



---

## IP/MPLSView API Guide



---

Modified: 2018-07-06

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

Juniper Networks, the Juniper Networks logo, Juniper, and Junos are registered trademarks of Juniper Networks, Inc. and/or its affiliates in the United States and other countries. All other trademarks may be 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.

*IP/MPLSView API Guide*

Copyright © 2018 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 <https://www.juniper.net/support/eula/>. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

# Table of Contents

	About the Documentation . . . . .	ix
	Documentation and Release Notes . . . . .	ix
	Documentation Conventions . . . . .	ix
	Documentation Feedback . . . . .	xi
	Requesting Technical Support . . . . .	xii
	Self-Help Online Tools and Resources . . . . .	xii
	Opening a Case with JTAC . . . . .	xii
<b>Chapter 1</b>	<b>Introduction . . . . .</b>	<b>15</b>
	WANDL IP/MPLSView RESTful Interfaces . . . . .	15
	Notes on Architecture . . . . .	15
	Root URL . . . . .	15
<b>Chapter 2</b>	<b>Supported APIs . . . . .</b>	<b>17</b>
	RESTful APIs Supported by WANDL IP/MPLSView . . . . .	17
	User/Tenant API Considerations . . . . .	17
	Networks and Topology . . . . .	18
	Query Parameters . . . . .	18
<b>Chapter 3</b>	<b>Authentication . . . . .</b>	<b>19</b>
	Understanding IP/MPLSView API Authentication . . . . .	19
<b>Chapter 4</b>	<b>User Administration . . . . .</b>	<b>21</b>
	Understanding the IP/MPLSView User Administration API . . . . .	21
	List of Users . . . . .	21
	User Details . . . . .	22
	List of Groups . . . . .	22
	Group Details . . . . .	23
	List of Features . . . . .	23
	Users Examples . . . . .	24
<b>Chapter 5</b>	<b>Networks . . . . .</b>	<b>25</b>
	Understanding the IP/MPLSView Networks API . . . . .	25
	List of Networks . . . . .	26
	Networks Example . . . . .	27
<b>Chapter 6</b>	<b>Nodes . . . . .</b>	<b>29</b>
	Understanding the IP/MPLSView Nodes API . . . . .	29
	List of Nodes . . . . .	29
	Node Details . . . . .	30
	Nodes Example . . . . .	30

<b>Chapter 7</b>	<b>Links</b> .....	<b>33</b>
	Understanding the IP/MPLSView Links API .....	33
	List of Links .....	33
	Link Details .....	34
	Links Example .....	34
<b>Chapter 8</b>	<b>Tunnels (TE-LSPs)</b> .....	<b>37</b>
	Understanding the IP/MPLSView Tunnels API .....	37
	List of TE-LSPs .....	37
	TE-LSP Details .....	38
	TE-LSP Node Tunnels .....	38
	TE-LSP Node Tunnel Details .....	39
	TE-LSPs Example .....	40
<b>Chapter 9</b>	<b>Interfaces</b> .....	<b>41</b>
	Understanding the IP/MPLSView Interfaces API .....	41
	List of Interfaces with Node Information .....	41
	List of Interfaces on a Node .....	42
	Node Interface Details .....	42
	Interfaces Example .....	43
<b>Chapter 10</b>	<b>Performance Management</b> .....	<b>45</b>
	Understanding the IP/MPLSView Performance Management API .....	45
	Categories .....	46
	Parameters .....	46
	Performance Management Query Parameters .....	47
	Supported Performance Management Parameters .....	47
	List of Performance Management Categories .....	47
	System Uptime Information .....	48
	CPU Temperature Information .....	48
	CPU Utilization Information .....	49
	Memory Usage Information .....	49
	Ping .....	50
	Advanced Ping .....	50
	LSP Ping .....	51
	SLA .....	51
	Link Latency .....	51
	Performance Management Examples .....	52
	System Uptime .....	52
	System Uptime from Specified Date .....	52
	System Uptime with Date and Offset .....	52
<b>Chapter 11</b>	<b>Hardware Inventory</b> .....	<b>55</b>
	Understanding the IP/MPLSView Hardware Inventory API .....	55
	Supported Hardware Inventory Categories .....	55
	List of Hardware Inventory Categories .....	56
	List of Hardware Inventory Category Dates .....	56
	Hardware Inventory for a Category on a Specific Date .....	57

---

	Hardware Inventory Examples . . . . .	57
	Hardware Inventory . . . . .	58
	Router Information . . . . .	58
	CAPEX with Date . . . . .	58
	Card Information with Date . . . . .	58
	Device Usage with Date . . . . .	59
	Extensive Part Information with Date . . . . .	59
	IC with Date . . . . .	59
	Interface with Date . . . . .	60
	Line Card Usage with Date . . . . .	60
	Miscellaneous Part with Date . . . . .	60
	Parts with Date . . . . .	60
	Router with Date . . . . .	61
	Transceiver with Date . . . . .	61
<b>Chapter 12</b>	<b>Traffic Data . . . . .</b>	<b>63</b>
	Understanding the IP/MPLSView Traffic Data API . . . . .	63
	Traffic Data Query Parameters . . . . .	65
	List of Available Nodes Traffic Data . . . . .	65
	Router Traffic Data Details . . . . .	65
	Interface Traffic Data Details . . . . .	66
	Tunnel Traffic Data Details . . . . .	67
	VPN Traffic Data Details . . . . .	68
	Traffic Collection Schema . . . . .	68
	Network Interface Traffic Data Information . . . . .	79
	Router Network Interface Traffic Data Details . . . . .	79
	Class of Service Traffic Statistics Data Details . . . . .	80
	Multicast Traffic Statistics Data Details . . . . .	81
	Traffic Data Examples . . . . .	82



# List of Tables

	<b>About the Documentation</b> . . . . .	<b>ix</b>
	Table 1: Notice Icons . . . . .	x
	Table 2: Text and Syntax Conventions . . . . .	x
<b>Chapter 2</b>	<b>Supported APIs</b> . . . . .	<b>17</b>
	Table 3: API Categories . . . . .	17
<b>Chapter 4</b>	<b>User Administration</b> . . . . .	<b>21</b>
	Table 4: Administration-Related APIs . . . . .	21
	Table 5: List of Users – Request Parameters . . . . .	22
	Table 6: User Details – Request Parameters . . . . .	22
	Table 7: List of Groups – Request Parameters . . . . .	23
	Table 8: Group – Request Parameters . . . . .	23
	Table 9: Features – Request Parameters . . . . .	23
<b>Chapter 5</b>	<b>Networks</b> . . . . .	<b>25</b>
	Table 10: Topology-Related API Categories . . . . .	25
	Table 11: Networks – Request Parameters . . . . .	26
<b>Chapter 6</b>	<b>Nodes</b> . . . . .	<b>29</b>
	Table 12: List of Nodes – Request Parameters . . . . .	29
	Table 13: Node – Request Parameters . . . . .	30
<b>Chapter 7</b>	<b>Links</b> . . . . .	<b>33</b>
	Table 14: List of Links – Request Parameters . . . . .	33
	Table 15: Link – Request Parameters . . . . .	34
<b>Chapter 8</b>	<b>Tunnels (TE-LSPs)</b> . . . . .	<b>37</b>
	Table 16: List of TE-LSPs – Request Parameters . . . . .	37
	Table 17: TE-LSP – Request Parameters . . . . .	38
	Table 18: TE-LSP Node Tunnels – Request Parameters . . . . .	39
	Table 19: TE-LSP Node Tunnel Details – Request Parameters . . . . .	39
<b>Chapter 9</b>	<b>Interfaces</b> . . . . .	<b>41</b>
	Table 20: Interface with Node – Request Parameters . . . . .	41
	Table 21: Interfaces on a Node – Request Parameters . . . . .	42
	Table 22: Node Interface – Request Parameters . . . . .	42
<b>Chapter 10</b>	<b>Performance Management</b> . . . . .	<b>45</b>
	Table 23: Performance Management-Related APIs . . . . .	45
	Table 24: Performance Management Categories – Request Parameters . . . . .	48
	Table 25: System Update – Request Parameters . . . . .	48
	Table 26: CPU Temperature – Request Parameters . . . . .	48

	Table 27: CPU Utilization – Request Parameters . . . . .	49
	Table 28: Memory Usage – Request Parameters . . . . .	49
	Table 29: Request Parameters . . . . .	50
	Table 30: Advanced Ping – Request Parameters . . . . .	50
	Table 31: LSP Ping – Request Parameters . . . . .	51
	Table 32: SLA – Request Parameters . . . . .	51
	Table 33: Link Latency – Request Parameters . . . . .	52
<b>Chapter 11</b>	<b>Hardware Inventory . . . . .</b>	<b>55</b>
	Table 34: Hardware Inventory-Related APIs . . . . .	55
	Table 35: Hardware Inventory – Request Parameters . . . . .	56
	Table 36: Hardware Inventory Category Dates – Request Parameters . . . . .	57
	Table 37: Hardware Inventory Category on a Specific Date – Request Parameters . . . . .	57
<b>Chapter 12</b>	<b>Traffic Data . . . . .</b>	<b>63</b>
	Table 38: Traffic Data Information-Related APIs . . . . .	63
	Table 39: Available Nodes Traffic Data – Request Parameters . . . . .	65
	Table 40: Router Traffic Data – Request Parameters . . . . .	66
	Table 41: Interface Traffic Data – Request Parameters . . . . .	66
	Table 42: Request Parameters . . . . .	67
	Table 43: Request Parameters . . . . .	79
	Table 44: Router Network Interface Traffic Data – Request Parameters . . . . .	79
	Table 45: Class of Service Traffic Statistics Data – Request Parameters . . . . .	80
	Table 46: Multicast Traffic Statistics Data – Request Parameters . . . . .	81



# About the Documentation

- Documentation and Release Notes on page ix
- Documentation Conventions on page ix
- Documentation Feedback on page xi
- Requesting Technical Support on page xii

## 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 <https://www.juniper.net/documentation/>.

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 <https://www.juniper.net/books>.

## Documentation Conventions

---

Table 1 on page x 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 x 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>
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>

Table 2: Text and Syntax Conventions (continued)

Convention	Description	Examples
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 <b>[edit protocols ospf area area-id]</b> hierarchy level.</li><li>The console port is labeled <b>CONSOLE</b>.</li></ul>
< > (angle brackets)	Encloses optional keywords or variables.	<b>stub &lt;default-metric <i>metric</i>&gt;;</b>
(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   multicast</b>  <b>(<i>string1</i>   <i>string2</i>   <i>string3</i>)</b>
# (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 [ <i>community-ids</i> ]</b>
Indentation and braces ( { } )	Identifies a level in the configuration hierarchy.	<pre>[edit] routing-options {   static {     route default {       nexthop <i>address</i>;       retain;     }   } }</pre>
;(semicolon)	Identifies a leaf statement at a configuration hierarchy level.	
GUI Conventions		
Bold text like this	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>
> (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 <https://www.juniper.net/documentation/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 <https://www.juniper.net/documentation/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 <https://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf>.
- Product warranties—For product warranty information, visit <https://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: <https://www.juniper.net/customers/support/>
- Search for known bugs: <https://prsearch.juniper.net/>
- Find product documentation: <https://www.juniper.net/documentation/>
- Find solutions and answer questions using our Knowledge Base: <https://kb.juniper.net/>
- Download the latest versions of software and review release notes: <https://www.juniper.net/customers/csc/software/>
- Search technical bulletins for relevant hardware and software notifications: <https://kb.juniper.net/InfoCenter/>
- Join and participate in the Juniper Networks Community Forum: <https://www.juniper.net/company/communities/>
- Open a case online in the CSC Case Management tool: <https://www.juniper.net/cm/>

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

## 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 <https://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 <https://www.juniper.net/support/requesting-support.html>.



## CHAPTER 1

# Introduction

- [WANDL IP/MPLSView RESTful Interfaces on page 15](#)

## WANDL IP/MPLSView RESTful Interfaces

---

WANDL IP/MPLSView RESTful interfaces allow you to work with data and accomplish many of the tasks that are available through both the application's graphical user interface (GUI) and CLI.

- [Notes on Architecture on page 15](#)
- [Root URL on page 15](#)

### Notes on Architecture

The IP/MPLSView application server uses Node.js to receive and process REST requests over HTTP using port 3000 (IP/MPLSView uses port 8091).



**NOTE:** IP/MPLSView requires that the database be running for full functionality.

### Root URL

After authentication, all REST URLs described in this document start at `/IPMPLSView/API/v1/tenant/1`.

#### Related Documentation

- [RESTful APIs Supported by WANDL IP/MPLSView on page 17](#)





## CHAPTER 2

# Supported APIs

- [RESTful APIs Supported by WANDL IP/MPLSView on page 17](#)

### RESTful APIs Supported by WANDL IP/MPLSView

---

The following is a high-level list of the RESTful APIs supported by WANDL IP/MPLSView. Network topology is broken out into separate API categories due to its size.

[Table 3 on page 17](#) list the API categories.

*Table 3: API Categories*

API	URL
User administration	<code>/IPMPLSView/API/v1/tenant/1/UserAdmin</code>
Networks	<code>/IPMPLSView/API/v1/tenant/1/topology/</code>
Nodes	<code>/IPMPLSView/API/v1/tenant/1/topology/1/nodes</code>
Links	<code>/IPMPLSView/API/v1/tenant/1/topology/1/links</code>
Tunnels	<code>/IPMPLSView/API/v1/tenant/1/topology/1/te-lsps</code>
Interfaces	<code>/IPMPLSView/API/v1/tenant/1/topology/1/interfaces</code>
Performance management	<code>/IPMPLSView/API/v1/tenant/1/pm</code>
Hardware inventory	<code>/IPMPLSView/API/v1/tenant/1/hardwareInventory</code>
Traffic data	<code>/IPMPLSView/API/v1/tenant/1/trafficData</code>

### User/Tenant API Considerations

Each application and user has an ID for accessing the API. This user ID value is fixed at 1 (one).

## Networks and Topology

Users and applications have access to several networks. The topology ID for the live network is 1 (one).

While IP/MPLSView supports a number of topologies, data collection is only performed on the live network.

## Query Parameters

While some query parameters, such as the tenant and topology ID, are expressed simply, others are URI parameters that are part of the URL path and in the format:

`?<parameter>=<value>`. For example, when a date parameter can be used, it is expressed as `?date=<date>`, where date is in *yyyymmdd* format, such as `?date=20150917`.

When applicable, these are called out in the API descriptions.

### Related Documentation

- [WANDL IP/MPLSView RESTful Interfaces on page 15](#)

## CHAPTER 3

# Authentication

- [Understanding IP/MPLSView API Authentication on page 19](#)

### Understanding IP/MPLSView API Authentication

Before accessing the REST interface, you must be authenticated using the IP/MPLSView login procedure.

To ensure a properly authenticated session:

1. Keep the session cookie JSESSIONID across requests.
2. Make a GET request to `/start.jsp` to create a session cookie.
3. Make a POST request to `/wandel/servlet/LoginServlet` as follows:
  - Set the Referrer header to `http://<server:port>/wandel/jsp/webMain.jsp`.
  - Send the **JSESSIONID** cookie.
  - Set the content-type to **application/x-www-form-urlencoded**.
  - The data should be:  
`userid=<user>`  
`passwd=<password>`
4. With valid credentials, the subsequent requests using the JSESSIONID cookie are authorized.

#### Related Documentation

- [WANDL IP/MPLSView RESTful Interfaces on page 15](#)



## CHAPTER 4

# User Administration

- [Understanding the IP/MPLSView User Administration API on page 21](#)
- [List of Users on page 21](#)
- [User Details on page 22](#)
- [List of Groups on page 22](#)
- [Group Details on page 23](#)
- [List of Features on page 23](#)
- [Users Examples on page 24](#)

## Understanding the IP/MPLSView User Administration API

The Users API root is `/IPMPLSView/API/v1/tenant/1/UserAdmin`. It only allows GET requests. [Table 4 on page 21](#) lists the user administration-related APIs.

*Table 4: Administration-Related APIs*

Object	URL	Description
Users	<code>/users</code>	List of users
User	<code>/users/&lt;user-cn&gt;</code>	A specific user, where <i>user-cn</i> is the user <i>cn</i> field
Groups	<code>/groups</code>	Return the list of groups
Group	<code>/groups/&lt;group-cn&gt;</code>	A specific group, where <i>group-cn</i> is the group <i>cn</i> field
Features	<code>/features</code>	The list of user administration features

- Related Documentation**
- [WANDL IP/MPLSView RESTful Interfaces on page 15](#)
  - [RESTful APIs Supported by WANDL IP/MPLSView on page 17](#)

## List of Users

Retrieves the full list of users. For example:

`v1/{tenant_id}/UserAdmin/users`

Normal response codes: 200

[Table 5 on page 22](#) lists the request parameters.

**Table 5: List of Users – Request Parameters**

Parameter	Style	Type	Description
<code>tenant_id</code>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.

**Related Documentation**

- [Understanding the IP/MPLSView User Administration API on page 21](#)

## User Details

Retrieves details for a specific user. For example:

`v1/{tenant_id}/UserAdmin/usersv1/{tenant_id}/UserAdmin/users/<user_cn>`

Normal response codes: 200

[Table 6 on page 22](#) lists the request parameters.

**Table 6: User Details – Request Parameters**

Parameter	Style	Type	Description
<code>tenant_id</code>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<code>&lt;user_cn&gt;</code>			The specific user for which information is requested, where <code>user_cn</code> is the user <code>cn</code> field.

**Related Documentation**

- [Understanding the IP/MPLSView User Administration API on page 21](#)

## List of Groups

Retrieves the full list of groups. For example:

`v1/{tenant_id}/UserAdmin/groups`

Normal response codes: 200

[Table 7 on page 23](#) lists the request parameters.

**Table 7: List of Groups – Request Parameters**

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.

**Related Documentation**

- [Understanding the IP/MPLSView User Administration API on page 21](#)

## Group Details

Retrieves details for a specific group. For example:

```
v1/{tenant_id}/UserAdmin/groups/<group_cn>
```

Normal response codes: 200

[Table 8 on page 23](#) lists the request parameters.

**Table 8: Group – Request Parameters**

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>&lt;group_cn&gt;</i>			The specific user for which information is requested, where <i>&lt;group_cn&gt;</i> is the group <i>cn</i> field.

**Related Documentation**

- [Understanding the IP/MPLSView User Administration API on page 21](#)

## List of Features

Retrieves the full list of features. For example:

```
v1/{tenant_id}/UserAdmin/features
```

Normal response codes: 200

[Table 9 on page 23](#) lists the request parameters.

**Table 9: Features – Request Parameters**

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.

**Related Documentation**

- [Understanding the IP/MPLSView User Administration API on page 21](#)

## Users Examples

---

### User

Details of a user.

```
{ "dn": "cn=username,ou=users,dc=wandl,dc=com", "controls": [], "dateModified": "1380549792743", "uid": "username", "type": "Full"
```

```
Access", "objectClass": "wandlUser", "dateCreated": "1380549792743", "sn": "Administrators", "cn": "username", "lastLogin": "1380751247424", "lastIpAddress": "192.10.21.135", "fullControl": "Full Control" }
```

### User Dates

Dates are specified in UNIX timestamp format.

### Group

Details of a group.

```
{ "dn": "cn=groupName,ou=groups,dc=wandl,dc=com", "controls": [], "perms": "101=0,102=0,103=0,104=0,105=0,106=0,107=0,108=0,201=0,202=0,203=0,204=0,205=0,206=0,207=0,208=0,209=0,301=0,302=0,303=0,304=0,305=0,306=0,307=0,308=0,309=0,381=0,382=0,387=0,388=0,389=0,383=0,384=0,385=0,386=0,401=0,402=0,403=0,501=0,502=0,503=0,504=0,505=0,506=0,507=0,601=0,602=0,603=0,604=0,605=0,606=0,607=0,608=0,701=0,702=0,703=0,704=0,705=0,706=0,707=0,708=0,801=0,802=0,803=0,804=0,805=0,806=0,807=0,808=0,809=0,811=0,812=0,813=0,824=0,814=0,815=0,816=0,817=0,818=0,826=0,819=0,825=0,821=0,822=0,823=0, ", "type": "Web Portal", "objectClass": "wandlGroup", "cn": " groupName" }
```

### Group Permissions

Comma-separated list of **feature-id=state**, containing the list of features for that group.

```
{ "dn": "id=100,ou=features,dc=wandl,dc=com", "controls": [], "objectClass": "feature", "id": "100", "name": "Desktop Client [All]", "mod": "1", "description": "These items apply to the Desktop Client for both offline and live network models." }
```

### Related Documentation

- [Understanding the IP/MPLSView User Administration API on page 21](#)



## CHAPTER 5

# Networks

- [Understanding the IP/MPLSView Networks API on page 25](#)
- [List of Networks on page 26](#)
- [Networks Example on page 27](#)

### Understanding the IP/MPLSView Networks API

The networks API root is **/IPMPLSView/API/v1/tenant/1/topology**. It only allows GET requests. The live network is 1. Traffic data collection is only collected for the live network.

The APIs related to nodes, links, tunnels, and interfaces are documented separately. [Table 10 on page 25](#) lists the network topology-related API categories.

*Table 10: Topology-Related API Categories*

Object	URL	Description
Topology List	/	Returns network topology information, including a list of network topologies.
Topology Views	/ <b>&lt;topo_id&gt;</b>	Returns the network topology of a view.
Nodes	/ <b>&lt;topo_id&gt;/nodes</b>	Returns network node information.
Node	/ <b>&lt;topo_id&gt;/nodes/&lt;node_id&gt;</b>	Returns details of a node.
Links	/ <b>&lt;topo_id&gt;/links</b>	Returns network link information.
Link	/ <b>&lt;topo_id&gt;/links/&lt;link_id&gt;</b>	Returns details of a link.
Tunnels	/ <b>&lt;topo_id&gt;/te-lsps</b>	Returns information about network tunnels.
Tunnels per Node	/ <b>&lt;topo_id&gt;/te-lsps/&lt;node_name&gt;</b>	Returns information about tunnels on a node.
Single Tunnel	/ <b>&lt;topo_id&gt;/te-lsps/&lt;node_name&gt;/&lt;lsp_id&gt;</b>	Returns details of a tunnel.
Interfaces	/ <b>&lt;topo_id&gt;/interfaces</b>	Returns details about network interfaces.

**Table 10: Topology-Related API Categories (continued)**

Object	URL	Description
Interfaces per Node	<code>/&lt;topo_id&gt;/interfaces/&lt;node_name&gt;</code>	Returns information about interfaces on a node.
Single Interface	<code>/&lt;topo_id&gt;/interfaces/&lt;node_name&gt;/&lt;interface_name&gt;</code>	Returns information about an interface.
Facilities	<code>/&lt;topo_id&gt;/facilities</code>	Returns details about network facilities.
specBridge	<code>/&lt;topo_id&gt;/specBridge</code>	Returns a list of legacy spec files available in JSON format.
Groups	<code>/&lt;topo_id&gt;/specBridge/groups</code>	Returns information about network groups.
Group	<code>/&lt;topo_id&gt;/specBridge/groups/&lt;group_name&gt;</code>	Returns details of a group, including group members.
Sites	<code>/&lt;topo_id&gt;/specBridge/sites</code>	Returns information about network sites.
Site	<code>/&lt;topo_id&gt;/specBridge/sites/&lt;site_name&gt;</code>	Returns details of a site, including site members.

**Related Documentation**

- [WANDL IP/MPLSView RESTful Interfaces on page 15](#)
- [RESTful APIs Supported by WANDL IP/MPLSView on page 17](#)

## List of Networks

Retrieves the network topology. For example:

```
v1/tenant/{tenant_id}/topology/
```

Normal response codes: 200

[Table 11 on page 26](#) lists the request parameters.

**Table 11: Networks – Request Parameters**

Parameter	Style	Type	Description
<code>tenant_id</code>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.

**Related Documentation**

- [Understanding the IP/MPLSView Networks API on page 25](#)

## Networks Example

---

```
[{"name":"livenetwork","topoObjectType":"topology","topologyIndex":1,"description":"","path":"/home/wandl/data/.network","runcode":"x","owner":"","modified":"2015-09-09T17:43:57.597Z","accessList":"*","links":[{"rel":"self","href":"1"}]}
```

- Related Documentation**
- [Understanding the IP/MPLSView Networks API on page 25](#)



## CHAPTER 6

# Nodes

- [Understanding the IP/MPLSView Nodes API on page 29](#)
- [List of Nodes on page 29](#)
- [Node Details on page 30](#)
- [Nodes Example on page 30](#)

### Understanding the IP/MPLSView Nodes API

---

The nodes API performs operations on nodes. The nodes API root is `/IPMPLSView/API/v1/tenant/1/topology/1/nodes`, and only allows GET requests.

The corresponding schema is `node.json`.

#### Related Documentation

- [WANDL IP/MPLSView RESTful Interfaces on page 15](#)
- [RESTful APIs Supported by WANDL IP/MPLSView on page 17](#)

### List of Nodes

---

Retrieves the full list of nodes.

For example:

`v1/{tenant_id}/topology/{topology_id}/nodes`

Normal response codes: 200

[Table 12 on page 29](#) lists the request parameters.

**Table 12: List of Nodes – Request Parameters**

Parameter	Style	Type	Description
<code>tenant_id</code>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<code>topology_id</code>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1.



NOTE: This operation does not accept a request body.

#### Related Documentation

- [Understanding the IP/MPLSView Nodes API on page 29](#)

## Node Details

Retrieve the details for a single node. For example:

```
v1/{tenant_id}/topology/{topology_id}/nodes/{nodeIndex}
```

Normal response codes: 200

Table 13 on page 30 lists the request parameters.

Table 13: Node – Request Parameters

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1.
<i>nodeIndex</i>	URI	xsd:int	The unique node index.



NOTE: This operation does not accept a request body.

#### Related Documentation

- [Understanding the IP/MPLSView Nodes API on page 29](#)

## Nodes Example

Following is a GET-only data enhanced example:

```
{
  "topoObjectType": "node",
  "topologyIndex": 1,
  "group": "SMALL",
  "protocols": {
    "OSPF": {},
    "BGP": {},
    "VPN": {}
  },
  "management": {
    "address": "172.16.0.108",
    "operatingSystem": "JUNOS",
    "operatingSystemVersion": "12.1X44.3",
    "vendor": "JUNIPER",
    "ISIS": {
      "isoAddress": "0622.0000.1008"
    },
    "nodeType": "JUNIPER",
    "name": "8_LYON",
    "nodeIndex": 1,
    "nodeId": "62.200.0.8",
    "ipv6": {
      "topoObjectType": "ipv6",
      "address": "4004:62:200::8",
      "hostName": "8_LYON",
      "source": {
        "file": "172.16.0.108.8_LYON.cfg"
      },
      "multicastRtAddress": {
        "topoObjectType": "ipv4",
        "address": "62.200.0.255"
      },
      "links": [
        {
          "rel": "self",
          "href": "nodes/1"
        },
        {
          "rel": "interfaces",
          "href": "interfaces/8_LYON"
        }
      ]
    }
  }
}
```

**Related Documentation** • [Understanding the IP/MPLSView Nodes API on page 29](#)





## CHAPTER 7

# Links

- [Understanding the IP/MPLSView Links API on page 33](#)
- [List of Links on page 33](#)
- [Link Details on page 34](#)
- [Links Example on page 34](#)

### Understanding the IP/MPLSView Links API

---

The links API performs operations on links. The links API root is `/IPMPLSView/API/v1/tenant/1/topology/1/links`. It only allows GET requests.

The corresponding schema is `link.json`.

#### Related Documentation

- [WANDL IP/MPLSView RESTful Interfaces on page 15](#)
- [RESTful APIs Supported by WANDL IP/MPLSView on page 17](#)

### List of Links

---

Retrieves the full list of links. For example:

```
v1/{tenant_id}/topology/topology_id/links
```

Normal response codes: 200

[Table 14 on page 33](#) lists the request parameters.

*Table 14: List of Links – Request Parameters*

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1.



**NOTE:** This operation does not accept a request body.

#### Related Documentation

- [Understanding the IP/MPLSView Links API on page 33](#)

## Link Details

Retrieve the details for a link. For example:

```
v1/{tenant_id}/topology/{topology_id}/links/{linkIndex}
```

Normal response codes: 200

[Table 15 on page 34](#) lists the request parameters.

**Table 15: Link – Request Parameters**

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1.
<i>linkIndex</i>	URI	xsd:int	The unique link index.



**NOTE:** This operation does not accept a request body.

#### Related Documentation

- [Understanding the IP/MPLSView Links API on page 33](#)

## Links Example

Following is a GET-only data enhanced example:

```
{
  "topoObjectType": "link",
  "topologyIndex": 1,
  "linkIndex": 1,
  "name": "8_LYON_ge_0/0/0.0",
  "endA": {
    "name": "ge-0/0/0.0",
    "ipv4Address": {
      "topoObjectType": "ipv4",
      "prefix": "24",
      "address": "172.16.0.108",
      "node": {
        "topoObjectType": "node",
        "name": "8_LYON",
        "topologyIndex": 1,
        "nodeIndex": 1,
        "links": [
          {
            "rel": "self",
            "href": "nodes/1"
          }
        ]
      }
    },
    "topoObjectType": "interface",
    "topologyIndex": 1,
    "bandwidth": "1.0G",
    "endZ": {
      "node": {
        "topoObjectType": "node",
        "name": "E172.16.0.0",
        "topologyIndex": 1,
        "nodeIndex": 110,
        "links": [
          {
            "rel": "self",
            "href": "nodes/110"
          }
        ]
      },
      "topoObjectType": "interface",
      "topologyIndex": 1,
      "MTU": 1500,
      "links": [
        {
          "rel": "self",
          "href": "links/1"
        }
      ]
    }
  }
}
```

- Related Documentation**
- [Understanding the IP/MPLSView Links API on page 33](#)



CHAPTER 8

# Tunnels (TE-LSPs)

- [Understanding the IP/MPLSView Tunnels API on page 37](#)
- [List of TE-LSPs on page 37](#)
- [TE-LSP Details on page 38](#)
- [TE-LSP Node Tunnels on page 38](#)
- [TE-LSP Node Tunnel Details on page 39](#)
- [TE-LSPs Example on page 40](#)

## Understanding the IP/MPLSView Tunnels API

The TE-LSPs API performs operations on tunnels (TE-LSPs). The TE-LSPs API root is `/IPMPLSView/API/v1/tenant/1/topology/1/te-lsps`. It only allows GET requests.

The corresponding schema is `lsp.json`.

- Related Documentation
- [WANDL IP/MPLSView RESTful Interfaces on page 15](#)
  - [RESTful APIs Supported by WANDL IP/MPLSView on page 17](#)

## List of TE-LSPs

Retrieves the full list of links. For example:

`v1/{tenant_id}/topology/{topology_id}/te-lsps`

Normal response codes: 200

Table 16 on page 37 lists the request parameters.

Table 16: List of TE-LSPs – Request Parameters

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently, it is fixed to 1.

Table 16: List of TE-LSPs – Request Parameters (continued)

Parameter	Style	Type	Description
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently, it is fixed to 1.



**NOTE:** This operation does not accept a request body.

#### Related Documentation

- [Understanding the IP/MPLSView Tunnels API on page 37](#)

## TE-LSP Details

Retrieves the details for a TE-LSP. For example:

`v1/{tenant_id}/topology/{topology_id}/te-lsp/{ISPIndex}`

Normal response codes: 200

[Table 17 on page 38](#) lists the request parameters.

Table 17: TE-LSP - Request Parameters

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1.
<i>ISPIndex</i>	URI	xsd:int	The unique TE-LSP index.



**NOTE:** This operation does not accept a request body.

#### Related Documentation

- [Understanding the IP/MPLSView Tunnels API on page 37](#)

## TE-LSP Node Tunnels

Retrieves the details for tunnels on a node. For example:

`v1/{tenant_id}/topology/{topology_id}/te-lsp/<node_name>`

Normal response codes: 200

Table 18 on page 39 lists the request parameters.

**Table 18: TE-LSP Node Tunnels – Request Parameters**

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1.
<i>node_name</i>	URI	xsd:int	The unique name of a node.



**NOTE:** This operation does not accept a request body.

**Related Documentation**

- [Understanding the IP/MPLSView Tunnels API on page 37](#)

## TE-LSP Node Tunnel Details

Retrieves the details for a specified tunnel on a node. For example:

```
v1/{tenant_id}/topology/{topology_id}/te-lsp/<node_name>/<lsp_id>
```

Normal response codes: 200

Table 19 on page 39 lists the request parameters.

**Table 19: TE-LSP Node Tunnel Details – Request Parameters**

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1.
<i>node_name</i>	URI	xsd:int	The unique name of a node.
<i>lsp_id</i>	URI	xsd:int	The unique identifier of an tunnel.



**NOTE:** This operation does not accept a request body.

- Related Documentation**
- [Understanding the IP/MPLSView Tunnels API on page 37](#)

## TE-LSPs Example

---

```
{
  "topoObjectType": "lsp",
  "topologyIndex": 1,
  "pathType": "primary",
  "name": "testlsp",
  "lspIndex": 1,
  "from": {
    "topoObjectType": "node",
    "name": "8_LYON",
    "topologyIndex": 1,
    "nodeIndex": 1,
    "links": [
      {
        "rel": "self",
        "href": "nodes/1"
      }
    ],
    "to": {
      "topoObjectType": "ipv4",
      "address": "62.200.0.1",
      "plannedProperties": {
        "bandwidth": 0,
        "setupPriority": 7,
        "holdingPriority": 7,
        "design": {
          "adminGroups": {
            "attributeIncludeAny": 10240
          }
        }
      }
    },
    "links": [
      {
        "rel": "self",
        "href": "te-lsps/1"
      }
    ]
  }
}
```

- Related Documentation**
- [Understanding the IP/MPLSView Tunnels API on page 37](#)



CHAPTER 9

# Interfaces

- [Understanding the IP/MPLSView Interfaces API on page 41](#)
- [List of Interfaces with Node Information on page 41](#)
- [List of Interfaces on a Node on page 42](#)
- [Node Interface Details on page 42](#)
- [Interfaces Example on page 43](#)

## Understanding the IP/MPLSView Interfaces API

The interfaces API provides access to interface data for nodes. The interfaces API root is `/IPMPLSView/API/v1/tenant/1/topology/1/interfaces`. It only allows GET requests.

By design, the set of all interfaces in the network is not provided due to its size and performance impact. As a result, the API requires per-node access.

The corresponding JSON schema file is `link.json#/definitions/interfaceConfiguration`.

- Related Documentation
- [WANDL IP/MPLSView RESTful Interfaces on page 15](#)
  - [RESTful APIs Supported by WANDL IP/MPLSView on page 17](#)

## List of Interfaces with Node Information

Returns the list of nodes providing interface information. For example:

`v1/{tenant_id}/topology/{topology_id}/interfaces`

Normal response codes: 200

[Table 20 on page 41](#) lists the request parameters.

Table 20: Interface with Node – Request Parameters

Parameter	Style	Type	Description
<code>tenant_id</code>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.

**Table 20: Interface with Node – Request Parameters (continued)**

Parameter	Style	Type	Description
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1.

**Related Documentation** • [Understanding the IP/MPLSView Interfaces API on page 41](#)

## List of Interfaces on a Node

Returns the list of interfaces on a given node. For example:

```
v1/{tenant_id}/topology/{topology_id}/interfaces/{nodeName}
```

[Table 21 on page 42](#) lists the request parameters.

**Table 21: Interfaces on a Node – Request Parameters**

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1.
<i>nodeName</i>	URI	xsd:string	The node name, expressed as <b>?nodeName=&lt;value&gt;</b> .

**Related Documentation** • [Understanding the IP/MPLSView Interfaces API on page 41](#)

## Node Interface Details

Returns details for a specific interface on a given node. For example:

```
v1/{tenant_id}/topology/{topology_id}/interfaces/{nodeName}/{interfaceName}
```

[Table 22 on page 42](#) lists the request parameters.

**Table 22: Node Interface – Request Parameters**

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.

Table 22: Node Interface – Request Parameters (continued)

Parameter	Style	Type	Description
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1.
<i>nodeName</i>	URI	xsd:string	The node name, expressed as <b>?nodeName=&lt;value&gt;</b> .
<i>interfaceName</i>	URI	xsd:string	The interface name, expressed as <b>?interfaceName=&lt;value&gt;</b> .

**Related Documentation** • [Understanding the IP/MPLSView Interfaces API on page 41](#)

## Interfaces Example

```
{
  "topoObjectType": "interfaceConfiguration",
  "topologyIndex": 1,
  "node": {
    "topoObjectType": "node",
    "name": "13_MILAN",
    "topologyIndex": 1,
    "nodeIndex": 11,
    "links": [
      {
        "rel": "self",
        "href": "nodes/11"
      }
    ],
    "name": "GigabitEthernet0/0/0.563",
    "ipv4Address": {
      "topoObjectType": "ipv4",
      "prefix": "30",
      "address": "62.200.163.2"
    },
    "bandwidth": "1.0G",
    "ethernet": {
      "MTU": 1500,
      "MACAddress": "00:50:56:94:09:CD",
      "VLANId": [563]
    },
    "adminStatus": "Active",
    "type": "gigabitEthernet",
    "protocols": {
      "OSPF": {
        "area": 40
      },
      "LDP": {},
      "RSVP": {},
      "MPLSTE": {}
    },
    "links": [
      {
        "rel": "self",
        "href": "13_MILAN/GigabitEthernet0/0/0.563"
      }
    ]
  }
}
```

**Related Documentation** • [Understanding the IP/MPLSView Interfaces API on page 41](#)



## CHAPTER 10

# Performance Management

- [Understanding the IP/MPLSView Performance Management API on page 45](#)
- [Performance Management Query Parameters on page 47](#)
- [List of Performance Management Categories on page 47](#)
- [System Uptime Information on page 48](#)
- [CPU Temperature Information on page 48](#)
- [CPU Utilization Information on page 49](#)
- [Memory Usage Information on page 49](#)
- [Ping on page 50](#)
- [Advanced Ping on page 50](#)
- [LSP Ping on page 51](#)
- [SLA on page 51](#)
- [Link Latency on page 51](#)
- [Performance Management Examples on page 52](#)

## Understanding the IP/MPLSView Performance Management API

---

The performance management API root is `/IPMPLSView/API/v1/tenant/1/pm/1`. It only allows GET requests. Performance management is valid for the live network only. The live network nodes and performance management nodes are disjointed, but they may intersect.

[Table 23 on page 45](#) lists the performance management-related APIs.

**Table 23: Performance Management-Related APIs**

Object	URL	Description
Categories	/	Lists the available performance management categories.

---

Table 23: Performance Management-Related APIs (continued)

Object	URL	Description
System uptime	<code>/devicePerformance/systemUptime</code>	Returns system uptime information. <ul style="list-style-type: none"> <li>• No parameters: available dates</li> <li>• from, to: start/end date in <i>yyyymmdd</i></li> <li>• router: router filtering</li> <li>• offset: beginning of data - line number</li> <li>• limit: number of lines to be returned, -1 for end of data</li> </ul>
CPU temperature	<code>/devicePerformance/CPUTemp</code>	Returns CPU temperature information.
CPU usage	<code>/devicePerformance/CPUUsage</code>	Returns CPU usage (load) data.
Memory usage	<code>/devicePerformance/memoryUsage</code>	Returns memory usage data.
Ping	<code>/networkPerformance/ping</code>	Performs a network ping operation.
Advanced ping	<code>/networkPerformance/advPing</code>	Performs an advanced network ping.
LSP ping	<code>/networkPerformance/LSPPing</code>	Performs an LSP ping.
SLA	<code>/networkPerformance/SLA</code>	Returns network SLA information.
Link latency	<code>/networkPerformance/LinkLatency</code>	Returns link latency information.

## Categories

- *categories*: List of category
- *category*: object, properties :
  - *name*: The category name, (devicePerformance/systemUptime, ...)

## Parameters

- *No parameters*: Returns available dates
- *from*: Start date in *yyyymmdd*
- *to*: End date in *yyyymmdd*, default is same value as from
- *router*: Router filtering
- *offset*: Beginning line of the data, must be positive
- *limit*: Number of lines to be returned, -1 for end of data

The URL also accepts start/end timestamp, the get totalObject (total number of result).

### Related Documentation

- [WANDL IP/MPLSView RESTful Interfaces on page 15](#)

- [RESTful APIs Supported by WANDL IP/MPLSView on page 17](#)

## Performance Management Query Parameters

It is possible to filter for a specific selection of performance management data. Where applicable, query parameters are presented in the format: `?<parameter>=<value>`.

### Supported Performance Management Parameters

- *No parameters*: The list of available dates is returned.
- *From*: Start date in `yyyymmdd` format. If no other parameter is specified, the call returns the list of times and router for that date.
- *To*: End date in `yyyymmdd` format. If no other parameter is specified, the call returns the list of times and router for that date.
- *Router*: Selects the specific router to be queried.
- *Offset*: Specifies the data offset by line number.
- *Limit*: The number of lines to be returned, -1 for end of data.

To retrieve data, the following parameters are required:

- From
- Offset
- Limit



**NOTE:** The URL also accepts the start/end timestamp and gets the total Object count, the total number of results.

#### Related Documentation

- [Understanding the IP/MPLSView Performance Management API on page 45](#)

## List of Performance Management Categories

Retrieves the full list of performance management categories. For example:

```
v1/{tenant_id}/pm/{topology_id}/
```

Normal response codes: 200

[Table 24 on page 48](#) lists the request parameters.

**Table 24: Performance Management Categories – Request Parameters**

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1

**Related Documentation** • [Understanding the IP/MPLSView Performance Management API on page 45](#)

## System Uptime Information

Retrieves the system uptime information. For example:

```
v1/{tenant_id}/pm/{topology_id}/devicePerformance/systemUptime
```

Normal response codes: 200

[Table 25 on page 48](#) lists the request parameters.

**Table 25: System Update – Request Parameters**

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1.

**Related Documentation** • [Understanding the IP/MPLSView Performance Management API on page 45](#)

## CPU Temperature Information

Retrieves the full list of performance management categories. For example:

```
v1/{tenant_id}/pm/{topology_id}/devicePerformance/CPUTemp
```

Normal response codes: 200

[Table 26 on page 48](#) lists the request parameters.

**Table 26: CPU Temperature – Request Parameters**

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.



*Table 26: CPU Temperature – Request Parameters (continued)*

Parameter	Style	Type	Description
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1

**Related Documentation** • [Understanding the IP/MPLSView Performance Management API on page 45](#)

## CPU Utilization Information

Returns CPU usage (load) information. For example:

```
v1/{tenant_id}/pm/{topology_id}/devicePerformance/CPUUsage
```

Normal response codes: 200

[Table 27 on page 49](#) lists the request parameters.

*Table 27: CPU Utilization – Request Parameters*

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1

**Related Documentation** • [Understanding the IP/MPLSView Performance Management API on page 45](#)

## Memory Usage Information

Returns memory usage data. For example:

```
v1/{tenant_id}/pm/{topology_id}/devicePerformance/memoryUsage
```

Normal response codes: 200

[Table 28 on page 49](#) lists the request parameters.

*Table 28: Memory Usage – Request Parameters*

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1

**Related Documentation**

- [Understanding the IP/MPLSView Performance Management API on page 45](#)

## Ping

Performs a network ping operation. For example:

```
v1/{tenant_id}/pm/{topology_id}/networkPerformance/ping
```

Normal response codes: 200

[Table 29 on page 50](#) lists the request parameters.

**Table 29: Request Parameters**

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1

**Related Documentation**

- [Understanding the IP/MPLSView Performance Management API on page 45](#)

## Advanced Ping

Performs an advanced network ping. For example:

```
v1/{tenant_id}/pm/{topology_id}/networkPerformance/advPing
```

Normal response codes: 200

[Table 30 on page 50](#) lists the request parameters.

**Table 30: Advanced Ping – Request Parameters**

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1

**Related Documentation**

- [Understanding the IP/MPLSView Performance Management API on page 45](#)

## LSP Ping

Performs an LSP ping. For example:

`v1/{tenant_id}/pm/{topology_id}/networkPerformance/LSPPing`

Normal response codes: 200

Table 31 on page 51 lists the request parameters.

Table 31: LSP Ping – Request Parameters

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1

**Related Documentation** • [Understanding the IP/MPLSView Performance Management API on page 45](#)

## SLA

Returns network SLA information. For example:

`v1/{tenant_id}/pm/{topology_id}/networkPerformance/SLA`

Normal response codes: 200

Table 32 on page 51 lists the request parameters.

Table 32: SLA – Request Parameters

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1

**Related Documentation** • [Understanding the IP/MPLSView Performance Management API on page 45](#)

## Link Latency

Returns network link latency information. For example:

`v1/{tenant_id}/pm/{topology_id}/networkPerformance/LinkLatency`

Normal response codes: 200

[Table 33 on page 52](#) lists the request parameters.

**Table 33: Link Latency – Request Parameters**

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1

**Related Documentation**

- [Understanding the IP/MPLSView Performance Management API on page 45](#)

## Performance Management Examples

- [System Uptime on page 52](#)
- [System Uptime from Specified Date on page 52](#)
- [System Uptime with Date and Offset on page 52](#)

### System Uptime

```
/IPMPLSView/API/v1/tenant/1/pm/1/devicePerformance/systemUptime

{
  "dates": ["150626","150625", ... ,"120922","120921" ]
}
```

### System Uptime from Specified Date

```
/IPMPLSView/API/v1/tenant/1/pm/1/devicePerformance/systemUptime?from=150611

{ "totalCount": "18",
  "fields": ["Router","Availability(%)","2015-06-11 00:00:00", ... ,"2015-06-11
23:50:00","2015-06-11
23:55:00"],
  "routers": "10_BARCELONA","11_MANCHESTER", ... ,"BRG_MX80_07" ]
}
```

### System Uptime with Date and Offset

```
/IPMPLSView/API/v1/tenant/1/pm/1/devicePerformance/systemUptime?from=150611&offset=1&limit=10

{ "fields": ["Router","Availability(%)","2015-06-11 00:00:00", ... ,"2015-06-11
23:50:00","2015-06-11
23:55:00"],
  "totalCount": "18",
  "count": "10",
  "data": [
["10_BARCELONA","100"," "," ","9192120.0 / 106 day(s) 09:22:00", ... ,"9277620.0
/ 107
day(s) 09:07:00"],
```

```
["11_MANCHESTER","100"," ","","9192120.0 / 106 day(s) 09:22:00", ... ,"9277620.0  
/ 107  
day(s) 09:07:00"],  
...  
]  
}
```

**Related Documentation** • [Understanding the IP/MPLSView Performance Management API on page 45](#)



## CHAPTER 11

# Hardware Inventory

- [Understanding the IP/MPLSView Hardware Inventory API on page 55](#)
- [Supported Hardware Inventory Categories on page 55](#)
- [List of Hardware Inventory Categories on page 56](#)
- [List of Hardware Inventory Category Dates on page 56](#)
- [Hardware Inventory for a Category on a Specific Date on page 57](#)
- [Hardware Inventory Examples on page 57](#)

## Understanding the IP/MPLSView Hardware Inventory API

The hardware inventory API root is `/IPMPLSView/API/v1/tenant/1/hardwareInventory/1`. It only allows GET requests. There is one hardware inventory for each IP/MPLSView instance. It is tied to the live network. The live network nodes and hardware inventory nodes are disjointed, but may intersect.

[Table 34 on page 55](#) lists the hardware inventory-related APIs.

*Table 34: Hardware Inventory-Related APIs*

Object	URL	Description
Categories	<code>/categories</code>	Lists the available hardware inventory categories.
<code>&lt;category&gt;</code>	<code>/&lt;category&gt;</code>	Returns a list of available hardware inventory dates for the specified category.
<code>&lt;category&gt; + &lt;date&gt;</code>	<code>/&lt;category&gt;?date=&lt;date&gt;</code>	Returns hardware inventory information for the specified category on the date in <code>yyyymmdd</code> format.

- Related Documentation**
- [WANDL IP/MPLSView RESTful Interfaces on page 15](#)
  - [RESTful APIs Supported by WANDL IP/MPLSView on page 17](#)

## Supported Hardware Inventory Categories

The supported hardware inventory categories include:

- CAPEX
- CARD
- DEVICE\_USAGE
- EXTENSIVE\_PART
- IC
- INTF
- LINECARD\_USAGE
- MISC\_PART
- PARTS
- ROUTER
- TRANSCEIVER

**Related Documentation**

- [Understanding the IP/MPLSView Hardware Inventory API on page 55](#)

## List of Hardware Inventory Categories

Returns the list of available hardware inventory categories. For example:

```
v1/{tenant_id}/hardwareInventory/{topology_id}/categories
```

Normal response codes: 200

[Table 35 on page 56](#) lists the request parameters.

**Table 35: Hardware Inventory – Request Parameters**

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1.

**Related Documentation**

- [Understanding the IP/MPLSView Hardware Inventory API on page 55](#)

## List of Hardware Inventory Category Dates

Returns the list of available dates for a specific hardware inventory category. For example:

```
v1/{tenant_id}/hardwareInventory/{topology_id}/<category>
```

Normal response codes: 200



Table 36 on page 57 lists the request parameters.

**Table 36: Hardware Inventory Category Dates – Request Parameters**

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1.
<category>			The specific hardware category for which to retrieve available date information.

**Related Documentation**

- [Understanding the IP/MPLSView Hardware Inventory API on page 55](#)

## Hardware Inventory for a Category on a Specific Date

Returns the list of available hardware inventory categories. For example:

```
v1/{tenant_id}/hardwareInventory/{topology_id}/<category>?date=<date>
```

Normal response codes: 200

Table 37 on page 57 lists the request parameters.

**Table 37: Hardware Inventory Category on a Specific Date – Request Parameters**

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1.
<category>			The specific hardware category for which to retrieve information.
?date=<date>	URI	xsd:string	The date, expressed as ?date=<date>, in <i>yyyymmdd</i> format

**Related Documentation**

- [Understanding the IP/MPLSView Hardware Inventory API on page 55](#)

## Hardware Inventory Examples

The following examples can be used:

- [Hardware Inventory on page 58](#)
- [Router Information on page 58](#)

- [CAPEX with Date on page 58](#)
- [Card Information with Date on page 58](#)
- [Device Usage with Date on page 59](#)
- [Extensive Part Information with Date on page 59](#)
- [IC with Date on page 59](#)
- [Interface with Date on page 60](#)
- [Line Card Usage with Date on page 60](#)
- [Miscellaneous Part with Date on page 60](#)
- [Parts with Date on page 60](#)
- [Router with Date on page 61](#)
- [Transceiver with Date on page 61](#)

## Hardware Inventory

IPMPLSView/API/v1/tenant/1/hardwareInventory/1

```
[{"description": "Hardware Inventory"}, {"availableTypes": ["CAPEX", "CARD", "DEVICE_USAGE", "EXTENSIVE_PART", "IC", "INTF", "LINECARD_USAGE", "MISC_PART", "PARTS", "ROUTER", "TRANSCIEVER"]}]
```

## Router Information

IPMPLSView/API/v1/tenant/1/hardwareInventory/1/ROUTER

```
[{"hwType": "ROUTER"}, {"availableDates": [20150616, 20150617, 20150618, 20150619, 20150620, 20150621, 20150622, 20150623]}]
```

## CAPEX with Date

IPMPLSView/API/v1/tenant/1/hardwareInventory/1/CAPEX?date=20150622

```
[{"Device": "10_BARCELONA", "Type": "", "Index": "", "Component_Name": "", "Cost_Estimated": "192,000.00"}, {"Device": "", "Type": "Chassis", "Index": "", "Component_Name": "JUNOSV-FIREFLY", "Cost_Estimated": ""}, {"Device": "", "Type": "Card", "Index": "S-0/0", "Component_Name": "Virtual GE", "Cost_Estimated": "192,000.00"}, {"Device": "11_MANCHESTER", "Type": "", "Index": "", "Component_Name": "", "Cost_Estimated": "144,000.00"}, {"Device": "", "Type": "Chassis", "Index": "", "Component_Name": "CSR1000V", "Cost_Estimated": ""}, {"Device": "", "Type": "Card", "Index": "S-0", "Component_Name": "6xGE ports(auto-assign by port)", "Cost_Estimated": "144,000.00"}, {"Device": "12_MUNICH", "Type": "", "Index": "", "Component_Name": "", "Cost_Estimated": "144,000.00"}, {"Device": "", "Type": "Chassis", "Index": "", "Component_Name": "CSR1000V", "Cost_Estimated": ""}, {"Device": "", "Type": "Card", "Index": "S-0", "Component_Name": "6xGE ports(auto-assign by port)", "Cost_Estimated": "144,000.00"}, {"Device": "13_MILAN", "Type": "", "Index": "", "Component_Name": "", "Cost_Estimated": "48,000.00"}, {"Device": "", "Type": "Chassis", "Index": "", "Component_Name": "IOS XRv Series", "Cost_Estimated": ""}]
```

## Card Information with Date

IPMPLSView/API/v1/tenant/1/hardwareInventory/1/CARD?date=20150622

```
[{"Name": "Virtual
```

```

CE", "Device_Name": "10_BARCELONA", "Device_Vendor": "Juniper", "Device_IP_Addr": "62.200.0.10", "Card_I
d": "S-
0/0", "Connected_Ports": "8", "Shutdown_Ports": "0", "No_of_Ports": "8", "Part": "", "S/N": "", "FRU_Line_Ca
rd": "", "FRU_Route_Mem": "", "FRU_Packet_Mem": "", "L3_Engine_Type": "", "L3_Engine": "", "TAN": "", "Descri
ption": "Virtual
CE", "Controller_Mem": "", "Reserved_Ports": "0", "Cost_Estimated": "192,000.00", "Device_Mgmt_IP_Addr":
"", "Version": "", "Hostname": "10_BARCELONA", "Operational_State": "", "Admin_State": "", "AS": "4064047"}
, {"Name": "6xGE ports(auto-assign by
port)", "Device_Name": "11_MANCHESTER", "Device_Vendor": "Cisco", "Device_IP_Addr": "22.22.0.103", "Card
_Id": "S-
0", "Connected_Ports": "4", "Shutdown_Ports": "2", "No_of_Ports": "6", "Part": "", "S/N": "", "FRU_Line_Card
": "", "FRU_Route_Mem": "", "FRU_Packet_Mem": "", "L3_Engine_Type": "", "L3_Engine": "", "TAN": "", "Descript
ion": "", "Controller_Mem": "", "Reserved_Ports": "0", "Cost_Estimated": "144,000.00", "Device_Mgmt_IP_Add
r": "", "Version": "", "Hostname": "11_MANCHESTER", "Operational_State": "", "Admin_State": "", "AS": "4064
047"}]}

```

## Device Usage with Date

IPMPLSView/API/v1/tenant/1/hardwareInventory/1/DEVICE\_USAGE?date=20150622

```

[{"Device_Name": "10_BARCELONA", "No_of_Line_Cards": "1", "Available_Slots": "0", "Total_Slots": "1", "Co
nnected_Ports": "8", "Shutdown_Ports": "0", "Reserved_Ports": "0", "No_of_Ports": "8", "Hostname": "10_BAR
CELONA"}, {"Device_Name": "11_MANCHESTER", "No_of_Line_Cards": "1", "Available_Slots": "0", "Total_Slots
": "1", "Connected_Ports": "4", "Shutdown_Ports": "2", "Reserved_Ports": "0", "No_of_Ports": "6", "Hostname
": "11_MANCHESTER"}, {"Device_Name": "12_MUNICH", "No_of_Line_Cards": "1", "Available_Slots": "0", "Total
_Slots": "1", "Connected_Ports": "4", "Shutdown_Ports": "2", "Reserved_Ports": "0", "No_of_Ports": "6", "Ho
stname": "12_MUNICH"}]

```

## Extensive Part Information with Date

IPMPLSView/API/v1/tenant/1/hardwareInventory/1/EXTENSIVE\_PART?date=20150622

```

[{"Name": "JUNOSV-
FIREFLY", "Device_Name": "10_BARCELONA", "Device_Vendor": "Juniper", "Device_IP_Addr": "62.200.0.10", "P
art": "", "S/N": "b349c63acebe", "Description": "", "Cost_Estimated": "", "Hostname": "10_BARCELONA", "AS":
"4064047"}, {"Name": "Virtual
CE", "Device_Name": "10_BARCELONA", "Device_Vendor": "Juniper", "Device_IP_Addr": "62.200.0.10", "Part":
"", "S/N": "", "Description": "", "Cost_Estimated": "192,000.00", "Hostname": "10_BARCELONA", "AS": "406404
7"}, {"Name": "Power Supply
0", "Device_Name": "10_BARCELONA", "Device_Vendor": "Juniper", "Device_IP_Addr": "62.200.0.10", "Part": "
", "S/N": "", "Description": "", "Cost_Estimated": "0.00", "Hostname": "10_BARCELONA", "AS": "4064047"}]

```

## IC with Date

IPMPLSView/API/v1/tenant/1/hardwareInventory/1/IC?date=20150622

```

[{"Device": "brg-mx80-07", "Severity": "Info", "Description": "Interface \"ge-0/0/8\"
does not fit in
the card 4x 10GE XFP"}, {"Device": "jacky-solx86-dev.wan-
controller.juniper.net", "Severity": "Warning", "Description": "Unrecognized
interface name syntax
\\\"lo0\\\""}, {"Device": "jacky-solx86-dev.wan-
controller.juniper.net", "Severity": "Warning", "Description": "Unrecognized
interface name syntax
\\\"atge0\\\""}, {"Device": "robert-solx86-test01.wan-
controller.juniper.net", "Severity": "Warning", "Description": "Unrecognized
interface name syntax
\\\"lo0\\\""}, {"Device": "robert-solx86-test01.wan-
controller.juniper.net", "Severity": "Warning", "Description": "Unrecognized
interface name syntax
\\\"gani0\\\""}]

```

## Interface with Date

IPMPLSView/API/v1/tenant/1/hardwareInventory/1/INTF?date=20150622

```
[{"Name": "ge-0/0/0", "Device_Name": "10_BARCELONA", "Device_Vendor": "Juniper", "Device_IP_Addr": "62.200.0.10", "Admin_Status": "active", "IP_Address": "172.16.0.110/24", "Media_Type": "GE", "MTU": "", "Bandwidth": "1000.0M", "Physical_Address": "", "Operational_Status": "active", "Description": "", "IPv6": "", "Device_Mgmt_IP_Addr": "172.16.0.110", "Card_Id": "S-0/0", "Aggregated_Link": "", "VLAN": "", "Switchport_Mode": "", "Hostname": "10_BARCELONA"}, {"Name": "ge-0/0/1", "Device_Name": "10_BARCELONA", "Device_Vendor": "Juniper", "Device_IP_Addr": "62.200.0.10", "Admin_Status": "active", "IP_Address": "", "Media_Type": "GE", "MTU": "", "Bandwidth": "1000.0M", "Physical_Address": "", "Operational_Status": "active", "Description": "", "IPv6": "", "Device_Mgmt_IP_Addr": "172.16.0.110", "Card_Id": "S-0/0", "Aggregated_Link": "", "VLAN": "", "Switchport_Mode": "", "Hostname": "10_BARCELONA"}]
```

## Line Card Usage with Date

IPMPLSView/API/v1/tenant/1/hardwareInventory/1/LINECARD\_USAGE?date=20150622

```
[{"Card_Name": "Virtual GE", "Device_Name": "10_BARCELONA", "Card_Id": "S-0/0", "Connected_Ports": "8", "Shutdown_Ports": "0", "Reserved_Ports": "0", "No_of_Ports": "8", "Hostname": "10_BARCELONA"}, {"Card_Name": "6xGE ports(auto-assign by port)", "Device_Name": "11_MANCHESTER", "Card_Id": "S-0", "Connected_Ports": "4", "Shutdown_Ports": "2", "Reserved_Ports": "0", "No_of_Ports": "6", "Hostname": "11_MANCHESTER"}, {"Card_Name": "6xGE ports(auto-assign by port)", "Device_Name": "12_MUNICH", "Card_Id": "S-0", "Connected_Ports": "4", "Shutdown_Ports": "2", "Reserved_Ports": "0", "No_of_Ports": "6", "Hostname": "12_MUNICH"}]
```

## Miscellaneous Part with Date

IPMPLSView/API/v1/tenant/1/hardwareInventory/1/MISC\_PART?date=20150622

```
[{"Name": "Power Supply 0", "Device_Name": "10_BARCELONA", "Device_Vendor": "Juniper", "Device_IP_Addr": "62.200.0.10", "Part": "", "S/N": "", "Description": "", "Cost_Estimated": "0.00", "Hostname": "10_BARCELONA", "AS": "4064047"}, {"Name": "Power Supply 0", "Device_Name": "1_DUBLIN", "Device_Vendor": "Juniper", "Device_IP_Addr": "62.200.0.1", "Part": "", "S/N": "", "Description": "", "Cost_Estimated": "0.00", "Hostname": "1_DUBLIN", "AS": "4064047"}, {"Name": "Power Supply 0", "Device_Name": "2_AMSTERDAM", "Device_Vendor": "Juniper", "Device_IP_Addr": "62.200.0.2", "Part": "", "S/N": "", "Description": "", "Cost_Estimated": "0.00", "Hostname": "2_AMSTERDAM", "AS": "4064047"}]
```

## Parts with Date

IPMPLSView/API/v1/tenant/1/hardwareInventory/1/PARTS?date=20150622

```
[{"Type": "Card", "Component_Name": "10x 1GE(LAN SFP", "Count": "12", "Cost_Estimated": "2,880,000.00"}, {"Type": "Card", "Component_Name": "2xGE ports(auto-assign by port)", "Count": "1", "Cost_Estimated": "48,000.00"}, {"Type": "Card", "Component_Name": "39x10GE ports(auto-assign by port)", "Count": "1", "Cost_Estimated": ""}, {"Type": "Card", "Component_Name": "39x10GE, GE ports(auto-assign by port)", "Count": "1", "Cost_Estimated": "24,000.00"}, {"Type": "Card", "Component_Name": "3xGE ports(auto-assign by port)", "Count": "5", "Cost_Estimated": "360,000.00"}]
```

## Router with Date

IPMPLSView/API/v1/tenant/1/hardwareInventory/1/ROUTER?date=20150622

```
[{"Name": "10_BARCELONA", "Vendor": "Juniper", "IP_Address": "62.200.0.10", "Source": "172.16.0.110.10_B
ARCELONA.cfg", "System_Name": "", "Description": "", "Contact": "", "Location": "", "Last_Update_by_CLI": "
06/22/15 20:12:25", "Chassis_Type": "JUNOSV-
FIREFLY", "Hardware_Version": "", "Hardware_Id": "b349c63acebe", "ROM_Version": "", "ROM_System_Version"
: "", "Configuration_Register": "", "Memory": "", "Configuration_Memory": "", "Configuration_Memory_in_Us
e": "", "Boot_Image": "", "Processor": "", "OS_Version": "12.1X44-
D20.3", "Mgmt_IP_Addr": "172.16.0.110", "Model": "JUNIPER", "OS_Family": "JUNOS", "Last_Update_by_SNMP":
"", "Hostname": "10_BARCELONA", "IPv6": "62:200::10", "AS": "4064047"}, {"Name": "11_MANCHESTER", "Vendor"
: "Cisco", "IP_Address": "22.22.0.103", "Source": "172.16.0.111.11_MANCHESTER.cfg", "System_Name": "", "D
escription": "", "Contact": "", "Location": "", "Last_Update_by_CLI": "06/22/15
20:12:25", "Chassis_Type": "CSR1000V", "Hardware_Version": "", "Hardware_Id": "9FQLW6NUTV4", "ROM_Versio
n": "IOS-XE
ROMMON", "ROM_System_Version": "", "Configuration_Register": "0x2102", "Memory": "1141424K/6147K", "Conf
iguration_Memory": "32768K", "Configuration_Memory_in_Use": "", "Boot_Image": "bootFlash:csr1000v-
packages-adventerprisek9.03.09.00a.S.153-
2.S", "Processor": "XE", "OS_Version": "15.3(2)S0a", "Mgmt_IP_Addr": "172.16.0.111", "Model": "CISCO", "O
S_Family": "IOS", "Last_Update_by_SNMP": "", "Hostname": "11_MANCHESTER", "IPv6": "", "AS": "4064047"}]
```

## Transceiver with Date

IPMPLSView/API/v1/tenant/1/hardwareInventory/1/TRANSCEIVER?date=20150622

```
[{"Name": "UNSUPPORTED", "Device_Name": "BRG_MX240_04", "Device_Vendor": "Juniper", "Device_IP_Addr": "4
1.0.0.4", "Index": "S-0/0/0", "Part": "740-
030076", "S/N": "APF13450014401", "Cost_Estimated": "", "Hostname": "brg-mx240-
04", "AS": ""}, {"Name": "UNSUPPORTED", "Device_Name": "BRG_MX240_04", "Device_Vendor": "Juniper", "Device
_IP_Addr": "41.0.0.4", "Index": "S-0/0/2", "Part": "740-
030077", "S/N": "APF1345002466B", "Cost_Estimated": "", "Hostname": "brg-mx240-
04", "AS": ""}, {"Name": "SFP-
T", "Device_Name": "BRG_MX240_04", "Device_Vendor": "Juniper", "Device_IP_Addr": "41.0.0.4", "Index": "S-
0/0/8", "Part": "740-013111", "S/N": "C351129", "Cost_Estimated": "", "Hostname": "brg-mx240-
04", "AS": ""}]
```

**Related Documentation**

- [Understanding the IP/MPLSView Hardware Inventory API on page 55](#)



## CHAPTER 12

# Traffic Data

- [Understanding the IP/MPLSView Traffic Data API on page 63](#)
- [List of Available Nodes Traffic Data on page 65](#)
- [Router Traffic Data Details on page 65](#)
- [Interface Traffic Data Details on page 66](#)
- [Tunnel Traffic Data Details on page 67](#)
- [VPN Traffic Data Details on page 68](#)
- [Network Interface Traffic Data Information on page 79](#)
- [Router Network Interface Traffic Data Details on page 79](#)
- [Class of Service Traffic Statistics Data Details on page 80](#)
- [Multicast Traffic Statistics Data Details on page 81](#)
- [Traffic Data Examples on page 82](#)

### Understanding the IP/MPLSView Traffic Data API

The traffic data API root is `/IPMPLSView/API/v1/tenant/1/trafficData/1`. It only allows GET requests.

There is one traffic data collection schema for each IP/MPLSView instance, tied to the live network. The live network nodes and traffic data collection nodes are disjointed, but may intersect. Specific matching is done through the name specification.

[Table 38 on page 63](#) lists the traffic data information-related APIs.

**Table 38: Traffic Data Information-Related APIs**

Object	URL	Description	Parameters
Nodes	<code>/devices</code>	Returns the list of all devices configured for traffic collection. Returns all rows in the <b>WRouter</b> table.	None
Node	<code>/routerDetails</code>	Returns traffic information specific to a router identified by number or name.	<code>routerID</code> : integer <code>routerName</code> : string

Table 38: Traffic Data Information-Related APIs (continued)

Object	URL	Description	Parameters
Interface traffic	<b>/interfaceTraffic</b>	Provides interface traffic details for the specified router. A router ID or name must be specified.	Required one of: <ul style="list-style-type: none"> <li><i>routerID</i>: integer</li> <li><i>routerName</i>: string</li> </ul> Optional Parameters: <ul style="list-style-type: none"> <li><i>interfaceDesc</i></li> <li><i>interfaceID</i></li> <li><i>startDate</i></li> <li><i>endDate</i></li> <li><i>limit</i></li> </ul>
Tunnel traffic	<b>/tunnelTraffic</b>	Returns tunnel traffic details for a router from the daily database tunnel traffic tables. A router ID or name must be specified.	Required one of: <ul style="list-style-type: none"> <li><i>routerID</i>: integer</li> <li><i>routerName</i>: string</li> </ul> Optional Parameters: <ul style="list-style-type: none"> <li><i>interfaceDesc</i></li> <li><i>interfaceID</i></li> <li><i>startDate</i></li> <li><i>endDate</i></li> <li><i>limit</i></li> </ul>
Interface Traffic File	<b>/network/intfTraffic</b>	Reads contents of the <b>/u/wand/data/network/interface.traffic</b> file.	Required: None
Network Interface	<b>/network/intfTraffic</b>	Returns traffic information for a network interface.	
Interface Details	<b>/interfaceDetails</b>	Returns network interface details for the specified router and all of its interfaces for the most recently completed traffic collection cycle. A router ID or name must be specified.	Required: <ul style="list-style-type: none"> <li><i>routerID</i>: integer - OR</li> <li><i>routerName</i>: string</li> </ul>
Class of Service Traffic Statistics	<b>/cosTraffic</b>	Returns database CoS traffic statistics collected for a specified device and interface. A router ID or name and interface description values are required.  Returns all stats for either the current day (the default) or the specified <b>startDate</b> .  Defaults to <i>egress</i> values.  <b>NOTE:</b> CoS traffic statistics are only available for Juniper, Cisco, Alcatel-Lucent, and Huawei devices.	Required: <ul style="list-style-type: none"> <li><i>routerID</i>: integer - OR -</li> <li><i>routerName</i>: string</li> <li><i>interfaceDesc</i>: interface Description value</li> </ul> Optional Parameters: <ul style="list-style-type: none"> <li><i>startDate</i>: YYYYMMDD</li> <li><i>limit</i>: N</li> <li><i>ingress</i>—if included in the URL, retrieves ingress values</li> </ul>



Table 38: Traffic Data Information-Related APIs (continued)

Object	URL	Description	Parameters
Multicast traffic statistics	/multiCastTraffic	Returns database multicast traffic statistics collected for a specified device and interface. A router ID or name must be specified.	Required: <ul style="list-style-type: none"><li>routerID: integer - OR -</li><li>routerName: string</li></ul> Optional Parameters: <ul style="list-style-type: none"><li>startDate: YYYYMMDD</li><li>limit: N</li></ul>

- [Traffic Data Query Parameters on page 65](#)

Traffic Data Query Parameters

Where applicable, query parameters are presented in the format: *?<parameter>=<value>*.

- Related Documentation
- [WANDL IP/MPLSView RESTful Interfaces on page 15](#)
  - [RESTful APIs Supported by WANDL IP/MPLSView on page 17](#)

List of Available Nodes Traffic Data

Returns the list of all devices configured for traffic collection. The query returns all rows in the **WRouter** table. For example:

v1/{tenant\_id}/trafficData/topology\_id/devices

Normal response codes: 200

[Table 39 on page 65](#) lists the request parameters.

Table 39: Available Nodes Traffic Data – Request Parameters

Parameter	Style	Type	Description
tenant_id	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
topology_id	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1.

- Related Documentation
- [Understanding the IP/MPLSView Traffic Data API on page 63](#)

Router Traffic Data Details

Returns traffic information specific to a router identified by number or name. A router ID or name must be specified. For example:

v1/{tenant\_id}/trafficData/{topology\_id}/routerDetails

Normal response codes: 200

Table 40 on page 66 lists the request parameters.

**Table 40: Router Traffic Data – Request Parameters**

Parameter	Style	Type	Description
tenant_id	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
topology_id	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1.
routerID	URI	xsd:int	The router ID number.
routerName	URI	xsd:string	The router name, expressed as ?routerName=<value>.

**Related Documentation**

- [Understanding the IP/MPLSView Traffic Data API on page 63](#)

## Interface Traffic Data Details

Returns interface traffic details for the specified router, with the information retrieved from the daily database traffic tables. A router ID or name must be specified. For example:

v1/{tenant\_id}/trafficData/{topology\_id}/interfaceTraffic

Normal response codes: 200

Table 41 on page 66 lists the request parameters.

**Table 41: Interface Traffic Data – Request Parameters**

Parameter	Style	Type	Description
tenant_id	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
topology_id	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1.
routerID	URI	xsd:int	The router ID number.
routerName	URI	xsd:string	The router name, expressed as ?routerName=<value>.

### Optional Parameters

Optional interface traffic data parameters include:

- **interfaceDesc**
- **tunnelID**
- **startDate**—Use YYYYMMDD format.
- **endDate**—Use YYYYMMDD format.
- **limit**—Use an integer value.

**Related Documentation** • [Understanding the IP/MPLSView Traffic Data API on page 63](#)

### Tunnel Traffic Data Details

Returns traffic details for the specified router tunnel, with the information retrieved from the daily database tunnel traffic tables. A router ID or name must be specified. All other parameters are optional. For example:

```
v1/{tenant_id}/trafficData/{topology_id}/tunnelTraffic
```

Normal response codes: 200

[Table 42 on page 67](#) lists the request parameters.

Table 42: Request Parameters

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1.
routerID	URI	xsd:int	The router ID number.
routerName	URI	xsd:string	The router name, expressed as ?routerName=<value>.

**Optional Parameters**

Optional interface traffic data parameters include:

- **interfaceDesc**
- **tunnelID**
- **startDate**—Use YYYYMMDD format.
- **endDate**—Use YYYYMMDD format.
- **limit**—Use an integer value.

- Related Documentation**
- [Understanding the IP/MPLSView Traffic Data API on page 63](#)

## VPN Traffic Data Details

Returns traffic details for the specified router VPN, with the information retrieved from the daily database VPN tables.

- Root URL: Same consideration as Performance Management and Hardware Inventory
- **Traffic collection specific:** `/tenant/1/trafficData /1/`

VPN Traffic Data	Parameter	Description
<code>IPMPLSView/API/v1/tenant/1/trafficData/1/vpnInterfacesByVPN</code>	<code>vpnname</code>	Returns list of device/interfaces on the specified <b>vpnName</b> .
<code>IPMPLSView/API/v1/tenant/1/trafficData/1/vpnfile</code>		Returns a formatted version of an attached <i>vpn.x</i> file.
<code>IPMPLSView/API/v1/tenant/1/trafficData/1/vpnfile</code>	<code>vpntype</code>	<p>Returns a list of Layer 2 Kompella VPN instance allowed values:</p> <ul style="list-style-type: none"> <li>• <code>vpntype</code> include 3 (L3 VPN)</li> <li>• 2M (L2 Martini)</li> <li>• 2K (L2 Kompella)</li> <li>• 2V (VPLS LDP based)</li> <li>• 2C (L2 CCC)</li> <li>• VPLS (VPLS BGP based)</li> <li>• VR (VRF Lite) Interface traffic per VPN</li> </ul>
<code>IPMPLSView/API/v1/tenant/1/trafficData/1/vpnInterfaceTraffic</code>	<code>vpnname</code>	Returns all device/interface traffic stats (from the DB) for the specified <b>vpnName</b> .

## Traffic Collection Schema

Example A:

`IPMPLSView/API/v1/tenant/1/trafficData/1/devices`

(partial sample output)

```
[ { routerID: 1,
  groupID: 1,
  routerIP: '172.16.0.105',
  routerName: '5_PARIS',
  routerType: 'JUNIPER_J',
  secondaryIP: '',
  snmpPort: 161,
  snmpRetry: 3,
  snmpTimeout: 3,
  snmpVersion: '2C',
```

```

snmpUser: '',
snmpContextEngine: null,
snmpAuth: 'NONE',
snmpPrivacy: 'NONE',
collectIFXinfo: 1,
collectionMethod: 'SNMP',
cliLogin: 'newlab',
cliAccessMethod: 'telnet',
cliTelnetTimeout: 300,
cliTelnetRetries: 3,
cliTelnetPort: 23,
cliSSHCommand: 'ssh',
cliAgents: '',
snmpGetBulkSize: 0,
maxCollectTime: 240,
collectTables: 'IF,IFX,TUNNEL,COS,MCAST',
cliSpecific: null,
ipv6Address: '' },
{ routerID: 2,
groupID: 1,
routerIP: '172.16.0.104',
routerName: '4_BERLIN',
routerType: 'JUNIPER_J',
secondaryIP: '',
snmpPort: 161,
snmpRetry: 3,
snmpTimeout: 3,
snmpVersion: '2C',
snmpUser: '',
snmpContextEngine: null,
snmpAuth: 'NONE',
snmpPrivacy: 'NONE',
collectIFXinfo: 1,
collectionMethod: 'SNMP',
cliLogin: 'newlab',
cliAccessMethod: 'telnet',
cliTelnetTimeout: 300,
cliTelnetRetries: 3,
cliTelnetPort: 23,
cliSSHCommand: 'ssh',
cliAgents: '',
snmpGetBulkSize: 0,
maxCollectTime: 240,
collectTables: 'IF,IFX,TUNNEL,COS,MCAST',
cliSpecific: null,
ipv6Address: '' },
{ routerID: 3,
groupID: 1,
routerIP: '172.16.0.107',
routerName: '7_MADRID',
routerType: 'JUNIPER_J',
secondaryIP: '',
snmpPort: 161,
snmpRetry: 3,
snmpTimeout: 3,
snmpVersion: '2C',
snmpUser: '',
snmpContextEngine: null,
snmpAuth: 'NONE',
snmpPrivacy: 'NONE',
collectIFXinfo: 1,

```

```
collectionMethod: 'SNMP',
cliLogin: 'newlab',
cliAccessMethod: 'telnet',
cliTelnetTimeout: 300,
cliTelnetRetries: 3,
cliTelnetPort: 23,
cliSSHCommand: 'ssh',
cliAgents: '',
snmpGetBulkSize: 0,
maxCollectTime: 240,
collectTables: 'IF,IFX,TUNNEL,COS,MCAST',
cliSpecific: null,
ipv6Address: '' },
{ routerID: 4,
groupID: 1,
routerIP: '172.16.0.106',
routerName: '6_FRANKFURT',
routerType: 'JUNIPER_J',
secondaryIP: '',
snmpPort: 161,
snmpRetry: 3,
snmpTimeout: 3,
snmpVersion: '2C',
snmpUser: '',
snmpContextEngine: null,
snmpAuth: 'NONE',
snmpPrivacy: 'NONE',
collectIFXinfo: 1,
collectionMethod: 'SNMP',
cliLogin: 'newlab',
cliAccessMethod: 'telnet',
cliTelnetTimeout: 300,
cliTelnetRetries: 3,
cliTelnetPort: 23,
cliSSHCommand: 'ssh',
cliAgents: '',
snmpGetBulkSize: 0,
maxCollectTime: 240,
collectTables: 'IF,IFX,TUNNEL,COS,MCAST',
cliSpecific: null,
ipv6Address: ''
},
. . .
. . .
]
```

Example B:

**IPMPLSView/API/v1/tenant/1/trafficData/1/routerDetails?routerName=5\_PARIS**

(partial sample output)

```
{"routerID":2,
"groupID":1,
"routerName":"4_BERLIN",
"routerIP":"172.16.0.104",
"routerType":"JUNIPER_J",
"collectTables":"IF,IFX",
"timeSig":20151114224,
```

```

"status":1,
"ingress":198153,
"egress":168965,
"tIngress":0,
"tEgress":0
},
{"routerID":3,
"groupID":1,
"routerName":"7_MADRID",
"routerIP":"172.16.0.107",
"routerType":"JUNIPER_J",
"collectTables":"IF,IFX",
"timeSig":20151114224,
"status":1,
"ingress":66626,
"egress":69420,
"tIngress":0,
"tEgress":0
},
{"routerID":4,
"groupID":1,
"routerName":"6_FRANKFURT",
"routerIP":"172.16.0.106",
"routerType":"JUNIPER_J",
"collectTables":"IF,IFX",
"timeSig":20151114224,
"status":1,
"ingress":458955,
"egress":428662,
"tIngress":0,
"tEgress":0},
. . .
. . .

```

Example C:

`/IPMPLSView/API/v1/tenant/1/trafficData/1/interfaceDetails?routerName=5_PARIS`

(partial sample output)

```

[
{ interfaceDesc: 'lo0.0',
  interfaceBW: 0,
  comment: '',
  interfaceID: 350,
  timeSig: 20141110121,
  collectedTS: 1418224231,
  status: 1,
  ingress: 0,
  egress: 0,
  ingressError: 0,
  egressError: 0,
  ingressDiscard: 0,
  egressDiscard: 0,
  ingressMax: 0,
  egressMax: 0 },
{

```

`/IPMPLSView/API/v1/tenant/1/trafficData/1/network/intfTraffic`

(partial sample output)

```
[ { StimeMMDDYY: '12/10/14',
  StimeHH: '9',
  StimeMM: '25',
  StimeAMPPM: 'AM',
  Interval: '5' },
  { routerName: '10_BARCELONA',
    intfName: 'ge-0/0/1.470',
    ipAddr: '172.16.0.110',
    type: '0',
    dataPoints:
      [ '16521',
        '16529',
        '16525',
        '16526',
        '16526',
        '16470',
        '16524',
        '16527',
        '16529',
        '16527',
        '16525',
        '16523',
        '16527',
        '16473',
        '16577',
        '16528',
        '16467',
        '16582',
        '16528',
        '16469',
        '16583',
        '16470',
        '16523',
        '16526' ] },
```

Example E:

**/IPMPLSView/API/v1/tenant/1/trafficData/1/network/tunTraffic**

(partial sample output)

Same as PM

Example F:

**/IPMPLSView/API/v1/tenant/1/trafficData/1/tunnelTraffic?routerName=5\_PARIS&limit=2**

(partial sample output)

```
[{"tunnelID":152,
"timeSig":0,
"collectedTS":0,
"status":2,
"recordedRoute":"null",
```



```

    "ingressBytes":0,
    "ingressBytesDelta":0,
    "ingressPackets":0,
    "ingressPacketsDelta":0,
    "routerID":1,
    "toRouterID":9,
    "tunnelName":"R5_PARIS1_DUBLIN_7"
  },
  {
    "tunnelID":153,
    "timeSig":0,
    "collectedTS":0,
    "status":-2,
    "recordedRoute":"null",
    "ingressBytes":0,
    "ingressBytesDelta":0,
    "ingressPackets":0,
    "ingressPacketsDelta":0,
    "routerID":1,
    "toRouterID":10,
    "tunnelName":"R5_PARIS2_AMSTERDAM_8"}
]

```

Example G:

`/IPMPLSView/API/v1/tenant/1/trafficData/1/interfaceTraffic?routerName=5_PARIS&limit=10`

(partial sample output)

```

[
  {
    "interfaceID": 1062,
    "interfaceDesc": "ge-0/0/1",
    "timeSig": 20151114239,
    "collectedTS": 1450123231,
    "status": 1,
    "ingressBytes": 1316525362950,
    "ingressBytesDelta": 98067,
    "ingressUtil": 0.0784536,
    "ingressPackets": 0,
    "ingressPacketsDelta": 0,
    "egressBytes": 1322019050049,
    "egressBytesDelta": 98299,
    "egressUtil": 0.0786392,
    "egressPackets": 0,
    "egressPacketsDelta": 0,
    "ifHCInUcastPkts": 3640909443,
    "ifHCInMulticastPkts": 0,
    "ifHCInBroadcastPkts": 0,
    "ifHCOutUcastPkts": 1066996921,
    "ifHCOutMulticastPkts": 0,
    "ifHCOutBroadcastPkts": 0,
    "ingressErrorCount": 2147483647,
    "egressErrorCount": 0,
    "ingressDiscardCount": 126336,
    "egressDiscardCount": 0
  },
  {
    "interfaceID": 1062,
    "interfaceDesc": "ge-0/0/1",
    "timeSig": 20151114240,

```

```
    "collectedTS": 1450123531,
    "status": 1,
    "ingressBytes": 1316554983037,
    "ingressBytesDelta": 98733,
    "ingressUtil": 0.0789864,
    "ingressPackets": 0,
    "ingressPacketsDelta": 0,
    "egressBytes": 1322048735396,
    "egressBytesDelta": 98951,
    "egressUtil": 0.0791608,
    "egressPackets": 0,
    "egressPacketsDelta": 0,
    "ifHCInUcastPkts": 3640964983,
    "ifHCInMulticastPkts": 0,
    "ifHCInBroadcastPkts": 0,
    "ifHCOutUcastPkts": 1067021777,
    "ifHCOutMulticastPkts": 0,
    "ifHCOutBroadcastPkts": 0,
    "ingressErrorCount": 2147483647,
    "egressErrorCount": 0,
    "ingressDiscardCount": 126336,
    "egressDiscardCount": 0
  }
]
```

Example H:

**/IPMPLSView/API/v1/tenant/1/trafficData/1/cosTraffic?routerName=4\_BERLIN  
&InterfaceDesc=ge-0/0/1.424**

(partial sample output of Juniper Networks device)

```
[
  {
    "interfaceDesc": "ge-0/0/1.424",
    "queueID": 45,
    "queueKey": "336+2+0",
    "routerID": 2,
    "interfaceID": 336,
    "queueNumber": 0,
    "queueName": "INTERNET",
    "inOutType": 2,
    "timeSig": 0,
    "queuedBytesCounter": 0,
    "queuedBytesDelta": 0,
    "transmittedBytesCounter": 0,
    "transmittedBytesDelta": 0,
    "totalDroppedBytesCounter": 0,
    "totalDroppedBytesDelta": 0
  },
  {
    "interfaceDesc": "ge-0/0/1.424",
    "queueID": 45,
    "queueKey": "336+2+0",
    "routerID": 2,
    "interfaceID": 336,
    "queueNumber": 0,
    "queueName": "INTERNET",
    "inOutType": 2,
  }
]
```

```

        "timeSig":20150915182,
        "queuedBytesCounter":45168572,
        "queuedBytesDelta":null,
        "transmittedBytesCounter":45168572,
        "transmittedBytesDelta":null,
        "totalDroppedBytesCounter":0,
        "totalDroppedBytesDelta":null
    },
    {
        "interfaceDesc":"ge-0/0/1.424",
        "queueID":45,
        "queueKey":"336+2+0",
        "routerID":2,
        "interfaceID":336,
        "queueNumber":0,
        "queueName":"INTERNET",
        "inOutType":2,
        "timeSig":20150915184,
        "queuedBytesCounter":45171148,
        "queuedBytesDelta":8,
        "transmittedBytesCounter":45171148,
        "transmittedBytesDelta":8,
        "totalDroppedBytesCounter":0,
        "totalDroppedBytesDelta":0
    },
    {
        "interfaceDesc":"ge-0/0/1.424",
        "queueID":45,
        "queueKey":"336+2+0",
        "routerID":2,
        "interfaceID":336,
        "queueNumber":0,
        "queueName":"INTERNET",
        "inOutType":2,
        "timeSig":20150915185,
        "queuedBytesCounter":45171352,
        "queuedBytesDelta":0,
        "transmittedBytesCounter":45171352,
        "transmittedBytesDelta":0,
        "totalDroppedBytesCounter":0,
        "totalDroppedBytesDelta":0
    },
    {
        "interfaceDesc":"ge-0/0/1.424",
        "queueID":46,
        "queueKey":"336+2+1",
        "routerID":2,
        "interfaceID":336,
        "queueNumber":1,
        "queueName":"BUSINESS",
        "inOutType":2,
        "timeSig":20150915237,
        "queuedBytesCounter":0,
        "queuedBytesDelta":0,
        "transmittedBytesCounter":0,
        "transmittedBytesDelta":0,
        "totalDroppedBytesCounter":0,
        "totalDroppedBytesDelta":0
    },
    {
        "interfaceDesc":"ge-0/0/1.424",

```

```
    "queueID":46,
    "queueKey":"336+2+1",
    "routerID":2,
    "interfaceID":336,
    "queueNumber":1,
    "queueName":"BUSINESS",
    "inOutType":2,
    "timeSig":20150915238,
    "queuedBytesCounter":0,
    "queuedBytesDelta":0,
    "transmittedBytesCounter":0,
    "transmittedBytesDelta":0,
    "totalDroppedBytesCounter":0,
    "totalDroppedBytesDelta":0

  {
    "interfaceDesc":"ge-0/0/1.424",
    "queueID":48,
    "queueKey":"336+2+3",
    "routerID":2,
    "interfaceID":336,
    "queueNumber":3,
    "queueName":"CONTROL",
    "inOutType":2,
    "timeSig":20150915185,
    "queuedBytesCounter":278741283,
    "queuedBytesDelta":15,
    "transmittedBytesCounter":278741283,
    "transmittedBytesDelta":15,
    "totalDroppedBytesCounter":0,
    "totalDroppedBytesDelta":0
  },
  . . .
]
```

Example I:

**/IPMPLSView/API/v1/tenant/1/trafficData/1/multiCastTraffic?routerName=12\_MUNICH**

(partial sample output)

```
[
  {
    "interfaceID":466,
    "interfaceDesc":"GigabitEthernet2.542",
    "timeSig":20150915182,
    "mcastInOctetsCounter":23666382708,
    "mcastInOctetsDelta":0,
    "mcastInUtil":0,
    "mcastOutOctetsCounter":0,
    "mcastOutOctetsDelta":0,
    "mcastOutUtil":0
  },
  {
    "interfaceID":466,
    "interfaceDesc":"GigabitEthernet2.542",
    "timeSig":20150915184,
```

```

        "mcastInOctetsCounter":23671007772,
        "mcastInOctetsDelta":15416,
        "mcastInUtil":0.0123328,
        "mcastOutOctetsCounter":0,
        "mcastOutOctetsDelta":0,
        "mcastOutUtil":0
    },
    {
        "interfaceID":466,
        "interfaceDesc":"GigabitEthernet2.542",
        "timeSig":20150915185,
        "mcastInOctetsCounter":23675541072,
        "mcastInOctetsDelta":15111,
        "mcastInUtil":0.0120888,
        "mcastOutOctetsCounter":0,
        "mcastOutOctetsDelta":0,
        "mcastOutUtil":0
    },
    {
        "interfaceID":466,
        "interfaceDesc":"GigabitEthernet2.542",
        "timeSig":20150915186,
        "mcastInOctetsCounter":23680071636,
        "mcastInOctetsDelta":15101,
        "mcastInUtil":0.0120808,
        "mcastOutOctetsCounter":0,
        "mcastOutOctetsDelta":0,
        "mcastOutUtil":0
    },
    {
        "interfaceID":466,
        "interfaceDesc":"GigabitEthernet2.542",
        "timeSig":20150915187,
        "mcastInOctetsCounter":23684540964,
        "mcastInOctetsDelta":14897,
        "mcastInUtil":0.0119176,
        "mcastOutOctetsCounter":0,
        "mcastOutOctetsDelta":0,
        "mcastOutUtil":0
    },
    {
        "interfaceID":466,
        "interfaceDesc":"GigabitEthernet2.542",
        "timeSig":20150915188,
        "mcastInOctetsCounter":23689124028,
        "mcastInOctetsDelta":15276,
        "mcastInUtil":0.0122208,
        "mcastOutOctetsCounter":0,
        "mcastOutOctetsDelta":0,
        "mcastOutUtil":0
    },
    {
        "interfaceID":466,
        "interfaceDesc":"GigabitEthernet2.542",
        "timeSig":20150915189,
        "mcastInOctetsCounter":23693670828,
        "mcastInOctetsDelta":15156,
        "mcastInUtil":0.0121248,
        "mcastOutOctetsCounter":0,
        "mcastOutOctetsDelta":0,
        "mcastOutUtil":0
    }

```

```
    },  
    {  
      "interfaceID":466,  
      "interfaceDesc":"GigabitEthernet2.542",  
      "timeSig":20150915190,  
      "mcastInOctetsCounter":23698244892,  
      "mcastInOctetsDelta":15246,  
      "mcastInUtil":0.0121968,  
      "mcastOutOctetsCounter":0,  
      "mcastOutOctetsDelta":0,  
      "mcastOutUtil":0  
    },  
    .....  
    .....  
    .....  
  ]
```

Example J:

**/IPMPLSView/API/v1/tenant/1/pm/1/devicePerformance/systemUptime**

(partial sample output)

```
{  
  "dates": ["150626", "150625", ... , "120922", "120921" ]  
}
```

Example K:

**/IPMPLSView/API/v1/tenant/1/pm/1/devicePerformance/systemUptime?from=150611**

(partial sample output)

```
{  
  "totalCount": "18",  
  "fields": ["Router", "Availability(%)", "2015-06-11 00:00:00", ... , "2015-06-11  
23:50:00", "2015-06-11 23:55:00"],  
  "routers": "10_BARCELONA", "11_MANCHESTER", ... , "BRG_MX80_07" ]  
}
```

Example L:

**/IPMPLSView/API/v1/tenant/1/pm/1/devicePerformance/systemUptime?from=150611&offset=1&limit=10**

(partial sample output)

```
{  
  "fields": ["Router", "Availability(%)", "2015-06-11 00:00:00", ... , "2015-06-11  
23:50:00", "2015-06-11 23:55:00"],  
  "totalCount": "18",  
  "count": "10",  
  "data": [  
    ["10_BARCELONA", "100", " ", " ", "9192120.0 / 106 day(s) 09:22:00", ... , "9277620.0 /  
107 day(s) 09:07:00"],
```

```
[
  ["11_MANCHESTER", "100", " ", " ", "9192120.0 / 106 day(s) 09:22:00", ... , "9277620.0 / 107 day(s) 09:07:00"],
  ...
]
```

**Related Documentation** • [Understanding the IP/MPLSView Traffic Data API on page 63](#)

## Network Interface Traffic Data Information

Returns traffic information for the network interface. For example:

```
v1/{tenant_id}/trafficData/{topology_id}/intfTraffic
```

Normal response codes: 200

[Table 43 on page 79](#) lists the request parameters.

**Table 43: Request Parameters**

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1.

**Related Documentation** • [Understanding the IP/MPLSView Traffic Data API on page 63](#)

## Router Network Interface Traffic Data Details

Returns network interface traffic data details for the specified router. A router ID or name must be provided. All other parameters are optional. For example:

```
v1/{tenant_id}/trafficData/{topology_id}/interfaceDetails
```

Normal response codes: 200

[Table 44 on page 79](#) lists the request parameters.

**Table 44: Router Network Interface Traffic Data – Request Parameters**

Parameter	Style	Type	Description
<i>tenant_id</i>	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
<i>topology_id</i>	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1.

Table 44: Router Network Interface Traffic Data – Request Parameters (continued)

Parameter	Style	Type	Description
routerID	URI	xsd:int	The router ID number.
routerName	URI	xsd:string	The router name, expressed as <code>?routerName=&lt;value&gt;</code> .

**Optional Parameters**

Optional interface traffic data parameters include:

- timesig
- sSearch
- status

**Related Documentation**

- [Understanding the IP/MPLSView Traffic Data API on page 63](#)

## Class of Service Traffic Statistics Data Details

Returns DB CoS traffic statistics collected for a specified device and an interface. Returns all statistics for either the current day (the default option) or for the specified start date.



**NOTE:** CoS traffic statistics are only available for Juniper Networks, Cisco Systems, Alcatel-Lucent, and Huawei devices.

`v1/{tenant_id}/trafficData/{topology_id}/cosTraffic`

Normal response codes: 200

Table 45 on page 80 lists the required parameters.

Table 45: Class of Service Traffic Statistics Data – Request Parameters

Parameter	Style	Type	Description
tenant_id	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
topology_id	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1.
routerID	URI	xsd:int	The router ID number.
routerName	URI	xsd:string	The router name, expressed as <code>?routerName=&lt;value&gt;</code> .
interfaceDesc	URI	xsd:string	The interface description value.



Optional Parameters

Optional interface traffic data parameters include:

- **startDate**—Use YYYYMMDD format.
- **limit**—Use an integer value.
- **ingress**—Ingress value is retrieved if included in the URL.

**Related Documentation** • [Understanding the IP/MPLSView Traffic Data API on page 63](#)

Multicast Traffic Statistics Data Details

Returns DB multicast traffic statistics collected for a specified device and an interface. Returns all statistics for either the current day (the default option) or for the specified start date. For example:

```
v1/{tenant_id}/trafficData/{topology_id}/multiCastTraffic
```

Normal response codes: 200

[Table 46 on page 81](#) lists the required parameters.

Table 46: Multicast Traffic Statistics Data – Request Parameters

Parameter	Style	Type	Description
tenant_id	URI	xsd:int	The unique identifier of the tenant or account. Currently it is fixed to 1.
topology_id	URI	xsd:int	The unique identifier of the topology. Currently it is fixed to 1.
routerID	URI	xsd:int	The router ID number.
routerName	URI	xsd:string	The router name, expressed as ?routerName=<value>.

Optional Parameters

Optional interface traffic data parameters include:

- **startDate**—Use YYYYMMDD format
- **limit**—Use an integer value.

**Related Documentation** • [Understanding the IP/MPLSView Traffic Data API on page 63](#)

## Traffic Data Examples

---

### Interface Details

```
/IPMPLSView/API/v1/tenant/1/trafficData/1/interfaceDetails?routerName=5_PARIS
[
{ interfaceDesc: 'lo0.0',
  interfaceBW: 0,
  comment: '',
  interfaceID: 350,
  timeSig: 20141110121,
  collectedTS: 1418224231,
  status: 1,
  ingress: 0,
  egress: 0,
  ingressError: 0,
  egressError: 0,
  ingressDiscard: 0,
  egressDiscard: 0,
  ingressMax: 0,
  egressMax: 0 },
{
```

### Devices Details

```
/IPMPLSView/API/v1/tenant/1/trafficData/1/devices
[ { routerID: 1,
  groupID: 1,
  routerIP: '172.16.0.105',
  routerName: '5_PARIS',
  routerType: 'JUNIPER_J',
  secondaryIP: '',
  snmpPort: 161,
  snmpRetry: 3,
  snmpTimeout: 3,
  snmpVersion: '2C',
  snmpUser: '',
  snmpContextEngine: null,
  snmpAuth: 'NONE',
  snmpPrivacy: 'NONE',
  collectIFXinfo: 1,
  collectionMethod: 'SNMP',
  cliLogin: 'newlab',
  cliAccessMethod: 'telnet',
  cliTelnetTimeout: 300,
  cliTelnetRetries: 3,
  cliTelnetPort: 23,
  cliSSHCommand: 'ssh',
  cliAgents: '',
  snmpGetBulkSize: 0,
  maxCollectTime: 240,
  collectTables: 'IF,IFX,TUNNEL,COS,MCAST',
  cliSpecific: null,
  ipv6Address: '' },
{ routerID: 2,
  groupID: 1,
  routerIP: '172.16.0.104',
  routerName: '4_BERLIN',
  routerType: 'JUNIPER_J',
  secondaryIP: '',
```

```

snmpPort: 161,
snmpRetry: 3,
snmpTimeout: 3,
snmpVersion: '2C',
snmpUser: '',
snmpContextEngine: null,
snmpAuth: 'NONE',
snmpPrivacy: 'NONE',
collectIFXinfo: 1,
collectionMethod: 'SNMP',
cliLogin: 'newlab',
cliAccessMethod: 'telnet',
cliTelnetTimeout: 300,
cliTelnetRetries: 3,
cliTelnetPort: 23,
cliSSHCommand: 'ssh',
cliAgents: '',
snmpGetBulkSize: 0,
maxCollectTime: 240,
collectTables: 'IF,IFX,TUNNEL,COS,MCAST',
cliSpecific: null,
ipv6Address: '' },
{ routerID: 3,
  groupID: 1,
  routerIP: '172.16.0.107',
  routerName: '7_MADRID',
  routerType: 'JUNIPER_J',
  secondaryIP: '',
  snmpPort: 161,
  snmpRetry: 3,
  snmpTimeout: 3,
  snmpVersion: '2C',
  snmpUser: '',
  snmpContextEngine: null,
  snmpAuth: 'NONE',
  snmpPrivacy: 'NONE',
  collectIFXinfo: 1,
  collectionMethod: 'SNMP',
  cliLogin: 'newlab',
  cliAccessMethod: 'telnet',
  cliTelnetTimeout: 300,
  cliTelnetRetries: 3,
  cliTelnetPort: 23,
  cliSSHCommand: 'ssh',
  cliAgents: '',
  snmpGetBulkSize: 0,
  maxCollectTime: 240,
  collectTables: 'IF,IFX,TUNNEL,COS,MCAST',
  cliSpecific: null,
  ipv6Address: '' },
{ routerID: 4,
  groupID: 1,
  routerIP: '172.16.0.106',
  routerName: '6_FRANKFURT',
  routerType: 'JUNIPER_J',
  secondaryIP: '',
  snmpPort: 161,
  snmpRetry: 3,
  snmpTimeout: 3,
  snmpVersion: '2C',
  snmpUser: '' },

```

```
snmpContextEngine: null,  
snmpAuth: 'NONE',  
snmpPrivacy: 'NONE',  
collectIFXinfo: 1,  
collectionMethod: 'SNMP',  
cliLogin: 'newlab',  
cliAccessMethod: 'telnet',  
cliTelnetTimeout: 300,  
cliTelnetRetries: 3,  
cliTelnetPort: 23,  
cliSSHCommand: 'ssh',  
cliAgents: '',  
snmpGetBulkSize: 0,  
maxCollectTime: 240,  
collectTables: 'IF,IFX,TUNNEL,COS,MCAST',  
cliSpecific: null,  
ipv6Address: ''  
},  
. . .  
. . .  
]
```

#### Router Details

```
/IPMPLSView/API/v1/tenant/1/trafficData/1/routerDetails?routerName=5_PARIS  
{  
  "routerID":2,  
  "groupID":1,  
  "routerName":"4_BERLIN",  
  "routerIP":"172.16.0.104",  
  "routerType":"JUNIPER_J",  
  "collectTables":"IF,IFX",  
  "timeSig":20151114224,  
  "status":1,  
  "ingress":198153,  
  "egress":168965,  
  "tIngress":0,  
  "tEgress":0  
},  
{  
  "routerID":3,  
  "groupID":1,  
  "routerName":"7_MADRID",  
  "routerIP":"172.16.0.107",  
  "routerType":"JUNIPER_J",  
  "collectTables":"IF,IFX",  
  "timeSig":20151114224,  
  "status":1,  
  "ingress":66626,  
  "egress":69420,  
  "tIngress":0,  
  "tEgress":0  
},  
{  
  "routerID":4,  
  "groupID":1,  
  "routerName":"6_FRANKFURT",  
  "routerIP":"172.16.0.106",  
  "routerType":"JUNIPER_J",  
  "collectTables":"IF,IFX",  
  "timeSig":20151114224,  
  "status":1,  
  "ingress":458955,  
  "egress":428662,
```

```

    "tIngress":0,
    "tEgress":0},
    . . .
    . . .

```

### Network Interface Traffic

```
/IPMPLSView/API/v1/tenant/1/trafficData/1/network/intfTraffic
```

```

[ { StimeMMDDYY: '12/10/14',
  StimeHH: '9',
  StimeMM: '25',
  StimeAMPPM: 'AM',
  Interval: '5' },
  { routerName: '10_BARCELONA',
    intfName: 'ge-0/0/1.470',
    ipAddr: '172.16.0.110',
    type: '0',
    dataPoints:
      [ '16521',
        '16529',
        '16525',
        '16526',
        '16526',
        '16470',
        '16524',
        '16527',
        '16529',
        '16527',
        '16525',
        '16523',
        '16527',
        '16473',
        '16577',
        '16528',
        '16467',
        '16582',
        '16528',
        '16469',
        '16583',
        '16470',
        '16523',
        '16526' ] },

```

### Tunnel Traffic

```
/IPMPLSView/API/v1/tenant/1/trafficData/1/tunnelTraffic?routerName=5_PARIS&limit=2
```

```

[{"tunnelID":152,"timeSig":0,"collectedTS":0,"status":-
2,"recordedRoute":"null","ingressBytes":0,"ingressBytesDelta":0,"ingressPackets":0,"ingressPacket
sDelta":0,"routerID":1,"toRouterID":9,"tunnelName":"R5_PARIS1_DUBLIN_7"},{"tunnelID":153,"timeSig
":0,"collectedTS":0,"status":-
2,"recordedRoute":"null","ingressBytes":0,"ingressBytesDelta":0,"ingressPackets":0,"ingressPacket
sDelta":0,"routerID":1,"toRouterID":10,"tunnelName":"R5_PARIS2_AMSTERDAM_8"}]

```

### Interface Traffic

```
/IPMPLSView/API/v1/tenant/1/trafficData/1/interfaceTraffic?routerName=5_PARIS&limit=10
(partial sample output)
```

```

[{"interfaceID":350,"timeSig":0,"collectedTS":0,"status":-
2,"ingressBytes":0,"ingressBytesDelta":0,"ingressUtil":0,"ingressPackets":0,"ingressPacketsDelta"

```

```
:0,"egressBytes":0,"egressBytesDelta":0,"egressUtil":0,"egressPackets":0,"egressPacketsDelta":0,"
interfaceDesc":"lo0.0"}, {"interfaceID":351,"timeSig":0,"collectedTS":0,"status":-
2,"ingressBytes":0,"ingressBytesDelta":0,"ingressUtil":0,"ingressPackets":0,"ingressPacketsDelta"
:0,"egressBytes":0,"egressBytesDelta":0,"egressUtil":0,"egressPackets":0,"egressPacketsDelta":0,
"interfaceDesc":"pimd"}, {"interfaceID":352,"timeSig":0,"collectedTS":0,"status":-
2,"ingressBytes":0,"ingressBytesDelta":0,"ingressUtil":0,"ingressPackets":0,"ingressPacketsDelta"
:0,"egressBytes":0,"egressBytesDelta":0,"egressUtil":0,"egressPackets":0,"egressPacketsDelta":0,"
interfaceDesc":"mtun"}, {"interfaceID":353,"timeSig":0,"collectedTS":0,"status":-
2,"ingressBytes":0,"ingressBytesDelta":0,"ingressUtil":0,"ingressPackets":0,"ingressPacketsDelta"
:0,"egressBytes":0,"egressBytesDelta":0,"egressUtil":0,"egressPackets":0,"egressPacketsDelta":0,"
interfaceDesc":"ge-0/0/1.32767"}, {"interfaceID":354,"timeSig":0,"collectedTS":0,"status":-
2,"ingressBytes":0,"ingressBytesDelta":0,"ingressUtil":0,"ingressPackets":0,"ingressPacketsDelta"
:0,"egressBytes":0,"egressBytesDelta":0,"egressUtil":0,"egressPackets":0,"egressPacketsDelta":0,"
interfaceDesc":"lsq-0/0/0"}, {"interfaceID":355,"timeSig":0,"collectedTS":0,"status":-
2,"ingressBytes":0,"ingressBytesDelta":0,"ingressUtil":0,"ingressPackets":0,"ingressPacketsDelta"
:0,"egressBytes":0,"egressBytesDelta":0,"egressUtil":0,"egressPackets":0,"egressPacketsDelta":0,"
interfaceDesc":"lo0.16384"}, {"interfaceID":356,"timeSig":0,"collectedTS":0,"status":-
2,"ingressBytes":0,"ingressBytesDelta":0,"ingressUtil":0,"ingressPackets":0,"ingressPacketsDelta"
:0,"egressBytes":0,"egressBytesDelta":0,"egressUtil":0,"egressPackets":0,"egressPacketsDelta":0,"
interfaceDesc":"ge-0/0/1.1457"}, {"interfaceID":357,"timeSig":0,"collectedTS":0,"status":-
2,"ingressBytes":0,"ingressBytesDelta":0,"ingressUtil":0,"ingressPackets":0,"ingressPacketsDelta"
:0,"egressBytes":0,"egressBytesDelta":0,"egressUtil":0,"egressPackets":0,"egressPacketsDelta":0,"
interfaceDesc":"ip-0/0/0"}, {"interfaceID":358,"timeSig":0,"collectedTS":0,"status":-
2,"ingressBytes":0,"ingressBytesDelta":0,"ingressUtil":0,"ingressPackets":0,"ingressPacketsDelta"
:0,"egressBytes":0,"egressBytesDelta":0,"egressUtil":0,"egressPackets":0,"egressPacketsDelta":0,"
interfaceDesc":"gr-0/0/0"}, {"interfaceID":359,"timeSig":0,"collectedTS":0,"status":-
2,"ingressBytes":0,"ingressBytesDelta":0,"ingressUtil":0,"ingressPackets":0,"ingressPacketsDelta"
:0,"egressBytes":0,"egressBytesDelta":0,"egressUtil":0,"egressPackets":0,"egressPacketsDelta":0,"
interfaceDesc":"sp-0/0/0"}]}
```

### Cos Traffic

```
/IPMPLSView/API/v1/tenant/1/trafficData/1/cosTraffic?routerName=4_BERLIN
&interfaceDesc=ge-0/0/1.424
{
  "interfaceDesc":"ge-0/0/1.424",
  "queueID":45,
  "queueKey":"336+2+0",
  "routerID":2,
  "interfaceID":336,
  "queueNumber":0,
  "queueName":"INTERNET",
  "inOutType":2,
  "timeSig":0,
  "queuedBytesCounter":0,
  "queuedBytesDelta":0,
  "transmittedBytesCounter":0,
  "transmittedBytesDelta":0,
  "totalDroppedBytesCounter":0,
  "totalDroppedBytesDelta":0
},
{
  "interfaceDesc":"ge-0/0/1.424",
  "queueID":45,
  "queueKey":"336+2+0",
  "routerID":2,
  "interfaceID":336,
  "queueNumber":0,
  "queueName":"INTERNET",
  "inOutType":2,
  "timeSig":20150915182,
```

```

"queuedBytesCounter":45168572,
"queuedBytesDelta":null,
"transmittedBytesCounter":45168572,
"transmittedBytesDelta":null,
"totalDroppedBytesCounter":0,
"totalDroppedBytesDelta":null
},
{
"interfaceDesc":"ge-0/0/1.424",
"queueID":45,
"queueKey":"336+2+0",
"routerID":2,
"interfaceID":336,
"queueNumber":0,
"queueName":"INTERNET",
"inOutType":2,
"timeSig":20150915184,
"queuedBytesCounter":45171148,
"queuedBytesDelta":8,
"transmittedBytesCounter":45171148,
"transmittedBytesDelta":8,
"totalDroppedBytesCounter":0,
"totalDroppedBytesDelta":0
},
{
"interfaceDesc":"ge-0/0/1.424",
"queueID":45,
"queueKey":"336+2+0",
"routerID":2,
6 Copyright © 2016, Juniper Networks, Inc.
"interfaceID":336,
"queueNumber":0,
"queueName":"INTERNET",
"inOutType":2,
"timeSig":20150915185,
"queuedBytesCounter":45171352,
"queuedBytesDelta":0,
"transmittedBytesCounter":45171352,
"transmittedBytesDelta":0,
"totalDroppedBytesCounter":0,
"totalDroppedBytesDelta":0
},
{
"interfaceDesc":"ge-0/0/1.424",
"queueID":46,
"queueKey":"336+2+1",
"routerID":2,
"interfaceID":336,
"queueNumber":1,
"queueName":"BUSINESS",
"inOutType":2,
"timeSig":20150915237,
"queuedBytesCounter":0,
"queuedBytesDelta":0,
"transmittedBytesCounter":0,
"transmittedBytesDelta":0,
"totalDroppedBytesCounter":0,
"totalDroppedBytesDelta":0
},
{
"interfaceDesc":"ge-0/0/1.424",

```

```
"queueID":46,
"queueKey":"336+2+1",
"routerID":2,
"interfaceID":336,
"queueNumber":1,
"queueName":"BUSINESS",
"inOutType":2,
"timeSig":20150915238,
"queuedBytesCounter":0,
"queuedBytesDelta":0,
"transmittedBytesCounter":0,
"transmittedBytesDelta":0,
"totalDroppedBytesCounter":0,
"totalDroppedBytesDelta":0
}
{
"interfaceDesc":"ge-0/0/1.424",
"queueID":48,
"queueKey":"336+2+3",
"routerID":2,
"interfaceID":336,
"queueNumber":3,
"queueName":"CONTROL",
"inOutType":2,
"timeSig":20150915185,
"queuedBytesCounter":278741283,
"queuedBytesDelta":15,
"transmittedBytesCounter":278741283,
"transmittedBytesDelta":15,
"totalDroppedBytesCounter":0,
"totalDroppedBytesDelta":0
},
. . .
]
```

### Multicast Traffic

```
/IPMPLSView/API/v1/tenant/1/trafficData/1/multiCastTraffic?routerName=12_MUNICH
[
{
"interfaceID":466,
"interfaceDesc":"GigabitEthernet2.542",
"timeSig":20150915182,
"mcastInOctetsCounter":23666382708,
"mcastInOctetsDelta":0,
"mcastInUtil":0,
"mcastOutOctetsCounter":0,
"mcastOutOctetsDelta":0,
"mcastOutUtil":0
},
{
"interfaceID":466,
"interfaceDesc":"GigabitEthernet2.542",
"timeSig":20150915184,
"mcastInOctetsCounter":23671007772,
"mcastInOctetsDelta":15416,
"mcastInUtil":0.0123328,
"mcastOutOctetsCounter":0,
"mcastOutOctetsDelta":0,
"mcastOutUtil":0
}
```



```

    },
    {
      "interfaceID":466,
      "interfaceDesc":"GigabitEthernet2.542",
      "timeSig":20150915185,
      "mcastInOctetsCounter":23675541072,
      "mcastInOctetsDelta":15111,
      "mcastInUtil":0.0120888,
      "mcastOutOctetsCounter":0,
      "mcastOutOctetsDelta":0,
      "mcastOutUtil":0
    },
    {
      "interfaceID":466,
      "interfaceDesc":"GigabitEthernet2.542",
      "timeSig":20150915186,
      "mcastInOctetsCounter":23680071636,
      "mcastInOctetsDelta":15101,
      "mcastInUtil":0.0120808,
      "mcastOutOctetsCounter":0,
      "mcastOutOctetsDelta":0,
      "mcastOutUtil":0
    },
    {
      "interfaceID":466,
      "interfaceDesc":"GigabitEthernet2.542",
      "timeSig":20150915187,
      "mcastInOctetsCounter":23684540964,
      "mcastInOctetsDelta":14897,
      "mcastInUtil":0.0119176,
      "mcastOutOctetsCounter":0,
      "mcastOutOctetsDelta":0,
      "mcastOutUtil":0
    },
    {
      "interfaceID":466,
      "interfaceDesc":"GigabitEthernet2.542",
      "timeSig":20150915188,
      "mcastInOctetsCounter":23689124028,
      "mcastInOctetsDelta":15276,
      "mcastInUtil":0.0122208,
      "mcastOutOctetsCounter":0,
      "mcastOutOctetsDelta":0,
      "mcastOutUtil":0
    },
    {
      "interfaceID":466,
      "interfaceDesc":"GigabitEthernet2.542",
      "timeSig":20150915189,
      "mcastInOctetsCounter":23693670828,
      "mcastInOctetsDelta":15156,
      "mcastInUtil":0.0121248,
      "mcastOutOctetsCounter":0,
      "mcastOutOctetsDelta":0,
      "mcastOutUtil":0
    },
    {
      "interfaceID":466,
      "interfaceDesc":"GigabitEthernet2.542",
      "timeSig":20150915190,

```

```
"mcastInOctetsCounter":23698244892,  
"mcastInOctetsDelta":15246,  
"mcastInUtil":0.0121968,  
"mcastOutOctetsCounter":0,  
"mcastOutOctetsDelta":0,  
"mcastOutUtil":0  
},  
.....  
.....  
.....  
]
```

**Related Documentation** • [Understanding the IP/MPLSView Traffic Data API on page 63](#)