Overview of AsyncOS API for Cisco Secure Web Appliance

The AsyncOS API for Cisco Secure Web Appliance (or AsyncOS API) is a representational state transfer (REST) based set of operations that provide secure and authenticated access to the Secure Web Appliance reports, report counters, and tracking. You can retrieve the Secure Web Appliance reporting and tracking data using the API. In this release you can query for configuration information.

Note You can configure Secure Web Appliance using Cisco Content Security Management appliance and REST APIs. If you use both these methods to configure the Secure Web Appliance, configurations done by the previous method are overwritten.

This chapter contains the following sections:

Prerequisites for Using AsyncOS API

To use AsyncOS API, you must have knowledge of:

- HTTP, which is the protocol used for API transactions. Secure communication over TLS.
- JavaScript Object Notation (JSON), which the API uses to construct resource representations.
- JSON Web Token (JWT).
- A client or programming library that initiates requests and receives responses from the AsyncOS API using HTTP or HTTPS, for example, cURL. The client or programming library must support JSON to interpret the response from the API.
- Authorization to access the AsyncOS API. See Authorization, on page 4.

- AsyncOS API enabled using web interface or CLI. See Enabling AsyncOS API, on page 2.

Enabling AsyncOS API

Before You Begin

Ensure you have access to the `interfaceconfig` command in the CLI. Access to the CLI is restricted only to authorized personnel, who are administrators, email administrators, cloud administrators, and operators.

You can enable the AsyncOS API using the `interfaceconfig` command in the CLI.

Step 1 Log in to the CLI and run the `interfaceconfig` command.

Step 2 Choose the interface that you want to edit.

Step 3 Answer the following questions to enable AsyncOS API (monitoring) HTTP:

- `Do you want to enable AsyncOS API (monitoring) HTTP on this interface? [Y]>` Enter Y.
- `Which port do you want to use for AsyncOS API (monitoring) HTTP?[6080]>` Enter the default port 6080 or the port you want to define.

Step 4 Answer the following questions to enable AsyncOS API (monitoring) HTTPS:

- `Do you want to enable AsyncOS API (Monitoring) HTTPS on this interface? [Y]>` Enter Y.
- `Which port do you want to use for AsyncOS API (Monitoring) HTTPS?[6443]>` Enter the default port 6443 or the port you want to define.

Note AsyncOS API communicates using HTTP / 1.1.

If you have selected HTTPS and want to use your own certificate for secure communication, see Securely Communicating with AsyncOS API, on page 2.

Note We recommend that you always use HTTPS in the production environment. Use HTTP only for troubleshooting and testing the API.

Step 5 Submit and commit the changes.

Securely Communicating with AsyncOS API

You can communicate with AsyncOS API over secure HTTP using your own certificate.

Note Do not perform this procedure if you are already running the web interface over HTTPS and using your own certificate for secure communication. AsyncOS API uses the same certificate as the web interface for communicating over HTTPS.

Step 1 Set up a certificate using the `certconfig` command in the CLI. For instructions, refer the User Guide or Online Help.

Step 2 Change the HTTPS certificate used by the IP interface to your certificate using the `interfaceconfig` command in CLI. For instructions, refer the User Guide or Online Help.

Step 3 Submit and commit your changes.

AsyncOS API Authentication and Authorization

This section explains the authentication methods, the user roles that can access APIs, and how to query for APIs accessible to a user.

- Authentication, on page 3
- Authorization, on page 4

Authentication

You can authenticate queries to the API using either of the following two methods:

- Submit the Secure Web Appliance's username and password with all the requests to the API, in the Base64-encoded format.
- Use a JSON Web Token (JWT) in an API request with the token key in the header.

The user inactivity timeout settings in the appliance apply to the validity of a JWT. If a request does not include valid credentials in the authorization header, the API sends a 401 error message. You can use any base64 library to convert your credentials into a base64-encoded format.

Authenticating API Queries with JSON Web Token

You can generate a JWT and use it with your API queries.

Note The user inactivity timeout settings in the appliance apply to the validity of a JWT. The Secure Web Appliance checks every API query with a JWT, for its time validity. If a JWT is found to be within 5 minutes of time validity, after which it will time out, a new refresh JWT is sent with the response header. You must use this new refresh JWT with API queries or generate a new one.

Synopsis	`POST /wsa/api/v2.0/login`
	Use the syntax below for two factor authentications:
	`POST /wsa/api/v2.0/login/two_factor`

Body Parameters	Use Base64 encoded credentials.
	`{`
	`    "data":`
	`    {`
	`        "userName":"YWRtaW4=",`
	`        "passphrase":"aXJvbnBvcnQ="`
	`    }`
	`}`

Request Headers		Host, Accept, Authorization
Response Headers		Content-Type, Content-Length, Connection

This example shows a query to log in with Base64 encoded credentials, and generate a JWT.

Sample Request

```
POST /wsa/api/v2.0/login
HTTP/1.1
Content-Type: application/json
cache-control: no-cache
User-Agent: curl/7.54.0
Accept: */*
Host: wsa.cisco.com:6080
accept-encoding: gzip, deflate
content-length: 95
Connection: keep-alive
{
    "data":
    {
        "userName":"YWRtaW4=",
        "passphrase":"aXJvbnBvcnQ="
    }
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: API/2.0
Date: Mon, 26 Nov 2018 07:22:47 GMT
Content-type: application/json
Content-Length: 618
Connection: close
Access-Control-Allow-Origin: *
Access-Control-Allow-Headers: content-type, jwttoken, mid, h, email
Access-Control-Allow-Credentials: true
Access-Control-Allow-Methods: GET, POST, DELETE, OPTIONS
Access-Control-Expose-Headers: Content-Disposition, jwtToken

{
    "data": {
        "userName": "admin",
        "is2FactorRedirectRequired": "false",
        "role": "Administrator",
        "email": [],
        "jwtToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiaXM
         yRmFjdG9yQ2hlY2tSZXF1aXJlZCI6ZmFsc2UsImNvb2tpZSI6IlRucEZOVTFFWTNwTlZFMDlDanRMYVR
         oeENqdFpiV1J6VFVSQk5VMURNWGRpTWxGMVdUSnNlbGt5T0hWWk1qbDBUMnBaZDA5RVFUMEtcbk8xVkh
         PWHBrUnpGb1lteEtNV0p1VW5CaVYxVjJUbmswTUV4cVFUMEtPMVJVUlhkTlJsazNUVlJKZFUxRE5IZE1
         WRWw1VFdwek1FMXFcblNUVlNhazVDVDBWRk1rOUVaM2xTUlVreVRYcGtSazFwTVVSTlZFMHpUbFZXUjA1
    }
}
```

Authorization

The AsyncOS API is a role based system, the scope of API queries is defined by the role of the user. Cisco Secure Web Appliance users with the following roles can access the AsyncOS API:

- Administrator
- Operator
- Technician
- Read-Only Operator
- Guest
- Web Administrator
- Web Policy Administrator
- URL Filtering Administrator
- Email Administrator
- Help Desk User

Note

- Externally authenticated users can access the API.
- Custom roles, delegated by the administrator, can also access the APIs.
- Only users with administrative privileges can use the REST APIs to modify the configurations. All other users like Operator or Read-Only Operator are allowed to only view these configurations.

AsyncOS API Requests and Responses

Note For complete list of APIs, see AsyncOS API - Addendum to the Getting Started Guide for Secure Web Appliance for more information.

AsyncOS API Requests

Requests made to the API have the following characteristics:

- Requests are sent over HTTP or HTTPS.
- Each request must contain a valid URI in the following format:

  ```
  http://{appliance}:{port}/wsa/api/v2.0/{resource}/{resource_attributes}
  ```

  ```
  https://{appliance}:{port}/wsa/api/v2.0/{resource}/{resource_attributes}
  ```

  where:

  - `{appliance}:{port}`

    is the FQDN or the IP address of the appliance and the TCP port number on which the appliance is listening.
  - `{resource}`

is the resource you are attempting to access, for example, reports, tracking, quarantine, configuration, or other counters.

- `{resource_attributes}`

  are the supported attributes for a resource, for example, duration, and so on.

- Each request must contain user credentials, or a valid authorization header.
- Use the JSON Web Token (JWT) generated earler in the API request with the token key in the header. For more information, see Authenticating API Queries with JSON Web Token.
- Each request must be set to accept:

  ```
  application/json
  ```

- Requests sent over HTTPS (using your own certificate) must contain your CA certificate. For example, in case of cURL, you can specify the CA certificate in the API request as follows:

  ```
  curl --cacert <ca_cert.crt> -u"username:password"

  https://<fqdn>:<port>/wsa/api/v2.0/{resource}/{resource_attributes}
  ```

Note API requests are case sensitive and should be entered as shown in this guide.

AsyncOS API Responses

This section explains the key components of the responses and various HTTP error codes.

- Key Components of Responses, on page 6
- HTTP Response Codes, on page 7

Key Components of Responses

Components		Values	Description
Status Code and Reason		See HTTP Response Codes, on page 7.	HTTP response code and the reason.
Message Header	Content-Type	`application/json`	Indicates the format of the message body.
	Content-Length	n/a	The length of the response body in octets.
	Connection	`close`	Options that are desired for the connection.

Components	Values	Description

Message Body	n/a	The message body is in the format defined by the Content-Type header. The following are the components of the message body:
		1. URI. The URI you specified in the request to the API.
		Example
		`:"/api/v2.0/config/"`
		2. Counter group and/or counter name
		Example
		`reporting/mail_security_summary`
		3. Query parameters
		Example
		`startDate=2017-01-30T00:00:00.000Z&endDate=2018-01-30T14:00:00.000Z`
		4. Error (Only for Error Events). This component includes three subcomponents—message, code, and explanation.
		Example
		`"error": {"message": "Unexpected attribute - starts_with.","code": "404", "explanation": "404 = Nothing matches the given URI."}`
		If the message body contains empty braces ({}), it means that the API could not find any records matching the query.
		Note **totalCount** is the number of data objects that are returned in a dataset (for results that are displayed as table format in the UI). For other queries, it returns -1 by default.

HTTP Response Codes

These are the list of HTTP response codes returned by AsyncOS API:

- 200
- 202
- 300
- 301
- 307

- 400
- 401
- 403
- 404
- 406
- 413
- 414
- 500
- 501
- 503
- 505

For descriptions of these HTTP response codes, refer to the following RFCs:

- RFC1945
- RFC7231

AsyncOS API Capabilities

You can use the AsyncOS API to retrieve information in the following categories:

- APIs for Web
- General Purpose APIs