname=top-app-summary&period=last-hour&topn=5
Last calendar week.	https://*firewall*/api/?type=report&reporttype=dynamic&reportname=top-app-summary&period=last-calendar-week&topn=5
Last 30 days	https://*firewall*/api/?type=report&reporttype=dynamic&reportname=top-app-summary&period=last-30-days&topn=5

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.	https://*firewall*/api/?type=report&reporttype=predefined
Top applications.	https://*firewall*/api/?type=report&async=yes&reporttype=predefined&reportname=top-application-categories
Top attackers.	https://*firewall*/api/?type=report&async=yes&reporttype=predefined&reportname=top-attackers
Top victims.	https://*firewall*/api/?type=report&async=yes&reporttype=predefined&reportname=top-victims

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.

Get a Custom Dynamic Report
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']

Get a Custom Dynamic Report

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 webbrowser, 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

▲ Export Threat, Filter, and Data Filtering PCAPs

▲ Export Certificates and Keys

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.	`https://firewall/api/?type=export&category=application-pcap`
List of files under a directory using the `from` parameter to indicate date.	`https://firewall/api/?type=export&category=application-pcap&from=yyyymmdd`

Application PCAP Type	API Request
Application PCAP file by name using the `from` parameter.	`https://firewall/api/?type=export&category=application-pcap&from=yyyymmdd/filename` The file will be retrieved and saved locally using the name yyyymmdd-filename.
Application PCAP file saved locally with a custom name using the `to` parameter.	`https://firewall/api/?type=export&category=application-pcap&from=yyyymmdd/filename&to=localfile`

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	`https://firewall/api/?type=export&category=threat-pcap&pcap-id=id&search-time=yyyy/mm/dd hr:min:sec`
List of filtered PCAPs	`https://firewall/api/?type=export&category=filters-pcap`
Specific filtered PCAP file	`https://firewall/api/?type=export&category=filters-pcap&from=filename`
List of data filtering PCAP file names	`https://firewall/api/?type=export&category=dlp-pcap&dlp-password=password`
Specific data filtering PCAP file	`https://firewall/api/?type=export&category=dlp-pcap&dlp-password=password&from=filename&to=localfile`

Export Certificates and Keys

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&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
- `format`—cerficate format, `pkcs12` or `pem`
- `include-key`—yes or no parameter to include or exclude the key
- `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.

Export Certificates and Keys

Step 2 Confirm that the XML response includes the certificate:

```
-----BEGIN CERTIFICATE-----
MIIDXTCCAkWgAwIBAgIJAJC1HiIAZAiIMA0GCSqGSIb3Df
BAYTAkFVMRMwEQYDVQQIDApTb21lLVN0YXRlMSEwHwYDVx
aWRnaXRzIFB0eSBMdGQwHhcNMTExMjMxMDg1OTQ0WhcNMT
<!-- 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.

Export Technical Support Data

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>
```

Export Technical Support Data (Continued)

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

Use cURL to import files to the firewall.

Import Files to a Firewall or Panorama
• 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 Certificates, Keys, Response Pages, or Custom Logos

- 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`

  The certificate file import (`category=certificate`) and keypair import (`category=keypair`) take these additional parameters.
  - `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_na
me&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 the firewall using the API with the `type=log` parameter. The type of logs to retrieve must be specified using the log-type parameter:

- `log-type=traffic`—traffic logs
- `log-type=threat`—threat logs
- `log-type=config`—config logs
- `log-type=system`—system logs
- `log-type=hipmatch`—HIP logs
- `log-type=wildfire`—WildFire logs
- `log-type=url`—URL filtering logs
- `log-type=data`—data filtering logs
- `log-type=corr`—correlated event logs as seen in the user interface within **Monitor** > **Automated Correlated Engine** > **Correlated Events**.
- `log-type=corr-detail`—correlated event details as seen in the user interface when you select an event within **Monitor** > **Automated Correlated Engine** > **Correlated Events**.
- `log-type=corr-categ`—correlated events by category, currently compromised hosts seen within **ACC** > **Threat Activity** > **Compromised Hosts**.

The other optional parameters to this request are:

- `query` parameter—Specify match criteria for the logs. This is similar to the query provided in the WebUI under the **Monitor** tab when viewing the logs. The query must be URL encoded.
- `nlogs` parameter—Specify the number of logs to retrieve. The default is 20 when the parameter is not specified. The maximum is 5000.
- `skip` parameter—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.
- `dir` parameter—specify whether logs are shown in oldest-first (`forward`) or newest-first (`backward`) order. The default direction is `backward`.

Since log data sizes can be large, the API uses an asynchronous job scheduling approach to retrieve log data. The initial query returns a Job ID that you can use for future queries with the action parameter. Learn more about Asynchronous and Synchronous Requests to the PAN-OS XML API. The values for the action parameter are:

- Unspecified—when the action parameter is not specified, the system creates a new job to retrieve log data.
- `action=get`—to check status and retrieve the log data when the status is `FIN`. (This is a slight difference from the asynchronous approach to retrieve tech support data where a separation status action was available)
- `action=finish`—to stop and active job.

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="100n>
<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 a 802.1x enabled wireless network, the User-ID API enables you to map users to groups so that you can capture login events and send them to the User-ID agent or directly to the firewall. In cases like this, you can use the API to capture login 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.

<table>
<tr><th>Mapping or Registration Action</th><th>API Request</th></tr>
<tr><td>User-ID mapping for a login, logout, or groups.</td><td>Use this input file format when providing a User-ID mapping for a login event, logout event, or for groups:
<pre>
<uid-message>
     <version>1.0</version>
     <type>update</type>
     <payload>
          <login>
              <entry name="domain\uid1" ip="10.1.1.1" timeout="20">
               </entry>
          </login>
          <groups>
               <entry name="group1">
                    <members>
                         <entry name="user1"/>
                         <entry name="user2"/>
                    </members>
               </entry>
               <entry name="group2">
                    <members>
                         <entry name="user3"/>
                    </members>
               </entry>
          </groups>
     </payload>
</uid-message></uid-message>
</pre>
You can include a HIP report by including a <code><hip-report></hip-report></code> XML container within an <code><entry></code> parent element.</td></tr>
</table>

<table>
<tr><th>Mapping or Registration Action</th><th>API Request</th></tr>
<tr><td>Multi-User System Entry</td><td>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.
<pre>
<uid-message>
     <payload>
          <multiusersystem>
              <entry ip="10.1.1.2" startport="xxxxx"
endport="xxxxx" blocksize="xxx">
          </multiusersystem>
     </payload>
     <type>update</type>
     <version>1.0</version>
</uid-message>
</pre></td></tr>
<tr><td>User-ID XML multiuser system login event</td><td>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.
<pre>
<uid-message>
     <payload>
          <login>
               <entry name="acme\jparker" ip="10.1.1.23"
blockstart="20100">
          </login>
     </payload>
     <type>update</type>
     <version>1.0</version>
</uid-message>
</pre></td></tr>
<tr><td>User-ID XML multiuser system logout</td><td>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.
<pre>
<uid-message>
     <payload>
          <logout>
              <entry user="domain\uid2" ip="10.1.1.2"
blockstart="xxxxx">
         </logout>
     </payload>
     <type>update</type>
     <version>1.0</version>
</uid-message>
</pre></td></tr>
</table>

Mapping or Registration Action	API Request

Dynamic Address Group IP address registration	`<uid-message>`
	`     <version>1.0</version>`
	`     <type>update</type>`
	`     <payload>`
	`         <register>`
	`             <entry ip="10.1.1.1">`
	`               <tag>`
	`<member>CBB09C3D-3416-4734-BE90-0395B7598DE3</member>`
	`   </tag>`
	`          </entry>`
	`         </register>`
	`          <unregister>`
	`               <entry ip="10.1.1.3"/>`
	`     <tag>`
	`<member>CBB09C3D-3416-4734-BE90-0395B7598DE5</member>`
	`     </tag>`
	`               </entry>`
	`          </unregister>`
	`     </payload>`
	`</uid-message>`

Get Version Info (API)

Use the `type=version` request type 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.

Get Version Info (API)

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 The XML response contains the software version, model, serial number, and whether multi-vsys mode is on:

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

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 Get Your API Key.
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.
21	Internal error	Check with technical support when seeing these errors.
22	Session timed out	The session for this query timed out.