

# Cloud Analytics Engine Compute Agent API Reference

Release

15.1X53



Modified: 2016-07-19

Juniper Networks, Inc.  
1133 Innovation Way  
Sunnyvale, California 94089  
USA  
408-745-2000  
[www.juniper.net](http://www.juniper.net)

Juniper Networks, Junos, Steel-Belted Radius, NetScreen, and ScreenOS are registered trademarks of Juniper Networks, Inc. in the United States and other countries. The Juniper Networks Logo, the Junos logo, and JunosE are trademarks of Juniper Networks, Inc. All other trademarks, service marks, registered trademarks, or registered service marks are the property of their respective owners.

Juniper Networks assumes no responsibility for any inaccuracies in this document. Juniper Networks reserves the right to change, modify, transfer, or otherwise revise this publication without notice.

*Cloud Analytics Engine Compute Agent API Reference*  
15.1X53  
Copyright © 2016, Juniper Networks, Inc.  
All rights reserved.

The information in this document is current as of the date on the title page.

#### YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. Junos OS has no known time-related limitations through the year 2038. However, the NTP application is known to have some difficulty in the year 2036.

#### END USER LICENSE AGREEMENT

The Juniper Networks product that is the subject of this technical documentation consists of (or is intended for use with) Juniper Networks software. Use of such software is subject to the terms and conditions of the End User License Agreement ("EULA") posted at <http://www.juniper.net/support/eula.html>. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

# Table of Contents

	About the Documentation . . . . .	vii
	Documentation and Release Notes . . . . .	vii
	Supported Platforms . . . . .	vii
	Using the Examples in This Manual . . . . .	vii
	Merging a Full Example . . . . .	viii
	Merging a Snippet . . . . .	viii
	Documentation Conventions . . . . .	ix
	Documentation Feedback . . . . .	xi
	Requesting Technical Support . . . . .	xi
	Self-Help Online Tools and Resources . . . . .	xi
	Opening a Case with JTAC . . . . .	xii
<b>Part 1</b>	<b>Using the Compute Agent API</b>	
<b>Chapter 1</b>	<b>Introducing the Compute Agent API . . . . .</b>	<b>3</b>
	Compute Agent API Resources Overview . . . . .	3
	Compute Agent API Quick Reference . . . . .	5
<b>Chapter 2</b>	<b>Managing CA Flow Path Analytics Data . . . . .</b>	<b>7</b>
	Listing Available Application Flows . . . . .	7
	GET list/flow Resource . . . . .	7
	List Flows Compute Agent API Example . . . . .	8
	List Flows Example with Non-Overlay Flows . . . . .	9
	List Flows Example For Overlay (Tunnel) Flows . . . . .	9
	Starting and Stopping a Flow Data Collection . . . . .	10
	GET start/flow Resource . . . . .	10
	GET stop/flow Resource . . . . .	12
	Compute Agent Flow Data Resource Usage Example . . . . .	13
	Requesting Collected Flow Data . . . . .	14
	GET data/flow Resource . . . . .	14
	Flow Path Response Data Object . . . . .	15
	Flow Data Compute Agent API Example . . . . .	18
	Bandwidth Flow Data Compute Agent API Example . . . . .	23
	Bandwidth Example with a TCP Flow . . . . .	23
	Bandwidth Example with a UDP Flow . . . . .	27
	Mirror Flow Data Compute Agent API Example . . . . .	32
	Retrieving Active Flows List . . . . .	37
	GET active/flow Resource . . . . .	37
	Active Flows Compute Agent API Example . . . . .	37

Starting and Stopping a Tunnel Flow Data Collection . . . . .	38
GET start/tunnel Resource . . . . .	38
GET stop/tunnel Resource . . . . .	40
Compute Agent Tunnel Data Resource Usage Example . . . . .	41
Requesting Collected Tunnel Flow Data . . . . .	42
GET data/tunnel Resource . . . . .	42
Flow Path Response Data Object . . . . .	43
Tunnel Data Compute Agent API Example . . . . .	46
Retrieving Active Tunnel Flows List . . . . .	51
GET active/tunnel Resource . . . . .	51
Active Tunnels Compute Agent API Example . . . . .	52

# List of Tables

	<b>About the Documentation</b> . . . . .	<b>vii</b>
	Table 1: Notice Icons . . . . .	ix
	Table 2: Text and Syntax Conventions . . . . .	ix
<b>Part 1</b>	<b>Using the Compute Agent API</b>	
<b>Chapter 1</b>	<b>Introducing the Compute Agent API</b> . . . . .	<b>3</b>
	Table 3: Compute Agent API Resources Quick-Reference . . . . .	5
<b>Chapter 2</b>	<b>Managing CA Flow Path Analytics Data</b> . . . . .	<b>7</b>
	Table 4: list/flow Resource JSON Response Elements . . . . .	8
	Table 5: GET start/flow Resource Request Parameters . . . . .	10
	Table 6: Flow Path Analytics Data JSON Response Object . . . . .	15
	Table 7: Flow Path Analytics JSON Response Data: Path Object Elements . . . . .	15
	Table 8: Flow Path Analytics JSON Response Data: Timestamp Object Elements . . . . .	16
	Table 9: Flow Path Analytics JSON Response Data: Ingress Logical Interface Attribute Object Elements . . . . .	16
	Table 10: Flow Path Analytics JSON Response Data: Egress Logical Interface Attribute Object Elements . . . . .	16
	Table 11: Flow Path Analytics JSON Response Data: Layer 3 ECMP Object Elements . . . . .	17
	Table 12: Flow Path Analytics JSON Response Data: Layer 2 ECMP List Object Elements . . . . .	17
	Table 13: Flow Path Analytics JSON Response Data: Logical Interface Object Elements . . . . .	17
	Table 14: Flow Path Analytics JSON Response Data: Interface Packet Statistics Object Elements . . . . .	17
	Table 15: GET start/tunnel Resource Parameters . . . . .	39
	Table 16: Flow Path Analytics Data JSON Response Object . . . . .	43
	Table 17: Flow Path Analytics JSON Response Data: Path Object Elements . . . . .	43
	Table 18: Flow Path Analytics JSON Response Data: Timestamp Object Elements . . . . .	44
	Table 19: Flow Path Analytics JSON Response Data: Ingress Logical Interface Attribute Object Elements . . . . .	44
	Table 20: Flow Path Analytics JSON Response Data: Egress Logical Interface Attribute Object Elements . . . . .	44
	Table 21: Flow Path Analytics JSON Response Data: Layer 3 ECMP Object Elements . . . . .	45
	Table 22: Flow Path Analytics JSON Response Data: Layer 2 ECMP List Object Elements . . . . .	45

Table 23: Flow Path Analytics JSON Response Data: Logical Interface Object Elements . . . . .	45
Table 24: Flow Path Analytics JSON Response Data: Interface Packet Statistics Object Elements . . . . .	45
Table 25: active/tunnel Resource JSON Response Tuple Elements . . . . .	52

# About the Documentation

- Documentation and Release Notes on page vii
- Supported Platforms on page vii
- Using the Examples in This Manual on page vii
- Documentation Conventions on page ix
- Documentation Feedback on page xi
- Requesting Technical Support on page xi

## Documentation and Release Notes

---

To obtain the most current version of all Juniper Networks® technical documentation, see the product documentation page on the Juniper Networks website at <http://www.juniper.net/techpubs/>.

If the information in the latest release notes differs from the information in the documentation, follow the product Release Notes.

Juniper Networks Books publishes books by Juniper Networks engineers and subject matter experts. These books go beyond the technical documentation to explore the nuances of network architecture, deployment, and administration. The current list can be viewed at <http://www.juniper.net/books>.

## Supported Platforms

---

For the features described in this document, the following platforms are supported:

- [QFX Series](#)

## Using the Examples in This Manual

---

If you want to use the examples in this manual, you can use the **load merge** or the **load merge relative** command. These commands cause the software to merge the incoming configuration into the current candidate configuration. The example does not become active until you commit the candidate configuration.

If the example configuration contains the top level of the hierarchy (or multiple hierarchies), the example is a *full example*. In this case, use the **load merge** command.

If the example configuration does not start at the top level of the hierarchy, the example is a *snippet*. In this case, use the **load merge relative** command. These procedures are described in the following sections.

## Merging a Full Example

To merge a full example, follow these steps:

1. From the HTML or PDF version of the manual, copy a configuration example into a text file, save the file with a name, and copy the file to a directory on your routing platform.

For example, copy the following configuration to a file and name the file **ex-script.conf**. Copy the **ex-script.conf** file to the **/var/tmp** directory on your routing platform.

```
system {
  scripts {
    commit {
      file ex-script.xml;
    }
  }
}
interfaces {
  fxp0 {
    disable;
    unit 0 {
      family inet {
        address 10.0.0.1/24;
      }
    }
  }
}
```

2. Merge the contents of the file into your routing platform configuration by issuing the **load merge** configuration mode command:

```
[edit]
user@host# load merge /var/tmp/ex-script.conf
load complete
```

## Merging a Snippet

To merge a snippet, follow these steps:

1. From the HTML or PDF version of the manual, copy a configuration snippet into a text file, save the file with a name, and copy the file to a directory on your routing platform.

For example, copy the following snippet to a file and name the file **ex-script-snippet.conf**. Copy the **ex-script-snippet.conf** file to the **/var/tmp** directory on your routing platform.

```
commit {
  file ex-script-snippet.xml; }
```

2. Move to the hierarchy level that is relevant for this snippet by issuing the following configuration mode command:

```
[edit]
user@host# edit system scripts
[edit system scripts]
```

3. Merge the contents of the file into your routing platform configuration by issuing the **load merge relative** configuration mode command:

```
[edit system scripts]
user@host# load merge relative /var/tmp/ex-script-snippet.conf
load complete
```

For more information about the **load** command, see the *CLI User Guide*.

## Documentation Conventions

Table 1 on page ix defines notice icons used in this guide.

Table 1: Notice Icons







Icon	Meaning	Description
	Informational note	Indicates important features or instructions.
	Caution	Indicates a situation that might result in loss of data or hardware damage.
	Warning	Alerts you to the risk of personal injury or death.
	Laser warning	Alerts you to the risk of personal injury from a laser.
	Tip	Indicates helpful information.
	Best practice	Alerts you to a recommended use or implementation.

Table 2 on page ix defines the text and syntax conventions used in this guide.

Table 2: Text and Syntax Conventions

Convention	Description	Examples
<b>Bold text like this</b>	Represents text that you type.	To enter configuration mode, type the <b>configure</b> command:  user@host> <b>configure</b>

Table 2: Text and Syntax Conventions (*continued*)

Convention	Description	Examples
Fixed-width text like this	Represents output that appears on the terminal screen.	user@host> <b>show chassis alarms</b>  No alarms currently active
<i>Italic text like this</i>	<ul style="list-style-type: none"><li>Introduces or emphasizes important new terms.</li><li>Identifies guide names.</li><li>Identifies RFC and Internet draft titles.</li></ul>	<ul style="list-style-type: none"><li>A policy <i>term</i> is a named structure that defines match conditions and actions.</li><li><i>Junos OS CLI User Guide</i></li><li>RFC 1997, <i>BGP Communities Attribute</i></li></ul>
<i>Italic text like this</i>	Represents variables (options for which you substitute a value) in commands or configuration statements.	Configure the machine's domain name:  [edit] root@# <b>set system domain-name</b> <i>domain-name</i>
Text like this	Represents names of configuration statements, commands, files, and directories; configuration hierarchy levels; or labels on routing platform components.	<ul style="list-style-type: none"><li>To configure a stub area, include the <b>stub</b> statement at the [edit protocols ospf area area-id] hierarchy level.</li><li>The console port is labeled <b>CONSOLE</b>.</li></ul>
< > (angle brackets)	Encloses optional keywords or variables.	<b>stub</b> <default-metric <i>metric</i> >;
(pipe symbol)	Indicates a choice between the mutually exclusive keywords or variables on either side of the symbol. The set of choices is often enclosed in parentheses for clarity.	<b>broadcast</b>   <b>multicast</b>  ( <i>string1</i>   <i>string2</i>   <i>string3</i> )
# (pound sign)	Indicates a comment specified on the same line as the configuration statement to which it applies.	<b>rsvp { # Required for dynamic MPLS only</b>
[ ] (square brackets)	Encloses a variable for which you can substitute one or more values.	<b>community name members</b> [ <i>community-ids</i> ]
Indentation and braces ( { } )	Identifies a level in the configuration hierarchy.	[edit] routing-options { static { route default { nexthop <i>address</i> ; retain; } } }
;(semicolon)	Identifies a leaf statement at a configuration hierarchy level.	
<b>GUI Conventions</b>		
<b>Bold text like this</b>	Represents graphical user interface (GUI) items you click or select.	<ul style="list-style-type: none"><li>In the Logical Interfaces box, select <b>All Interfaces</b>.</li><li>To cancel the configuration, click <b>Cancel</b>.</li></ul>

Table 2: Text and Syntax Conventions (*continued*)

Convention	Description	Examples
> (bold right angle bracket)	Separates levels in a hierarchy of menu selections.	In the configuration editor hierarchy, select <b>Protocols&gt;Ospf</b> .

## Documentation Feedback

We encourage you to provide feedback, comments, and suggestions so that we can improve the documentation. You can provide feedback by using either of the following methods:

- Online feedback rating system—On any page of the Juniper Networks TechLibrary site at <http://www.juniper.net/techpubs/index.html>, simply click the stars to rate the content, and use the pop-up form to provide us with information about your experience. Alternately, you can use the online feedback form at <http://www.juniper.net/techpubs/feedback/>.
- E-mail—Send your comments to [techpubs-comments@juniper.net](mailto:techpubs-comments@juniper.net). Include the document or topic name, URL or page number, and software version (if applicable).

## Requesting Technical Support

Technical product support is available through the Juniper Networks Technical Assistance Center (JTAC). If you are a customer with an active J-Care or Partner Support Service support contract, or are covered under warranty, and need post-sales technical support, you can access our tools and resources online or open a case with JTAC.

- JTAC policies—For a complete understanding of our JTAC procedures and policies, review the *JTAC User Guide* located at <http://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf>.
- Product warranties—For product warranty information, visit <http://www.juniper.net/support/warranty/>.
- JTAC hours of operation—The JTAC centers have resources available 24 hours a day, 7 days a week, 365 days a year.

## Self-Help Online Tools and Resources

For quick and easy problem resolution, Juniper Networks has designed an online self-service portal called the Customer Support Center (CSC) that provides you with the following features:

- Find CSC offerings: <http://www.juniper.net/customers/support/>
- Search for known bugs: <http://www2.juniper.net/kb/>
- Find product documentation: <http://www.juniper.net/techpubs/>
- Find solutions and answer questions using our Knowledge Base: <http://kb.juniper.net/>

- Download the latest versions of software and review release notes:  
<http://www.juniper.net/customers/csc/software/>
- Search technical bulletins for relevant hardware and software notifications:  
<http://kb.juniper.net/InfoCenter/>
- Join and participate in the Juniper Networks Community Forum:  
<http://www.juniper.net/company/communities/>
- Open a case online in the CSC Case Management tool: <http://www.juniper.net/cm/>

To verify service entitlement by product serial number, use our Serial Number Entitlement (SNE) Tool: <https://tools.juniper.net/SerialNumberEntitlementSearch/>

## Opening a Case with JTAC

You can open a case with JTAC on the Web or by telephone.

- Use the Case Management tool in the CSC at <http://www.juniper.net/cm/>.
- Call 1-888-314-JTAC (1-888-314-5822 toll-free in the USA, Canada, and Mexico).

For international or direct-dial options in countries without toll-free numbers, see <http://www.juniper.net/support/requesting-support.html>.

## PART 1

# Using the Compute Agent API

- [Introducing the Compute Agent API on page 3](#)
- [Managing CA Flow Path Analytics Data on page 7](#)



## CHAPTER 1

# Introducing the Compute Agent API

- [Compute Agent API Resources Overview on page 3](#)
- [Compute Agent API Quick Reference on page 5](#)

### Compute Agent API Resources Overview

---

The Compute Agent (CA) API is a Web API that enables access to network application flow path emulation and analytics data collection functions of the CA component of Juniper Networks Cloud Analytics Engine. CA enables collection of application-specific network data by generating probes that emulate application traffic, which triggers per-hop traces of network activity on the devices along the application's flow path. Subsequently, CA collates and stores resulting end-to-end flow path statistics from the devices, including data from encapsulated flows, enabling underlay-overlay correlation.

The CA API has resources with which you can:

- Get information about the applications running in the network and learn which application flow paths are available to trace.
- Request to start or stop application flow path data collection, or change flow data collection parameters for an active flow.
- Retrieve collected flow path analytics data.
- Get information about active flow data collections.
- Request to start or stop encapsulated (tunnel) flow path data collection, or change data collection parameters for an active tunnel flow.
- Retrieve collected tunnel flow path analytics data.
- Get information about active tunnel flow data collections.

See [“Compute Agent API Quick Reference” on page 5](#) for the list of CA API resources by request name, which includes quick reference links to usage details.

In general, in the context of the functions of the CA API resources:

- A *flow* is represented by an N-tuple identifying the end-to-end devices for which data is being collected, specifically, the following 5-tuple:  
*{ source IP address, destination IP address, transport protocol, source port, destination port }*
- A *tunnel* represents an encapsulation type for a flow. CA supports the VXLAN encapsulation type. A tunnel is identified by all or part of the following 6-tuple:  
*{ VTEP source IP address, VTEP destination IP address, VXLAN network identifier, source port, source IP address of the application, destination IP address of the application }*
- Flow or tunnel CA API resource *request parameters* include the flow or tunnel identifier. Other required or optional parameters can be specified when starting or updating a flow data collection, such as setting sampling frequencies, mirroring, and bandwidth measurement options.
- CA API resources support only transport protocols TCP and UDP—set and received as 'tcp' (or 'TCP') and 'udp' (or 'UDP') in request parameters and response data. Note, however, that response data might alternatively specify transport protocol values as the registered Internet Assigned Numbers Authority (IANA) protocol number (TCP=6, UDP=17). See the response data object descriptions to interpret the correct protocol value format when using API requests that include a transport protocol value.
- Except for **list/flow**, all CA APIs also accept a non-standard HTTP header field **Originator**.
  - Specifying this field when starting a flow or tunnel flow tags it with an identifier string representing the authorized owner, and controls resource access when subsequent requests are made related to that flow.
  - If a request to access information about an active flow does not have the matching **Originator** field, the request is rejected with the error code **E\_UNAUTHORIZED** (see list of all return codes next).

Numeric values of CA API request return codes are as follows:

- **EOK** = 0
- **EAGAIN** = 1
- **EFAIL** = 2
- **E\_UNAUTHORIZED** = 3
- **E\_AoF** = 4

**Related  
Documentation**

- *Cloud Analytics Engine Compute Agent API Reference*
- *Installing and Configuring Cloud Analytics Engine Compute Agent*
- *Cloud Analytics Engine Overview*

## Compute Agent API Quick Reference

Table 3 on page 5 lists the resource requests available in the CA Web API. Note that the GET method is used for both control and data requests.

**Table 3: Compute Agent API Resources Quick-Reference**

Resource	Description
"GET list/flow Resource" on page 7	Return a list of application flows detected on the CA host that might be traced.
"GET start/flow Resource" on page 10	Start generating probes and collecting analytics information about a flow from the network, or modify existing data collection parameters for an active flow.
"GET stop/flow Resource" on page 12	Stop an active flow collector.
"GET data/flow Resource" on page 14	Return network analytics data for an active flow.
"GET active/flow Resource" on page 37	Return the list of active flows for the requesting client or <b>Originator</b> field in the request header.
"GET start/tunnel Resource" on page 38	Start collecting network analytics information about a tunnel flow, or modify existing data collection parameters for an active tunnel flow.
"GET stop/tunnel Resource" on page 40	Stop an active tunnel collector.
"GET data/tunnel Resource" on page 42	Return network analytics data for an active tunnel flow.
"GET active/tunnel Resource" on page 51	Return the list of active tunnel flows for the requesting client or <b>Originator</b> field in the request header.

- Related Documentation**
- *Cloud Analytics Engine Compute Agent API Reference*
  - *Cloud Analytics Engine Overview*



## CHAPTER 2

# Managing CA Flow Path Analytics Data

- [Listing Available Application Flows on page 7](#)
- [Starting and Stopping a Flow Data Collection on page 10](#)
- [Requesting Collected Flow Data on page 14](#)
- [Retrieving Active Flows List on page 37](#)
- [Starting and Stopping a Tunnel Flow Data Collection on page 38](#)
- [Requesting Collected Tunnel Flow Data on page 42](#)
- [Retrieving Active Tunnel Flows List on page 51](#)

### **Listing Available Application Flows**

---

Use the Cloud Analytics Engine CA API list flows resource to retrieve information about application flows CA has discovered are currently active and can be emulated to collect application flow statistics.

- [GET list/flow Resource on page 7](#)
- [List Flows Compute Agent API Example on page 8](#)

### **GET list/flow Resource**

Use the GET list/flow CA API to retrieve a list of all active application flows that CA has “sniffed” (detected) that are running on CA's host. CA clients can then request to start tracing an application flow of interest using CA probes to collect flow path statistics (see [“GET start/flow Resource” on page 10.](#))

## Request Parameters:

- **active:** (Boolean, Optional) If “True”, return flows active only for the last 60 seconds. Default is “False”, return all available flows.

## Return Codes:

- **EFAIL**—Failed to fetch the data.

## Response Data:

- **FLOWS:** List of flow identifier objects for all flows that were sniffed in the CA host, in JSON format, described in [Table 4 on page 8](#).

See “[List Flows Compute Agent API Example](#)” on [page 8](#) for a sample request and response data.

Table 4: list/flow Resource JSON Response Elements

Element Name	Description
SIP	Source IP address
DIP	Destination IP address
PROTO	Protocol (“tcp” or “udp”)
PORTS	<p>List of flow port information tuples, including the timestamp when CA recorded detection of the application flow.</p> <ul style="list-style-type: none"> <li>• For non-overlay flow, this tuple consists of the following 3 values, in the order shown: <i>{timestamp, destination port, source port}</i>.</li> <li>• For overlay (tunnel) flow, this tuple consists of the following 11 values, in the order shown: <i>{timestamp, destination port, source port, VXLAN identifier, VM source MAC address, VM destination MAC address, VM source IP address, VM destination IP address, VM application protocol, VM application source port, VM application destination port}</i>.</li> </ul>

## List Flows Compute Agent API Example

These examples show the JSON-format response data returned from CA API requests to list all application flows detected by the target CA server. Note that the response data differs for identifying either non-overlay or overlay (tunnel) flows, depending on what flows are available at the time of the request. These examples show results filtered to only return flows active within the past 60 seconds ( the **active** parameter is set to “True”). See “[GET list/flow Resource](#)” on [page 7](#) for details on the supported request parameters and response data elements.

- [List Flows Example with Non-Overlay Flows on page 9](#)
- [List Flows Example For Overlay \(Tunnel\) Flows on page 9](#)

### List Flows Example with Non-Overlay Flows

Retrieve a list of flows active within the past 60 seconds (**GET list/flow** request):

**http://10.94.201.11:8080/list/flow?active=True**

Response data:

```
{
  'FLOWS': [
    {
      'SIP': '10.1.1.1',
      'DIP': '10.1.7.1',
      'PORTS': [
        [
          1442515268.272547,
          22,
          46047
        ],
        [
          1442515282.1547129,
          9000,
          59011
        ],
        [
          1442515282.25717,
          9000,
          59010
        ],
        [
          1442515282.3083589,
          9000,
          59012
        ]
      ],
      'PROTO': 'TCP'
    }
  ]
}
```

### List Flows Example For Overlay (Tunnel) Flows

Retrieve a list of flows active within the past 60 seconds, including overlay (tunnel) flows (**GET list/flow** request):

**http://rod-ix-pc-04:8080/list/flow?active=True**

Response data:

```
{
  'FLOWS': [
    {
      'SIP': '10.10.11.11',
      'DIP': '192.168.5.11',
      'PORTS': [
```

```

    [
      1442623012.2390749,
      4789,
      53812,
      200,
      '52: 54: 0: b8: 9b: 2e',
      '52: 54: 0: 6e: cc: dc',
      '10.0.0.2',
      '10.0.0.3',
      'UDP',
      41504,
      9000
    ]
  ],
  'PROTO': 'UDP'
}
]
}

```

- Related Documentation**
- [Compute Agent API Resources Overview on page 3](#)
  - [Cloud Analytics Engine Feature Guide for the QFX Series](#)

## Starting and Stopping a Flow Data Collection

Use the Cloud Analytics Engine CA API flow data resource to start or stop generating application flow path analytics data.

- [GET start/flow Resource on page 10](#)
- [GET stop/flow Resource on page 12](#)
- [Compute Agent Flow Data Resource Usage Example on page 13](#)

### GET start/flow Resource

Use the **GET start/flow** CA API to instruct CA to generate probes and start collecting information from the network about a flow. You can also use this API to modify the specified data collection parameters for a flow that is already active. This is an asynchronous API.

Request Parameters:

- [Table 5 on page 10](#) describes the **GET start/flow** request parameters, which include the 5-tuple identifying the application flow, and optional parameters specifying what to collect from the available flow data options.

**Table 5: GET start/flow Resource Request Parameters**

Flow Resource Parameters	Description
dst	(String) Destination IPv4 address for the flow of interest.  Required parameter.

Table 5: GET start/flow Resource Request Parameters (*continued*)

Flow Resource Parameters	Description
src	(String) Source IPv4 address of the flow of interest.  Required parameter.
proto	(String) Protocol type: "tcp" or "udp".  Optional parameter. Default is "udp".
sport	(Numeric) Source port corresponding to the protocol type for the flow.  Optional parameter. Default value is 33434.
dport	(Numeric) Destination port corresponding to the protocol type for the flow.  Optional parameter. Default value is 33434.
sfreq	(Numeric) Sampling frequency for collecting network data (in seconds).  Optional parameter. Default value is 1 second.
nhops	(Numeric) Number of network hops from which to collect analytics data. Data is gathered across all hops until the destination ( $\leq 255$ ) if this parameter is not specified.  Optional parameter. Default value is 255.
ntry	(Numeric) Number of tries to attempt in case of a timeout on a given hop. If this parameter is not specified, only a single attempt is made to gather data for the hop.  Optional parameter. Default value is 1 (no retries).
timeout	(Numeric) Timeout (in ms) to wait for the device at a given hop to respond. If the device does not respond within the timeout, any delayed response is ignored by CA.  Optional parameter. Default value is 30000 ms (30 seconds).
chksum	(Boolean) Enable ("true") or disable ("false") checksum computation.  Optional parameter. Default value is "true" or enabled.
bw_ingress	(Boolean) If specified as "true", measure flow bandwidth at each hop ingress.  Optional parameter. Default value is "false", or do not include bandwidth measurements.
mirror_type	(String) Install mirroring of the indicated type for this flow. Supported mirror type values: "ERSPAN".  Optional parameter. Default if not specified is no mirror installation.
mirror_dir	(String) Mirror direction. Supported values: "ingress".  Optional parameter with no default value. If <b>mirror_type</b> is specified, this parameter must also be specified.

Table 5: GET start/flow Resource Request Parameters (*continued*)

Flow Resource Parameters	Description
mirror_analyzer	(String) Mirror analyzer IP address.  Optional parameter with no default value. If <b>mirror_type</b> is specified, this parameter must also be specified.

## Return Codes:

- **EOK**—Request was serviced.
- **EAGAIN**—Exceeding pending updates for the same flow. This might happen when the same flow is modified, or there is a previous stop request pending on this flow.
- **EFAIL**—Invalid parameters were provided.
- **E\_UNAUTHORIZED**—Flow was initiated by a different client and cannot be started or modified by the originator of this request.

## Response Data:

- None

See “[Compute Agent Flow Data Resource Usage Example](#)” on page 13 for a simple example of a request to start a flow with required parameters specifying the 5-tuple that identifies the flow, the same parameters used subsequently to get collected data from that flow, and the same parameters used again to stop the flow.

You can also see these **GET data/flow** examples that show **GET start/flow** requests to start a flow with different data collection options, and the response data that is returned:

- [Flow Data Compute Agent API Example on page 18](#)
- [Bandwidth Flow Data Compute Agent API Example on page 23](#)
- [Mirror Flow Data Compute Agent API Example on page 32](#)

## GET stop/flow Resource

Use the **GET stop/flow** CA API to instruct CA to stop collecting information from the network about a flow. This is an asynchronous API.

## Request Parameters:

- The **GET stop/flow** request parameters are the following 5-tuple, which identifies the flow that is active between two endpoints (source and destination):
  - src: (String) Source IPv4 address.
  - dst: (String) Destination IPv4 address.
  - proto: (String, Optional) Protocol type: “tcp” or “udp”. Default is “udp”.

- **sport:** (Numeric, Optional) Source port corresponding to the protocol type for the flow. Default value is 33434.
- **dport:** (Numeric, Optional) Destination port corresponding to the protocol type for the flow. Default value is 33434.

Return Codes:

- **EOK**—A stop signal has been initiated to the flow thread.
- **EFAIL**—Invalid parameters were provided.
- **E\_UNAUTHORIZED**—Flow was initiated by a different client and cannot be stopped by the originator of this request.
- **E\_AoF**—No such flow.

Response Data:

- None

See “[Compute Agent Flow Data Resource Usage Example](#)” on page 13 for a simple example of a request to stop a flow data collection, using the same parameters as the **GET start/flow** request that started the flow.

## Compute Agent Flow Data Resource Usage Example

The following is a simple example of using the flow data resource with the required parameters that specify the flow identifier 5-tuple (**src**, **dst**, **sport**, **dport**, **proto**). This example shows the sequence of CA API web requests to the CA server at IP address 10.94.201.11 and port 8080 to start a flow data collection, retrieve collected data from that flow, and stop the flow data collection. All requests use the GET method.

- Start a flow data collection (**GET start/flow**):  
`http://10.94.201.11:8080/start/flow?dport=9000&src=10.1.1.1&dst=10.1.7.1&sport=4567&proto=tcp`
- Request collected data for the flow (**GET data/flow**):  
`http://10.94.201.11:8080/data/flow?dport=9000&src=10.1.1.1&dst=10.1.7.1&sport=4567&proto=tcp`
- Stop flow data collection (**GET stop/flow**):  
`http://10.94.201.11:8080/stop/flow?dport=9000&src=10.1.1.1&dst=10.1.7.1&sport=4567&proto=tcp`

### Related Documentation

- [Compute Agent API Resources Overview on page 3](#)
- *Cloud Analytics Engine Feature Guide for the QFX Series*

## Requesting Collected Flow Data

---

Use the Cloud Analytics Engine CA API flow data resource to retrieve application flow path analytics data.

- [GET data/flow Resource on page 14](#)
- [Flow Path Response Data Object on page 15](#)
- [Flow Data Compute Agent API Example on page 18](#)
- [Bandwidth Flow Data Compute Agent API Example on page 23](#)
- [Mirror Flow Data Compute Agent API Example on page 32](#)

### GET data/flow Resource

Use the **GET data/flow** CA API to request analytics data for a flow you have started. This is a synchronous API.

The available data depends on the parameters used to start the flow, described in [“GET start/flow Resource” on page 10](#).

Request Parameters:

- The 5-tuple identifying the flow from which to retrieve available analytics data:
  - src: (String) Source IPv4 address.
  - dst: (String) Destination IPv4 address.
  - proto: (String, Optional) Protocol type: “tcp” or “udp”. Default is “udp”.
  - sport: (Numeric, Optional) Source port corresponding to the protocol type for the flow. Default value is 33434.
  - dport: (Numeric, Optional) Destination port corresponding to the protocol type for the flow. Default value is 33434.

Return Code:

- None—CA returns the requested flow data.
- **EFAIL**—No data is present for the specified flow.
- **E\_UNAUTHORIZED**—Flow was initiated by a different client and cannot be retrieved by the originator of this request.
- **E\_AoF**—No such flow.

Response Data:

- The response data object containing statistics from the devices in the specified flow path, in JSON format. See [“Flow Path Response Data Object” on page 15](#).

See the following for examples of using the data/flow resource with different flow data collection parameters:

- [Flow Data Compute Agent API Example on page 18](#)
- [Bandwidth Flow Data Compute Agent API Example on page 23](#)
- [Mirror Flow Data Compute Agent API Example on page 32](#)

## Flow Path Response Data Object

[Table 6 on page 15](#) lists the elements in the response data object (JSON format) for requests to retrieve CA flow path analytics data. See [“GET data/flow Resource” on page 14](#) and [“GET data/tunnel Resource” on page 42](#), which describe the CA API resource requests that return this data.

Note that the Cloud Analytics Engine DLE component collects, stores, and processes this data as a client of CA **data/flow** and **data/tunnel** resources, and supports an analytics data subscription service that can stream this CA flow path data in bulk directly to subscribed DLE clients. See [POST /api/v1/subscription/subscribeAll](#) for details on the DLE API you can use to subscribe to this service to receive this data automatically.

**Table 6: Flow Path Analytics Data JSON Response Object**

Element Name	Description
Path	Path data object that contains statistics from the devices in the flow path, containing the elements listed in <a href="#">Table 7 on page 15</a> .
ID	<p>A 5-tuple in JSON format identifying the flow or tunnel flow. The tuple values are as follows, in the order shown:</p> <ul style="list-style-type: none"> <li>• For non-overlay (<b>data/flow</b> resource):  {source IP address, destination IP address, protocol, source port, destination port}</li> <li>• For overlay (<b>data/tunnel</b> resource):  {VTEP source IP address, VTEP destination IP address, VXLAN ID, application source IP address, application destination IP address}</li> </ul>
Time	(Numeric) Time (epoch time in seconds) when the probe run for the flow was started.

[Table 7 on page 15](#) lists the elements in the Path data object of flow path data responses.

**Table 7: Flow Path Analytics JSON Response Data: Path Object Elements**

Element Name	Description
HOP	(Numeric) Number identifying the position of node in the path.
DevMEMUtil	(Numeric, decimal) Memory utilization of the node expressed as a percentage.
DevCPUUtil	(Numeric, decimal) CPU utilization of the node expressed as a percentage.
Bandwidth_Counter	(Numeric) Counter to track the flow packet count at the node.
Bandwidth_ID	(Numeric) Bandwidth measurement ID to track reset of bandwidth filters.
Latency	(Numeric, decimal) Latency for the node from CA server.

**Table 7: Flow Path Analytics JSON Response Data: Path Object Elements (*continued*)**

Element Name	Description
DevName	(String) Node name.
DevSerialID	(String) Node serial id.
TIMESTAMP	Timestamp object representing the time at which flow packet was received at the node. See <a href="#">Table 8 on page 16</a> .
IngressIFLAttribute	Ingress Logical Interface Attribute data object. See <a href="#">Table 9 on page 16</a> .
EgressIFLAttribute	Egress Logical Interface Attribute data object. See <a href="#">Table 10 on page 16</a> .

[Table 8 on page 16](#) lists the elements in the Timestamp data object of the Path data object in flow path data responses.

**Table 8: Flow Path Analytics JSON Response Data: Timestamp Object Elements**

Element Name	Description
TimeStamp	(Numeric) Seconds portion of the timestamp.
TimeStamp_usec	(Numeric) Microseconds portion of the timestamp.
Type	(String) Timestamp type. Accepted values are "PTP", "LCPU" or "NTP".

[Table 9 on page 16](#) lists the elements in the Ingress Logical Interface Attribute data object, part of the Path data object of flow path data responses.

**Table 9: Flow Path Analytics JSON Response Data: Ingress Logical Interface Attribute Object Elements**

Element Name	Description
IFLList	List of Logical Interface data objects. See <a href="#">Table 13 on page 17</a> .

[Table 10 on page 16](#) lists the elements in the Egress Logical Interface Attribute data object, part of the Path data object of flow path data responses.

**Table 10: Flow Path Analytics JSON Response Data: Egress Logical Interface Attribute Object Elements**

Element Name	Description
L3Ecmp	Layer 3 ECMP data object. See <a href="#">Table 11 on page 17</a> .

[Table 11 on page 17](#) lists the elements in the Layer 3 ECMP data object in the Egress Logical Interface Attribute data object (part of the Path data object) in flow path data responses.

**Table 11: Flow Path Analytics JSON Response Data: Layer 3 ECMP Object Elements**

Element Name	Description
IFLList	List of Logical Interface data objects. See <a href="#">Table 13 on page 17</a> .
NumBuckets	(Numeric) Number of buckets in the ECMP.
L2EcmpList	List of Layer 2 ECMP data objects. See <a href="#">Table 12 on page 17</a> .

[Table 12 on page 17](#) lists the elements in the Layer 2 ECMP List data object within the Layer 3 ECMP data object of flow path data responses.

**Table 12: Flow Path Analytics JSON Response Data: Layer 2 ECMP List Object Elements**

Element Name	Description
AE_IFL_NAME	(String) Aggregated logical interface name.
NumBuckets	(Numeric) Number of buckets in the ECMP.
IFLList	List of Logical Interface data objects. See <a href="#">Table 13 on page 17</a> .

[Table 13 on page 17](#) lists the elements in the Logical Interface data object (part of the Layer 2 ECMP and Layer 3 ECMP data objects) of flow path data responses.

**Table 13: Flow Path Analytics JSON Response Data: Logical Interface Object Elements**

Element Name	Description
IFLName	(String) Interface name.
IFLStats	Interface Packet Statistics data object. See <a href="#">Table 14 on page 17</a> .

[Table 14 on page 17](#) lists the elements in the Interface Packet Statistics data object (part of the Logical Interface data object) of flow path data responses.

**Table 14: Flow Path Analytics JSON Response Data: Interface Packet Statistics Object Elements**

Element Name	Description
TX_PKTS	(Numeric) Total transmitted (Tx) packets on the interface.
RX_PKTS	(Numeric) Total received (Rx) packets on the interface.
TX_UCPKTS	(Numeric) Total number of egress/transmitted unicast packets on the interface.
RX_UCPKTS	(Numeric) Total number of ingress/received unicast packets on the interface.
TX_BCPKTS	(Numeric) Total number of outgoing broadcast packets.

**Table 14: Flow Path Analytics JSON Response Data: Interface Packet Statistics Object Elements** (*continued*)

Element Name	Description
RX_BCPKTS	(Numeric) Total number of incoming broadcast packets.
TX_MCPKTS	(Numeric) Total outgoing multicast packets.
RX_MCPKTS	(Numeric) Total incoming multicast packets.
TX_PPS	(Numeric) Transmit bandwidth (packets per second).
RX_PPS	(Numeric) Receive bandwidth (packets per second).
TX_CRCERROR	(Numeric) Total CRC align errors while transmitting.
RX_CRCERROR	(Numeric) Total CRC align errors while receiving.
TX_DROP_PKT	(Numeric) Cumulative Tx packet drop in all the queues for the interface.
RX_DROP_PKT	(Numeric) Ingress Rx packet drops on the interface. This could be due to a buffer full condition.
TX_BYTES	(Numeric) Total transmit bytes.
RX_BYTES	(Numeric) Total receive bytes.
TX_BW	(Numeric) Transmit bandwidth (bytes per second), the average rate at which packet bytes are transmitted.
RX_BW	(Numeric) Receive bandwidth (bytes per second), the average rate at which packet bytes are received.

## Flow Data Compute Agent API Example

This example shows the JSON-format response data returned from a CA API request to retrieve flow data results for the specified flow. See [“GET data/flow Resource” on page 14](#) for details on response data elements.

Also see [“GET start/flow Resource” on page 10](#) for request parameters you can use to specify different options for the data collection when setting up the flow. This example illustrates tracing a simple TCP application flow with no extra instructions specified to CA for collecting flow data.

Start a flow data collection with these parameters (**GET start/flow** request):

```
http://10.94.201.11:8080/start/flow?dport=9000&src=10.1.1.1&dst=10.1.7.1&
sport=4567&proto=tcp
```

Request collected data for the flow (**GET data/flow** request):

**http://10.94.201.11:8080/data/flow?dport=9000&src=10.1.1.1&dst=10.1.7.1&sport=4567&proto=tcp**

Response data:

```
{
  'Path': [
    {
      'Latency': 0.0077555179595947266,
      'DevMEMUtil': 15.0,
      'IngressIFLAttribute': {
        'IFLList': [
          {
            'IFLStats': {
              'RX_BYTES': 149355029120,
              'RX_PKTS': 98481376,
              'TX_BCPKTS': 46,
              'RX_MCPKTS': 6103,
              'TX_PKTS': 7321110,
              'RX_UCPKTS': 98475269,
              'RX_BCPKTS': 4,
              'TX_UCPKTS': 7321064,
              'TX_BYTES': 577408637
            },
            'IFLName': 'ge-0/0/0.0'
          }
        ]
      },
      'DevCPUUtil': 20.0,
      'TIMESTAMP': {
        'TimeStamp': 1442517373,
        'Type': 'LCPU',
        'TimeStamp_usec': 259821
      },
      'DevName': 'device02-example-01',
      'EgressIFLAttribute': {
        'L3Ecmp': {
          'IFLList': [
            {
              'IFLStats': {
                'RX_BYTES': 162959304,
                'RX_PKTS': 2312326,
                'TX_BCPKTS': 33,
                'RX_MCPKTS': 5526,
                'TX_MCPKTS': 5213,
                'TX_PKTS': 29285830,
                'RX_UCPKTS': 2306784,
                'RX_BCPKTS': 16,
                'TX_UCPKTS': 29280584,
                'TX_BYTES': 44441537737,
                'RX_BW': 28
              },
              'IFLName': 'xe-0/0/18.0'
            }
          ],
          {
            'IFLStats': {
```

```
'RX_BYTES': 280504269,
'RX_PKTS': 3399293,
'TX_BCPKTS': 79,
'RX_MCPKTS': 20148,
'TX_MCPKTS': 23222,
'TX_PKTS': 38627454,
'RX_UCPKTS': 3379057,
'RX_BCPKTS': 88,
'TX_UCPKTS': 38604153,
'TX_BYTES': 58522788624
},
'IFLName': 'xe-0/0/8.0'
}
],
'NumBuckets': 3,
'L2EcmpList': [
{
  'IFLList': [
    {
      'IFLStats': {
        'RX_BYTES': 74487320,
        'RX_PKTS': 1054789,
        'TX_BCPKTS': 2,
        'TX_MCPKTS': 10,
        'RX_UCPKTS': 1054789,
        'TX_PKTS': 16197298,
        'TX_UCPKTS': 16197286,
        'TX_BYTES': 24587188634
      },
      'IFLName': 'xe-0/0/14.0'
    },
    {
      'IFLStats': {
        'TX_BW': 28,
        'RX_BYTES': 41279186,
        'RX_PKTS': 573084,
        'TX_BCPKTS': 23,
        'RX_MCPKTS': 5340,
        'TX_MCPKTS': 5229,
        'TX_PKTS': 14371104,
        'RX_UCPKTS': 567720,
        'RX_BCPKTS': 24,
        'TX_UCPKTS': 14365852,
        'TX_BYTES': 21803910373
      },
      'IFLName': 'xe-0/0/16.0'
    }
  ],
  'AE_IFL_NAME': 'ae0.0',
  'NumBuckets': 2
}
]
}
},
'HOP': 1,
'DevSerialID': 'TA3714040533'
```

```

},
{
  'Latency': 0.015787482261657715,
  'DevMEMUtil': 15.0,
  'IngressIFLAttribute': {
    'IFLList': [
      {
        'IFLStats': {
          'RX_BYTES': 44441535195,
          'RX_PKTS': 29285802,
          'TX_BCPKTS': 16,
          'RX_MCPKTS': 5188,
          'TX_MCPKTS': 5526,
          'TX_PKTS': 2312326,
          'RX_UCPKTS': 29280584,
          'RX_BCPKTS': 30,
          'TX_UCPKTS': 2306784,
          'TX_BYTES': 162958884
        },
        'IFLName': 'xe-0/0/2: 0.0'
      }
    ]
  },
  'DevCPUUtil': 15.0,
  'TIMESTAMP': {
    'TimeStamp': 1442517373,
    'Type': 'LCPU',
    'TimeStamp_usec': 291136
  },
  'DevName': 'device02-example-02',
  'EgressIFLAttribute': {
    'IFLList': [
      {
        'IFLStats': {
          'RX_UCPKTS': 3923286,
          'TX_BW': 28,
          'RX_DROP_PKT': 1,
          'RX_PKTS': 3944333,
          'TX_BCPKTS': 40,
          'RX_MCPKTS': 21011,
          'TX_MCPKTS': 8090,
          'TX_PKTS': 59851103,
          'RX_BYTES': 278004527,
          'RX_BCPKTS': 36,
          'TX_UCPKTS': 59842973,
          'TX_BYTES': 90832345724
        },
        'IFLName': 'xe-0/0/4: 0.0'
      }
    ]
  },
  'HOP': 2,
  'DevSerialID': 'TB3714120508'
},
{
  'Latency': 0.0077500343322753906,

```

```
'DevMEMUtil': 15.0,
'IngressIFLAttribute': {
  'IFLList': [
    {
      'IFLStats': {
        'RX_PPS': 3,
        'TX_BW': 263,
        'RX_BYTES': 90832346125,
        'RX_PKTS': 59851103,
        'TX_BCPKTS': 39,
        'RX_MCPKTS': 8090,
        'TX_MCPKTS': 21036,
        'TX_PKTS': 3944361,
        'RX_UCPKTS': 59842973,
        'RX_BCPKTS': 40,
        'TX_UCPKTS': 3923286,
        'TX_BYTES': 278007069,
        'RX_BW': 785
      },
      'IFLName': 'xe-0/0/46.0'
    }
  ]
},
'DevCPUUtil': 10.0,
'TIMESTAMP': {
  'TimeStamp': 1442517373,
  'Type': 'LCPU',
  'TimeStamp_usec': 316698
},
'DevName': 'device02-example-03',
'EgressIFLAttribute': {
  'IFLList': [
    {
      'IFLStats': {
        'TX_BW': 343,
        'RX_BYTES': 506694775,
        'RX_PKTS': 7236139,
        'TX_BCPKTS': 59,
        'TX_PKTS': 98391432,
        'RX_UCPKTS': 7236138,
        'RX_BCPKTS': 1,
        'TX_UCPKTS': 98391373,
        'TX_PPS': 1,
        'TX_BYTES': 149348425859,
        'RX_BW': 369
      },
      'IFLName': 'ge-0/0/12.0'
    }
  ]
},
'HOP': 3,
'DevSerialID': 'VB3113350021'
},
>ID': "('10.1.1.1', '10.1.7.1', 6, 4567, 9000)",
```

```
'Time': 1442517345.8826571
}
```

## Bandwidth Flow Data Compute Agent API Example

These examples show the JSON-format response data returned from CA API requests to retrieve flow data results that include bandwidth measurements for TCP and UDP flows. See [“GET data/flow Resource” on page 14](#) for details on response data elements.

Also see [“GET start/flow Resource” on page 10](#) for request parameters you can use to specify different options for the data collection when setting up the flow, such as the **bw\_ingress** option used here to also trace ingress bandwidth statistics.

- [Bandwidth Example with a TCP Flow on page 23](#)
- [Bandwidth Example with a UDP Flow on page 27](#)

### Bandwidth Example with a TCP Flow

Start a flow data collection with these parameters for a TCP flow (**GET start/flow** request):

```
http://10.94.201.11:8080/start/flow?src=10.1.1.1&bw_ingress=true&proto=tcp&dst=10.1.7.1&dport=9000&sport=58991
```

Request collected data for the TCP flow (**GET data/flow** request):

```
http://10.94.201.11:8080/data/flow?src=10.1.1.1&proto=tcp&dst=10.1.7.1&dport=9000&sport=58991
```

Response data:

```
{
  'Path': [
    {
      'Latency': 0.015927553176879883,
      'DevMEMUtil': 15.0,
      'IngressIFLAttribute': {
        'IFLList': [
          {
            'IFLStats': {
              'RX_PPS': 81274,
              'TX_BW': 387242,
              'RX_BYTES': 52602682875,
              'RX_PKTS': 34737472,
              'TX_BCPKTS': 44,
              'RX_MCPKTS': 6030,
              'TX_PKTS': 2696789,
              'RX_UCPKTS': 34731438,
              'RX_BCPKTS': 4,
              'TX_UCPKTS': 2696745,
              'TX_PPS': 5490,
              'TX_BYTES': 251476888,
              'RX_BW': 123370657
            },
            'IFLName': 'ge-0/0/0.0'
          }
        ]
      }
    }
  ]
}
```

```
]
},
'DevCPUUtil': 19.0,
'TIMESTAMP': {
  'TimeStamp': 1442515155,
  'Type': 'LCPU',
  'TimeStamp_usec': 395570
},
'DevName': 'device02-example-01',
'Bandwidth_Counter': 798465,
'EgressIFLAttribute': {
  'L3Ecmp': {
    'IFLList': [
      {
        'IFLStats': {
          'TX_BW': 123379964,
          'RX_BYTES': 32821992,
          'RX_PKTS': 463681,
          'TX_BCPKTS': 32,
          'RX_MCPKTS': 5245,
          'TX_MCPKTS': 4948,
          'TX_PKTS': 6040171,
          'RX_UCPKTS': 458421,
          'RX_BCPKTS': 15,
          'TX_UCPKTS': 6035191,
          'TX_PPS': 81279,
          'TX_BYTES': 9161164227
        },
        'IFLName': 'xe-0/0/18.0'
      },
      {
        'IFLStats': {
          'TX_BW': 28,
          'RX_BYTES': 149296221,
          'RX_PKTS': 1535534,
          'TX_BCPKTS': 78,
          'RX_MCPKTS': 19877,
          'TX_MCPKTS': 22954,
          'TX_PKTS': 22465642,
          'RX_UCPKTS': 1515570,
          'RX_BCPKTS': 87,
          'TX_UCPKTS': 22442610,
          'TX_BYTES': 33990330357
        },
        'IFLName': 'xe-0/0/8.0'
      }
    ],
    'NumBuckets': 3,
    'L2EcmpList': [
      {
        'IFLList': [
          {
            'IFLStats': {
              'RX_PPS': 5482,
              'RX_BYTES': 9662714,
              'RX_PKTS': 135509,
```

```

        'TX_BCPKTS': 2,
        'TX_MCPKTS': 10,
        'RX_UCPKTS': 135509,
        'TX_PKTS': 6250423,
        'TX_UCPKTS': 6250411,
        'TX_BYTES': 9488051662,
        'RX_BW': 384481
    },
    'IFLName': 'xe-0/0/14.0'
},
{
    'IFLStats': {
        'RX_BYTES': 40185445,
        'RX_PKTS': 568824,
        'TX_BCPKTS': 23,
        'RX_MCPKTS': 5068,
        'TX_MCPKTS': 4960,
        'TX_PKTS': 10405,
        'RX_UCPKTS': 563733,
        'RX_BCPKTS': 23,
        'TX_UCPKTS': 5422,
        'TX_BYTES': 5343023,
        'RX_BW': 725
    },
    'IFLName': 'xe-0/0/16.0'
}
],
'AE_IFL_NAME': 'ae0.0',
'NumBuckets': 2
}
]
}
},
'HOP': 1,
'DevSerialID': 'TA3714040533',
'Bandwidth_ID': 612761622
},
{
    'Latency': 0.016196966171264648,
    'DevMEMUtil': 15.0,
    'IngressIFLAttribute': {
        'IFLList': [
            {
                'IFLStats': {
                    'RX_PPS': 81273,
                    'RX_BYTES': 9098071939,
                    'RX_PKTS': 5998580,
                    'TX_BCPKTS': 15,
                    'RX_MCPKTS': 4923,
                    'TX_MCPKTS': 5245,
                    'TX_PKTS': 463681,
                    'RX_UCPKTS': 5993628,
                    'RX_BCPKTS': 29,
                    'TX_UCPKTS': 458421,
                    'TX_BYTES': 32821572,
                    'RX_BW': 123371832
                }
            }
        ]
    }
}

```

```
    },
    'IFLName': 'xe-0/0/2: 0.0'
  }
]
},
'DevCPUUtil': 10.0,
'TIMESTAMP': {
  'TimeStamp': 1442515155,
  'Type': 'LCPU',
  'TimeStamp_usec': 435787
},
'DevName': 'device02-example-02',
'Bandwidth_Counter': 799209,
'EgressIFLAttribute': {
  'IFLList': [
    {
      'IFLStats': {
        'RX_UCPKTS': 1154727,
        'RX_PPS': 5486,
        'TX_BW': 123371390,
        'RX_DROP_PKT': 1,
        'RX_PKTS': 1175501,
        'TX_BCPKTS': 39,
        'RX_MCPKTS': 20739,
        'TX_MCPKTS': 7823,
        'TX_PKTS': 12149595,
        'RX_BYTES': 83261636,
        'RX_BCPKTS': 35,
        'TX_UCPKTS': 12141733,
        'TX_PPS': 81273,
        'TX_BYTES': 18428173956,
        'RX_BW': 384659
      },
      'IFLName': 'xe-0/0/4: 0.0'
    }
  ]
},
'HOP': 2,
'DevSerialID': 'TB3714120508',
'Bandwidth_ID': 3274244103
},
{
  'Latency': 0.015960097312927246,
  'DevMEMUtil': 15.0,
  'IngressIFLAttribute': {
    'IFLList': [
      {
        'IFLStats': {
          'RX_PPS': 81280,
          'TX_BW': 384200,
          'RX_BYTES': 18647080829,
          'RX_PKTS': 12293804,
          'TX_BCPKTS': 38,
          'RX_MCPKTS': 7824,
          'TX_MCPKTS': 20763,
          'TX_PKTS': 1168943,
```

```

        'RX_UCPKTS': 12285941,
        'RX_BCPKTS': 39,
        'TX_UCPKTS': 1148142,
        'TX_PPS': 5477,
        'TX_BYTES': 82802361,
        'RX_BW': 123381858
    },
    'IFLName': 'xe-0/0/46.0'
}
]
},
'DevCPUUtil': 12.0,
'TIMESTAMP': {
    'TimeStamp': 1442515155,
    'Type': 'LCPU',
    'TimeStamp_usec': 461993
},
'DevName': 'device02-example-03',
'Bandwidth_Counter': 800689,
'EgressIFLAttribute': {
    'IFLList': [
        {
            'IFLStats': {
                'RX_PPS': 5478,
                'TX_BW': 123381264,
                'RX_BYTES': 183327408,
                'RX_PKTS': 2617174,
                'TX_BCPKTS': 59,
                'RX_UCPKTS': 2617174,
                'TX_PKTS': 34481916,
                'TX_UCPKTS': 34481857,
                'TX_PPS': 81278,
                'TX_BYTES': 52339522340,
                'RX_BW': 383476
            },
            'IFLName': 'ge-0/0/12.0'
        }
    ]
},
'HOP': 3,
'DevSerialID': 'VB3113350021',
'Bandwidth_ID': 3148349462
}
],
>ID': "('10.1.1.1', '10.1.7.1', 6, 58991, 9000)",
'Time': 1442515128.0384891
}

```

### Bandwidth Example with a UDP Flow

Start a flow data collection with these parameters for a UDP flow (**GET start/flow** request):

**[http://10.94.201.11:8080/start/flow?src=10.1.1.1&bw\\_ingress=true&proto=udp&dst=10.1.7.1&dport=9000&sport=54676](http://10.94.201.11:8080/start/flow?src=10.1.1.1&bw_ingress=true&proto=udp&dst=10.1.7.1&dport=9000&sport=54676)**

Request collected data for the UDP flow (**GET data/flow** request):

**http://10.94.201.11:8080/data/flow?src=10.1.1.1&proto=udp&dst=10.1.7.1&dport=9000&sport=54676**

Response data:

```
{
  'Path': [
    {
      'Latency': 0.0078104734420776367,
      'DevMEMUtil': 15.0,
      'IngressIFLAttribute': {
        'IFLList': [
          {
            'IFLStats': {
              'RX_PPS': 56,
              'TX_BW': 6112,
              'RX_BYTES': 54888012436,
              'RX_PKTS': 36245359,
              'TX_BCPKTS': 44,
              'RX_MCPKTS': 6031,
              'TX_PKTS': 2799842,
              'RX_UCPKTS': 36239324,
              'RX_BCPKTS': 4,
              'TX_UCPKTS': 2799798,
              'TX_PPS': 6,
              'TX_BYTES': 258867317,
              'RX_BW': 12943
            },
            'IFLName': 'ge-0/0/0.0'
          }
        ]
      },
      'DevCPUUtil': 19.0,
      'TIMESTAMP': {
        'TimeStamp': 1442515206,
        'Type': 'LCPU',
        'TimeStamp_usec': 413577
      },
      'DevName': 'device02-example-01',
      'Bandwidth_Counter': 529,
      'EgressIFLAttribute': {
        'L3Ecmp': {
          'IFLList': [
            {
              'IFLStats': {
                'TX_BW': 12907,
                'RX_BYTES': 32822580,
                'RX_PKTS': 463687,
                'TX_BCPKTS': 32,
                'RX_MCPKTS': 5251,
                'TX_MCPKTS': 4954,
                'TX_PKTS': 7518908,
                'RX_UCPKTS': 458421,
                'RX_BCPKTS': 15,
```

```

        'TX_UCPKTS': 7513922,
        'TX_PPS': 55,
        'TX_BYTES': 11404639015
    },
    'IFLName': 'xe-0/0/18.0'
},
{
    'IFLStats': {
        'TX_BW': 28,
        'RX_BYTES': 149297823,
        'RX_PKTS': 1535549,
        'TX_BCPKTS': 78,
        'RX_MCPKTS': 19883,
        'TX_MCPKTS': 22960,
        'TX_PKTS': 22465666,
        'RX_UCPKTS': 1515579,
        'RX_BCPKTS': 87,
        'TX_UCPKTS': 22442628,
        'TX_BYTES': 33990333262
    },
    'IFLName': 'xe-0/0/8.0'
}
],
'NumBuckets': 3,
'L2EcmpList': [
    {
        'IFLList': [
            {
                'IFLStats': {
                    'RX_PPS': 2,
                    'RX_BYTES': 16782331,
                    'RX_PKTS': 236628,
                    'TX_BCPKTS': 2,
                    'TX_MCPKTS': 10,
                    'RX_UCPKTS': 236628,
                    'TX_PKTS': 6250423,
                    'TX_UCPKTS': 6250411,
                    'TX_BYTES': 9488051662,
                    'RX_BW': 1910
                },
                'IFLName': 'xe-0/0/14.0'
            },
            {
                'IFLStats': {
                    'RX_PPS': 2,
                    'RX_BYTES': 40235547,
                    'RX_PKTS': 568900,
                    'TX_BCPKTS': 23,
                    'RX_MCPKTS': 5074,
                    'TX_MCPKTS': 4966,
                    'TX_PKTS': 10411,
                    'RX_UCPKTS': 563803,
                    'RX_BCPKTS': 23,
                    'TX_UCPKTS': 5422,
                    'TX_BYTES': 5343611,
                    'RX_BW': 1919
                }
            }
        ]
    }
]

```

```

        },
        'IFLName': 'xe-0/0/16.0'
    }
],
'AE_IFL_NAME': 'ae0.0',
'NumBuckets': 2
}
]
}
},
'HOP': 1,
'DevSerialID': 'TA3714040533',
'Bandwidth_ID': 612761623
},
{
'Latency': 0.01627051830291748,
'DevMEMUtil': 15.0,
'IngressIFLAttribute': {
'IFLList': [
{
'IFLStats': {
'RX_PPS': 54,
'RX_BYTES': 11404653969,
'RX_PKTS': 7518954,
'TX_BCPKTS': 15,
'RX_MCPKTS': 4929,
'TX_MCPKTS': 5251,
'TX_PKTS': 463687,
'RX_UCPKTS': 7513996,
'RX_BCPKTS': 29,
'TX_UCPKTS': 458421,
'TX_BYTES': 32822160,
'RX_BW': 12844
},
'IFLName': 'xe-0/0/2: 0.0'
}
]
},
'DevCPUUtil': 11.0,
'TIMESTAMP': {
'TimeStamp': 1442515206,
'Type': 'LCPU',
'TimeStamp_usec': 442902
},
'DevName': 'device02-example-02',
'Bandwidth_Counter': 519,
'EgressIFLAttribute': {
'IFLList': [
{
'IFLStats': {
'RX_UCPKTS': 1258544,
'RX_PPS': 1,
'TX_BW': 12739,
'RX_DROP_PKT': 1,
'RX_PKTS': 1279324,
'TX_BCPKTS': 39,

```

```

        'RX_MCPKTS': 20745,
        'TX_MCPKTS': 7829,
        'TX_PKTS': 13777264,
        'RX_BYTES': 90574797,
        'RX_BCPKTS': 35,
        'TX_UCPKTS': 13769396,
        'TX_PPS': 52,
        'TX_BYTES': 20897798936,
        'RX_BW': 1267
    },
    'IFLName': 'xe-0/0/4: 0.0'
}
]
},
'HOP': 2,
'DevSerialID': 'TB3714120508',
'Bandwidth_ID': 3274244104
},
{
    'Latency': 0.0077755451202392578,
    'DevMEMUtil': 15.0,
    'IngressIFLAttribute': {
        'IFLList': [
            {
                'IFLStats': {
                    'RX_PPS': 52,
                    'TX_BW': 1423,
                    'RX_BYTES': 20897786653,
                    'RX_PKTS': 13777210,
                    'TX_BCPKTS': 38,
                    'RX_MCPKTS': 7829,
                    'TX_MCPKTS': 20769,
                    'TX_PKTS': 1279347,
                    'RX_UCPKTS': 13769342,
                    'RX_BCPKTS': 39,
                    'TX_UCPKTS': 1258540,
                    'TX_PPS': 1,
                    'TX_BYTES': 90573885,
                    'RX_BW': 12699
                },
                'IFLName': 'xe-0/0/46.0'
            }
        ]
    },
    'DevCPUUtil': 13.0,
    'TIMESTAMP': {
        'TimeStamp': 1442515206,
        'Type': 'LCPU',
        'TimeStamp_usec': 466649
    },
    'DevName': 'device02-example-03',
    'Bandwidth_Counter': 510,
    'EgressIFLAttribute': {
        'IFLList': [
            {
                'IFLStats': {

```

```

        'TX_BW': 12593,
        'RX_BYTES': 190339083,
        'RX_PKTS': 2717302,
        'TX_BCPKTS': 59,
        'RX_UCPKTS': 2717302,
        'TX_PKTS': 36157163,
        'TX_UCPKTS': 36157104,
        'TX_PPS': 51,
        'TX_BYTES': 54881536331
      },
      'IFLName': 'ge-0/0/12.0'
    }
  ]
},
'HOP': 3,
'DevSerialID': 'VB3113350021',
'Bandwidth_ID': 3148349463
}
],
>ID': "('10.1.1.1', '10.1.7.1', 17, 54676, 9000)",
'Time': 1442515179.0630369
}

```

## Mirror Flow Data Compute Agent API Example

This example shows the JSON-format response data returned from a CA API request to retrieve flow data results for the specified flow that includes mirroring. See [“GET data/flow Resource” on page 14](#) for details on response data elements.

Also see [“GET start/flow Resource” on page 10](#) for request parameters you can use to specify different options for the data collection when setting up the flow, such as the mirror options used here to set up and trace mirroring. This flow also includes the **bw\_ingress** option to include bandwidth measurements.

Start a flow data collection with mirroring and bandwidth measurements (**GET start/flow** request):

```
http://10.94.201.11:8080/start/flow?src=10.1.1.1&mirror_dir=ingress&
mirror_analyzer=10.2.1.1&bw_ingress=true&proto=tcp&dst=10.1.7.1&mirror_type=ERSPAN&
dport=9000&sport=59047
```

Request collected data for the flow (**GET data/flow** request):

```
http://10.94.201.11:8080/data/flow?src=10.1.1.1&proto=tcp&dst=10.1.7.1&
dport=9000&sport=59047
```

Response data:

```

{
  'Path': [
    {
      'Latency': 0.016028523445129395,
      'DevMEMUtil': 15.0,
      'IngressIFLAttribute': {
        'IFLList': [

```

```

{
  'IFLStats': {
    'RX_PPS': 81282,
    'TX_BW': 364260,
    'RX_BYTES': 73316904545,
    'RX_PKTS': 48386313,
    'TX_BCPKTS': 44,
    'RX_MCPKTS': 6042,
    'TX_PKTS': 3710505,
    'RX_UCPKTS': 48380267,
    'RX_BCPKTS': 4,
    'TX_UCPKTS': 3710461,
    'TX_PPS': 5161,
    'TX_BYTES': 323540027,
    'RX_BW': 123382993
  },
  'IFLName': 'ge-0/0/0.0'
}
]
},
'DevCPUUtil': 20.0,
'TIMESTAMP': {
  'TimeStamp': 1442515533,
  'Type': 'LCPU',
  'TimeStamp_usec': 174849
},
'DevName': 'device02-example-01',
'Bandwidth_Counter': 801594,
'EgressIFLAttribute': {
  'L3Ecmp': {
    'IFLList': [
      {
        'IFLStats': {
          'RX_PPS': 5163,
          'RX_BYTES': 52389977,
          'RX_PKTS': 741687,
          'TX_BCPKTS': 32,
          'RX_MCPKTS': 5289,
          'TX_MCPKTS': 4991,
          'TX_PKTS': 7520651,
          'RX_UCPKTS': 736383,
          'RX_BCPKTS': 15,
          'TX_UCPKTS': 7515628,
          'TX_BYTES': 11405042894,
          'RX_BW': 362203
        },
        'IFLName': 'xe-0/0/18.0'
      },
      {
        'IFLStats': {
          'RX_PPS': 2,
          'TX_BW': 123380461,
          'RX_BYTES': 183663201,
          'RX_PKTS': 2023973,
          'TX_BCPKTS': 78,
          'RX_MCPKTS': 19920,

```

```

        'TX_MCPKTS': 22997,
        'TX_PKTS': 26230297,
        'RX_UCPKTS': 2003966,
        'RX_BCPKTS': 87,
        'TX_UCPKTS': 26207222,
        'TX_PPS': 81279,
        'TX_BYTES': 39704643063,
        'RX_BW': 972
    },
    'IFLName': 'xe-0/0/8.0'
}
],
'NumBuckets': 3,
'L2EcmpList': [
{
    'IFLList': [
        {
            'IFLStats': {
                'RX_BYTES': 27517777,
                'RX_PKTS': 387681,
                'TX_BCPKTS': 2,
                'TX_MCPKTS': 10,
                'RX_UCPKTS': 387681,
                'TX_PKTS': 10035134,
                'TX_UCPKTS': 10035122,
                'TX_BYTES': 15233232856
            },
            'IFLName': 'xe-0/0/14.0'
        },
        {
            'IFLStats': {
                'RX_BYTES': 40383598,
                'RX_PKTS': 569116,
                'TX_BCPKTS': 23,
                'RX_MCPKTS': 5111,
                'TX_MCPKTS': 5004,
                'TX_PKTS': 4618539,
                'RX_UCPKTS': 563982,
                'RX_BCPKTS': 23,
                'TX_UCPKTS': 4613512,
                'TX_BYTES': 7000005140
            },
            'IFLName': 'xe-0/0/16.0'
        }
    ],
    'AE_IFL_NAME': 'ae0.0',
    'NumBuckets': 2
}
]
}
},
'HOP': 1,
'DevSerialID': 'TA3714040533',
'Bandwidth_ID': 612761628
},
{

```

```

'Latency': 0.016282558441162109,
'DevMEMUtil': 15.0,
'IngressIFLAttribute': {
  'IFLList': [
    {
      'IFLStats': {
        'RX_PPS': 81284,
        'TX_BW': 1033,
        'RX_BYTES': 39777173103,
        'RX_PKTS': 26278078,
        'TX_BCPKTS': 87,
        'RX_MCPKTS': 22997,
        'TX_MCPKTS': 19920,
        'TX_PKTS': 2023967,
        'RX_UCPKTS': 26255003,
        'RX_BCPKTS': 78,
        'TX_UCPKTS': 2003960,
        'TX_PPS': 2,
        'TX_BYTES': 183661035,
        'RX_BW': 123387391
      },
      'IFLName': 'xe-0/0/44.0'
    }
  ]
},
'DevCPUUtil': 8.0,
'TIMESTAMP': {
  'TimeStamp': 1442515533,
  'Type': 'LCPU',
  'TimeStamp_usec': 211202
},
'DevName': 'device03-example-01',
'Bandwidth_Counter': 802300,
'EgressIFLAttribute': {
  'IFLList': [
    {
      'IFLStats': {
        'TX_BW': 123352971,
        'RX_BYTES': 161350530,
        'RX_PKTS': 1998868,
        'TX_BCPKTS': 86,
        'RX_MCPKTS': 23151,
        'TX_MCPKTS': 19837,
        'TX_PKTS': 26229091,
        'RX_UCPKTS': 1975640,
        'RX_BCPKTS': 77,
        'TX_UCPKTS': 26209168,
        'TX_PPS': 81261,
        'TX_BYTES': 39746534033,
        'RX_BW': 30
      },
      'IFLName': 'xe-0/0/94.0'
    }
  ]
},
'HOP': 2,

```

```
'DevSerialID': 'VB3714010003',
'Bandwidth_ID': 361365522
},
{
  'Latency': 0.015943408012390137,
  'DevMEMUtil': 15.0,
  'IngressIFLAttribute': {
    'IFLList': [
      {
        'IFLStats': {
          'RX_PPS': 81275,
          'RX_BYTES': 39743375075,
          'RX_PKTS': 26225556,
          'TX_BCPKTS': 77,
          'RX_MCPKTS': 19837,
          'TX_MCPKTS': 23151,
          'TX_PKTS': 1998868,
          'RX_UCPKTS': 26205633,
          'RX_BCPKTS': 86,
          'TX_UCPKTS': 1975640,
          'TX_BYTES': 161350530,
          'RX_BW': 123374219
        },
        'IFLName': 'xe-0/0/84.0'
      }
    ]
  },
  'DevCPUUtil': 13.0,
  'TIMESTAMP': {
    'TimeStamp': 1442515533,
    'Type': 'LCPU',
    'TimeStamp_usec': 244050
  },
  'DevName': 'device02-example-03',
  'Bandwidth_Counter': 802957,
  'EgressIFLAttribute': {
    'IFLList': [
      {
        'IFLStats': {
          'RX_PPS': 5193,
          'TX_BW': 123373897,
          'RX_BYTES': 253631556,
          'RX_PKTS': 3621325,
          'TX_BCPKTS': 59,
          'RX_UCPKTS': 3621325,
          'TX_PKTS': 48212178,
          'TX_UCPKTS': 48212119,
          'TX_PPS': 81274,
          'TX_BYTES': 73178766632,
          'RX_BW': 363553
        },
        'IFLName': 'ge-0/0/12.0'
      }
    ]
  },
  'HOP': 3,
```

```

        'DevSerialID': 'VB3113350021',
        'Bandwidth_ID': 3148349468
    }
],
    'ID': "('10.1.1.1', '10.1.7.1', 6, 59047, 9000)",
    'Time': 1442515505.8140769
}

```

**Related  
Documentation**

- [Compute Agent API Resources Overview on page 3](#)
- [Cloud Analytics Engine Feature Guide for the QFX Series](#)

## Retrieving Active Flows List

Use the Cloud Analytics Engine CA API active flows resource to retrieve information about flows you started that are currently active.

- [GET active/flow Resource on page 37](#)
- [Active Flows Compute Agent API Example on page 37](#)

### GET active/flow Resource

Use the **GET active/flow** CA API to retrieve a list of all active application flows for the client who originated the request.

Request Parameters:

- None

Return Codes:

- **EOK**—CA found and returned active flows for this client.
- **EFAIL**—Failed to find or return active flows for this client.

Response Data:

- **flows**: List of flow identifier tuples for all active flows initiated by this client, in JSON format with tuple values in order as follows:
  - `{DIP, SIP, PROTO, SPORT, DPORT}`

See [“Active Flows Compute Agent API Example” on page 37](#) for a sample request and response.

### Active Flows Compute Agent API Example

This example shows the JSON-format response data returned from a CA API request to retrieve all active flows for the client issuing the request. See [“GET active/flow Resource” on page 37](#) for details on response data elements.

Request (**GET active/flow**):

**http://10.94.201.11:8080/active/flow**

Response data:

```
{
  'flows': [
    [
      '10.1.7.1',
      '10.1.1.1',
      'tcp',
      59010,
      9000
    ],
    [
      '10.1.7.1',
      '10.1.1.1',
      'tcp',
      59011,
      9000
    ],
    [
      '10.1.7.1',
      '10.1.1.1',
      'tcp',
      59012,
      9000
    ]
  ]
}
```

**Related  
Documentation**

- [Compute Agent API Resources Overview on page 3](#)
- [Cloud Analytics Engine Feature Guide for the QFX Series](#)

---

## Starting and Stopping a Tunnel Flow Data Collection

Use the Cloud Analytics Engine CA API tunnel data resource to start or stop generating application flow path analytics data over a tunnel.

- [GET start/tunnel Resource on page 38](#)
- [GET stop/tunnel Resource on page 40](#)
- [Compute Agent Tunnel Data Resource Usage Example on page 41](#)

### GET start/tunnel Resource

Use the **GET start/tunnel** CA API to instruct CA to generate probes and start collecting information from the network about an overlay tunnel flow. You can also use this API call to modify the specified data collection parameters for a tunnel flow that is already active. This is an asynchronous API.

## Request Parameters:

- [Table 15 on page 39](#) describes the **GET start/tunnel** request parameters, which include tuple values identifying the tunnel flow, and optional parameters specifying what to collect from the available tunnel flow data options.

Table 15: GET start/tunnel Resource Parameters

Flow Resource Parameters	Description
src	(String) VTEP Source IPv4 address.
dst	(String) VTEP endpoint Ipv4 address.
sport	(Numeric) Source port corresponding to the protocol type for the encapsulated application flow.
vnid	(Numeric) VXLAN network identifier.
appsrc	(String) Source IP address of the application communicating via the tunnel VTEP.
appdst	(String) Destination IP address of the application corresponding to <b>appsrc</b> .
appsmac	(String) Application source MAC address.
appdmac	(String) Application destination MAC address.
sfreq	(Numeric) Sampling frequency for collecting network data (in seconds). Optional parameter. Default value is 1 second.
nhops	(Numeric) Number of network hops from which to collect analytics data. Data is gathered across all hops until the destination ( $\leq 255$ ) if this parameter is not specified. Optional parameter. Default value is 255.
ntry	(Numeric) Number of tries to attempt in case of a timeout on a given hop. If this parameter is not specified, only a single attempt is made to gather data for the hop. Optional parameter. Default value is 1 (no retries).
timeout	(Numeric) Timeout (in ms) to wait for the device at a given hop to respond. If the device does not respond back within the timeout, any delayed response is ignored by CA. Optional parameter. Default value is 30000 ms (30 seconds).
chksum	(Boolean) Enable ("true") or disable ("false") checksum computation. Optional parameter. Default value is "true" or enabled.
bw-ingress	(Boolean) If specified as "true", measure flow bandwidth at each hop ingress. Optional parameter. Default value is "false" or do not include bandwidth measurements.

Table 15: GET start/tunnel Resource Parameters (*continued*)

Flow Resource Parameters	Description
mirror_type	(String) Install mirroring of the indicated type for this flow. Supported mirror type values: "ERSPAN".  Optional parameter. Default if not specified is no mirror installation.
mirror_dir	(String) Mirror direction. Supported values: "ingress".  Optional parameter with no default value. If <b>mirror_type</b> is specified, this parameter must also be specified.
mirror_analyzer	(String) Mirror analyzer IP address.  Optional parameter with no default value. If <b>mirror_type</b> is specified, this parameter must also be specified.

## Return Codes:

- **EOK**—Request was serviced.
- **EAGAIN**—Exceeding pending updates for the same tunnel flow. This might happen when the same flow is modified, or there is a previous stop request pending on this tunnel flow.
- **EFAIL**—Invalid parameters were provided.
- **E\_UNAUTHORIZED**—Tunnel flow was initiated by a different client and cannot be started or modified by the originator of this request.

## Response Data:

- None

See [“Compute Agent Tunnel Data Resource Usage Example” on page 41](#) for a simple example of a request to start a tunnel flow with required parameters identifying the tunnel flow, the same parameters used subsequently to get collected data from that tunnel flow, and the same parameters used again to stop the flow.

You can also see the following **GET data/tunnel** example that shows starting an overlay tunnel flow data collection in a KVM topology, and the response data that is returned:

- [Tunnel Data Compute Agent API Example on page 46](#)

## GET stop/tunnel Resource

Use the **GET stop/tunnel** CA API to instruct CA to stop collecting information from the network about a tunnel flow. This is an asynchronous API.

## Request Parameters:

- The **GET stop/tunnel** request parameters are the following tuple values that identify the overlay tunnel:

- **src:** (String) VTEP source IPv4 address.
- **dst:** (String) VTEP endpoint IPv4 address.
- **vnid:** (Numeric) VXLAN network identifier.
- **sport:** (Numeric) Source port corresponding to the protocol type for the encapsulated application flow.
- **appsrc:** (String) Application source IP address communicating on the given tunnel's VTEP.
- **appdst:** (String) Application destination IP address corresponding to **appsrc**.

Return Codes:

- **EOK**—Request was serviced.
- **EFAIL**—Invalid parameters were provided.
- **E\_UNAUTHORIZED**—Tunnel flow was initiated by a different client and cannot be stopped by the originator of this request.
- **E\_AoF**—No such tunnel flow.

Response Data:

- None

See “[Compute Agent Tunnel Data Resource Usage Example](#)” on page 41 for a simple example of a request to stop a tunnel flow data collection, using the same parameters as the **GET start/tunnel** request that started the tunnel flow.

## Compute Agent Tunnel Data Resource Usage Example

The following is a simple example showing a sequence of tunnel data resource CA API web requests to a CA server (IP address 10.94.201.11, port 8080) to start a tunnel flow data collection, retrieve collected data from that tunnel flow, and stop the tunnel flow data collection. The minimum required elements of the tunnel identifier tuple are provided for each request. All requests use the GET method.

- Start tunnel flow data collection (**GET start/tunnel**):

```
http://10.94.201.11:8080/start/tunnel?src=10.1.1.1&sport=4567&vnid=5&dst=10.1.7.1&appdst=10.1.2.5&appmac=44:1e:a1:02:25:c0&dport=9000&appsrc=10.1.2.3&appsmac=9c:8e:99:12:88:2e
```

- Request collected tunnel data (**GET data/tunnel**):

```
http://10.94.201.11:8080/data/tunnel?src=10.1.1.1&sport=4567&vnid=5&dst=10.1.7.1&appdst=10.1.2.5&appsrc=10.1.2.3
```

- Stop tunnel flow data collection (**GET stop/tunnel**):

```
http://10.94.201.11:8080/stop/tunnel?src=10.1.1.1&sport=4567&vnid=5&dst=10.1.7.1&appdst=10.1.2.5&appsrc=10.1.2.3
```

- Related Documentation**
- [Compute Agent API Resources Overview on page 3](#)
  - [Cloud Analytics Engine Feature Guide for the QFX Series](#)

## Requesting Collected Tunnel Flow Data

---

Use the Cloud Analytics Engine CA API tunnel data resource to retrieve application flow path analytics data flowing over a tunnel.

- [GET data/tunnel Resource on page 42](#)
- [Flow Path Response Data Object on page 43](#)
- [Tunnel Data Compute Agent API Example on page 46](#)

### GET data/tunnel Resource

Use the **GET data/tunnel** CA API to request analytics data from the network for a tunnel flow you have started. This is an asynchronous API.

The available data depends on the parameters used to start the tunnel flow, described in [“GET start/tunnel Resource” on page 38](#).

Request Parameters:

- The following tuple values identifying the overlay tunnel:
  - **src**: (String) VTEP source IPv4 address.
  - **dst**: (String) VTEP endpoint IPv4 address.
  - **vnid**: (Numeric) VXLAN network identifier.
  - **appsrc**: (String) Application source IP address communicating on the given tunnel's VTEP.
  - **appdst**: (String) Application destination IP address corresponding to **appsrc**.

Return Codes:

- **None**—CA returns the requested tunnel flow data.
- **EFAIL**—No data is present for the specified tunnel flow.
- **E\_UNAUTHORIZED**—Flow was initiated by a different client and cannot be retrieved by the originator of this request.
- **E\_AoF**—No such flow.

Response Data:

- A data/tunnel response data object with statistics from the devices in the tunnel flow path, in JSON format. See [“Flow Path Response Data Object” on page 15](#).

See [“Tunnel Data Compute Agent API Example” on page 46](#) for an example of using the data/tunnel resource.

## Flow Path Response Data Object

Table 6 on page 15 lists the elements in the response data object (JSON format) for requests to retrieve CA flow path analytics data. See “GET data/flow Resource” on page 14 and “GET data/tunnel Resource” on page 42, which describe the CA API resource requests that return this data.

Note that the Cloud Analytics Engine DLE component collects, stores, and processes this data as a client of CA **data/flow** and **data/tunnel** resources, and supports an analytics data subscription service that can stream this CA flow path data in bulk directly to subscribed DLE clients. See *POST /api/v1/subscription/subscribeAll* for details on the DLE API you can use to subscribe to this service to receive this data automatically.

**Table 16: Flow Path Analytics Data JSON Response Object**

Element Name	Description
Path	Path data object that contains statistics from the devices in the flow path, containing the elements listed in Table 7 on page 15.
ID	<p>A 5-tuple in JSON format identifying the flow or tunnel flow. The tuple values are as follows, in the order shown:</p> <ul style="list-style-type: none"> <li>For non-overlay (<b>data/flow</b> resource):  {source IP address, destination IP address, protocol, source port, destination port}</li> <li>For overlay (<b>data/tunnel</b> resource):  {VTEP source IP address, VTEP destination IP address, VXLAN ID, application source IP address, application destination IP address}</li> </ul>
Time	(Numeric) Time (epoch time in seconds) when the probe run for the flow was started.

Table 7 on page 15 lists the elements in the Path data object of flow path data responses.

**Table 17: Flow Path Analytics JSON Response Data: Path Object Elements**

Element Name	Description
HOP	(Numeric) Number identifying the position of node in the path.
DevMEMUtil	(Numeric, decimal) Memory utilization of the node expressed as a percentage.
DevCPUUtil	(Numeric, decimal) CPU utilization of the node expressed as a percentage.
Bandwidth_Counter	(Numeric) Counter to track the flow packet count at the node.
Bandwidth_ID	(Numeric) Bandwidth measurement ID to track reset of bandwidth filters.
Latency	(Numeric, decimal) Latency for the node from CA server.
DevName	(String) Node name.
DevSerialID	(String) Node serial id.

**Table 17: Flow Path Analytics JSON Response Data: Path Object Elements (*continued*)**

Element Name	Description
TIMESTAMP	Timestamp object representing the time at which flow packet was received at the node. See <a href="#">Table 8 on page 16</a> .
IngressIFLAttribute	Ingress Logical Interface Attribute data object. See <a href="#">Table 9 on page 16</a> .
EgressIFLAttribute	Egress Logical Interface Attribute data object. See <a href="#">Table 10 on page 16</a> .

[Table 8 on page 16](#) lists the elements in the Timestamp data object of the Path data object in flow path data responses.

**Table 18: Flow Path Analytics JSON Response Data: Timestamp Object Elements**

Element Name	Description
TimeStamp	(Numeric) Seconds portion of the timestamp.
TimeStamp_usec	(Numeric) Microseconds portion of the timestamp.
Type	(String) Timestamp type. Accepted values are "PTP", "LCPU" or "NTP".

[Table 9 on page 16](#) lists the elements in the Ingress Logical Interface Attribute data object, part of the Path data object of flow path data responses.

**Table 19: Flow Path Analytics JSON Response Data: Ingress Logical Interface Attribute Object Elements**

Element Name	Description
IFLList	List of Logical Interface data objects. See <a href="#">Table 13 on page 17</a> .

[Table 10 on page 16](#) lists the elements in the Egress Logical Interface Attribute data object, part of the Path data object of flow path data responses.

**Table 20: Flow Path Analytics JSON Response Data: Egress Logical Interface Attribute Object Elements**

Element Name	Description
L3Ecmp	Layer 3 ECMP data object. See <a href="#">Table 11 on page 17</a> .

[Table 11 on page 17](#) lists the elements in the Layer 3 ECMP data object in the Egress Logical Interface Attribute data object (part of the Path data object) in flow path data responses.

**Table 21: Flow Path Analytics JSON Response Data: Layer 3 ECMP Object Elements**

Element Name	Description
IFLList	List of Logical Interface data objects. See <a href="#">Table 13 on page 17</a> .
NumBuckets	(Numeric) Number of buckets in the ECMP.
L2EcmpList	List of Layer 2 ECMP data objects. See <a href="#">Table 12 on page 17</a> .

[Table 12 on page 17](#) lists the elements in the Layer 2 ECMP List data object within the Layer 3 ECMP data object of flow path data responses.

**Table 22: Flow Path Analytics JSON Response Data: Layer 2 ECMP List Object Elements**

Element Name	Description
AE_IFL_NAME	(String) Aggregated logical interface name.
NumBuckets	(Numeric) Number of buckets in the ECMP.
IFLList	List of Logical Interface data objects. See <a href="#">Table 13 on page 17</a> .

[Table 13 on page 17](#) lists the elements in the Logical Interface data object (part of the Layer 2 ECMP and Layer 3 ECMP data objects) of flow path data responses.

**Table 23: Flow Path Analytics JSON Response Data: Logical Interface Object Elements**

Element Name	Description
IFLName	(String) Interface name.
IFLStats	Interface Packet Statistics data object. See <a href="#">Table 14 on page 17</a> .

[Table 14 on page 17](#) lists the elements in the Interface Packet Statistics data object (part of the Logical Interface data object) of flow path data responses.

**Table 24: Flow Path Analytics JSON Response Data: Interface Packet Statistics Object Elements**

Element Name	Description
TX_PKTS	(Numeric) Total transmitted (Tx) packets on the interface.
RX_PKTS	(Numeric) Total received (Rx) packets on the interface.
TX_UCPKTS	(Numeric) Total number of egress/transmitted unicast packets on the interface.
RX_UCPKTS	(Numeric) Total number of ingress/received unicast packets on the interface.
TX_BCPKTS	(Numeric) Total number of outgoing broadcast packets.

**Table 24: Flow Path Analytics JSON Response Data: Interface Packet Statistics Object Elements** (*continued*)

Element Name	Description
RX_BCPKTS	(Numeric) Total number of incoming broadcast packets.
TX_MCPKTS	(Numeric) Total outgoing multicast packets.
RX_MCPKTS	(Numeric) Total incoming multicast packets.
TX_PPS	(Numeric) Transmit bandwidth (packets per second).
RX_PPS	(Numeric) Receive bandwidth (packets per second).
TX_CRCERROR	(Numeric) Total CRC align errors while transmitting.
RX_CRCERROR	(Numeric) Total CRC align errors while receiving.
TX_DROP_PKT	(Numeric) Cumulative Tx packet drop in all the queues for the interface.
RX_DROP_PKT	(Numeric) Ingress Rx packet drops on the interface. This could be due to a buffer full condition.
TX_BYTES	(Numeric) Total transmit bytes.
RX_BYTES	(Numeric) Total receive bytes.
TX_BW	(Numeric) Transmit bandwidth (bytes per second), the average rate at which packet bytes are transmitted.
RX_BW	(Numeric) Receive bandwidth (bytes per second), the average rate at which packet bytes are received.

### Tunnel Data Compute Agent API Example

This example shows the JSON-format response data returned from a CA API request to retrieve tunnel flow data results for the specified tunnel. See [“GET data/tunnel Resource” on page 42](#) for details on response data elements.

Also see [“GET start/tunnel Resource” on page 38](#) for request parameters you can use to specify different options for the data collection when setting up the tunnel flow.

Start a tunnel flow data collection with these parameters (**GET start/tunnel** request):

```
http://10.94.201.11:8080/start/tunnel?src=10.1.1.1&sport=4567&
vnid=5&dst=10.1.7.1&appdst=10.1.2.5&appdmac=44:1e:a1:02:25:c0&dport=9000&appsrc=10.1.2.3&
appsmac=9c:8e:99:12:88:2e
```

Request collected data for the tunnel flow (**GET data/tunnel** request):

```
http://10.94.201.11:8080/data/tunnel?src=10.1.1.1&sport=4567&vnid=5&
```

dst=10.1.7.1&appdst=10.1.2.5&appsrc=10.1.2.3

Response data:

```
{
  'Path': [
    {
      'Latency': 0.0078845024108886719,
      'DevMEMUtil': 15.0,
      'IngressIFLAttribute': {
        'IFLList': [
          {
            'IFLStats': {
              'RX_PPS': 81293,
              'TX_BW': 356385,
              'RX_BYTES': 67449363756,
              'RX_PKTS': 44520918,
              'TX_BCPKTS': 44,
              'RX_MCPKTS': 6040,
              'TX_PKTS': 3388998,
              'RX_UCPKTS': 44514874,
              'RX_BCPKTS': 4,
              'TX_UCPKTS': 3388954,
              'TX_PPS': 5043,
              'TX_BYTES': 300920892,
              'RX_BW': 123398583
            },
            'IFLName': 'ge-0/0/0.0'
          }
        ]
      },
      'DevCPUUtil': 20.0,
      'TIMESTAMP': {
        'TimeStamp': 1442515460,
        'Type': 'LCPU',
        'TimeStamp_usec': 968290
      },
      'DevName': 'device02-example-01',
      'EgressIFLAttribute': {
        'L3Ecmp': {
          'IFLList': [
            {
              'IFLStats': {
                'TX_BW': 28,
                'RX_BYTES': 43869305,
                'RX_PKTS': 620675,
                'TX_BCPKTS': 32,
                'RX_MCPKTS': 5280,
                'TX_MCPKTS': 4983,
                'TX_PKTS': 7520635,
                'RX_UCPKTS': 615380,
                'RX_BCPKTS': 15,
                'TX_UCPKTS': 7515620,
                'TX_BYTES': 11405041270,
                'RX_BW': 724
              }
            }
          ]
        }
      }
    }
  ]
}
```

```
'IFLName': 'xe-0/0/18.0'
},
{
  'IFLStats': {
    'RX_PPS': 5044,
    'TX_BW': 251,
    'RX_BYTES': 169032019,
    'RX_PKTS': 1816664,
    'TX_BCPKTS': 78,
    'RX_MCPKTS': 19911,
    'TX_MCPKTS': 22989,
    'TX_PKTS': 24828768,
    'RX_UCPKTS': 1796666,
    'RX_BCPKTS': 87,
    'TX_UCPKTS': 24805701,
    'TX_PPS': 1,
    'TX_BYTES': 37577256330,
    'RX_BW': 353791
  },
  'IFLName': 'xe-0/0/8.0'
}
],
'NumBuckets': 3,
'L2EcmpList': [
  {
    'IFLList': [
      {
        'IFLStats': {
          'TX_BW': 123368103,
          'RX_BYTES': 27482374,
          'RX_PKTS': 387466,
          'TX_BCPKTS': 2,
          'TX_MCPKTS': 10,
          'RX_UCPKTS': 387466,
          'TX_PKTS': 7563459,
          'TX_UCPKTS': 7563447,
          'TX_PPS': 81270,
          'TX_BYTES': 11481235302
        },
        'IFLName': 'xe-0/0/14.0'
      },
      {
        'IFLStats': {
          'RX_BYTES': 40382814,
          'RX_PKTS': 569108,
          'TX_BCPKTS': 23,
          'RX_MCPKTS': 5103,
          'TX_MCPKTS': 4995,
          'TX_PKTS': 4618530,
          'RX_UCPKTS': 563982,
          'RX_BCPKTS': 23,
          'TX_UCPKTS': 4613512,
          'TX_BYTES': 7000004258
        },
        'IFLName': 'xe-0/0/16.0'
      }
    ]
  }
]
```

```

        ],
        'AE_IFL_NAME': 'ae0.0',
        'NumBuckets': 2
    }
]
}
},
'HOP': 1,
'DevSerialID': 'TA3714040533'
},
{
    'Latency': 0.015943527221679688,
    'DevMEMUtil': 15.0,
    'IngressIFLAttribute': {
        'IFLList': [
            {
                'IFLStats': {
                    'RX_PPS': 1,
                    'TX_BW': 353914,
                    'RX_BYTES': 37577256586,
                    'RX_PKTS': 24828770,
                    'TX_BCPKTS': 87,
                    'RX_MCPKTS': 22989,
                    'TX_MCPKTS': 19911,
                    'TX_PKTS': 1816849,
                    'RX_UCPKTS': 24805703,
                    'RX_BCPKTS': 78,
                    'TX_UCPKTS': 1796851,
                    'TX_PPS': 5046,
                    'TX_BYTES': 169043989,
                    'RX_BW': 225
                },
                'IFLName': 'xe-0/0/44.0'
            }
        ]
    },
    'DevCPUUtil': 9.0,
    'TIMESTAMP': {
        'TimeStamp': 1442515460,
        'Type': 'LCPU',
        'TimeStamp_usec': 996042
    },
    'DevName': 'device03-example-01',
    'EgressIFLAttribute': {
        'IFLList': [
            {
                'IFLStats': {
                    'RX_PPS': 5045,
                    'TX_BW': 112,
                    'RX_BYTES': 147180897,
                    'RX_PKTS': 1797068,
                    'TX_BCPKTS': 86,
                    'RX_MCPKTS': 23142,
                    'TX_MCPKTS': 19828,
                    'TX_PKTS': 24798650,
                    'RX_UCPKTS': 1773849,

```

```
        'RX_BCPKTS': 77,
        'TX_UCPKTS': 24778736,
        'TX_BYTES': 37575212868,
        'RX_BW': 353243
      },
      'IFLName': 'xe-0/0/94.0'
    }
  ]
},
'HOP': 2,
'DevSerialID': 'VB3714010003'
},
{
  'Latency': 0.016468524932861328,
  'DevMEMUtil': 15.0,
  'IngressIFLAttribute': {
    'IFLList': [
      {
        'IFLStats': {
          'TX_BW': 353331,
          'RX_BYTES': 37575212996,
          'RX_PKTS': 24798651,
          'TX_BCPKTS': 77,
          'RX_MCPKTS': 19828,
          'TX_MCPKTS': 23142,
          'TX_PKTS': 1798630,
          'RX_UCPKTS': 24778737,
          'RX_BCPKTS': 86,
          'TX_UCPKTS': 1775411,
          'TX_PPS': 5047,
          'TX_BYTES': 147290237,
          'RX_BW': 127
        },
        'IFLName': 'xe-0/0/84.0'
      }
    ]
  },
  'DevCPUUtil': 11.0,
  'TIMESTAMP': {
    'TimeStamp': 1442515461,
    'Type': 'LCPU',
    'TimeStamp_usec': 26941
  },
  'DevName': 'device02-example-03',
  'EgressIFLAttribute': {
    'IFLList': [
      {
        'IFLStats': {
          'RX_PPS': 5047,
          'TX_BW': 123374250,
          'RX_BYTES': 232358129,
          'RX_PKTS': 3317472,
          'TX_BCPKTS': 59,
          'RX_UCPKTS': 3317472,
          'TX_PKTS': 44421507,
          'TX_UCPKTS': 44421448,
```

```

        'TX_PPS': 81274,
        'TX_BYTES': 67424575227,
        'RX_BW': 353319
      },
      'IFLName': 'ge-0/0/12.0'
    }
  ]
},
'HOP': 3,
'DevSerialID': 'VB3113350021'
}
],
>ID': "('10.1.1.1', '10.1.7.1', 4567, 5)",
'Time': 1442515433.609627
}

```

**Related  
Documentation**

- [Compute Agent API Resources Overview on page 3](#)
- [Cloud Analytics Engine Feature Guide for the QFX Series](#)

## Retrieving Active Tunnel Flows List

Use the Cloud Analytics Engine CA API active tunnels resource to retrieve information about tunnel flows you started that are currently active.

- [GET active/tunnel Resource on page 51](#)
- [Active Tunnels Compute Agent API Example on page 52](#)

### GET active/tunnel Resource

Use the **GET active/tunnel** CA API to retrieve a list of all active tunnel flows for the client who originated the request.

Request Parameters:

- None

Return Codes:

- **EOK**—CA found and returned active tunnel flows for this client.
- **EFAIL**—CA failed to find or return active tunnel flows for this client.

Response Data:

- **flows**: List of tunnel identifier tuples for all active tunnel flows initiated by this client, in JSON format, with tuple values in order as follows (see descriptions in [Table 25 on page 52](#)):
  - {VTEP src IP, VTEP dst IP, VM dst IP, VM src IP, VXLAN ID, outer source port}

See “[Active Tunnels Compute Agent API Example](#)” on [page 52](#) for a sample request and response.

Table 25: active/tunnel Resource JSON Response Tuple Elements

Tuple Value	Description
VTEP src IP	VTEP source IP address
VTEP dst IP	VTEP destination IP address
VM dst IP	VM destination IP address
VM src IP	VM source IP address
VXLAN ID	Encapsulation VXLAN ID
outer source port	VTEP-provided UDP source port for the VXLAN encapsulation

### Active Tunnels Compute Agent API Example

This example shows the JSON-format response data returned from a CA API request to retrieve all active tunnel flows for the client issuing the request. See [“GET active/tunnel Resource” on page 51](#) for details on response data elements.

Request (GET active/tunnel):

`http://rod-ix-pc-04:8080/active/tunnel`

Response data showing one currently-active tunnel flow:

```
{
  'flows': [
    [
      '10.10.11.11',
      '192.168.5.11',
      '10.0.0.3',
      '10.0.0.2',
      200,
      53812
    ]
  ]
}
```

#### Related Documentation

- [Compute Agent API Resources Overview on page 3](#)
- *Cloud Analytics Engine Feature Guide for the QFX Series*