FORESCOUT

Forescout

eyeExtend Connect Module: Web API Plugin

Configuration Guide

Versions 1.5.2 and 1.5.3

Contact Information

Forescout Technologies, Inc.

190 West Tasman Drive

San Jose, CA 95134 USA

https://www.Forescout.com/support/

Toll-Free (US): 1.866.377.8771

Tel (Intl): 1.408.213.3191

Support: 1.708.237.6591

About the Documentation

- Refer to the Technical Documentation page on the Forescout website for additional documentation: https://www.Forescout.com/company/technical-documentation/
- Have feedback or questions? Write to us at documentation@forescout.com

Legal Notice

2020-11-05 13:33

Table of Contents

About eyeExtend Connect Module: Web API 4

How to Work with Web API Plugin 4

How to Install Web API Plugin 7

Configure Web API Plugin 8

Program Your Web Service Application 11

RESTful Web Service Interaction 11

Web Service Transactions for Data Retrieval 14

Error Responses from Web Service 24

Web Service Logs 25

Change the Validity Period for Web Service Communication 26

Appendix 1: Forescout Property and Data Types 26

About eyeExtend Connect Module: Web API

The Forescout eyeExtend Connect Module: Web API Plugin lets external entities communicate with the Forescout platform using simple, yet powerful web service requests based on HTTP interaction. Web API provides web service functionality. Install this plugin on Enterprise Manager or standalone Appliances to work with the CounterACT® web service.

Information can be submitted to the Forescout platform using web service requests with an XML data body. The CounterACT Web Service parses the data to update Forescout platform host properties.

Supported Forescout Platform Version

The following table lists the Forescout platform version that works with each version covered by this guide.

Version	Forescout Platform Version
1.5.2	Minimum version: 8.1.4 or 8.2.1
1.5.3	Minimum version: 8.1.4 or 8.2.1

About Certification Compliance Mode

Forescout eyeExtend Connect Module: Web API Plugin supports Certification Compliance mode. For information about this mode, refer to "Certification Compliance" in the *Forescout Installation Guide.*

About Support for Dual Stack Environments

The Forescout platform detects endpoints and interacts with network devices based on both IPv4 and IPv6 addresses. However, ***IPv6 addresses are not yet supported by this eyeExtend module.*** The functionality described in this document is based only on IPv4 addresses. IPv6-only endpoints are typically ignored or not detected by the properties, actions, and policies provided by this eyeExtend module.

How to Work with Web API Plugin

This topic describes how to work with the module and module requirements.

What to Do

Perform the following steps to set up your system.

1. Install the plugin. See How to Install Web API Plugin.
2. Configure web service users and security features. See Configure Web API Plugin.
3. Script, test, and launch your platform's interaction with the CounterACT web service. See Program Your Web Service Application.
4. Examine logs to troubleshoot the service, if necessary. See Web Service Logs.

Requirements

This section describes system requirements, including:

- Forescout Requirements
- Forescout eyeExtend Connect Licensing Requirements
- Networking Requirements

Forescout Requirements

Install Forescout eyeExtend Connect Module: Web API Plugin only on Enterprise Manager or a standalone Appliance. Appliances managed by an Enterprise Manager cannot host the CounterACT web service.

Web API requires the following:

- A module license for the Open Integration Module or the eyeExtend Connect Module. See Forescout eyeExtend Connect Licensing Requirements.

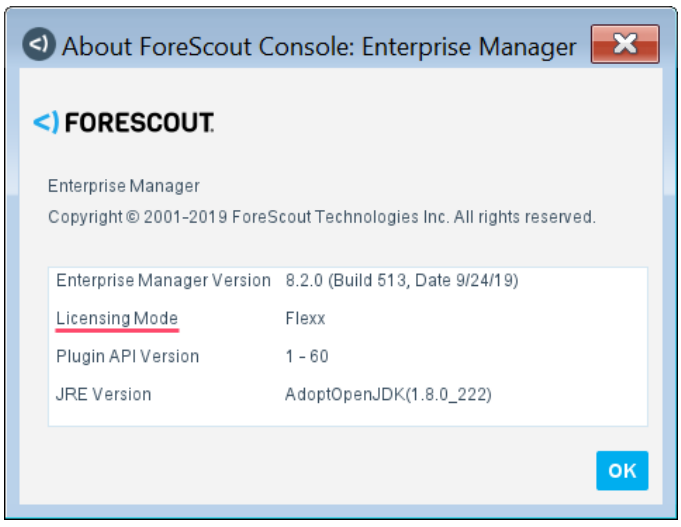
Forescout eyeExtend Connect Licensing Requirements

This Forescout eyeExtend module requires a valid license. Licensing requirements differ based on which licensing mode your deployment is operating in:

- Per-Appliance Licensing Mode
- Flexx Licensing Mode

To identify your licensing mode:

- From the Console, select **Help > About Forescout**.



Per-Appliance Licensing Mode

When installing the module, you are provided with a 90-day demo license.

If you would like to continue exploring the module before purchasing a permanent license, you can request a demo license extension. Consult with your Forescout representative before requesting the extension. You will receive email notification and alerts at the Console before the demo period expires.

To continue working with the module after the demo period expires, you must purchase a permanent module license.

Demo license extension requests and permanent license requests are made from the Console.

> *This module may have been previously packaged as a component of an Integration Module which contained additional modules. If you already installed this module as a component of an Integration Module, you can continue to use it as such. Refer to the section about module packaging in the Forescout Administration Guide for more information.*

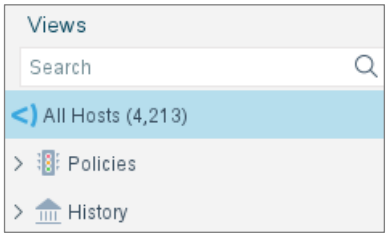
Requesting a License

When requesting a demo license extension or permanent license, you are asked to provide the device *capacity* requirements. This is the number of devices that you want this license to handle. You must define at least the number of devices currently detected by the Forescout platform. You can request a license that handles more to ensure that you are licensed for support on additional devices as your deployment grows.

Enter this number in the **Devices** pane of the Module License Request wizard, in the Console Modules pane.

To view the number of currently detected devices:

1. Select the **Home** tab.
2. In the Views pane, select the **All Hosts** folder. The number in parentheses displayed next to the **All Hosts** folder is the number of devices currently detected.



Flexx Licensing Mode

When you set up your Forescout deployment, you must activate a license file containing valid licenses for each feature you want to work with in your deployment, including eyeExtend modules. After the initial license file has been activated, you can update the file to add additional eyeExtend licenses or change endpoint capacity for existing eyeExtend modules. For more information on obtaining eyeExtend licenses, contact your Forescout sales representative.

- *No demo license is automatically installed during system installation.*

License entitlements are managed in the Forescout Customer Portal. After an entitlement has been allocated to a deployment, you can activate or update the relevant licenses for the deployment in the Console.

Each eyeExtend license has an associated capacity, indicating the number of endpoints the license can handle. The capacity of each eyeExtend license varies by module but does not exceed the capacity of the Forescout eyeSight license.

- *Integration Modules, which package together groups of related licensed modules, are not supported when operating in Flexx Licensing Mode. Only eyeExtend modules, packaging individual licensed modules are supported.* The eyeExtend Connect Module is an *eyeExtend* module even though it packages more than one module.

More License Information

For more information on eyeExtend (Extended Module) licenses:

- **Per-Appliance Licensing**. Refer to the *Forescout Administration Guide*.
- **Flexx Licensing**. Refer to the *Flexx Licensing How-to Guide*.

You can also contact your Forescout sales representative for more information.

Networking Requirements

By default, the web service receives inbound requests to Enterprise Manager or a standalone Appliance using HTTPS on port 443/TCP.

How to Install Web API Plugin

This topic describes how to install Web API.

To install the module:

1. Navigate to one of the following Forescout download portals, depending on the licensing mode your deployment is using:
   - Product Updates Portal - ***Per-Appliance Licensing Mode***
   - Customer Portal, Downloads Page - ***Flexx Licensing Mode***

   To identify your licensing mode, select **Help > About ForeScout** from the Console.
2. Download the module `.fpi` file.

3. Save the file to the machine where the Console is installed.
4. Log into the Console and select **Options** from the **Tools** menu.
5. Select **Modules**. The Modules pane opens.
6. Select **Install**. The Open dialog box opens.
7. Browse to and select the saved module `.fpi` file.
8. Select **Install**. The Installation screen opens.
9. Select **I agree to the License Agreement** to confirm that you have read and agree to the terms of the License Agreement and select **Install**. The installation cannot proceed unless you agree to the license agreement.

   - *The installation begins immediately after selecting Install and cannot be interrupted or canceled.*
   - *In modules that contain more than one component, the installation proceeds automatically one component at a time.*

10. When the installation completes, select **Close** to close the window. The installed module is displayed in the Modules pane.

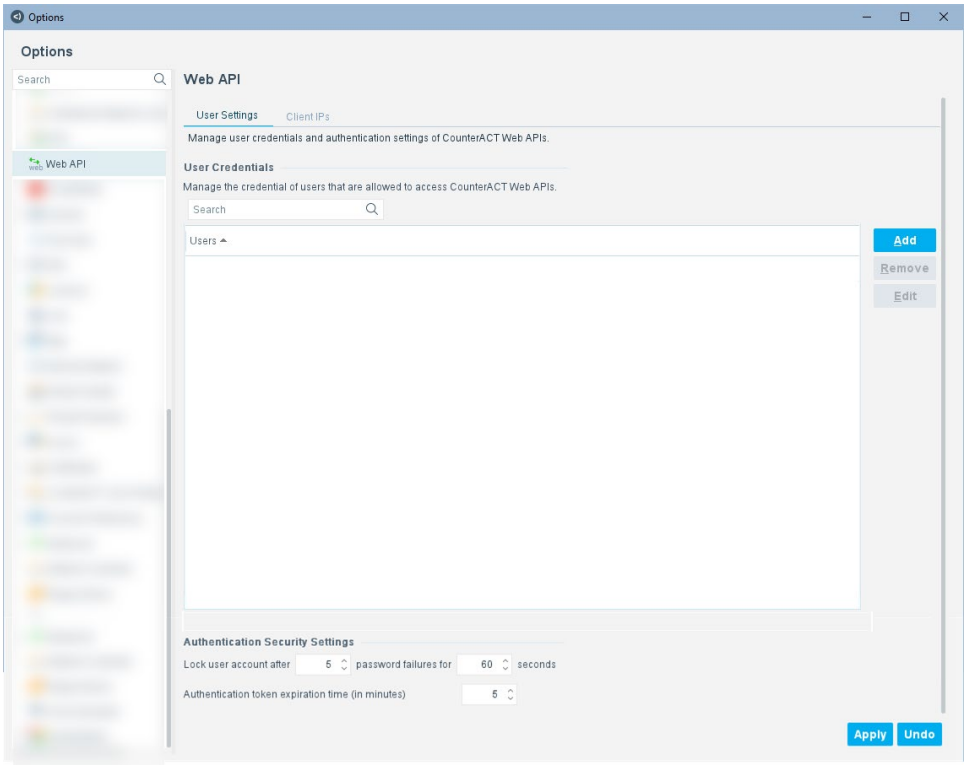
    - *Some components are not automatically started following installation.*

Configure Web API Plugin

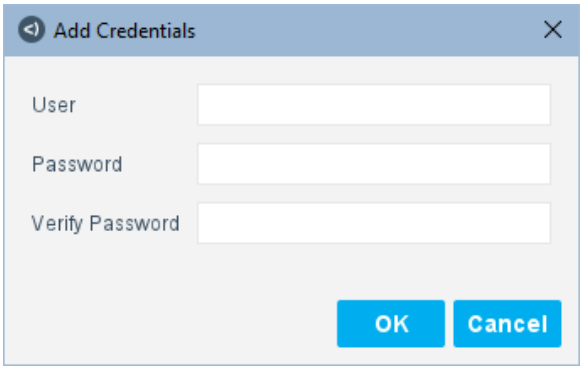
Configure Web API to define the users that can contact the web service, and to enable optional features that secure API access.

To configure Web API:

1. Select **Options** from the Console **Tools** menu.
2. Select **Web API**.



3. Define Web Service User Accounts. Client platforms must use the credentials you define in this step when they communicate with the web service. The Forescout platform uses these credentials to authenticate HTTP request messages sent to the service. In the User Credentials area of the User Settings tab, select **Add**.

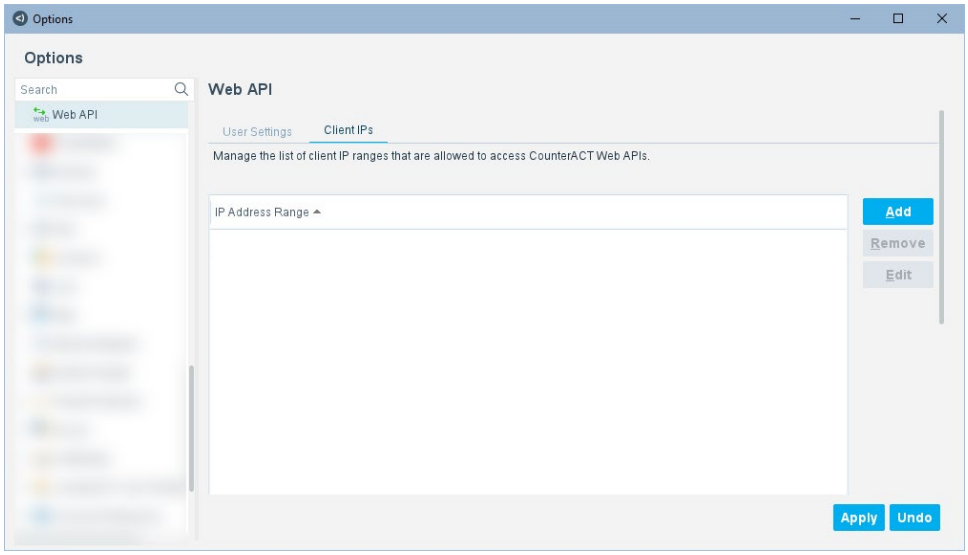


4. Define a user name and password for the web service client. Select **OK**. The user is displayed in the Users table.
5. Define Authentication Security Settings. These settings protect the service from brute-force login attacks and define the timeout behavior for valid API sessions.

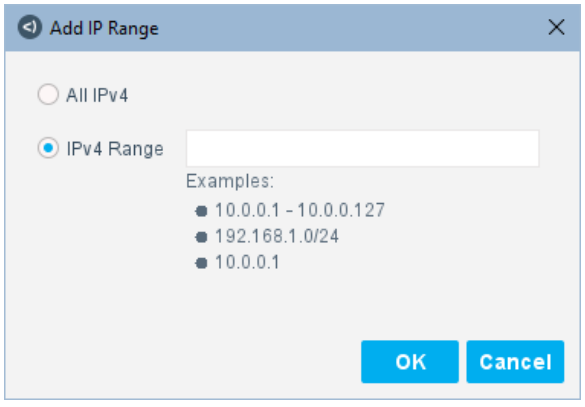
Lock user account after _ password failures	The number of times a user login can fail before the service blocks the account.

for _ seconds	The length of time (in seconds) to block a user account after the specified number of failed login attempts.
Authentication token expiration time (in minutes)	The validity period of a client session after successful user login. Client sessions are terminated after this interval, and clients must log in again.

6. To define the web access IP ranges, go to **Options > Access > Web**. Refer to "Define Web Access" in the *Forescout Administration Guide* for details.
7. To define the IP addresses that can connect to the service, select the Client IPs tab.



8. Select **Add**.



9. Do one of the following:
   - To allow all IPv4 addresses to connect to the service, select **All IPv4**.
   - To specify a range of IP addresses that can access the service, select **IPv4 Range** and enter values.
10. Select **OK**. The IP address range is added.

Login attempts from IP addresses not within the specified range(s) are rejected, even if the user credentials are correct.

Program Your Web Service Application

Applications on external platforms submit requests to the web service using simple HTTPS messages. Developers define and test the commands or routines that generate and submit messages to the web service.

- The web service is available while Web API is installed and running on Enterprise Manager or a standalone Appliance.
- Clients must use the user credentials defined in the configuration to communicate with the web service.
- In large deployments, only Enterprise Manager hosts the web service. Request messages must be addressed only to Enterprise Manager.

Refer to "Appendix 4: Submit Data with the Forescout Web API" in the Forescout eyeExtend Connect Module: Data Exchange Plugin Configuration Guide.

The following topics describe web service interactions:

- RESTful Web Service Interaction
- Web Service Transactions for Data Retrieval
- Error Responses from Web Service
- Web Service Logs
- Change the Validity Period for Web Service Communication

RESTful Web Service Interaction

This topic provides a general overview of the HTTP request and response messages exchanged by external platforms and the CounterACT Web Service.

- Request Messages
- Responses
- JSON Web Token (JWT) Authentication
- Using Entity Tags in Follow-up Requests
- Service Interaction Considerations

Request Messages

These general points apply to the Web Service requests described in this guide:

- Requests use secured HTTP (HTTPS)
- All URIs are built on the network address of the Enterprise Manager or standalone Appliance. In sample URIs this is abbreviated as *{EM.IP}*.

- Requests are authorized by a JSON Web Token (JWT) issued by the service.
- The optional header specifies the expected MIME type of data in the response:

  `Accept: application/hal+json`
- When a request is repeated, the **`If-None-Match`** header can be used with the entity tag of the previous transaction to check if previously reported data has been updated in the Forescout platform. See Using Entity Tags in Follow-up Requests.

The following example shows a typical request message.

```
GET https://{EM.IP}/api/hosts
Host: {EM.IP}
Authorization: u7jV0PcoXLh5KKAJOQm4ek6jsqpuEaILKQ3MnOiHoYcyI1yZgLTxWkqqCH3A
emv4MlAjL9zvyjJ.geOP9LCLtbUCVK6wApa16Ymqm1vHBDALtrtydTx2kzAhJGrFr3HFgrTbZTj
z.d3KJFWYWhuAwU0brDKkMTcRjmzbZovqamn6bgYhMIf-None-Match:
"0cb3980dc881b0b4e47165546"
Accept: application/hal+json
```

Responses

The following general points apply to the CounterACT Web Service response messages described in this guide.

- A standard HTTP response message is used. Standard HTTP Status Codes indicate success or failure of the transaction.
- The Last-Modified header indicates the date and time the response was sent.
- An Entity Tag (ETag) identifies the transaction. See Using Entity Tags in Follow-up Requests for details.
- The response message includes a JSON message body containing the requested information in HAL format. This is indicated by the Content-Type line of the response header.

The following example shows a typical response message.

```
HTTP/1.1 200 OK
Date: Wed, 15 Nov 2013 04:58:08 GMT
Content-Type: application/hal+json
Last-Modified: Wed, 15 Nov 2013 04:58:08 GMT
ETag: "0cb3980dc881b0b4e47165546"
[
    {
        "name": "in-group",
        "label": "Member of Group",
        "description": "Indicates host is a member of a CounterACT group."
    }
]
```

JSON Web Token (JWT) Authentication

Requests are authorized using a JSON Web Token (JWT) issued by the web service.

A client initially logs in to web service using the credentials provided by the Forescout administrator.

- This login is an HTTPS POST message

- The target URI is **`https://{EM.IP}/api/login`**

The body of the message includes the user name and password, as shown in the following example.

```
POST https://{EM.IP}/api/login
Host: {EM.IP}
Content-Type: application/x-www-form-urlencoded
username={user}&password={pwd}
```

Where {user} and {pwd} are the credentials provided by the Forescout administrator.

The response message includes a JSON web token, as shown in the following example.

```
HTTP/1.1 200 OK
Date: Wed, 15 Nov 2013 04:58:08 GMT
Content-Type: text/plain;charset=ISO-8859-1
Last-Modified: Wed, 15 Nov 2013 04:58:08 GMT
u7jV0PcoXLh5KKAJOQm4ek6jsqpuEaILKQ3MnOiHoYcyI1yZgLTxWkqqCH3Aemv4MlAjL9zvyjJ
.geOP9LCLtbUCVK6wApa16Ymqm1vHBDALtrtydTx2kzAhJGrFr3HFgrTbZTjz.d3KJFWYWhuAwU
0brDKkMTcRjmzbZovqamn6bgYhM
```

This token must be specified in the Authorization header of all subsequent request messages, as shown here and in other sample request messages in this guide.

```
GET https://{EM.IP}/api/hosts
Host: {EM.IP}
Authorization: u7jV0PcoXLh5KKAJOQm4ek6jsqpuEaILKQ3MnOiHoYcyI1yZgLTxWkqqCH3A
emv4MlAjL9zvyjJ.geOP9LCLtbUCVK6wApa16Ymqm1vHBDALtrtydTx2kzAhJGrFr3HFgrTbZTj
z.d3KJFWYWhuAwU0brDKkMTcRjmzbZovqamn6bgYhM
If-None-Match: "0cb3980dc881b0b4e47165546"
Accept: application/hal+json
```

By default, the token is valid for five minutes after it is issued. To receive a new token, the client must log in again to the web service.

The Forescout administrator can change this default validity period. See Change the Validity Period for Web Service Communication.

Using Entity Tags in Follow-up Requests

The CounterACT Web Service supports use of Entity Tags (ETags) as described in the HTTP/1.1 specification. Entity tags reduce bandwidth by eliminating transactions when previously reported data has not changed, as in the following typical sequence:

1. The client submits a request to the web service.
2. The response message from the web service include an ETag header with a hash value that identifies the transaction:
3. When clients repeat a query, they can submit the ETag hash value using the **`If-None-Match`** header in their request message.
4. The web service checks the hash and responds as follows:
   - If the queried data has changed in the Forescout platform since the last request, the web service sends a full response message with new data. This message includes a new ETag hash to identify the new transaction.

  - If the queried data has not changed in the Forescout platform since the last request, clients receive an HTTP 304 (Not Modified) return message. Use the existing ETag hash the next time you repeat the query.

Service Interaction Considerations

Consider the following when you plan your web service integration:

- The web service reports current values from the Forescout platform. Endpoints are not queried to update or resolve host properties in response to web service requests.
- The following factors may affect web service latency:
  - When web service requests are received by Enterprise Manager, endpoint data is retrieved on demand from managed Appliances.
  - The web service does not impose limits on bandwidth or response payload. Submit well-formed queries that limit the results and use ETag functionality to reduce bandwidth.
  - When internal timeouts are exceeded:

    When the response includes a list of endpoints, host properties, or policies, the web service may truncate the response.

    When the response provides details for a single endpoint, the web service returns an HTTP error code. No partial data is returned.

Web Service Transactions for Data Retrieval

This topic lists the transactions provided by the CounterACT Web Service that retrieve information from the Forescout platform and gives implementation details for each one.

- Retrieve an Index of Forescout Host Properties
- Retrieve a List of Active Endpoints
- Retrieve Specified Host Property Values for an Endpoint
- Retrieve Forescout Policy Information

Retrieve an Index of Forescout Host Properties

This transaction returns a list of all host properties defined in the Forescout platform. This list includes core properties provided by all Forescout platform implementations, and properties provided by the optional plugins and modules installed in the target Forescout environment.

Use this list to determine what information you want to retrieve from the Forescout platform.

When you retrieve information from the Forescout platform, the internal identity strings in this list are used to identify host properties in data returned by the web service. Use the host properties listing as a lookup table to translate these strings.

Request

The request message is addressed to the following URI:

https://{EM.IP}/api/hostfields

Transaction Data

This request does not contain a message body.

Basic Request Example

This example shows the basic request message to retrieve a list of host properties.

```
GET https://{EM.IP}/api/hostfields HTTP/1.1
Host: {EM.IP}
Authorization: u7jV0PcoXLh5KKAJOQm4ek6jsqpuEaILKQ3MnOiHoYcyI1yZgLTxWkqqCH3A
emv4MlAjL9zvyjJ.geOP9LCLtbUCVK6wApa16Ymqm1vHBDALtrtydTx2kzAhJGrFr3HFgrTbZTj
z.d3KJFWYWhuAwU0brDKkMTcRjmzbZovqamn6bgYhMIf-None-Match:
"0cb3980dc881b0b4e47165546"
Accept: application/hal+json
```

Response

This transaction returns a payload in JSON format, as shown in the following example.

```
HTTP/1.1 200 OK
Date: Wed, 15 Nov 2013 04:58:08 GMT
Content-Type: application/hal+json
Last-Modified: Wed, 15 Nov 2013 04:58:08 GMT
ETag: "0cb3980dc881b0b4e47165546"
{
  "hostFields": [
        {
      "label": "accept_id_change",
      "description": "",
      "name": "accept_id_change",
      "type": "boolean"
       },
       {
      "label": "Channel",
      "description": "",
      "name": "channel",
      "type": "string"
       },
       {
      "label": "Logged On",
      "description": "Indicates if a user is logged on to the host.",
      "name": "is_logged_in",
      "type": "boolean"
       },
       {
      "label": "Device is NAT",
      "description": "The host performs Network Address Translation,
potentially hiding other devices behind it",
      "name": "nat",
      "type": "boolean"
       }
   ]
}
```

The JSON text has the following structure:

- The header "hostFields" identifies the output as the result of a query to **`api/hosts`**.
- Host properties are presented as an array of unlabeled records.
- Each host property record contains the following field-value pairs:

label	The name of the property in the Console views.
description	The text description of the property that was entered when the property was defined.
name	The internal identity string of the property.
type	Indicates the type of data in the field. See About Data Types.

Retrieve a List of Active Endpoints

This transaction returns a list of endpoints currently listed by the Forescout platform as present in the network, and their associated IP and MAC addresses. The response also includes a URI for each endpoint that points to detailed information for that endpoint.

By default, the transaction returns a list of all active endpoints in the Forescout platform. You can return a filtered list based on the following criteria:

- Host property values of the endpoint
- Policies that select the endpoint

Request

The basic request message addresses the following URI:

`https://{EM.IP}/api/hosts`

This request returns a list of all active endpoints in the Forescout platform.

List Endpoints Selected by Forescout Policies

Use the optional **`?matchRuleId`** parameter to return endpoints that are selected by the specified policies or policy sub-rules:

`https://{EM.IP}/api/hosts?matchRuleId=`*{rule_ID},…,{rule_ID_n}*

Where *{rule_ID},…,{rule_ID_n}* are internal identifier strings of active policies defined in the Forescout platform, or individual rules of policies. See Retrieve Forescout Policy Information for details.

Only endpoints that are selected by ALL the policies or rules are returned.

Select Endpoints by Host Property Values

Use the following optional syntax to filter returned endpoints based on host property values:

`https://{EM.IP}/api/hosts?`*{prop}={val}*

Where

{prop} is the internal identity string of a host property in the Forescout platform.

{val} is a string representation of the property value to be matched.

Both these terms are case-sensitive.

Endpoints with property values that contain the specified {*val*} string are returned.

See Retrieve an Index of Forescout Host Properties for more information about the host property identifier string.

The following example returns endpoints that contain the string *Windows* in the Network Function host property:

`https://{EM.IP}/api/hosts?va_netfunc=Windows`

The following advanced matching patterns are supported:

- Filter based on several host properties – Concatenate property-value pairs with the ampersand & symbol as shown in the following example:

  `?{prop}={val}&{prop_2}={val_2}` … &{*prop_*n}={*val_*n}

  Where:

  {*prop*} and {prop_*n*} are internal identifiers of host properties.

  {*val*} and {*val_n*} are desired matching values for the corresponding host properties.

  Only endpoints that match ALL the conditions are returned.

- Match several values of a list property – Specify a comma-separated list of values for a list property. Only endpoints that have ALL specified values for the property are returned.

  `?{list_prop}={val},..,{val_n}`

  Where:

  {*list_prop*} is the internal identifier string of the list property.

  {*val*} and {*val_n*} are desired matching values for the corresponding sub-fields.

See Appendix 1: Forescout Property and Data Types for details.

- Specify sub-fields of a composite property – Specify a comma-separated list of field-value pairs to match sub-field values of a composite host property. The following syntax options are supported:
  - To return endpoints when a record of the composite property matches ALL the specified values, enclose all the field-value pairs in a single set of square brackets:

    `?{composite_prop}=[{field},{val},..,{field_n},{val_n}]`
  - To return endpoints when a record of the composite property contains ANY of the specified values, enclose each field-value pair in square brackets:

    `?{composite_prop}=[{field},{val}],..,[{field_n},{val_n}]`

  Where:

  {*composite_prop*} is the internal identifier string of the composite property.

  {*field*} and {*field_n*} are internal identifiers of sub-fields of the composite property.

{*val*} and {*val_n*} are desired matching values for the corresponding sub-fields.

See Appendix 1: Forescout Property and Data Types for more information about composite properties in the Forescout platform.

Combined Selection Syntax

You can combine selection criteria in a single request. The following example returns endpoints that match the conditions of the specified policy rule and have the value *True* for the Host is Online host property.

https://{EM.IP}/api/hosts?matchedRuleId=987987&online=true

Transaction Data

This request does not contain a message body.

Basic Request Example

The following example shows the basic request message to retrieve a list of endpoints with the Network Function host property value of *Windows*.

```
GET https://{EM.IP}/api/hosts?va_netfunc=Windows HTTP/1.1
Host: {EM.IP}
Authorization: u7jV0PcoXLh5KKAJOQm4ek6jsqpuEaILKQ3MnOiHoYcyI1yZgLTxWkqqCH3A
emv4MlAjL9zvyjJ.geOP9LCLtbUCVK6wApa16Ymqm1vHBDALtrtydTx2kzAhJGrFr3HFgrTbZTj
z.d3KJFWYWhuAwU0brDKkMTcRjmzbZovqamn6bgYhMIf-None-Match:
"0cb3980dc881b0b4e47165546"
Accept: application/hal+json
```

Response

This transaction returns a payload in JSON format, as shown in the following example.

```
HTTP/1.1 200 OK
Date: Wed, 15 Nov 2013 04:58:08 GMT
Content-Type: application/hal+json
Last-Modified: Wed, 15 Nov 2013 04:58:08 GMT
ETag: "0cb3980dc881b0b4e47165546"
{
  "hosts": [
    {
      "hostId": 170525084,
      "ip": "10.42.1.156",
      "mac": "0050568b007f",
      "_links": {
        "self": {
          "href": "https://10.42.1.45/api/hosts/170525084"
        }
      }
    },
    {
      "hostId": 170524982,
      "ip": "10.42.1.54",
      "mac": null,
      "_links": {
        "self": {
          "href": "https://10.42.1.45/api/hosts/170524982"
        }
```

```
        }
      }
   ]
}
```

The JSON text has the following structure:

- The header "hosts" identifies the output as the result of a query to **`api/hosts`**.
- Endpoints are presented as an array of records.
- Each endpoint record contains the following data objects:

  hostId The unique host key that identifies the endpoint in the Forescout platform database.

  ip The IP address of the specified endpoint. If no IP address exists, the string "undefined" is returned.

  mac The MAC address of the specified endpoint. If no MAC address exists, the string "undefined" is returned.

- Each endpoint record contains a **`_links`** object with the following link:

  href A self link to the web service node with a fully detailed listing of host property values for the endpoint. This URI specifies the persistent object ID of the node. For the equivalent standalone web service interaction, see Retrieve Specified Host Property Values for an Endpoint.

Retrieve Specified Host Property Values for an Endpoint

By default, this transaction returns all known property values for a specified endpoint. Typically, you use optional parameters to return specified host properties and their values.

If a specified host property is not found, the corresponding field:value pair is omitted from the returned JSON object structure.

Request

Use one of the following URIs to specify the target endpoint and properties to report:

- Based on endpoint IP address:

  `https://{EM.IP}/api/hosts/ip/`*{ipv4}*

  `https://{EM.IP}/api/hosts/ip/`*{ipv4}***`?fields=`***{prop},..,{prop_n}*

- Based on endpoint MAC address:

  `https://{EM.IP}/api/hosts/mac/`*{mac}*

  `https://{EM.IP}/api/hosts/mac/`*{mac}***`?fields=`***{prop},..,{prop_n}*

- Based on the web service object ID of the full endpoint listing; the object ID is specified in the self link that is returned for each endpoint when you Retrieve a List of Active Endpoints.

  `https://{EM.IP}/api/hosts/`*{obj_ID}*

  `https://{EM.IP}/api/hosts/`*{obj_ID}***`?fields=`***{prop},..,{prop_n}*

Where

{ipv4} is the unique IP address of the endpoint

{mac} is a unique MAC address of the endpoint. All standard formats are supported.

{obj_ID} is the object ID of the full endpoint record in the web service.

{prop},..,{prop_n} is a comma-separated list of host property identifier strings. See Retrieve an Index of Forescout Host Properties for details.

Transaction Data

This request does not contain a message body.

Basic Request Example

The following example shows the basic request message to retrieve host property information for the endpoint with IP address 180.10.10.24.

```
GET https://{EM.IP}/api/hosts/ip/180.10.10.24/?fields=va_funct HTTP/1.1
Host: {EM.IP}
Authorization: u7jV0PcoXLh5KKAJOQm4ek6jsqpuEaILKQ3MnOiHoYcyI1yZgLTxWkqqCH3A
emv4MlAjL9zvyjJ.geOP9LCLtbUCVK6wApa16Ymqm1vHBDALtrtydTx2kzAhJGrFr3HFgrTbZTj
z.d3KJFWYWhuAwU0brDKkMTcRjmzbZovqamn6bgYhMIf-None-Match:
"0cb3980dc881b0b4e47165546"
Accept: application/hal+json
```

Response

This transaction returns a payload in JSON format, with the following structure:

```
HTTP/1.1 200 OK
Date: Wed, 15 Nov 2013 04:58:08 GMT
Content-Type: application/hal+json
Last-Modified: Wed, 15 Nov 2013 04:58:08 GMT
ETag: "0cb3980dc881b0b4e47165546"
{
    "host": {
        "ip": "10.42.1.143",
        "mac": "0050568b0091",
        "fields": {. . .
                  },
    "links": {
            "rel": "self",
            "href": "https://10.42.1.45/api/hosts/170525071"
        }
}
```

The **`host`** object is an envelope for the entire returned text.

- The following header fields identify the endpoint for which values are reported:

  ip the unique IP address of the endpoint

  mac a unique MAC address of the endpoint

- The **`fields`** object contains an unordered list of host properties and their values. See Representation of Host Property Values for details.
- A **`links`** object at the end of the host object contains the following link:

href A self link to the web service node with a full listing of host property values for the endpoint. This URI specifies the persistent object ID of the node. For the equivalent standalone web service interaction, see Retrieve Specified Host Property Values for an Endpoint.

Representation of Host Property Values

Response messages format host properties and values as field:value pairs.

Single-value host property values are presented as in the following example, which shows the Host is Online property. The **`timestamp`** field indicates how recently the value was updated in the Forescout platform.

```
"online": {
          "timestamp": 1407233459,
          "value": "true"
          },
```

List properties return a list of **`value`** fields, each with its own **`timestamp`** field. The following example shows the Open Ports host property:

```
"openports": [
                {
                "timestamp": 1407211212,
                "value": "22/TCP"
                },
                {
                "timestamp": 1407211212,
                "value": "137/TCP"
                },
                {
                "timestamp": 1407211212,
                "value": "80/TCP"
                }
            ],
```

Composite properties are represented as an array of record objects. The following example shows the Applications Installed host property:

```
"comp_application": [
{
"timestamp": 1407199458,
"value": {
      "app_name": "Microsoft Visual C++ Redistributable - x86"
      "app_version": "9.0.30729.4148"
      "app_user": "All Users"
        }
}
{
"timestamp": 1407199458
"value": {
      "app_name": "Symantec Endpoint Protection"
```

```
      "app_version": "12.1.2015.2015"
      "app_user": "All Users"
         }
}
],
```

Error Codes

If the specified IP or MAC address is not found, an HTTP 404 Status Code is returned.

If a specified host property is not found, the corresponding field:value pair is omitted from the returned JSON object structure.

Retrieve Forescout Policy Information

This transaction returns a list of all policies defined in the Forescout platform and their sub-rules.

Request

The basic request message addresses the following URI:

`https://{EM.IP}/api/policies`

Transaction Data

This request does not contain a message body.

Request Example

The following example shows the basic request message to retrieve a list of policies.

```
GET https://{EM.IP}/api/policies HTTP/1.1
Host: {EM.IP}
Authorization: u7jV0PcoXLh5KKAJOQm4ek6jsqpuEaILKQ3MnOiHoYcyI1yZgLTxWkqqCH3A
emv4MlAjL9zvyjJ.geOP9LCLtbUCVK6wApa16Ymqm1vHBDALtrtydTx2kzAhJGrFr3HFgrTbZTj
z.d3KJFWYWhuAwU0brDKkMTcRjmzbZovqamn6bgYhMIf-None-Match:
"0cb3980dc881b0b4e47165546"
Accept: application/hal+json
```

Response

The response message for this transaction returns a payload in JSON format. The following example reports two policies – Asset Classification and Corporate/Guest Control.

```
HTTP/1.1 200 OK
Date: Wed, 15 Nov 2013 04:58:08 GMT
Content-Type: application/hal+json
Last-Modified: Wed, 15 Nov 2013 04:58:08 GMT
ETag: "0cb3980dc881b0b4e47165546"
{
  "policies": [
    {
      "policyId": 5423840174599379000,
      "rules": [
        {
          "ruleId": 7269473298755804000,
          "name": "NAT Devices",
          "description": "When a device is NAT, its other classifications
may be inaccurate. Therefore, we put the NAT detection first."
```

```
        },
        {
          "ruleId": 6476980249580685000,
          "name": "Windows",
          "description": ""
        },
        {
          "ruleId": -1585781564882007000,
          "name": "Linux/Unix",
          "description": ""
        },
        {
          "ruleId": -4195435042842094000,
          "name": "Macintosh",
          "description": ""
        },
        {
          "ruleId": 5674605010550634000,
          "name": "Network Devices",
          "description": ""
        },
        {
          "ruleId": -6018692461488323000,
          "name": "Unclassified",
          "description": ""
        }
      ],
      "name": "Asset Classification",
      "description": "Classify the hosts into the following groups for
easier management:\n\n1. NAT devices\n2. Windows systems\n3. Linux/Unix
systems\n4. Macintosh systems\n5. Network gear\n6. Unclassified\n\nRequired
parameters:\nNetwork Segment\n\nMethod:\nRe-classification is done on every
admission and at least once a day using passive and active fingerprinting."
    },
    {
      "policyId": 7680533389306876000,
      "rules": [
        {
          "ruleId": 8670050299727505000,
          "name": "Corporate Hosts",
          "description": ""
        },
        {
          "ruleId": -2228014476500593400,
          "name": "Signed In Hosts",
          "description": ""
        },
        {
          "ruleId": -938034771476122400,
          "name": "Guest Hosts",
          "description": ""
        }
      ],
      "name": "Corporate/Guest Control",
      "description": ""
    }
  ]
}
```

The JSON text has the following structure:

- The header "policies" identifies the output as the result of a query to **`api/policies`**.
- Each policy object contains the following header field:

  policyId The internal ID of the policy. This can be a negative integer.

- The **`rules`** object contains an ordered list of rules in the policy, beginning with the main rule of the policy. Each rule is an unlabeled record that contains **`ruleId`**, **`name`**, and **`description`** fields.
- Each policy object contains the following footer fields:

  name The name of the policy displayed in the Policy view of the Console.

  description The text description of the policy that is displayed in the Policy view of the Console.

Error Responses from Web Service

The following table lists standard error responses from the web service.

URL	Use case	Message	HTTP CODE
/api/login/	Login to managed Appliance.	Service not available from this server.	503
	Error while checking if web service is available.	Service unavailable. See log for more details.	500
	User is not authenticated.	Authentication failed for user {}	401
	Error while authenticating user.	Unable to authenticate user {}. See log for more details.	500
/api/policies	Error occurred while fetching policies	Unable to retrieve policy information. See log for more details.	500
/api/hostfields	Error occurred while fetching host fields	Unable to retrieve host property information. See log for more details.	500
/api/hosts/	Error occurred while fetching hosts	Unable to retrieve endpoint information. See log for more details.	500
/api/hosts?property=value	Host property not found.	The following host properties are invalid or not found: {} {} See log for more details.	500
/api/hosts?matchRuleId=32132	Invalid policy or rule ID	Policy ID {} invalid or not found.	500

URL	Use case	Message	HTTP CODE
/api/hosts?matchRuleId=987987&online=true&nbt_bost=2	Error occurred while building filter	The following host properties are invalid or not found: {} {} The following Policy ID values are invalid or not found: {} See log for more details.	500
/api/hosts?matchRuleId=	Empty policy or rule ID	Policy ID invalid, must not be NULL or empty. See log for more details.	500
/api/hosts/ip/	Invalid IP	Invalid IP address {}	400
	Error occurred while fetching host	Unable to retrieve endpoint data by IP address. See log for more details.	500
	Host not found	Unable to find endpoint with IP address {}	404
/api/hosts/mac/	Invalid MAC	Invalid MAC address {}	400
	Error occurred while fetching host	Unable to retrieve endpoint data by MAC address. See log for more details.	500
	Host not found	Unable to find endpoint with MAC address {}.	404
/api/hosts/id/	Invalid Host Id	Invalid Object ID {}	400
	Error occurred while fetching host	Unable to find endpoint data by Object ID. See log for more details.	500
	Host not found	Unable to find endpoint with Object ID {}.	404
Authentication Errors			
All URIs	Invalid token	Authentication token is invalid.	401
	Error while validating token.	Failed to validate authentication token.	500
	Private key generation failed.	Unable to generate authentication token.	500

Web Service Logs

The following log files are maintained on the Enterprise Manager or standalone Appliance. Use these logs to troubleshoot Web service connection or service issues reported by client users.

`/usr/local/forescout/log/plugin/webapi/webapi.log`

`/usr/local/tomcat/logs/webapi-tomcat.log`

`/usr/local/tomcat/logs/catalina.out`

Change the Validity Period for Web Service Communication

Requests to the web service are authorized using a JSON Web Token (JWT) issued by the Web Service. When a client initially logs in to the web service using the credentials defined in the configuration (see Configure Web API Plugin), the web service responds with an authentication token. This token must be included in request messages sent to the web service.

By default, the token is valid for five minutes after it is issued. To receive a new token, the client must log in again to the web service. You can change this default validity period.

To change the validity period of the JSON web token:

1. Log in to the Enterprise Manager.
2. Edit the local properties file for Web API at the following location:

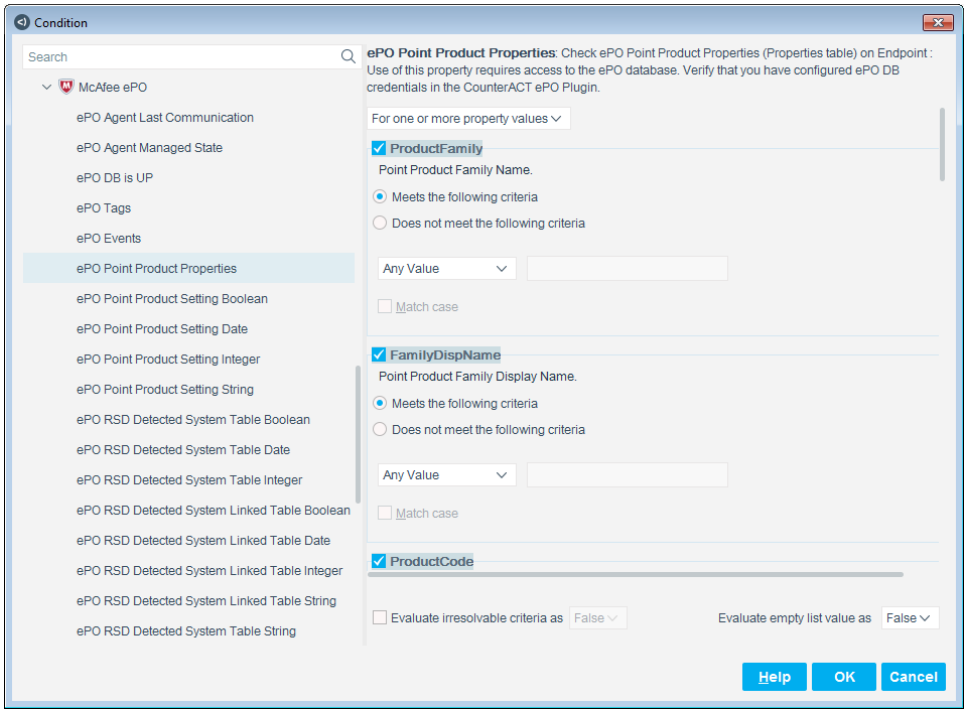
   `/usr/local/forescout/plugin/webapi/local.properties`
3. Add the property `token.expiration.time` and specify a value for this property, in seconds. The following example sets the token validity period to 12 minutes:

   ```
   token.expiration.time = 720
   ```
4. Save changes to the file.
5. Restart the Forescout platform service on the Enterprise Manager using the following command:

   ```
   fstool service restart
   ```

Appendix 1: Forescout Property and Data Types

Host properties store information that the Forescout platform discovers for each endpoint.



The Forescout platform provides information using the following types of properties:

Single-value properties contain one value. For example:

- A string property that contains the GUID of the endpoint

List properties contain a list of unique values. All items in the list are the same type of data. For example, a list property can contain:

- A list of all users in a directory group
- A list of previous host logins

Composite properties are like database tables, with several rows and columns. For example, a composite property can contain data from a help desk server listing recent service calls for a host. Retrieved columns would include:

- Date
- Contact
- Severity
- Status
- Description

About Data Types

Forescout platform host properties can contain various types of data. When you define a property, you specify the type of data that the property contains. This determines the matching options that the Forescout platform provides when you use the property in a policy condition. For example, the Forescout platform offers *Segment* and *IP range* options to match IP address values, and *Older Than* and *Before* options to match Date values.

The following table lists the data types your custom property can hold, and typical external data sources.

Forescout Property Data Type	Typical Expected SQL Data Types	Typical Expected LDAP Syntaxes
String	CHAR, VARCHAR, TEXT, LONGTEXT, TINYTEXT MEDIUMTEXT	Unicode String Distinguished Name
MAC Address	Standard notation with period (.) or colon (:) separators	
IP Address	Standard IPv4 notation : xxx.xxx.xxx.xxx	
Integer	INTEGER	Integer Octet String
Date	NUMBER type with Epoch Time in seconds, or DATE, DATETIME, TIMESTAMP	UTC Coded Timestamp
Boolean	TRUE, FALSE, BIT, BINARY	Boolean

When the Forescout platform reports the data type of a property, the following additional designations may be included:

- List – A property containing multiple values of the same data type.
- Composite – These properties resemble a database record and contain multiple fields. Each field can contain a different type of data.
- Change – These properties track changes in the value of another property.