# Cisco ASA REST API Quick Start Guide

**First Published: December 24, 2014**
**Revised Date: June 2, 2015**

## Contents

## Overview

Several options are available for configuring and managing individual Cisco ASAs:

- Command Line Interface (CLI) – you send control commands directly to the ASA via a connected console.
- Adaptive Security Device Manager (ASDM) – an "on-box" management application with a graphical user interface that you can use to configure, manage and monitor an ASA.
- Cisco Security Manager – while intended for medium to large networks of many security devices, this graphical application can be used to configure, manage and monitor individual ASAs.

With the release of Cisco's ASA REST API, you now have another light-weight, easy-to-use option. This is an application programming interface (API), based on "RESTful" principles, which you can quickly download and enable on any ASA on which the API is running.

After installing a REST client in your browser, you can contact the specific ASA's REST agent and use standard HTTP methods to access current configuration information, and issue additional configuration parameters.

**Caution** When the REST API is enabled on an ASA, connections by other security management protocols are not blocked. This means others using CLI, ASDM, or Security Manager could be altering the ASA configuration while you are doing the same.

## ASA REST API Requests and Responses

The ASA REST API gives you programmatic access to managing individual ASAs through a Representational State Transfer (REST) API. The API allows external clients to perform CRUD (Create, Read, Update, Delete) operations on ASA resources; it is based on the HTTPS protocol and REST methodology.

All API requests are sent over HTTPS to the ASA, and a response is returned.

This section provides an overview of how requests are structured, and the expected responses,

### Request Structure

Available request methods are:

- GET – Retrieves data from the specified object.
- PUT – Adds the supplied information to the specified object; returns a 404 Resource Not Found error if the object does not exist.
- POST – Creates the object with the supplied information.
- DELETE – Deletes the specified object.
- PATCH – Applies partial modifications to the specified object.

### Response Structure

Each request produces an HTTPS response from the ASA with the standard headers, response content, and status code.

The response structure can be:

- LOCATION – Newly created resource ID; for POST only—holds the new resource ID (as a URI representation).
- CONTENT-TYPE – Media type describing the response message body; describes the representation and syntax of the response message body.

Each response includes an HTTP status or error code. Available codes fall into these categories:

- `20x` - A two-hundred series code indicates successful operation, including:
  - `200 OK` – Standard response for successful requests.
  - `201 Created` – Request completed; new resource created.
  - `202 Accepted` – Request accepted, but processing not complete.
  - `204 No Content` – Server successfully processed request; no content is being returned.

- `4xx` - A four-hundred series code indicates a client-side error, including:
  - `400 Bad Request` – Invalid query parameters, including unrecognized parameters, missing parameters, or invalid values.
  - `404 Not Found` – The provided URL does not match an existing resource. For example, an HTTP DELETE may fail because the resource is unavailable.
  - `405 Method not Allowed` – An HTTP request was presented that is not allowed on the resource; for example, a POST on a read-only resource.
- `5xx` - A five-hundred series code indicates a server-side error.

In the case of an error, in addition to the error code, the return response may include an error object containing more details about the error. The JSON error/warning response schema is as follows:

```
[
   { "code"   : "string",
       "details": "string",
       "context": attribute name,
       "level"  : <Error/Warning/Info>
   },
   ...
]
```

where the object properties are:

| Property | Type | Description |
|---|---|---|
| `messages` | List of Dictionaries | List of error or warning messages |
| `code` | String | Error/Warning/Info code |
| `details` | String | Detailed message corresponding to Error/Warning/Info |

**Note** Changes to the ASA configuration made by REST API calls are not persisted to the startup configuration; that is, changes are assigned only to the running configuration. To save changes to the startup configuration, you can POST a `writemem` API request; see Write Memory API for more information.

# Install and Configure the ASA REST API Agent and Client

These sections described installing and enabling the REST API agent on an ASA, and adding and configuring a REST API client on your local host.

## Download and Install the REST API Agent

Using the CLI, follow these steps to download and install the ASA REST API agent on a specific ASA:

**Step 1** On the desired ASA, issue the `copy <package> disk0:` command to download the current ASA REST API package from cisco.com to the ASA's flash memory. For example:

```
copy tftp://10.7.0.80/asa-restapi-111-lfbff-k8.SPA disk0:
```

**Step 2** Issue the `rest-api image disk0:/<package>` command to verify and install the package.
For example:

```
rest-api image disk0:/asa-restapi-111-lfbff-k8.SPA
```

The installer will perform compatibility and validation checks, and then install the package. The ASA will not reboot.

## Enable and Configure the REST API Agent

Follow these steps to enable and configure the ASA REST API Agent on the specific ASA:

**Step 1** Ensure the correct software image is installed on the ASA.

For the ASA 55*nn*-X, this is `asa100-12-16-15-smp-k8.bin` or later. For an ASAv, this is `asav100-12-16-15.ova`. If a `.ova` image is already installed on your ASAv, you can use `asa100-12-16-15-smp-k8.bin` to update the VM image.

**Step 2** Using the CLI, ensure the HTTP server is enabled on the ASA, and that API clients can connect to the management interface. For example:

```
http server enable
http 0.0.0.0 0.0.0.0 <management interface nameif>
```

**Step 3** Using the CLI, define HTTP authentication for the API connections. For example:

```
aaa authentication http console LOCAL
```

**Step 4** Using the CLI, create a local user with read/write privileges, in this case, privilege level 15. For example:

```
username <user name> password <password> encrypted privilege 15
```

Privilege level 3 or greater is required to invoke monitoring requests, while Privilege level 5 or greater is required for invoking GET requests, and Privilege level 15 is necessary for invoking PUT/POST/DELETE operations.

**Step 5** Using the CLI, create a static route on the ASA for API traffic. For example:

```
route <management interface nameif> 0.0.0.0 0.0.0.0 <gwip> 1
```

**Step 6** Using the CLI, enable the ASA REST API Agent on the ASA. For example:

```
rest-api agent
```

## Configure Your REST API Client

Follow these steps to install and configure a REST API client on your local-host browser:

**Step 1** Acquire and install a REST API client for your browser

For Chrome, install the REST client from Google. For Firefox, install the RESTClient add-on. Internet Explorer is not supported.

**Step 2** Initiate the following request using your browser:

```
https:<asa management ip address>/api/objects/networkobjects
```

If you receive a non-error response, you have reached the REST API agent functioning on the ASA.

If you are having issues with the agent request, you can enable display of debugging information on the CLI console, as described in Enabling REST API Debugging on the ASA, page 6.

**Step 3** Optionally, you can test your connection to the ASA by performing a POST operation.

For example:

Provide basic authorization credentials (<*username*><*password*>), or an authentication token (see Token Authentication for additional information).

Target request address: `https://<asa management ipaddress>/api/objects/networkobjects`

Body content type: `application/json`

Raw body of the operation:

```
{
  "kind": "object#NetworkObj",
  "name": "TestNetworkRangeObj",
  "host": {
    "kind": "IPv4Network",
    "value": "12.12.12.0/24"
  }
}
```

You can now use the ASA REST API to configure and monitor the ASA. Refer the API documentation for call descriptions and examples.

## The Documentation Console and Exporting API Scripts

You also can use the REST API on-line documentation console (referred to as the "Doc UI"), available at *host:port*/doc/ as a "sandbox" for learning about and trying the API calls directly on the ASA.

Further, you can use the **Export Operation** button in the Doc UI to save the displayed method example as a JavaScript, Python, or Perl script file to your local host. You can then apply this script to your ASA, and edit it for application on other ASAs and other network devices. This meant primarily as an educational and bootstrapping tool.

### JavaScript

Using a JavaScript file requires installation of `node.js`, which can be found at `http://nodejs.org/`. Using `node.js`, you can execute a JavaScript file, typically written for a browser, like a command-line script. Simply follow the installation instructions, and then run your script with `node script.js`.

## Python

The Python scripts require you to install Python, available from `https://www.python.org/`. Once you've installed Python, you can run your script with `python script.py` *`username password`*.

## Perl

Using the Perl scripts requires some additional set-up—you need five components: Perl itself, and four Perl libraries:

- Perl package, found at `http://www.perl.org/`
- Bundle::CPAN, found at `http://search.cpan.org/~andk/Bundle-CPAN-1.861/CPAN.pm`
- REST::Client, found at `http://search.cpan.org/~mcrawfor/REST-Client-88/lib/REST/Client.pm`
- MIME::Base64, found at `http://perldoc.perl.org/MIME/Base64.html`
- JSON, found at `http://search.cpan.org/~makamaka/JSON-2.90/lib/JSON.pm`

Here is an example of bootstrapping Perl on a Macintosh:

```
$ sudo perl -MCPAN e shell
cpan> install Bundle::CPAN
cpan> install REST:: Client
cpan> install MIME::Base64
cpan> install JSON
```

After installing the dependencies, you can run your script using `perl script.pl` *`username password`*.

## Enabling REST API Debugging on the ASA

If you are having problems configuring or connecting to the REST API on the ASA, you can use the following CLI command to enable display of debugging messages on your console. Use the **no** form of the command to disable the debug messages.

**debug rest-api [agent | cli | client | daemon | process | token-auth] [error | event]**

**no debug rest-api**

**Syntax Description**

| | |
|---|---|
| **agent** | (Optional) Enable REST API Agent debugging information. |
| **cli** | (Optional) Enable debugging messages for REST API CLI Daemon-to-Agent communications. |
| **client** | (Optional) Enable debugging information for Message routing between the REST API Client and the REST API Agent. |
| **daemon** | (Optional) Enable debugging messages for REST API Daemon-to-Agent communications. |
| **process** | (Optional) Enable REST API Agent process start/stop debugging information. |
| **token-auth** | (Optional) REST API token authentication debugging information. |

| | |
|---|---|
| **error** | (Optional) Use this keyword to limit debug messages to only errors logged by the API. |
| **event** | (Optional) Use this keyword to limit debug messages to only events logged by the API. |

**Usage Guidelines** If you do not provide a specific component keyword (that is, if you simply issue the command **debug rest-api**), debug messages are displayed for all component types. If you do not provide either the **event** or **error** keyword, both event and error messages are displayed for the specified component. For example, **debug rest-api daemon event** will show only event debug messages for API Daemon-to-Agent communications.

**Related Commands**

| Command | Description |
|---|---|
| **debug http** | Use this command to view detailed information about HTTP traffic. |

## ASA REST API-related Syslog Messages

The ASA REST API-related system-log messages are described in this section.

### 342001

**Error Message** `%ASA-7-342001: REST API Agent started successfully.`

**Explanation** The REST API Agent must be successfully started before a REST API Client can configure the ASA.

**Recommended Action** None.

### 342002

**Error Message** `%ASA-3-342002: REST API Agent failed, reason: reason`

**Explanation** The REST API Agent could fail to start or crash for various reasons, and the reason is specified.

- *reason*—The cause for the REST API failure

**Recommended Action** The actions taken to resolve the issue vary depending on the reason logged. For example, the REST API Agent crashes when the Java process runs out of memory. If this occurs, you need to restart the REST API Agent. If the restart is not successful, contact the Cisco TAC to identify the root cause fix.

## 342003

**Error Message** `%ASA-3-342003: REST API Agent failure notification received. Agent will be restarted automatically.`

**Explanation** A failure notification from the REST API Agent has been received and a restart of the Agent is being attempted.

**Recommended Action** None.

## 342004

**Error Message** `%ASA-3-342004: Failed to automatically restart the REST API Agent after 5 unsuccessful attempts. Use the 'no rest-api agent' and 'rest-api agent' commands to manually restart the Agent.`

**Explanation** The REST API Agent has failed to start after many attempts.

**Recommended Action** See syslog %ASA-3-342002 (if logged) to better understand the reason behind the failure. Try to disable the REST API Agent by entering the **no rest-api agent** command and re-enable the REST API Agent using the **rest-api agent** command.

## Related Documentation

Use the following link to find more information about the ASA, and its configuration and management:

- *Navigating the Cisco ASA Series Documentation*: http://www.cisco.com/go/asadocs

Use the following link to view a list of ASA features not supported on the ASAv:

- http://www.cisco.com/c/en/us/td/docs/security/asa/asa92/configuration/general/asa-general-cli/intro-asav.html#pgfId-1156883