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://docs.paloaltonetworks.com or search the documentation.
- For access to the knowledge base and community forums, 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://docs.paloaltonetworks.com/pan-os/7-1/pan-os-release-notes

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

**About the PAN-OS XML API** ................................................. 5
PAN-OS XML API Components .............................................. 6
Structure of a PAN-OS XML API Request ............................... 7
    API Authentication and Security ..................................... 7
    XML and XPath .......................................................... 8
    XPath Node Selection .................................................... 8

**Get Started with the PAN-OS XML API** ............................. 11
Enable API Access ............................................................ 12
Get Your API Key ............................................................ 13
Make Your First API Call .................................................... 14
Explore the API ............................................................... 15
    Use the API Browser ................................................... 15
    Use the CLI to Find XML API Syntax ................................ 18
    Use the Web Interface to Find XML API Syntax ..................... 19

**PAN-OS XML API Use Cases** ............................................ 21
Upgrade a Firewall to the Latest PAN-OS Version (API) ............. 22
Show and Manage GlobalProtect Users (API) ........................... 25
Query a Firewall from Panorama (API) .................................... 27
Upgrade PAN-OS on Multiple HA Firewalls through Panorama (API) .............................................................................. 29

**PAN-OS XML API Request Types** ...................................... 35
PAN-OS XML API Request Types and Actions ........................... 36
    Request Types ............................................................. 36
    Configuration Actions .................................................... 36
Asynchronous and Synchronous Requests to the PAN-OS XML API .............................................................. 38
Configuration (API) ............................................................. 39
    Get Active Configuration ............................................... 40
    Get Candidate Configuration ......................................... 41
    Set Configuration .......................................................... 42
    Edit Configuration .......................................................... 43
    Delete Configuration ....................................................... 44
    Rename Configuration ..................................................... 44
    Clone Configuration ......................................................... 44
    Move Configuration .......................................................... 45
    Override Configuration .................................................... 45
    Multi-Move or Multi-Clone Configuration ............................ 46
    View Configuration Node Values for XPath .......................... 46
Table of Contents

Commit Configuration (API) ................................................................. 47
  Commit ................................................................. 47
  Commit-all ................................................................. 48

Run Operational Mode Commands (API) ............................................ 49

Get Reports (API) ................................................................. 50
  Dynamic Reports ................................................................. 50
  Predefined Reports ................................................................. 51
  Custom Reports ................................................................. 51

Export Files (API) ................................................................. 53
  Export Packet Captures ................................................................. 53
  Export Certificates and Keys ................................................................. 54
  Export Technical Support Data ................................................................. 55

Import Files (API) ................................................................. 57
  Importing Basics ................................................................. 57
  Import Files ................................................................. 58

Retrieve Logs (API) ................................................................. 59

Apply User-ID Mapping and Populate Dynamic Address Groups (API) ........ 61

Get Version Info (API) ................................................................. 64

PAN-OS XML API Error Codes ................................................................. 65
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 when you want to automate tasks you need to perform, such as:

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

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

▲ PAN-OS XML API Components
▲ Structure of a PAN-OS XML API Request
PAN-OS XML API Components

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

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full access to PAN-OS functionality</td>
<td>The PAN-OS XML API allows you to access almost all of the functionality normally provided through the firewall web interface and CLI.</td>
</tr>
<tr>
<td>Secure authentication and access using API key and admin roles</td>
<td>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.</td>
</tr>
<tr>
<td>Options to view XML syntax through API browser, CLI and web interface debug mode</td>
<td>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.</td>
</tr>
</tbody>
</table>
Structure of a PAN-OS XML API Request

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

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

- **API key (key-)**: The API key allows you to authenticate yourself to the API when making requests. Learn about API Authentication and Security and how to Get Your API Key.
- **Request type (type-)**: Because the XML API allows you to perform a wide array of requests, you must first specify the type of request you want, ranging from configuration to operation, importing to exporting, and from reports to user ID. Learn more about Request Types.
- **Action (action-)**: When the request type is `config` (configuration) or `op` (operational mode command), you must also specify an associated action, such as `edit`, `delete`, or `move`. Learn more about Configuration Actions.

XML and XPath elements (xpath- or cmd-): When using configuration or operational mode commands on the firewall, you include only the XML or the XPath that specifies the XML node. Learn more about XML and XPath and XPath Node Selection.

You use the GET or POST method to make requests to the PAN-OS XML API. 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 an Authorization header field:

```
Authorization: Basic amJPbLxpbw9UaTpxb3JrKJtwMDA=
```

You cannot use basic authentication when you Get Your API Key.
XML and XPath

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

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

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

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

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

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

Explore the API with the API browser, CLI, or debug console to learn how to construct XML requests.

XPath Node Selection

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

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

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

<table>
<thead>
<tr>
<th>XML Node</th>
<th>XPath Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>entry name=&quot;guest&quot;</td>
<td>/config/mgt-config/users/entry/permissions/role-based/superuser[text()='yes']</td>
</tr>
<tr>
<td>XML Node</td>
<td>XPath Syntax</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>entry @name:guest</td>
<td>/config/mgt-config/users/entry/permissions/role-based/custom/profile[text()='NewUser']</td>
</tr>
<tr>
<td>permissions</td>
<td></td>
</tr>
<tr>
<td>role-based</td>
<td></td>
</tr>
<tr>
<td>custom</td>
<td></td>
</tr>
<tr>
<td>profile:NewUser</td>
<td></td>
</tr>
</tbody>
</table>
Get Started with the PAN-OS XML API

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

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

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

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 Administrator’s Guide.

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

<table>
<thead>
<tr>
<th>Enable API Access</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> Select an Admin Role profile.</td>
</tr>
</tbody>
</table>
| **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.

<table>
<thead>
<tr>
<th>Step 1</th>
<th>To generate an API key, make a URL request to the firewall’s hostname or IP addresses using the administrative credentials and type=keygen:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>curl -X GET '<a href="https://firewall/api/?type=keygen&amp;user=username&amp;password=password">https://firewall/api/?type=keygen&amp;user=username&amp;password=password</a>'</td>
</tr>
<tr>
<td></td>
<td>A successful API call returns status=&quot;success&quot; along with the API key within the key element:</td>
</tr>
<tr>
<td></td>
<td>&lt;response status=&quot;success&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;result&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;key&gt;gJlQE56987nBxIqyfa62sZeRtYuIo2BgzEA9U0nlZBu&lt;/key&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/result&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/response&gt;</td>
</tr>
</tbody>
</table>

| Step 2 | (Optional) Revoke an API key. |
|        | |
| Step 3 | 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. |

Change the master key on your firewall in order to generate a unique API key. If you have not changed the firewall master key from the default, all firewalls with the same username/password will return the same API key.
Make Your First API Call

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

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

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

**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>
      <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-release-date>2016/02/09 15:26:53</av-release-date>
      <threat-release-date>2016/02/09 16:15:02</threat-release-date>
      <url-filtering-version>2016.01.08.407</url-filtering-version>
      <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.

<table>
<thead>
<tr>
<th>Use the API Browser to Explore the API</th>
<th></th>
</tr>
</thead>
</table>
| 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)

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

The browser shows the resulting XML response in the browser:

```
<report reportname="top-application-categories" logtype="SOAP">
  <xmltree name="top-application-categories" logtype="predefined">
    <entry>
      <category-of-name networkipfiling/category-of-name>
        <category-of-name="192.0.2.0/24"/>
        <category-of-name="192.0.2.0/256"/>
      </entry>
      <entry>
        <category-of-name=domain-business-system/category-of-name>
          <category-of-name="example.com"/>
          <category-of-name="example.org"/>
        </entry>
      <entry>
        <category-of-name=general-internet/category-of-name>
          <category-of-name="example.com"/>
          <category-of-name="example.org"/>
        </entry>
      <entry>
        <category-of-name=collaboration/category-of-name>
          <category-of-name="example.com"/>
          <category-of-name="example.org"/>
        </entry>
      </entry>
    </xmltree>
  </report>
```
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**

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

![XPath Example](image)

### Use the CLI to Find XML API Syntax and XPath

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td>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.</td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td>Enable debug mode. Enter the following command: <code>debug cli on</code></td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td>Run a CLI command. Enter and run a CLI command. Example: <code>test url http://paloaltonetworks.com &lt;request cmd=&quot;op&quot; cookie=&quot;7581536015878829&quot; uid=&quot;1206&quot;&gt;&lt;operations&gt;&lt;test&gt;&lt;url&gt;http://paloaltonetworks.com&lt;/url&gt;&lt;/test&gt;&lt;/operations&gt;&lt;/request&gt;</code></td>
</tr>
</tbody>
</table>
### 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:

```plaintext
https://firewall/api/?type=op&cmd=<test><url>https://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:

  ```plaintext
  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

<table>
<thead>
<tr>
<th>Step 1</th>
<th>Launch the web interface.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 2</td>
<td>Launch the debug console.</td>
</tr>
</tbody>
</table>
Use the Web Interface and Debug Console to Find XML API Syntax and XPath (Continued)

<table>
<thead>
<tr>
<th>Step 3</th>
<th>Perform the action you want to replicate through the API.</th>
<th>In the web browser, navigate to the menu and item or action that you want to perform.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>In the web browser, navigate to the menu and item or action that you want to perform.</td>
<td>To aid in finding the relevant XML, select <strong>Clear</strong> in the debug console just before you select the final menu or action.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 4</th>
<th>View the resulting XML syntax in the debug console.</th>
<th>In the debug console, select <strong>Refresh</strong> and then navigate through the console to the syntax related to your choice or action:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Example XML within debug console:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;request cmd=&quot;op&quot; cookie=&quot;3885378180190727&quot;&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;operations xml=&quot;yes&quot;&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;show&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;system&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;info/&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;/info&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;/system&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;/show&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;/operations&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;/request&gt;</td>
<td></td>
</tr>
</tbody>
</table>

**The corresponding API call looks like this:**

http://firewall/api/?type=op&cmd=<show><system><info/></system></show>&key=apikey
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)
- Show and Manage GlobalProtect Users (API)
- Query a Firewall from Panorama (API)
- Upgrade PAN-OS on Multiple HA Firewalls through Panorama (API)
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>
<thead>
<tr>
<th>Upgrade a Firewall to the Latest PAN-OS Version</th>
<th>Use the following request to first download the latest content update:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1  Download the latest content update.</td>
<td>curl -X GET '<a href="https://firewall/api/?type=op&amp;cmd=">https://firewall/api/?type=op&amp;cmd=</a>&lt;request&gt;&lt;content&gt;&lt;upgrade&gt;&lt;download&gt;&lt;latest/&gt;&lt;/download&gt;&lt;/upgrade&gt;&lt;/content&gt;&lt;/request&gt;&amp;key=ap ikey'</td>
</tr>
<tr>
<td></td>
<td>If successful, the response contains a <strong>jobid</strong> that you can use to check on the status of your request.</td>
</tr>
<tr>
<td>Step 2  Check on the content download status.</td>
<td>Use the <strong>jobid</strong> to ensure that the content download completes successfully:</td>
</tr>
<tr>
<td></td>
<td>curl -X GET '<a href="https://firewall/api/?type=op&amp;action=get&amp;job-id=2&amp;key=ap">https://firewall/api/?type=op&amp;action=get&amp;job-id=2&amp;key=ap</a> ikey'</td>
</tr>
<tr>
<td></td>
<td>The response should include the following:</td>
</tr>
<tr>
<td>Step 3  Install the latest content update.</td>
<td>Use the following request to install the newly downloaded content:</td>
</tr>
<tr>
<td></td>
<td>curl -X GET '<a href="https://firewall/api/?type=op&amp;cmd=">https://firewall/api/?type=op&amp;cmd=</a>&lt;request&gt;&lt;content&gt;&lt;upgrade&gt;&lt;install&gt;&lt;version&gt;latest&lt;/version&gt;&lt;/install&gt;&lt;/upgrade&gt;&lt;/content&gt;&lt;/request&gt;&amp;key=ap ikey'</td>
</tr>
<tr>
<td></td>
<td>If successful, the response contains a <strong>jobid</strong> that you can use to check on the status of your request.</td>
</tr>
<tr>
<td></td>
<td>&lt;response status=&quot;success&quot; code=&quot;19&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;result&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;msg&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;line&gt;Download job enqueued with jobid 2&lt;/line&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/msg&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;job&gt;2&lt;/job&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/result&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/response&gt;</td>
</tr>
</tbody>
</table>

---

Step 1: Download the latest content update.

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

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

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

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

Step 2: Check on the content download status.

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

```bash
curl -X GET 'https://firewall/api/?type=op&action=get&job-id=2&key=ap ikey'
```

The response should include the following:

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

Step 3: Install the latest content update.

Use the following request to install the newly downloaded content:

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

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

```xml
<response status="success" code="19">
  <result>
    <msg>
      <line>Content install job enqueued with jobid 3</line>
    </msg>
    <job>3</job>
  </result>
</response>
```
<table>
<thead>
<tr>
<th>Step 4</th>
<th>Check on the content installation status.</th>
<th>Use the <code>jobid</code> to ensure that the content installation completes successfully:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><code>curl -X GET 'https://firewall/api/?type=op&amp;action=get&amp;job-id=3&amp;key=apikey'</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>The response should include the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>&lt;response status=&quot;success&quot;&gt;...</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 5</th>
<th>Check for the latest PAN-OS software update.</th>
<th>After installing the latest Content Release update, check for the latest available PAN-OS software updates:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><code>curl -X GET 'https://firewall/api/?type=op&amp;cmd=&lt;request&gt;&lt;system&gt;&lt;software&gt;&lt;check&gt;&lt;/check&gt;&lt;/software&gt;&lt;/system&gt;&lt;/request&gt;&amp;key=apikey'</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>In the response, the first entry is the latest version of PAN-OS:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>&lt;response status=&quot;success&quot;&gt;</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>&lt;result&gt;</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>&lt;sw-updates last-updated-at=&quot;2015/10/20 14:16:30&quot;&gt;</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>&lt;msg /&gt;</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>&lt;versions&gt;</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>&gt;</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>&lt;version&gt;7.1.0&lt;/version&gt;</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>&lt;filename&gt;PanOS_3000-7.1.0-c65&lt;/filename&gt;</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>&lt;size&gt;720&lt;/size&gt;</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>&lt;size-kb&gt;737504&lt;/size-kb&gt;</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>&lt;released-on&gt;2015/10/20 13:23:11&lt;/released-on&gt;</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 6</th>
<th>Download the latest PAN-OS software update.</th>
<th>1. In this case, the latest version is 7.1.0-c65, so download that version:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><code>curl -X GET 'https://firewall/api/?type=op&amp;cmd=&lt;request&gt;&lt;system&gt;&lt;software&gt;&lt;download&gt;&lt;version&gt;7.1.0-c65&lt;/version&gt;&lt;/download&gt;&lt;/software&gt;&lt;/system&gt;&lt;/request&gt;&amp;key=apikey'</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Use the <code>jobid</code> in the response to ensure that the system update download completes successfully:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>curl -X GET 'https://firewall/api/?type=op&amp;action=get&amp;job-id=318&amp;key=apikey'</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>The response should include the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>&lt;response status=&quot;success&quot;&gt;...</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 7</th>
<th>Install the latest PAN-OS software update.</th>
<th>To install the latest system update, include the version in a software install request:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><code>curl -X GET 'https://firewall/api/?type=op&amp;cmd=&lt;request&gt;&lt;system&gt;&lt;software&gt;&lt;install&gt;&lt;version&gt;7.1.0-c65&lt;/version&gt;&lt;/install&gt;&lt;/software&gt;&lt;/system&gt;&lt;/request&gt;&amp;key=apikey'</code></td>
</tr>
</tbody>
</table>
### Upgrade a Firewall to the Latest PAN-OS Version (Continued)

<table>
<thead>
<tr>
<th>Step 8</th>
<th>Check on the software installation status.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Use the <code>jobid</code> in the response to ensure that the system update installs successfully:</td>
</tr>
<tr>
<td></td>
<td><code>curl -X GET https://firewall/api/?type=op&amp;action=get&amp;job-id=320&amp;key=apikey</code></td>
</tr>
<tr>
<td></td>
<td>The response should include the following:</td>
</tr>
<tr>
<td></td>
<td><code>&lt;response status=&quot;success&quot;&gt;…</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 9</th>
<th>Reboot the firewall.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>After the system update installs successfully, trigger:</td>
</tr>
<tr>
<td></td>
<td><code>curl -X GET 'https://firewall/api/?type=op&amp;cmd=&lt;request&gt;&lt;restart&gt;&lt;system&gt;&lt;/system&gt;&lt;/restart&gt;&lt;request&gt;&amp;key=apikey'</code></td>
</tr>
</tbody>
</table>
Show and Manage GlobalProtect Users (API)

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

<table>
<thead>
<tr>
<th>Show and Manage GlobalProtect Users</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> View all GlobalProtect users.</td>
</tr>
<tr>
<td>Make a request to view all GlobalProtect users:</td>
</tr>
<tr>
<td>curl -X GET '<a href="https://firewall/api/?type=op&amp;cmd=">https://firewall/api/?type=op&amp;cmd=</a>&lt;show&gt;&lt;global-protect-gateway&gt;&lt;current-user/&gt;&lt;/global-protect-gateway&gt;&lt;/show&gt;&amp;key=apikey'</td>
</tr>
<tr>
<td>The response contains a list of users along with related information including IP addresses, logins, and client information:</td>
</tr>
<tr>
<td>&lt;response status=&quot;success&quot;&gt;</td>
</tr>
<tr>
<td>&lt;result&gt;</td>
</tr>
<tr>
<td>&lt;domain /&gt;</td>
</tr>
<tr>
<td>&lt;islocal&gt;yes&lt;/islocal&gt;</td>
</tr>
<tr>
<td>&lt;username&gt;dward&lt;/username&gt;</td>
</tr>
<tr>
<td>&lt;computer&gt;Dan’s iPhone&lt;/computer&gt;</td>
</tr>
<tr>
<td>&lt;client&gt;Apple iOS 8.1.2&lt;/client&gt;</td>
</tr>
<tr>
<td>&lt;vpn-type&gt;Device Level VPN&lt;/vpn-type&gt;</td>
</tr>
<tr>
<td>&lt;virtual-ip&gt;192.168.2.1&lt;/virtual-ip&gt;</td>
</tr>
<tr>
<td>&lt;public-ip&gt;166.173.63.240&lt;/public-ip&gt;</td>
</tr>
<tr>
<td>&lt;tunnel-type&gt;SSL&lt;/tunnel-type&gt;</td>
</tr>
<tr>
<td>&lt;login-time&gt;Jan.22 01:50:36&lt;/login-time&gt;</td>
</tr>
<tr>
<td>&lt;login-time-utc&gt;1421916636&lt;/login-time-utc&gt;</td>
</tr>
<tr>
<td>&lt;lifetime&gt;2592000&lt;/lifetime&gt;</td>
</tr>
<tr>
<td>&lt;/entry&gt;</td>
</tr>
<tr>
<td>&lt;/result&gt;</td>
</tr>
<tr>
<td>&lt;/response&gt;</td>
</tr>
</tbody>
</table>

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 (API)  

Step 2  Disconnect a GlobalProtect user.  

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

```
curl -X GET 'https://firewall/api/?type=op&cmd=<request><global-protect-gateway><client-logout><gateway>Home-N</gateway><user>dward</user><reason>force-logout</reason><computer>Dan's%20iPhone</computer></client-logout></global-protect-gateway></request>&key=apikey'
```

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

```
<response status="success">
  <result>
    <response status="success">
      <gateway>Home-N</gateway>
      <domain>(null)</domain>
      <user>dward</user>
      <computer>Dan’s iPhone</computer>
    </response>
  </result>
</response>
```
The target parameter on Panorama allows you to redirect queries to a managed firewall. Redirecting queries to firewalls helps to reduce time and the number of steps required to issue repetitive commands. Use the scripting language of your choice to store firewall serial numbers and use them to issue a query to several firewalls.

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

### Query a Firewall from Panorama

<table>
<thead>
<tr>
<th>Step 1</th>
<th>Get a list of managed firewalls.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Get a list of firewalls that Panorama manages:</td>
</tr>
<tr>
<td></td>
<td><a href="https://panorama/api/?type=op&amp;cmd=">https://panorama/api/?type=op&amp;cmd=</a>&lt;show&gt;&lt;devices&gt;&lt;all&gt;&lt;/all&gt;&lt;/devices&gt;&lt;/show&gt;</td>
</tr>
<tr>
<td></td>
<td>The response includes the serial number (serial) of each firewall.</td>
</tr>
<tr>
<td></td>
<td>&lt;response status=&quot;success&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;result&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;devices&gt;</td>
</tr>
<tr>
<td></td>
<td>name=&quot;007200002517&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;serial&gt;007200002342&lt;/serial&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;connected&gt;yess&lt;/connected&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;unsupported-version&gt;no&lt;/unsupported-version&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;deactivated&gt;no&lt;/deactivated&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;hostname&gt;PM-6-1-VMK&lt;/hostname&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;ip-address&gt;10.3.4.137&lt;/ip-address&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;mac-addr /&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;uptime&gt;81 days, 20:39:41&lt;/uptime&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;family&gt;vm&lt;/family&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;model&gt;PA-VM&lt;/model&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;sw-version&gt;6.1.3&lt;/sw-version&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;av-version&gt;555-3129&lt;/av-version&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;wildfire-version&gt;91873-101074&lt;/wildfire-version&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;url-db&gt;paloaltonetworks&lt;/url-db&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;url-filtering-version&gt;2016.02.02.416&lt;/url-filtering-version&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;vpnclient-package-version &gt;/</td>
</tr>
<tr>
<td></td>
<td>&lt;logdb-version&gt;6.1.3&lt;/logdb-version&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;vpnclient-package-version &gt;/</td>
</tr>
<tr>
<td></td>
<td>&lt;logdb-version&gt;6.1.3&lt;/logdb-version&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;global-protect-client-package-version&gt;0.0.0&lt;/global-protect-c</td>
</tr>
<tr>
<td></td>
<td>&lt;client-package-version&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;vpn-disable-mode&gt;no&lt;/vpn-disable-mode&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;operational-mode&gt;normal&lt;/operational-mode&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;multi-vsys&gt;no&lt;/multi-vsys&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;vsys&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;name=&quot;vsys1&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;display-name&gt;vsys1&lt;/display-name&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;shared-policy-status /&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;shared-policy-md5sum&gt;4a0913667df83ff098492e2e2ec1756&lt;/shared-policy-md5sum&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/vsys&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/device&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/result&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/response&gt;</td>
</tr>
<tr>
<td></td>
<td>The response contains a &lt;serial&gt; XML element for each firewall.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 2</th>
<th>Collect firewall serial numbers.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>In your script or code, store the firewall serial numbers returned in the response to the previous request.</td>
</tr>
</tbody>
</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:fffe: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-c154</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>
            <url-filtering-version>2015.10.27.226</url-filtering-version>
            <global-protect-datafile-version>1445974904</global-protect-datafile-version>
            <logdb-version>7.6.9</logdb-version>
            <platform-family>3000</platform-family>
            <vpn-disable-mode>off</vpn-disable-mode>
            <multi-avsyon</multi-avsyon>
            <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.

### Upgrade PAN-OS on Multiple Firewalls through Panorama

<table>
<thead>
<tr>
<th>Step 1</th>
<th>Check for the latest PAN-OS software update through Panorama</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Check for the latest available PAN-OS software updates. Include the firewall serial number in your request:</strong>&lt;br&gt;<a href="https://panorama/api/?type=op&amp;cmd=">https://panorama/api/?type=op&amp;cmd=</a>&lt;request&gt;&lt;system&gt;&lt;software&gt;&lt;check&gt;&lt;/check&gt;&lt;/software&gt;&lt;/system&gt;&amp;target=007200002517&amp;key=apikey&lt;br&gt;The response contains an array of results sorted to show the latest version first:</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td>Download the latest PAN-OS software update.</td>
</tr>
<tr>
<td><strong>1.</strong></td>
<td>In this case, the latest version is 7.1.0-c65, so download that version: &lt;br&gt;curl -X GET '<a href="https://firewall/api/?type=op&amp;cmd=">https://firewall/api/?type=op&amp;cmd=</a>&lt;request&gt;&lt;system&gt;&lt;software&gt;&lt;download&gt;&lt;version&gt;7.1.0-c65&lt;/version&gt;&lt;/download&gt;&lt;/software&gt;&lt;/system&gt;&lt;/request&gt;&amp;key=apikey'</td>
</tr>
</tbody>
</table>
| **2.** | Use the jobid in the response to ensure that the system update download completes successfully: <br>curl -X GET 'https://firewall/api/?type=op&action=get&job-id=318&key=apikey' <br>The response should include the following: <br><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:  

```bash
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:  

```bash
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:  

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

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

```
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:

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

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

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

<table>
<thead>
<tr>
<th>Step 7</th>
<th>Download the latest PAN-OS software update.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>After determining the latest system update, download it to both firewalls in the HA pair:</td>
</tr>
<tr>
<td></td>
<td><a href="https://panorama/api/?type=op&amp;cmd=">https://panorama/api/?type=op&amp;cmd=</a>&lt;request&gt;&lt;system&gt;&lt;software&gt;&lt;download&gt;&lt;version&gt;7.1&lt;/version&gt;&lt;download&gt;&lt;/software&gt;&lt;/system&gt;&lt;/request&gt;&amp;target=serialnumber&amp;key=apikey</td>
</tr>
<tr>
<td></td>
<td>The response contains a job ID:</td>
</tr>
<tr>
<td></td>
<td>&lt;response status=&quot;success&quot; code=&quot;19&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;result&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;msg&gt;</td>
</tr>
<tr>
<td></td>
<td>Download job enqueued with jobid 3448&lt;/msg&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/msg&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;job&gt;3448&lt;/job&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/result&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/response&gt;</td>
</tr>
<tr>
<td></td>
<td>Use the job ID to check on the download status:</td>
</tr>
<tr>
<td></td>
<td><a href="https://panorama/api/?type=op&amp;cmd=">https://panorama/api/?type=op&amp;cmd=</a>&lt;show&gt;&lt;jobs&gt;&lt;id&gt;3448&lt;/id&gt;&lt;/jobs&gt;&lt;/show&gt;&amp;target=serialnumber&amp;key=apikey</td>
</tr>
<tr>
<td></td>
<td>The response contains a job status of FIN when the download is complete:</td>
</tr>
<tr>
<td></td>
<td>&lt;response status=&quot;success&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;result&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;job&gt;</td>
</tr>
<tr>
<td></td>
<td>2016/02/03 08:32:00&lt;/tenq&gt;</td>
</tr>
<tr>
<td></td>
<td>3448&lt;/id&gt;</td>
</tr>
<tr>
<td></td>
<td>/user /&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;Downld&lt;/type&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;status&gt;FIN&lt;/status&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;stoppable&gt;no&lt;/stoppable&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;result&gt;OK&lt;/result&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;tfin&gt;08:32:10&lt;/tfin&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;progress&gt;08:32:10&lt;/progress&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;details&gt;</td>
</tr>
<tr>
<td></td>
<td>Successfully downloaded</td>
</tr>
<tr>
<td></td>
<td>Preloading into software manager</td>
</tr>
<tr>
<td></td>
<td>Successfully loaded into software manager</td>
</tr>
<tr>
<td></td>
<td>&lt;/details&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;warnings /&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/job&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/result&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/response&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 8</th>
<th>Suspend the active HA firewall.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Suspend the active firewall in your high-availability firewall pair:</td>
</tr>
<tr>
<td></td>
<td><a href="https://panorama/api/?type=op&amp;cmd=">https://panorama/api/?type=op&amp;cmd=</a>&lt;request&gt;&lt;high-availability&gt;&lt;state&gt;&lt;suspend&gt;&lt;/suspend&gt;&lt;/state&gt;&lt;/high-availability&gt;&lt;/request&gt;&amp;target=serialnumber&amp;key=apikey</td>
</tr>
<tr>
<td></td>
<td>The response confirms the active firewall has been suspended:</td>
</tr>
<tr>
<td></td>
<td>&lt;response status=&quot;success&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;result&gt;Successfully changed HA state to suspended&lt;/result&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/response&gt;</td>
</tr>
</tbody>
</table>
### 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=<restart><system></system></restart>&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>&target=serialnumber&key=apikey  
The response confirms the active firewall is now active:  
<response status="success">  
<result>Successfully changed HA state to functional</result>  
</response> |
| Step 14 | Install the system update on the passive HA peer. | Once the suspended HA firewall is active, you can then repeat steps 5-8 on the now passive HA peer. |
PAN-OS XML API Request Types

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

- PAN-OS XML API Request Types and Actions
- Asynchronous and Synchronous Requests to the PAN-OS XML API
- Configuration (API)
- Commit Configuration (API)
- Run Operational Mode Commands (API)
- Get Reports (API)
- Export Files (API)
- Import Files (API)
- Retrieve Logs (API)
- Apply User-ID Mapping and Populate Dynamic Address Groups (API)
- Get Version Info (API)
PAN-OS XML API Request Types and Actions

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

- **Request Types**
- **Configuration Actions**

### Request Types

You can currently use the following request types:

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

### Configuration Actions

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

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

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

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

<table>
<thead>
<tr>
<th>Configuration Action Type</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get active configuration</td>
<td>action=show</td>
</tr>
<tr>
<td>Get candidate configuration</td>
<td>action=get</td>
</tr>
</tbody>
</table>

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.
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

- Use XPath to Get Active Configuration
- Use XPath to Get ARP Information
Use XPath to Get Active Configuration

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

**Step 1** Use the `xpath` parameter to target a specific portion of the configuration. For example, to retrieve just the `security` rulebase:

```
xpath=/config/devices/entry/vsys/entry/rulebase/security
```

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

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

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

```
<response status="success">
  <result>
    <security>
      <rules>
        <entry name="IT DNS Services">
          <profile-setting>
            <group>
              <member>best-practice</member>
            </group>
            <profile-setting>
              <to>
                <member>untrust</member>
              </to>
              <from>
                <member>trust</member>
              </from>
              <source>
                <member>any</member>
              </source>
              <destination>
                <member>Data Center</member>
              </destination>
              <source-user>
                <member>any</member>
              </source-user>
              <category>
                <member>any</member>
              </category>
              <application>
                <member>dns</member>
              </application>
              <service>
                <member>application-default</member>
              </service>
              <hip-profiles>
                <member>any</member>
              </hip-profiles>
              <action>allow</action>
              <tag>
                <member>Best Practice</member>
              </tag>
              <log-start>no</log-start>
              <log-setting>default</log-setting>
            </entry>
          ...
        </rules>
      </security>
    </result>
  </response>
```
Use XPath to Get ARP Information

**Step 1** Use the following request to retrieve ARP information:

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

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

```
<response status="success">
  <result>
    <max>3000</max>
    <total>16</total>
    <timeout>1800</timeout>
    <dp>dp0</dp>
    <entries>
      <entry>
        <status>c</status>
        <ip>10.47.0.1</ip>
        <mac>00:1b:17:00:2f:13</mac>
        <ttl>1743</ttl>
        <interface>ethernet1/1</interface>
        <port>ethernet1/1</port>
      </entry>
      <entry>
        <status>c</status>
        <ip>10.47.0.10</ip>
        <mac>00:50:56:93:68:6f</mac>
        <ttl>386</ttl>
        <interface>ethernet1/1</interface>
        <port>ethernet1/1</port>
      </entry>
    </entries>
  </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

<table>
<thead>
<tr>
<th>Configuration Node</th>
<th>API Request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address objects in a VSYS.</td>
<td><a href="https://firewall/api/?type=config&amp;action=get&amp;xpath=/config/devices/entry/vsys/entry%5B@name='vsys1'%5D/address">https://firewall/api/?type=config&amp;action=get&amp;xpath=/config/devices/entry/vsys/entry[@name='vsys1']/address</a></td>
</tr>
<tr>
<td>The response looks similar to the following:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;response status=&quot;success&quot; code=&quot;19&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;result total-count=&quot;1&quot; count=&quot;1&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;address admin=&quot;name&quot; dirtyId=&quot;8&quot; time=&quot;2015/10/20 15:32:36&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;entry name=&quot;testobject&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;ip-netmask&gt;2.2.2.2&lt;/ip-netmask&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/entry&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;entry name=&quot;test1&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;ip-netmask&gt;1.1.1.1&lt;/ip-netmask&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/entry&gt;</td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>&lt;/address&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/result&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/response&gt;</td>
</tr>
<tr>
<td>Pre-rules pushed from Panorama.</td>
<td><a href="https://firewall/api/?type=config&amp;action=get&amp;xpath=/config/panorama/vsys/entry%5B@name='vsys'%5D/pre-rulebase/security">https://firewall/api/?type=config&amp;action=get&amp;xpath=/config/panorama/vsys/entry[@name='vsys']/pre-rulebase/security</a></td>
</tr>
<tr>
<td>Detailed information on Applications and Threats from the firewall.</td>
<td><a href="https://firewall/api/?type=config&amp;action=get&amp;xpath=/config/predefined/threats/vulnerability/entry%5B@name='30003">https://firewall/api/?type=config&amp;action=get&amp;xpath=/config/predefined/threats/vulnerability/entry[@name='30003</a>']</td>
</tr>
<tr>
<td>Full list of all applications.</td>
<td><a href="https://firewall/api/?type=config&amp;action=get&amp;xpath=/config/predefined/application">https://firewall/api/?type=config&amp;action=get&amp;xpath=/config/predefined/application</a></td>
</tr>
<tr>
<td>Details on the specific application.</td>
<td><a href="https://firewall/api/?type=config&amp;action=get&amp;xpath=/config/predefined/application/entry%5B@name='hotmail">https://firewall/api/?type=config&amp;action=get&amp;xpath=/config/predefined/application/entry[@name='hotmail</a>']</td>
</tr>
</tbody>
</table>
Set Configuration

Use `action=set` to 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`).

---

**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><negate-source>yes-or-no</negate-source></option><negate-destination>yes-or-no</negate-destination><disabled>yes-or-no</disabled><log-start>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**

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

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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>` |
| 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` |
| 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/vsys1/address-group/entry[@name='test']/static/member[text()='abc']/member=abc&text='xyz'/member=xyz` |

---

**Delete Configuration**

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

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
</table>
| 1    | 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']` |
| 2    | 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/vsys1/address-group/entry[@name='test']/static/member[text()='abc']` |
Rename Configuration

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

| Step 1 | Use the following API query to rename an address object called `old_address` to `new_address`:
| | `https://firewall/api/?type=config&action=rename&xpath=/config/devices/entry/vsys/entry[@name='vsys1']/address/entry[@name='old_address']&newname=new_address`
| Step 2 | Confirm that the XML response for the request looks like the following:

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

Clone Configuration

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

| Step 1 | Use the following API query to clone a security policy called `rule1` to `rule2`:
| | `https://firewall/api/?type=config&action=clone&xpath=/config/devices/entry/vsys/entry[@name='vsys1']/rulebase/security/rules&from=/config/devices/entry/vsys/entry[@name='vsys1']/rulebase/security/rules/entry[@name='rule1']&newname=rule2`
| Step 2 | Confirm that the XML response for the request looks like the following:

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

A corresponding success log is recorded in the Configuration log:

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

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

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

**Move Configuration**

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

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

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

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

**Override Configuration**

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

**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/snmtrappelement<entry name="snmp" src="tpl"><version src="tpl"><v2c src="tpl"><server src="tpl"><entry name="test" src="tpl"><manager src="tpl">2.2.2.2</manager><community src="tpl">test</community></entry></server></v2c></version></entry>
```

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

```
<response status="success" code="20"><msg>command succeeded</msg></response>
```
Multi-Move or Multi-Clone Configuration

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

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

<table>
<thead>
<tr>
<th>Multi-Move or Multi-Clone Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Move addresses addr1, addr2, to device group norcal from device group socal:</td>
</tr>
<tr>
<td><a href="https://firewall/api/?type=config&amp;action=multimove&amp;xpath=/config/devices/entry%5B@name='localhost.localdomain'%5D/devicegroup/entry%5B@name='norcal'%5D/address">https://firewall/api/?type=config&amp;action=multimove&amp;xpath=/config/devices/entry[@name='localhost.localdomain']/devicegroup/entry[@name='norcal']/address</a></td>
</tr>
</tbody>
</table>

| • 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-list><source xpath="/config/devices/entry[@name='localhost.localdomain']/devicegroup/entry[@name='socal']/address"><member>addr1</member><member>addr2</member></source></selected-list><all-errors>no</all-errors> |

View Configuration Node Values for XPath

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

<table>
<thead>
<tr>
<th>View Configuration Node Values for XPath</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1 View the possible values, such as network interfaces, for multi-vsys firewalls, use the following command:</td>
</tr>
<tr>
<td><a href="https://firewall/api/?type=config&amp;action=complete&amp;xpath=/api/?type=config&amp;action=get&amp;xpath=/config/devices/entry%5B@name='localhost.localdomain'%5D/vsys&amp;key=apikey">https://firewall/api/?type=config&amp;action=complete&amp;xpath=/api/?type=config&amp;action=get&amp;xpath=/config/devices/entry[@name='localhost.localdomain']/vsys&amp;key=apikey</a></td>
</tr>
</tbody>
</table>

| Step 2 Confirm that the XML response for the request looks like the following: |
| <response status="success" code="19"> |
| <completions> |
| <completion value="vsys1" vxpath="/config/devices/entry[@name='localhost.localdomain']/vsys/entry[@name='vsys1']" current="yes" help-string="vsys1"/> |
| </completions> |
| </response> |
Commit Configuration (API)

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

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

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 (Continued)

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

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

Commit-all

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

<table>
<thead>
<tr>
<th>Commit Type</th>
<th>API Request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-commit policy validation.</td>
<td><a href="https://panorama/api/?type=commit&amp;action=all&amp;cmd=">https://panorama/api/?type=commit&amp;action=all&amp;cmd=</a>&lt;commit-all&gt;&lt;shared-policy&gt;&lt;validate-only/&gt;&lt;/validate-only&gt;commit-all&gt;</td>
</tr>
<tr>
<td>Device group commit.</td>
<td><a href="https://panorama/api/?type=commit&amp;action=all&amp;cmd=">https://panorama/api/?type=commit&amp;action=all&amp;cmd=</a>&lt;commit-all&gt;&lt;shared-policy&gt;&lt;device-group&gt;&lt;entry%20name=&quot;device-group-name&quot;/&gt;&lt;/device-group&gt;&lt;/shared-policy&gt;commit-all&gt;</td>
</tr>
<tr>
<td>VSYS commit.</td>
<td><a href="https://panorama/api/?type=commit&amp;action=all&amp;cmd=">https://panorama/api/?type=commit&amp;action=all&amp;cmd=</a>&lt;commit-all&gt;&lt;shared-policy&gt;&lt;device-group&gt;&lt;entry%20name=&quot;device-group-name&quot;/&gt;&lt;/device-group&gt;&lt;devices&gt;&lt;entry%20name=&quot;serial_number&quot;&gt;&lt;/vsys&gt;&lt;member&gt;vsys-name&lt;/member&gt;&lt;/vsys&gt;&lt;/devices&gt;&lt;/device-group&gt;&lt;/shared-policy&gt;commit-all&gt;</td>
</tr>
<tr>
<td>Specific firewall commit.</td>
<td><a href="https://panorama/api/?type=commit&amp;action=all&amp;cmd=">https://panorama/api/?type=commit&amp;action=all&amp;cmd=</a>&lt;commit-all&gt;&lt;shared-policy&gt;&lt;device-group&gt;&lt;entry%20name=&quot;device-group-name&quot;/&gt;&lt;/device-group&gt;&lt;devices&gt;&lt;entry%20name=&quot;serial_number&quot;&gt;&lt;/device-group&gt;&lt;/shared-policy&gt;commit-all&gt;</td>
</tr>
</tbody>
</table>

Use the API Browser to find other options available for granular commit operations on Panorama. In the cmd parameter, you must replace the XML element for the corresponding commit-all operation.
Run Operational Mode Commands (API)

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

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

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

<table>
<thead>
<tr>
<th>Operational Command</th>
<th>API Request</th>
</tr>
</thead>
<tbody>
<tr>
<td>System restart.</td>
<td><a href="https://firewall/api/?type=op&amp;cmd=">https://firewall/api/?type=op&amp;cmd=</a>&lt;request&gt;&lt;restart&gt;&lt;system&gt;&lt;/system&gt;/&lt;request&gt;</td>
</tr>
<tr>
<td>System software version installation.</td>
<td><a href="https://firewall/api/?type=op&amp;cmd=">https://firewall/api/?type=op&amp;cmd=</a>&lt;request&gt;&lt;system&gt;&lt;software&gt;&lt;install&gt;&lt;version&gt;version_number&lt;/version&gt;&lt;/install&gt;&lt;/software&gt;&lt;/system&gt;/&lt;request&gt;</td>
</tr>
<tr>
<td>Multi-vsys mode.</td>
<td><a href="https://firewall/api/?type=op&amp;cmd=">https://firewall/api/?type=op&amp;cmd=</a>&lt;set&gt;&lt;system&gt;&lt;setting&gt;&lt;multi-vsys&gt;&lt;/multi-vsys&gt;&lt;/setting&gt;&lt;/system&gt;/&lt;set&gt;</td>
</tr>
<tr>
<td>Full configuration validation.</td>
<td><a href="https://firewall/api/?type=op&amp;cmd=">https://firewall/api/?type=op&amp;cmd=</a>&lt;validate&gt;&lt;full&gt;&lt;/full&gt;&lt;/validate&gt;</td>
</tr>
<tr>
<td>Configuration saving.</td>
<td><a href="https://firewall/api/?type=op&amp;cmd=">https://firewall/api/?type=op&amp;cmd=</a>&lt;save&gt;&lt;config&gt;&lt;to&gt;filename&lt;/to&gt;&lt;/config&gt;/&lt;save&gt;</td>
</tr>
<tr>
<td>Configuration loading.</td>
<td><a href="https://firewall/api/?type=op&amp;cmd=">https://firewall/api/?type=op&amp;cmd=</a>&lt;load&gt;&lt;config&gt;&lt;from&gt;filename&lt;/from&gt;&lt;/config&gt;/&lt;load&gt;</td>
</tr>
</tbody>
</table>

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 `type=report` parameter.

You can access three kinds of reports:

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

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

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

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

▲ Dynamic Reports
▲ Predefined Reports
▲ Custom Reports

Dynamic Reports

You can use the API to view a number of dynamic reports, such as `top-applications-summary`, `top-blocked-url-summary`, and `top-spyware-threats-summary`. For dynamic reports, provide either a specific period using the `period` or a time frame using `starttime` and `endtime` options (use a + instead of a space between the date and timestamp). Use `topn` to determine the number of rows.

<table>
<thead>
<tr>
<th>Dynamic Report Type</th>
<th>API Request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full dynamic report list.</td>
<td><a href="https://firewall/api/?type=report&amp;reporttype=dynamic">https://firewall/api/?type=report&amp;reporttype=dynamic</a></td>
</tr>
<tr>
<td>Last 60 seconds.</td>
<td><a href="https://firewall/api/?type=report&amp;reporttype=dynamic&amp;reportname=top-app-summary&amp;period=last-60-seconds&amp;topn=5">https://firewall/api/?type=report&amp;reporttype=dynamic&amp;reportname=top-app-summary&amp;period=last-60-seconds&amp;topn=5</a></td>
</tr>
<tr>
<td>Last hour.</td>
<td><a href="https://firewall/api/?type=report&amp;reporttype=dynamic&amp;reportname=top-app-summary&amp;period=last-hour&amp;topn=5">https://firewall/api/?type=report&amp;reporttype=dynamic&amp;reportname=top-app-summary&amp;period=last-hour&amp;topn=5</a></td>
</tr>
<tr>
<td>Last 12 hours.</td>
<td><a href="https://firewall/api/?type=report&amp;reporttype=dynamic&amp;reportname=top-app-summary&amp;period=last-12-hrs&amp;topn=5">https://firewall/api/?type=report&amp;reporttype=dynamic&amp;reportname=top-app-summary&amp;period=last-12-hrs&amp;topn=5</a></td>
</tr>
<tr>
<td>Last calendar day.</td>
<td><a href="https://firewall/api/?type=report&amp;reporttype=dynamic&amp;reportname=top-app-summary&amp;period=last-calendar-day&amp;topn=5">https://firewall/api/?type=report&amp;reporttype=dynamic&amp;reportname=top-app-summary&amp;period=last-calendar-day&amp;topn=5</a></td>
</tr>
<tr>
<td>Last 7 days</td>
<td><a href="https://firewall/api/?type=report&amp;reporttype=dynamic&amp;reportname=top-app-summary&amp;period=last-7-days&amp;topn=5">https://firewall/api/?type=report&amp;reporttype=dynamic&amp;reportname=top-app-summary&amp;period=last-7-days&amp;topn=5</a></td>
</tr>
</tbody>
</table>
Get Reports (API)  

PAN-OS XML API Request Types

<table>
<thead>
<tr>
<th>Dynamic Report Type</th>
<th>API Request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Last 7 calendar days</td>
<td><a href="https://firewall/api/?type=report&amp;reporttype=dynamic&amp;reportname=top-app-summary&amp;period=last-hour&amp;topn=5">https://firewall/api/?type=report&amp;reporttype=dynamic&amp;reportname=top-app-summary&amp;period=last-hour&amp;topn=5</a></td>
</tr>
<tr>
<td>Last calendar week.</td>
<td><a href="https://firewall/api/?type=report&amp;reporttype=dynamic&amp;reportname=top-app-summary&amp;period=last-calendar-week&amp;topn=5">https://firewall/api/?type=report&amp;reporttype=dynamic&amp;reportname=top-app-summary&amp;period=last-calendar-week&amp;topn=5</a></td>
</tr>
<tr>
<td>Last 30 days</td>
<td><a href="https://firewall/api/?type=report&amp;reporttype=dynamic&amp;reportname=top-app-summary&amp;period=last-30-days&amp;topn=5">https://firewall/api/?type=report&amp;reporttype=dynamic&amp;reportname=top-app-summary&amp;period=last-30-days&amp;topn=5</a></td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>Dynamic Report Type</th>
<th>API Request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full predefined report list.</td>
<td><a href="https://firewall/api/?type=report&amp;reporttype=predefined">https://firewall/api/?type=report&amp;reporttype=predefined</a></td>
</tr>
<tr>
<td>Top applications</td>
<td><a href="https://firewall/api/?type=report&amp;async=yes&amp;reporttype=predefined&amp;reportname=top-application-categories">https://firewall/api/?type=report&amp;async=yes&amp;reporttype=predefined&amp;reportname=top-application-categories</a></td>
</tr>
<tr>
<td>Top attackers.</td>
<td><a href="https://firewall/api/?type=report&amp;async=yes&amp;reporttype=predefined&amp;reportname=top-attackers">https://firewall/api/?type=report&amp;async=yes&amp;reporttype=predefined&amp;reportname=top-attackers</a></td>
</tr>
<tr>
<td>Top victims.</td>
<td><a href="https://firewall/api/?type=report&amp;async=yes&amp;reporttype=predefined&amp;reportname=top-victims">https://firewall/api/?type=report&amp;async=yes&amp;reporttype=predefined&amp;reportname=top-victims</a></td>
</tr>
</tbody>
</table>

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 (Continued)

**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:

```xml
<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`
- **Technical support data**—`category=tech-support`
- **Device State**—`category=device-state`

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

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

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

- ▲ Export Packet Captures
- ▲ Export Certificates and Keys
- ▲ Export Technical Support Data

Export Packet Captures

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

- ▲ Export Application PCAPS
- ▲ Export Threat, Filter, and Data Filtering PCAPs

Export Application PCAPS

Application PCAPS are organized by a directory/filename structure where the directory is a date in `yyyy/mm/dd` format. Filenames for application pcaps use a `SourceIP-SourcePort-DestinationIP-DestinationPort-SessionID.pcap` format.

<table>
<thead>
<tr>
<th>Application PCAP Type</th>
<th>API Request</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application PCAP directory list.</td>
<td><code>https://firewall/api/?type=export&amp;category=application-pcap</code></td>
</tr>
<tr>
<td>List of files under a directory using the <code>from</code> parameter to indicate date.</td>
<td><code>https://firewall/api/?type=export&amp;category=application-pcap&amp;from=yyyy/mm/dd</code></td>
</tr>
</tbody>
</table>
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.

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

Export Certificates and Keys

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

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

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

```xml
<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>0K</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:

```xml
<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>
- **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`

- (category=certificate or category=keypair only) Specify these additional parameters for the certificate file and keypair imports:
  - `certificate-name`—name of the certificate object on the firewall
  - `format`—certificate format, pkcs12 or pem
  - `passphrase`—required when including the certificate key
  - `vsys`—virtual system where the certificate object is used. Ignore this parameter if the certificate is a shared object.

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

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

- Import custom logos to different locations based on the `where` parameter:
  ```
  where=<login-screen | main-ui | pdf-report-footer | pdf-report-header>
  ```
Retrieve Logs (API)

Retrieve logs from a firewall using the API.

- **API Log Retrieval Parameters**
- **Example: Use the API to Retrieve Traffic Logs**

## API Log Retrieval Parameters

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

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>log-type</td>
<td>The type of logs to retrieve:</td>
</tr>
<tr>
<td></td>
<td>- log-type=traffic—Traffic logs</td>
</tr>
<tr>
<td></td>
<td>- log-type=threat—Threat logs</td>
</tr>
<tr>
<td></td>
<td>- log-type=config—Config logs</td>
</tr>
<tr>
<td></td>
<td>- log-type=system—System logs</td>
</tr>
<tr>
<td></td>
<td>- log-type=hipmatch—HIP logs</td>
</tr>
<tr>
<td></td>
<td>- log-type=wildfire—WildFire logs</td>
</tr>
<tr>
<td></td>
<td>- log-type=url—URL filtering logs</td>
</tr>
<tr>
<td></td>
<td>- log-type=data—Data filtering logs</td>
</tr>
<tr>
<td></td>
<td>- log-type=corr—Correlated event logs as seen in the user interface within Monitor &gt; Automated Correlated Engine &gt; Correlated Events.</td>
</tr>
<tr>
<td></td>
<td>- log-type=corr-detail—Correlated event details as seen in the user interface when you select an event within Monitor &gt; Automated Correlated Engine &gt; Correlated Events.</td>
</tr>
<tr>
<td></td>
<td>- log-type=corr-categ—Correlated events by category, currently compromised hosts seen within ACC &gt; Threat Activity &gt; Compromised Hosts.</td>
</tr>
<tr>
<td>query</td>
<td>(Optional) Specify the match criteria for the logs. This is similar to the query provided in the web interface under the Monitor tab when viewing the logs. The query must be URL encoded.</td>
</tr>
<tr>
<td>nlogs</td>
<td>(Optional) Specify the number of logs to retrieve. The default is 20 when the parameter is not specified. The maximum is 5000.</td>
</tr>
<tr>
<td>skip</td>
<td>(Optional) Specify the number of logs to skip when doing a log retrieval. The default is 0. This is useful when retrieving logs in batches where you can skip the previously retrieved logs.</td>
</tr>
<tr>
<td>dir</td>
<td>(Optional) Specify whether logs are shown oldest first (forward) or newest first (backward). Default is backward.</td>
</tr>
</tbody>
</table>
Example: Use the API to Retrieve Traffic Logs

Use the API to 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:

```xml
<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
Use the API to Retrieve Traffic Logs (Continued)

Step 3  Confirm that the XML response looks similar to the following:

```xml
<response status="success">
  <result>
    <job>...</job>
    <log>
      <logs count="20" progress="100">
        <entry logid="575304543500710425">
          <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>
        </entry>
      </logs>
    </log>
  </result>
</response>
```

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 an active log retrieval job. To delete an active log retrieval job, run the following query:

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

A successful completion returns a job ID.
Apply User-ID Mapping and Populate Dynamic Address Groups (API)

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

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

or

```
curl --data-urlencode key=apikey --data-urlencode "cmd=xml-document" https://firewall/api/
```

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

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

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

<table>
<thead>
<tr>
<th>Mapping or Registration Action</th>
<th>API Request</th>
</tr>
</thead>
<tbody>
<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:</td>
</tr>
<tr>
<td></td>
<td><code>&lt;uid-message&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;version&gt;1.0&lt;/version&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;type&gt;update&lt;/type&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;payload&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;login&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;entry name=&quot;domain\uid1&quot; ip=&quot;10.1.1.1&quot; timeout=&quot;20&quot;&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;/entry&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;/login&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;groups&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;entry name=&quot;group1&quot;&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;members&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;entry name=&quot;user1&quot;/&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;entry name=&quot;user2&quot;/&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;/members&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;/entry&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;entry name=&quot;group2&quot;&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;members&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;entry name=&quot;user3&quot;/&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;/members&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;/entry&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;/groups&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;/payload&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;/uid-message&gt;</code></td>
</tr>
</tbody>
</table>

You can include a HIP report by including a `<hip-report>` element within an `<entry>` parent element.
<table>
<thead>
<tr>
<th>Mapping or Registration Action</th>
<th>API Request</th>
</tr>
</thead>
</table>
| Multi-User System Entry       | 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.  
<br><uid-message>  
<br><payload>  
<br>  
<br><multiusersystem>  
<br>      <entry ip="10.1.1.2" startport="xxxxx" endport="xxxxx" blocksize="xxx">  
<br>        </entry>  
<br>      </multiusersystem>  
<br></payload>  
<br>  
<br><type>update</type>  
<br><version>1.0</version>  
<br></uid-message> |
| User-ID XML multiuser system login event | 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.  
<br><uid-message>  
<br><payload>  
<br>      <login>  
<br>          <entry name="acme\jparker" ip="10.1.1.23" blockstart="20100">  
<br>            </entry>  
<br>          </login>  
<br>      </payload>  
<br>  
<br><type>update</type>  
<br><version>1.0</version>  
<br></uid-message> |
| User-ID XML multiuser system logout | 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.  
<br><uid-message>  
<br><payload>  
<br>      <logout>  
<br>          <entry user="domain\uid2" ip="10.1.1.2" blockstart="xxxxx">  
<br>            </entry>  
<br>          </logout>  
<br>      </payload>  
<br>  
<br><type>update</type>  
<br><version>1.0</version>  
<br></uid-message> |
<table>
<thead>
<tr>
<th>Mapping or Registration Action</th>
<th>API Request</th>
</tr>
</thead>
</table>
| 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 to show the PAN-OS version for a firewall or Panorama. In addition to the PAN-OS version, this request provides a direct way to obtain the serial number and model number.

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

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>Bad request</td>
<td>A required parameter is missing, an illegal parameter value is used.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden</td>
<td>Authentication or authorization errors including invalid key or insufficient admin access rights. Learn how to <a href="#">Get Your API Key</a>.</td>
</tr>
<tr>
<td>1</td>
<td>Unknown command</td>
<td>The specific config or operational command is not recognized.</td>
</tr>
<tr>
<td>2-5</td>
<td>Internal errors</td>
<td>Check with technical support when seeing these errors.</td>
</tr>
<tr>
<td>6</td>
<td>Bad Xpath</td>
<td>The xpath specified in one or more attributes of the command is invalid.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check the API browser for proper xpath values.</td>
</tr>
<tr>
<td>7</td>
<td>Object not present</td>
<td>Object specified by the xpath is not present.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example, entry[@name='value'] where no object with name 'value' is present.</td>
</tr>
<tr>
<td>8</td>
<td>Object not unique</td>
<td>For commands that operate on a single object, the specified object is not unique.</td>
</tr>
<tr>
<td>10</td>
<td>Reference count not zero</td>
<td>Object cannot be deleted as there are other objects that refer to it.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example, address object still in use in policy.</td>
</tr>
<tr>
<td>11</td>
<td>Internal error</td>
<td>Check with technical support when seeing these errors.</td>
</tr>
<tr>
<td>12</td>
<td>Invalid object</td>
<td>Xpath or element values provided are not complete.</td>
</tr>
<tr>
<td>14</td>
<td>Operation not possible</td>
<td>Operation is allowed but not possible in this case.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example, moving a rule up one position when it is already at the top.</td>
</tr>
<tr>
<td>15</td>
<td>Operation denied</td>
<td>Operation is allowed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example, Admin not allowed to delete own account, Running a command that is not allowed on a passive device.</td>
</tr>
<tr>
<td>16</td>
<td>Unauthorized</td>
<td>The API role does not have access rights to run this query.</td>
</tr>
<tr>
<td>17</td>
<td>Invalid command</td>
<td>Invalid command or parameters.</td>
</tr>
<tr>
<td>18</td>
<td>Malformed command</td>
<td>The XML is malformed.</td>
</tr>
<tr>
<td>19-20</td>
<td>Success</td>
<td>Command completed successfully.</td>
</tr>
<tr>
<td>21</td>
<td>Internal error</td>
<td>Check with technical support when seeing these errors.</td>
</tr>
<tr>
<td>22</td>
<td>Session timed out</td>
<td>The session for this query timed out.</td>
</tr>
</tbody>
</table>