API REFERENCE GUIDE

Extreme Virtual TAP 2.0.0 REST API Guide

9035940-00
13 February 2019

Legal Notice

Extreme Networks, Inc. reserves the right to make changes in specifications and other information contained in this document and its website without prior notice. The reader should in all cases consult representatives of Extreme Networks to determine whether any such changes have been made.

The hardware, firmware, software or any specifications described or referred to in this document are subject to change without notice.

Trademarks

Extreme Networks and the Extreme Networks logo are trademarks or registered trademarks of Extreme Networks, Inc. in the United States and/or other countries.

All other names (including any product names) mentioned in this document are the property of their respective owners and may be trademarks or registered trademarks of their respective companies/owners.

For additional information on Extreme Networks trademarks, please see: www.extremenetworks.com/company/legal/trademarks

Open Source Declarations

Some software files have been licensed under certain open source or third-party licenses. End-user license agreements and open source declarations can be found at: www.extremenetworks.com/support/policies/software-licensing

Contents

Preface

This section discusses the conventions used in this guide, ways to provide feedback, additional help, and other Extreme Networks® publications.

Conventions

This section discusses the conventions used in this guide.

Notes, cautions, and warnings

Notes, cautions, and warning statements may be used in this document. They are listed in the order of increasing severity of potential hazards.

NOTE
A Note provides a tip, guidance, or advice, emphasizes important information, or provides a reference to related information.

ATTENTION
An Attention statement indicates a stronger note, for example, to alert you when traffic might be interrupted or the device might reboot.

CAUTION
A Caution statement alerts you to situations that can be potentially hazardous to you or cause damage to hardware, firmware, software, or data.

DANGER
A Danger statement indicates conditions or situations that can be potentially lethal or extremely hazardous to you. Safety labels are also attached directly to products to warn of these conditions or situations.

Text formatting conventions

Text formatting conventions such as boldface, italic, or Courier font may be used to highlight specific words or phrases.

Format	Description
bold text	Identifies command names.
	Identifies keywords and operands.
	Identifies the names of GUI elements.
	Identifies text to enter in the GUI.
italic text	Identifies emphasis.
	Identifies variables.
	Identifies document titles.

Format	Description
`Courier font`	Identifies CLI output. Identifies command syntax examples.

Command syntax conventions

Bold and italic text identify command syntax components. Delimiters and operators define groupings of parameters and their logical relationships.

Convention	Description
bold text	Identifies command names, keywords, and command options.
italic text	Identifies a variable.
[ ]	Syntax components displayed within square brackets are optional. Default responses to system prompts are enclosed in square brackets.
{ **x** | **y** | **z** }	A choice of required parameters is enclosed in curly brackets separated by vertical bars. You must select one of the options.
x | **y**	A vertical bar separates mutually exclusive elements.
< >	Nonprinting characters, for example, passwords, are enclosed in angle brackets.
...	Repeat the previous element, for example, *member*[*member*...].
\	Indicates a "soft" line break in command examples. If a backslash separates two lines of a command input, enter the entire command at the prompt without the backslash.

Documentation and Training

To find Extreme Networks product guides, visit our documentation pages at:

Current Product Documentation	www.extremenetworks.com/documentation/
Archived Documentation (for earlier versions and legacy products)	www.extremenetworks.com/support/documentation-archives/
Release Notes	www.extremenetworks.com/support/release-notes
Hardware/Software Compatibility Matrices	https://www.extremenetworks.com/support/compatibility-matrices/
White papers, data sheets, case studies, and other product resources	https://www.extremenetworks.com/resources/

Training

Extreme Networks offers product training courses, both online and in person, as well as specialized certifications. For more information, visit www.extremenetworks.com/education/.

Getting Help

If you require assistance, contact Extreme Networks using one of the following methods:

Extreme Portal	Search the GTAC (Global Technical Assistance Center) knowledge base, manage support cases and service contracts, download software, and obtain product licensing, training, and certifications.
The Hub	A forum for Extreme Networks customers to connect with one another, answer questions, and share ideas and feedback. This community is monitored by Extreme Networks employees, but is not intended to replace specific guidance from GTAC.
Call GTAC	For immediate support: 1-800-998-2408 (toll-free in U.S. and Canada) or +1 408-579-2826. For the support phone number in your country, visit: www.extremenetworks.com/support/contact

Before contacting Extreme Networks for technical support, have the following information ready:

- Your Extreme Networks service contract number and/or serial numbers for all involved Extreme Networks products
- A description of the failure
- A description of any action(s) already taken to resolve the problem
- A description of your network environment (such as layout, cable type, other relevant environmental information)
- Network load at the time of trouble (if known)
- The device history (for example, if you have returned the device before, or if this is a recurring problem)
- Any related RMA (Return Material Authorization) numbers

Subscribing to Service Notifications

You can subscribe to email notifications for product and software release announcements, Vulnerability Notices, and Service Notifications.

1. Go to www.extremenetworks.com/support/service-notification-form.
2. Complete the form with your information (all fields are required).
3. Select the products for which you would like to receive notifications.

   NOTE
   You can modify your product selections or unsubscribe at any time.

4. Click **Submit**.

Providing Feedback to Us

Quality is our first concern at Extreme Networks, and we have made every effort to ensure the accuracy and completeness of this document. We are always striving to improve our documentation and help you work better, so we want to hear from you! We welcome all feedback but especially want to know about:

- Content errors or confusing or conflicting information.
- Ideas for improvements to our documentation so you can find the information you need faster.
- Broken links or usability issues.

If you would like to provide feedback to the Extreme Networks Information Development team, you can do so in two ways:

- Use our short online feedback form at https://www.extremenetworks.com/documentation-feedback/.
- Email us at documentation@extremenetworks.com.

Please provide the publication title, part number, and as much detail as possible, including the topic heading and page number if applicable, as well as your suggestions for improvement.

Getting started

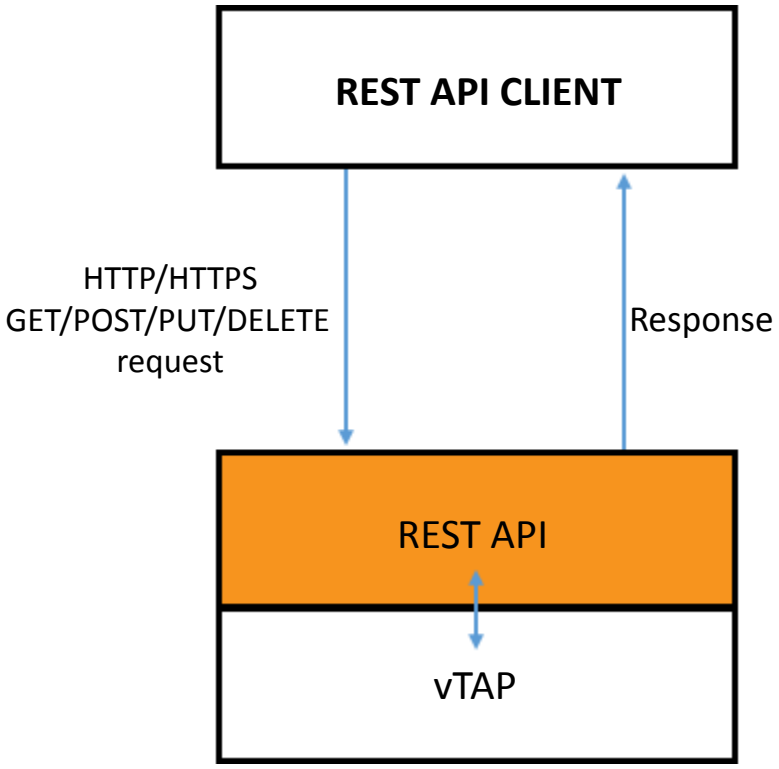
Overview

The Virtual TAP (vTAP) REST API provides a Web services interface for configuring vTAP.

The API adheres to Representational State Transfer (REST) principles where possible, and uses the JavaScript Object Notation (JSON) format for data representation. You can use third-party REST API tools, such as cURL for command line, or browser extensions like Postman or RestClient to execute APIs.

The vTAP REST API is based on HTTP and HTTPS requests. The default HTTP port number is 80 and HTTPS port number is 443.

FIGURE 1 vTAP API architecture



Protocol support

The vTAP REST API supports both HTTP and HTTPS protocols.

> NOTE
> For HTTPS to work on clients such as Firefox or Postman, the vTAP IP needs to be added as a security exception.

By default, the HTTP port number is 80 and HTTPS port number is 443. If required, these ports can be changed by updating the file `nginx.conf`:

> NOTE
> In the `nginx.conf` file, there are two occurrences of HTTPS port number and one occurrence of HTTP port number.

1. Open the file `/etc/vtap/nginx/nginx.conf`.
2. For changing HTTPS port, edit the value of the property `listen`.

   Change 443 to any other port number in `listen 443 default_server ssl;`:

```
server {
        listen       443 default_server ssl;
        server_name  10.37.129.91;

        ssl_certificate /etc/ssl/certs/nginx-selfsigned.crt;
        ssl_certificate_key /etc/ssl/private/nginx-selfsigned.key;
        include /etc/vtap/nginx/ssl-params.conf;

        ssl_session_timeout  5m;

        ssl_ciphers  HIGH:!aNULL:!MD5;
        ssl_prefer_server_ciphers  on;

        location /{
            root   html;
            index  index.html index.htm;
        }
    }
```

   Change 443 to any other port number in `listen 443 ssl;`:

```
server {
        listen       80;
        listen       443 ssl;
        server_name  10.37.129.91;

        client_body_buffer_size      2M;
        client_max_body_size         100M;
        client_header_buffer_size    100M;
        large_client_header_buffers  4 40k;
        client_body_in_single_buffer on;
        output_buffers               1 32k;
        postpone_output              1460;
        proxy_send_timeout 159s;
        proxy_read_timeout 360s;
        location /vtap {
          vtap;
        }
```

3. For changing HTTP port, edit the value of the property `listen`.

   Change 80 to any other port number in `listen 80;`:

```
server {
        listen       80;
```

```
listen       443 ssl;
server_name  10.37.129.91;

client_body_buffer_size      2M;
client_max_body_size         100M;
client_header_buffer_size    100M;
large_client_header_buffers  4 40k;
client_body_in_single_buffer on;
output_buffers               1 32k;
postpone_output              1460;
proxy_send_timeout 159s;
proxy_read_timeout 360s;
location /vtap {
  vtap;
}
```

4. Save file and exit.
5. Run the following command to restart nginx:

```
systemctl restart vtap_nginx
```

URI

You can invoke an API command or request by sending an HTTP or HTTPS message to vTAP using a Uniform Resource Identifier (URI).

```
{http | https}://<host>:<port>/vtap/
```

The URI consists of two parts:

- **Base URI**: The base URI is the entry point to access and manage all the resources defined in the system, and is used to perform PUT, POST and DELETE requests.

  Example: `http://10.37.129.91:80/vtap`
- **Request URI**: The request URI, along with the base URI is used to perform GET requests.

  Example: `http://10.37.129.91:80/vtap/flow-info@all.json`

  NOTE
  URIs are case-sensitive.

The components of the URI are as follows:

- **http://** or **https://**: This specifies if the request is HTTP or HTTPS.
- **host**: This specifies the IP address of vTAP.
- **port**: This specifies the HTTP or HTTPS port number. For information about configuring the port number, see the section Protocol support on page 12.
- **/vtap**: This is the extension for vTAP.

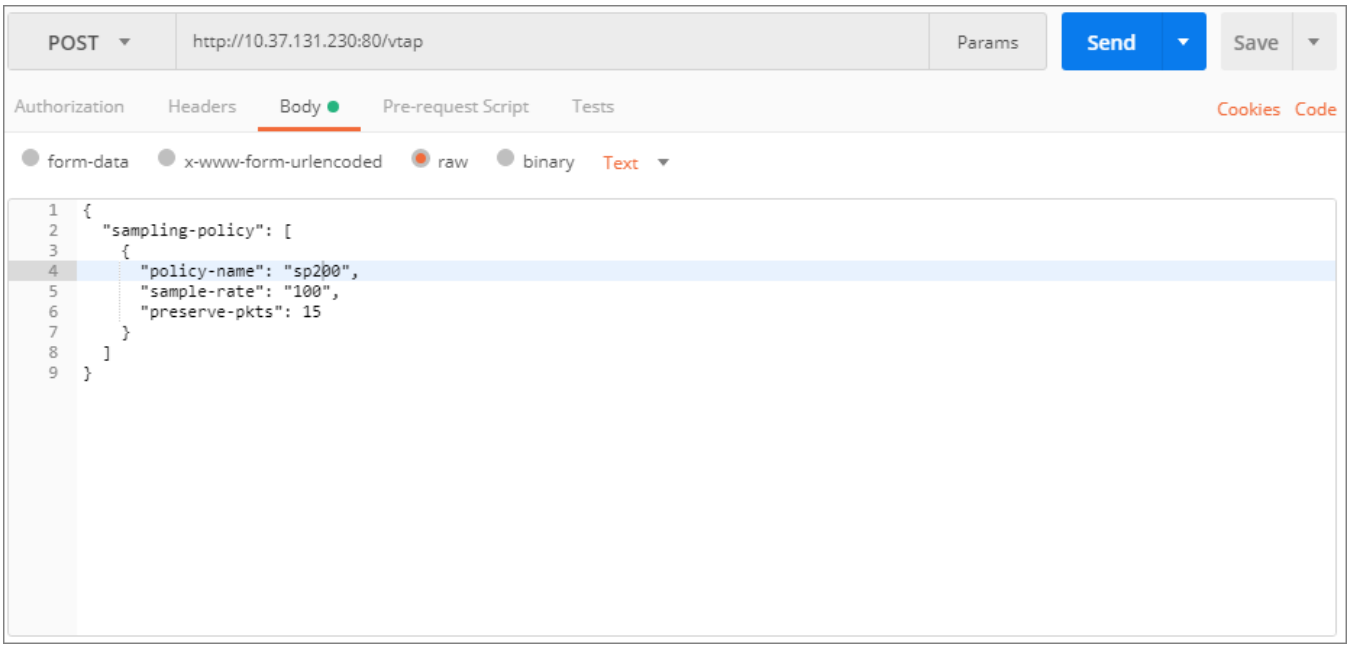
Supported operations

All create, read, update, and delete (CRUD) operations are supported and performed using the following methods:

- **POST**: Create a new resource in the specific resource location identified by the URI specified in the given request.

The image below is an example of how a REST client can be used to send a POST request:

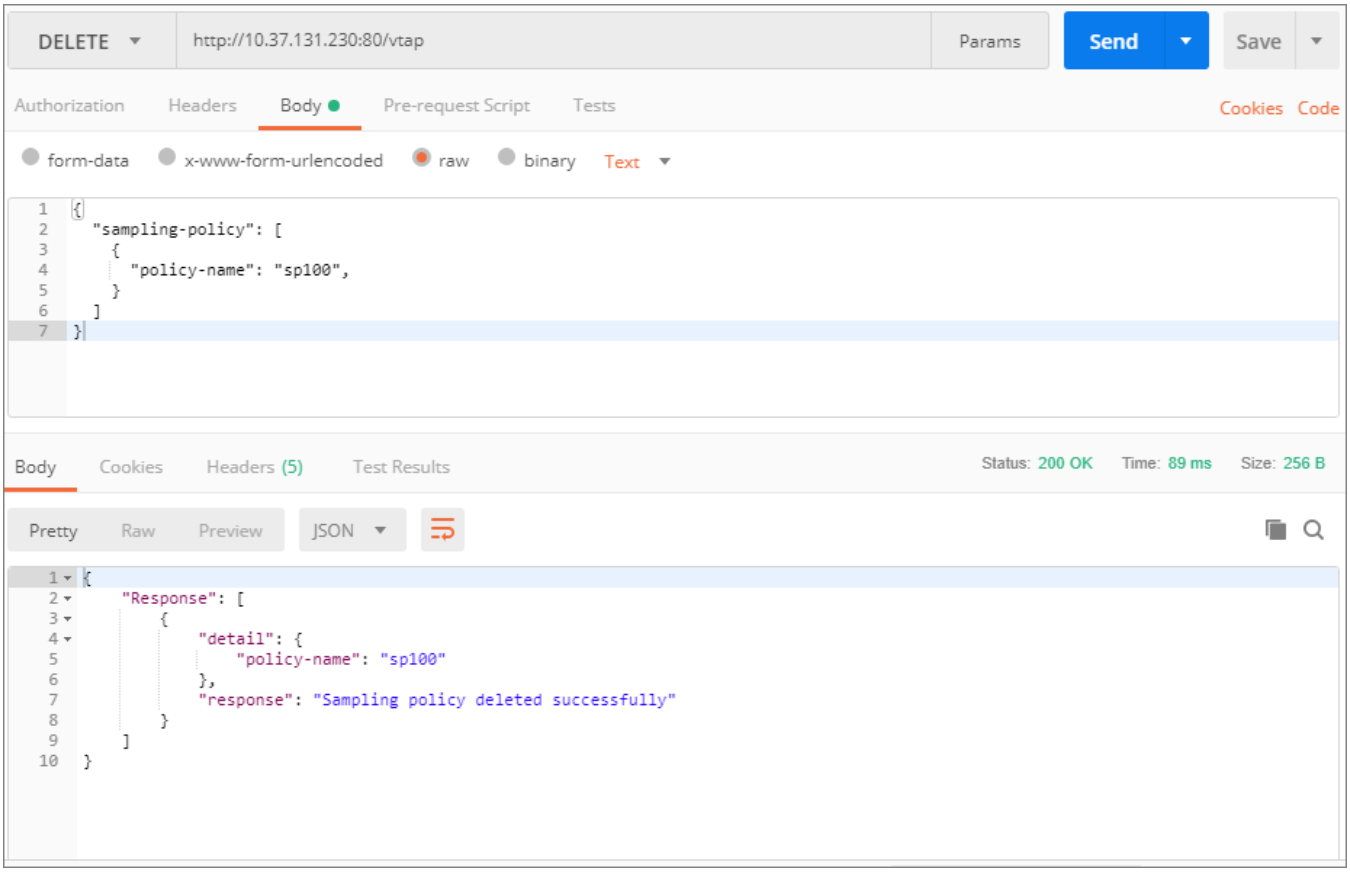
FIGURE 2 POST method using REST client



- **DELETE**: Delete an existing resource.

  The image below is an example of how a REST client can be used to send a DELETE request:

FIGURE 3 DELETE method using REST client

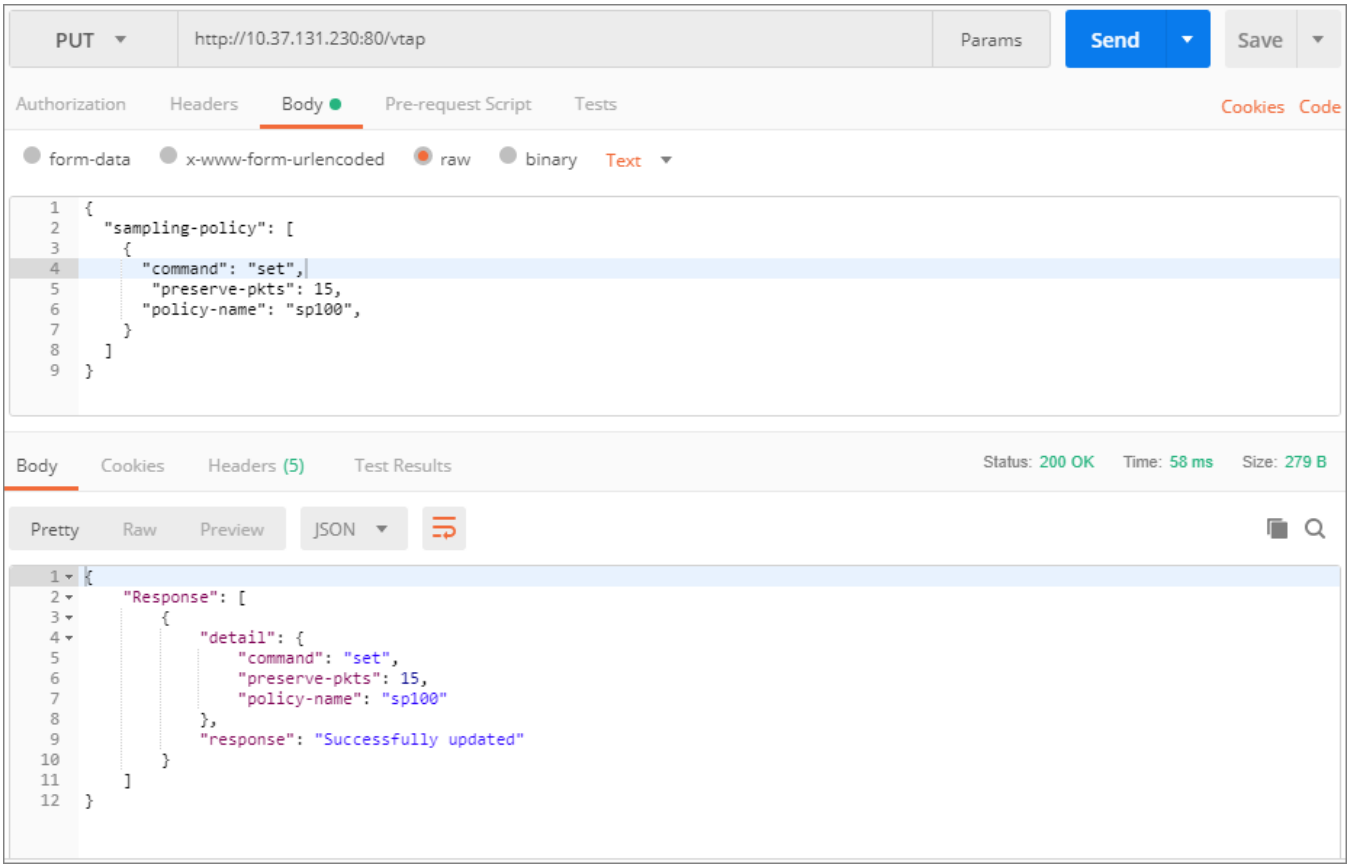


- **PUT**: Apply changes to a resource. Where the resource exists, only those properties specified in the request are modified; all others remain unchanged.

  The image below is an example of how a REST client can be used to send a PUT request:

  NOTE
  When using PUT method for clearing properties, the value (in the key:value pair) for a property in the request body can either be empty or contain any character. The property is cleared irrespective of the value.

FIGURE 4 PUT method using REST client



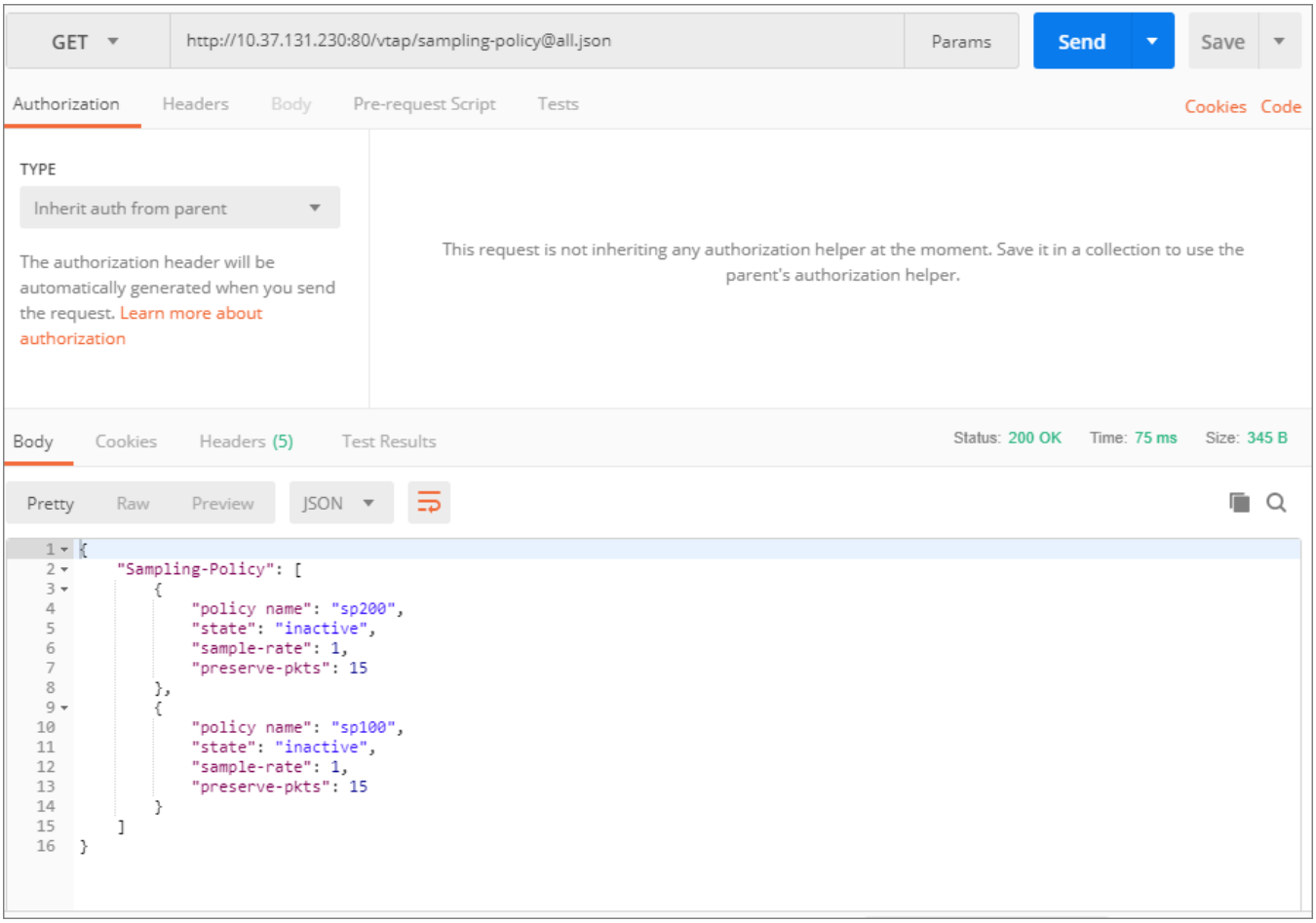
- **GET**: Obtain a representation of the resource/information without modifying anything on the system.

  The image below is an example of how a REST client can be used to send a GET request.

  NOTE

  A request payload is not required for a GET operation. As shown in the image below, the **Body** field is empty.

FIGURE 5 GET method using REST client



cURL commands

You can invoke an API request by sending an HTTP or HTTPS message to vTAP using cURL commands.

cURL command structure

The cURL command can vary depending on whether it is a configuration command (POST, PUT and DELETE) or a show/clear command (GET). In addition, HTTPS requests take an extra option (--insecure).

Following is the basic structure of a cURL command:

- **HTTP request**

```
curl -v -X <method> -H "Expect:" "http://<host>:<port>/vtap" -d @/<filename>.json
```

- **HTTPS request**

```
curl -v -X <method> -H "Expect:" "https://<host>:<port>/vtap" -d @/<filename>.json --insecure
```

Where:

- **method**: Is the API method, PUT, POST, DELETE or GET.
- **host**: Specifies the IP address of vTAP.
- **port**: Specifies the HTTP or HTTPS port number. For information about configuring the port number, see the section Protocol support on page 12.
- **-d:** Is required for PUT, POST and DELETE requests only. `-d` with `@/<filename>.json` indicates to send the content of `<filename>.json` to the server.
- **<filename>.json**: Is required for PUT, POST and DELETE requests only. This is a user-defined JSON file that contains the request body for the API request. Provide the full path to the file. This file must be located on the server on which cURL is hosted.

  NOTE
  For GET requests, do not include `-d @/<filename>.json` in the command.

- **--insecure**: Is required for HTTPS requests only.

POST, PUT and DELETE cURL command

Perform the following steps to send a POST, PUT or DELETE cURL call:

1. Create a local file that contains the JSON command body.

   The JSON command body is based on the key-value pair defined for each POST, PUT and DELETE method.

2. Save the file as `.json`. For example, `sp11.json`.
3. On the cURL command line, use the following format to specify the HTTP/HTTPS method (such as POST), the URI, and the path and name of the local file. Make sure to include the @ symbol before the filename:
   - **HTTP request**

     ```
     curl -v -X <method> -H "Expect:" "http://<host>:<port>/vtap" -d @/<filename>.json
     ```

   - **HTTPS request**

     ```
     curl -v -X <method> -H "Expect:" "https://<host>:<port>/vtap" -d @/<filename>.json --insecure
     ```

Example

The following is a POST method for a Sampling policy.

The filename is `sp1.json`, located in the `/root` directory. The file contains the request body for a sampling policy:

```
{
  "sampling-policy": [
    {
      "policy-name": "sp100",
      "sample-rate": "100",
      "preserve-pkts": 15
    }
  ]
}
```

After saving the file, an HTTPS POST method is executed as follows:

```
curl -v -X POST -H "Expect:" "https://10.37.129.91:443/vtap" -d @/root/sp1.json --insecure
```

GET cURL command

GET requests are used for fetching information or for clearing statistics and hence, do not require a user-defined file. To send a GET request, enter the cURL command in the following format:

- **HTTP request**

```
curl -v -X <method> -H "Expect:" "http://<host>:<port>/vtap/<json_request>"
```

- **HTTPS request**

```
curl -v -X <method> -H "Expect:" "https://<host>:<port>/vtap/<json_request>" --insecure
```

Example 1

The following is a GET request for retrieving information about the interface configuration. Unlike PUT, POST and DELETE requests, in this GET request, `interface-config@all.json` is a keyword and not a user-defined filename.

```
curl -v -X POST -H "Expect:" "https://10.37.129.91:80/vtap/interface-config@all.json" --insecure
```

Example 2

The following is a GET method for retrieving information about a specific egress alias named `egress-1`. In this case, it is assumed that the egress alias `egress-1` was created earlier using the POST method.

```
curl -v -X POST -H "Expect:" "https://10.37.129.91:80/vtap/egress-config@egress-1.json" --insecure
```

Interacting with the API

Interactions with vTAP consist of an HTTP/HTTPs request to the vTAP system followed by a response from the system. Each request includes in its HTTP/HTTPS header a command, the address of the remote system, and the format of the response body that is expected.

HTTP status code and messages

Both success and error status are reported to the client through HTTP/HTTPS Status-Line, which contains the HTTP/HTTPS status code.

TABLE 1 Status code

Status-Line	Description
200 OK	Success with response body
204 No Content	Success without response body
400 Bad Request	Invalid request message

Before you begin

Before using vTAP REST API:

- Ensure that vTAP is already installed.

NOTE
For information about installing vTAP, see *Extreme Virtual TAP Installation Guide*.

- Ensure that all vTAP services are running. Run the `status all` CLI command to verify.
- Ensure that you have a tool for interacting with REST APIs.

Egress

Configuring egress

Method

POST

Resource URI

```
{http | https}://<host>:<port>/vtap
```

NOTE

- For IPv4, the parameters `local-ip` and `nexthop-mac` are optional. For IPv6, the parameter `local-ip` is optional, but `nexthop-mac` is mandatory. If the optional parameters are not configured, vTAP auto-detects the value.
- The destination port for VXLAN is set to 4789 by default and cannot be changed.
- vTAP supports a maximum of 10 egress tunnels.

Examples

URI

```
http://10.37.129.91:80/vtap
```

Request Body - For GRE

```
{
  "egress-config": [
    {
      "egress-alias": "egress-1",
      "local-ip": "10.37.129.4",
      "destination-ip": "10.37.129.193",
      "nexthop-mac": "11:12:13:14:15:16",
      "tunnel-type": "gre"
    }
  ]
}
```

Request Body - For VXLAN

```
{
  "egress-config": [
    {
      "egress-alias": "egress-3",
      "local-ip": "1.1.1.1",
      "destination-ip": "2.2.2.2",
      "vni": "4023"
    }
  ]
}
```

cURL command

```
curl -v -X POST -H "Expect:" "https://10.37.129.91:443/vtap" -d @/root/egress_gre1.json --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Response": [
        {
            "detail": {
                "egress-alias": "egress-1",
                "local-ip": "10.37.129.4",
                "destination-ip": "10.37.129.193",
                "nexthop-mac": "11:12:13:14:15:16",
                "tunnel-type": "gre"
            },
            "response": "Successfully configured"
        }
    ]
}
```

Error Response Body

```
400 Bad Request
{
  "Response": [
    {
      "detail": {
        "egress-alias": "egress-1",
        "local-ip": "10.37.129.4",
        "destination-ip": "10.37.129.193",
        "nexthop-mac": "11:12:13:14:15:16",
        "tunnel-type": "gre"
      },
      "error-code": -2,
      "error-message": "egress-alias:egress-1 is already configured"
    }
  ]
}
```

Deleting egress alias

Method

DELETE

Resource URI

```
{http | https}://<host>:<port>/vtap
```

Examples

URI

```
http://10.37.129.91:80/vtap
```

Request Body

```
{
  "egress-config": [
    {
      "egress-alias": "egress-1"
    }
  ]
}
```

cURL command

```
curl -v -X DELETE -H "Expect:" "https://10.37.129.91:443/vtap" -d @/root/delete_egress_alias.json --
insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Response": [
        {
            "detail": {
                "egress-alias": "egress-1"
            },
            "response": "Successfully Deleted"
        }
    ]
}
```

Error Response Body

Port deletion failed because the requested port is not available.

```
400 Bad Request
{
    "Response": [
        {
            "detail": {
                "egress-alias": "egress-2"
            },
            "error-code": -2,
            "error-message": "Error: egress-alias egress-2 not configured"
        }
      ]
}
```

Getting egress alias configuration

Retrieve configuration information for all or specific egress alias from vTAP.

Method

GET

All egress aliases

Resource URI

```
{http | https}://<host>:<port>/vtap/egress-config@all.json
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get information about all egress aliases.

URI

```
http://10.37.129.91:80/vtap/egress-config@all.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/egress-config@all.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "egress-config": [
        {
            "egress alias": "egress-1",
            "state": "inactive",
            "local-ip": "1.1.1.1",
            "destination-ip": "2.2.2.2",
            "tunnel-type": "gre",
            "nexthop-mac": "11:12:13:14:15:16"
        },
        {
            "egress alias": "egress-2",
            "state": "inactive",
            "local-ip": "10.37.129.4",
            "destination-ip": "10.37.129.193",
            "tunnel-type": "gre",
            "nexthop-mac": "11:12:13:14:15:16"
        }
    ]
}
```

Individual egress alias

Resource URI

{http | https}://<host>:<port>/vtap/egress-config@*<egress_alias_name>*.json

Examples

URI

```
http://10.37.129.91:80/vtap/egress-config@egress-1.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/egress-config@egress-1.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "egress-config": [
        {
            "egress alias": "egress-1",
            "state": "inactive",
            "local-ip": "1.1.1.1",
            "destination-ip": "2.2.2.2",
            "tunnel-type": "gre",
            "nexthop-mac": "11:12:13:14:15:16"
        }
    ]
}
```

Error Response Body

```
400 Bad Request
{
  "Response": [
    {
      "egress-name": "egress-1"
    }
  ],
  "error code": 6,
  "error message": "not configured"
}
```

Getting egress tunnel information

Retrieve information about the tunnels received by vTAP.

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/tunnel-info@all.json
```

Examples

URI

```
http://10.37.129.91:80/vtap/tunnel-info@all.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/tunnel-info@all.json" --insecure
```

Success Response Body

```
204 No Content
```

or


```
{
    "tunnel-info": [
        {
            "tunnel-id": 2,
            "parent-tunnel-id": 0,
            "tunnel-type": "gre",
            "smatch rule id": "-",
            "host1-ip": "10.0.0.1",
            "host2-ip": "10.0.0.2",
            "child tunnels": 0,
            "flows": 1,
            "dir1 rx packets": 5,
            "dir1 rx bytes": 690,
            "dir2 rx packets": 5,
            "dir2 rx bytes": 690,
            "egress-action": "drop"
        }
    ]
}
```

Sampling policy

Configuring sampling policy

Method

POST

Resource URI

```
{http | https}://<host>:<port>/vtap
```

Examples

URI

```
http://10.37.129.91:80/vtap
```

Request Body

```
{
  "sampling-policy": [
    {
      "policy-name": "sp100",
      "sample-rate": "100",
      "preserve-pkts": 15
    }
  ]
}
```

cURL command

```
curl -v -X POST -H "Expect:" "https://10.37.129.91:443/vtap" -d @/root/sampling1.json --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
  "Response": [
    {
      "detail": {
        "policy-name": "sp100",
        "sample-rate": "100",
        "preserve-pkts": 15
      },
      "response": "Successfully configured"
    }
  ]
}
```

Error Response Body

```
400 Bad Request
{
  "Response": [
    {
      "detail": {
        "policy-name": "sp100",
        "sample-rate": "100",
        "preserve-pkts": 15
      },
      "error-code": -2,
      "error-message": "Policy sp100 already exists"
    }
  ]
}
```

Setting/clearing sampling policy parameters

Method

PUT

Resource URI

```
{http | https}://<host>:<port>/vtap
```

Examples

URI

```
http://10.37.129.91:80/vtap
```

Request Body - Set

```
{
  "sampling-policy": [
    {
      "command": "set",
      "policy-name": "sp100",
       "sample-rate": 10,
       "preserve-pkts": 15,

    }
  ]
}
```

Request Body - Clear

```
{
  "sampling-policy": [
    {
      "command": "clear",
      "policy-name": "sp100",
    }
  ]
}
```

cURL command

```
curl -v -X PUT -H "Expect:" "https://10.37.129.91:443/vtap" -d @/root/put_sampling1.json --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
  "Response": [
    {
      "detail": {
        "policy-name": "sp100",
        "command": "set",
        "sample-rate": "100",
        "preserve-pkts": 15
      },
      "response": "Successfully updated"
    }
  ]
}
```

Error Response Body

```
400 Bad Request
{
  "Response": [
    {
      "detail": {
        "policy-name": "sp100",
        "command": "set",
        "sample-rate": "100",
        "preserve-pkts": 15
      },
      "error-code": -2,
      "error-message": "Info: Sampling rate already set"
    }
  ]
}
```

Deleting sampling policy

Method

DELETE

Resource URI

```
{http | https}://<host>:<port>/vtap
```

Examples

URI

```
http://10.37.129.91:80/vtap
```

Request Body

```
{
  "sampling-policy": [
    {
      "policy-name": "sp100",
    }
  ]
}
```

cURL command

```
curl -v -X DELETE -H "Expect:" "https://10.37.129.91:443/vtap" -d @/root/delete_sampling1.json --
insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
  "Response": [
    {
      "detail": {
        "policy-name": "sp10"
      },
      "response": "Sampling policy deleted successfully"
    }
  ]
}
```

Error Response Body

```
400 Bad Request
{
  "Response": [
    {
      "detail": null,
      "error-code": -2,
      "error-message": "Sampling policy doesn't exists"
    }
  ]
}
```

Getting sampling policy configuration

Retrieve configuration information for all or specific sampling policies from vTAP.

Method

GET

All sampling policies

Resource URI

```
{http | https}://<host>:<port>/vtap/sampling-policy@all.json
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get information about all sampling policies.

URI

```
http://10.37.129.91:80/vtap/sampling-policy@all.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/sampling-policy@all.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Sampling-Policy": [
        {
            "policy name": "sp100",
            "state": "inactive",
            "sample-rate": 100,
            "preserve-pkts": 15
        }
    ]
}
```

Individual sampling policy

Resource URI

```
{http | https}://<host>:<port>/vtap/sampling-policy@<sampling_policy_name>.json
```

Examples

URI

```
http://10.37.129.91:80/vtap/sampling-policy@sp1.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/sampling-policy@sp1.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Sampling-Policy": [
        {
            "policy name": "sp1",
            "state": "active",
            "sample-rate": 10,
            "preserve-pkts": 0
        }
    ]
}
```

Error Response Body

```
400 Bad Request
{
    "Sampling-Policy": [
        {
            "detail": "sp2",
            "response": "sampling policy not present"
        }
    ]
}
```

Getting sampling statistics

Retrieve sampling statistics from vTAP.

Method

GET

All sampling policies

Resource URI

```
{http | https}://<host>:<port>/vtap/sampling-stats@all.json
```

Examples

URI

```
http://10.37.129.91:80/vtap/sampling-stats@all.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/sampling-stats@all.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Sampling-Stats": [
        {
            "policy name": "sp1",
            "sample-rate": 10,
            "total-flow": 30,
            "drop-flow": 3,
            "drop-pkts-count": 2,
            "preserve-pkts-count": 0
        }
    ]
}
```

Individual sampling policy

Resource URI

{http | https}://<host>:<port>vtap/sampling-stats@<*sampling_policy_name*>.json

Examples

URI

http://10.37.129.91:80/vtap/sampling-stats@sp1.json

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/sampling-stats@sp1.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Sampling-Stats": [
        {
            "policy name": "sp1",
            "sample-rate": 10,
            "total-flow": 30,
            "drop-flow": 3,
            "drop-pkts-count": 2,
            "preserve-pkts-count": 0
        }
    ]
}
```

SMARTMatch policy

Configuring alias

Method

POST

Resource URI

```
{http | https}://<host>:<port>/vtap
```

Examples

URI

```
http://10.37.129.91:80/vtap
```

Request Body - Example 1

```
{
  "alias-config": [
    {
      "alias": "alias-1",
      "flex-match": "32:10:cloud,0:0:0x3456"
    }
  ]
}
```

Request Body - Example 2

cURL command

```
curl -v -X POST -H "Expect:" "https://10.37.129.91:443/vtap" -d @/root/alias1.json --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Response": [
        {
            "detail": {
                "alias": "alias-1",
                "flex-match": "32:10:`cloud`,0:0:0x3456"
            },
            "response": "Alias configured successfully"
        }
    ]
}
```

Error Response Body

```
400 Bad Request
{
  "Response": [
    {
      "detail": {
        "alias": "alias-1",
        "flex-match": "32:10:`cloud`,0:0:0x3456"
      },
      "error-code": -2,
      "error-message": "already configured"
    }
  ]
}
```

Deleting alias

Method

DELETE

Resource URI

```
{http | https}://<host>:<port>/vtap
```

Examples

URI

```
http://10.37.129.91:80/vtap
```

Request Body

```
{
  "alias-config": [
    {
      "alias": "alias-1"
    }
  ]
}
```

cURL command

```
curl -v -X DELETE -H "Expect:" "https://10.37.129.91:443/vtap" -d @/root/delete_alias1.json --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Response": [
        {
            "detail": {
                "alias": "alias-1"
            },
            "response": "Alias deleted successfully"
        }
    ]
}
```

Error Response Body

```
400 Bad Request
{
    "Response": [
        {
            "detail": {
                "alias": "alias-1"
            },
            "error-code": -2,
            "error-message": "Match criteria with name alias-1 doesn't exists"
        },
        {
            "detail": {
                "alias": "alias-1"
            },
            "error-code": -2,
            "error-message": "Alias deletion failed"
        }
    ]
}
```

Getting alias

Retrieve configuration information for all or specific aliases from vTAP.

Method

GET

All aliases

Resource URI

```
{http | https}://<host>:<port>/vtap/alias-config@all.json
```

Example 1

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get information about all aliases.

URI

```
http://10.37.129.91:80/vtap/alias-config@all.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/alias@all.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
  "Configured aliases": [
    {
      "alias name": "alias-2",
      "Number of flex matches": 4,
      "  Flex match1 byte offset": 32,
      "  Flex match1 length to search": 10,
      "  Flex match1 pattern": "cloud",
      "  Flex match2 byte offset": 0,
      "  Flex match2 length to search": 0,
      "  Flex match2 pattern": "0x3456",
      "  Flex match3 byte offset": 0,
      "  Flex match3 length to search": 0,
      "  Flex match3 pattern": "cl.*.d",
      "  Flex match4 byte offset": 10,
      "  Flex match4 length to search": 20,
      "  Flex match4 pattern": "0x5645"
    }
  ]
}
```

Individual alias

Resource URI

{http | https}://<host>:<port>/vtap/alias-config@<*alias_name*>.json

Example 1

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get information about alias named `alias1`.

URI

```
http://10.37.129.91:80/vtap/alias-config@alias1.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/alias-config@alias1.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
  "Configured aliases": [
    {
      "alias name": "alias-2",
      "Number of flex matches": 4,
      "  Flex match1 byte offset": 32,
      "  Flex match1 length to search": 10,
      "  Flex match1 pattern": "cloud",
      "  Flex match2 byte offset": 0,
      "  Flex match2 length to search": 0,
      "  Flex match2 pattern": "0x3456",
      "  Flex match3 byte offset": 0,
      "  Flex match3 length to search": 0,
      "  Flex match3 pattern": "cl.*.d",
      "  Flex match4 byte offset": 10,
      "  Flex match4 length to search": 20,
      "  Flex match4 pattern": "0x5645"
    }
  ]
}
```

Getting SMARTMatch alias stats

Retrieve SMARTMatch alias statistics.

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/smart-match@alias-stats
```

Examples

URI

```
http://10.37.129.91:80/vtap/smart-match@alias-stats
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/smart-match@alias-stats" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "smatch-alias": [
        {
            "flexmatch alias ": "IPV4_outergre",
            "policy name ": "smatch-1",
            "rule id ": "2",
            "packets matched ": "0",
            "packets forwarded/dropped ": "-",
            "egress ": "-"
        },
        {
            "flexmatch alias ": "a2",
            "policy name ": "smatch-1",
            "rule id ": "1",
            "packets matched ": "0",
            "packets forwarded/dropped ": "-",
            "egress ": "-"
                  }
          ]
       }
```

Configuring ingress

Method

POST

Resource URI

```
{http | https}://<host>:<port>/vtap
```

NOTE

If both `svlan` and `cvlan` are configured, tunnel is considered as QINQ.

Examples

URI

```
http://10.37.129.91:80/vtap
```

Request Body - For GRE

```
{
  "ingress-tunnel": [
    {
      "tunnel-name": "ingress-1",
      "host1-ip": "10.37.129.190",
      "host2-ip": "10.37.129.193",
      "tunnel-type": "gre"
    }
  ]
}
```

Request Body - For IPIP

```
{
  "ingress-tunnel": [
    {
      "tunnel-name": "ingress-ipip",
      "host1-ip": "10.37.129.190",
      "host2-ip": "10.37.129.193",
      "tunnel-type": "ipip"
    }
  ]
}
```

Request Body - VxLAN

```
{
  "ingress-tunnel": [
    {
      "tunnel-name": "ingress-ipip",
      "host1-ip": "10.37.129.190",
      "host2-ip": "10.37.129.193",
      "vni": "123"
    }
  ]
}
```

Request Body - VLAN

```
{
  "ingress-tunnel": [
    {
      "tunnel-name": "ingress-3",
      "svlan": "2323"
    }
  ]
}
```

Request Body - QINQ

```
{
  "ingress-tunnel": [
    {
      "tunnel-name": "ingress-qinq",
      "svlan": "2323",
      "cvlan": "3232"
    }
  ]
}
```

cURL command

```
curl -v -X POST -H "Expect:" "https://10.37.129.91:443/vtap" -d @/root/ingress1.json --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Response": [
        {
            "detail": {
                "tunnel-name": "ingress-1",
                "host1-ip": "10.37.129.190",
                "host2-ip": "10.37.129.193",
                "tunnel-type": "gre"
            },
            "response": "Successfully configured"
        }
    ]
}
```

Error Response Body

```
400 Bad Request
{
    "Response": [
        {
            "detail": {
                "tunnel-name": "ingress-1",
                "host1-ip": "10.37.129.190",
                "host2-ip": "10.37.129.193",
                "tunnel-type": "gre"
            },
            "error-code": -2,
            "error-message": "Tunnel ingress-1 is
             already  configured"
        }
      ]
}
```

Deleting ingress

Method

DELETE

Resource URI

```
{http | https}://<host>:<port>/vtap
```

Examples

URI

```
http://10.37.129.91:80/vtap
```

Request Body

```
{
  "ingress-tunnel": [
    {
      "tunnel-name": "tunnel-1"
    }
  ]
}
```

cURL command

```
curl -v -X DELETE -H "Expect:" "https://10.37.129.91:443/vtap" -d @/root/delete_ingress.json --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Response": [
        {
            "detail": {
                "tunnel-name": " tunnel-1"
            },
            "response": "Successfully Deleted"
        }
    ]
}
```

Error Response Body

```
400 Bad Request
{
    "Response": [
        {
            "detail": {
                "tunnel-name": "tunnel-1"
            },
            "error-code": -2,
            "error-message": "Error: in-tunnel tunnel-1 not configured"
        }
    ]
}
```

or

```
{
  "Response": [
    {
      "detail": {
        "tunnel-name": "tunnel-1"
      },
      "error-code":7,
      "response": "still in use"
    }
  ]
}
```

Getting ingress tunnel configuration

Retrieve configuration information for all or specific ingress tunnels from vTAP.

Method

GET

All ingress tunnels

Resource URI

```
{http | https}://<host>:<port>/vtap/ingress-tunnel@all.json
```

Examples

URI

```
http://10.37.129.91:80/vtap/ingress-tunnel@all.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/ingress-tunnel@all.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
  "ingress-tunnel": [
    {
      "tunnel-name": "ingress-ipip",
      "host1-ip": "10.37.129.190",
      "host2-ip": "10.37.129.193",
      "destination-port": 4789,
      "tunnel-type": "vxlan",
      "vni": 123
    },
    {
      "ingress-tunnel": [
        {
          "tunnel-name": "ingress-1",
          "host1-ip": "10.37.129.190",
          "host2-ip": "10.37.129.193",
          "tunnel-type": "gre"
        }
      ]
    }
  ]
}
```

Individual ingress tunnel

Resource URI

{http | https}://<host>:<port>/vtap/ingress-tunnel@<*ingress_tunnel_name*>.json

Examples

URI

```
http://10.37.129.91:80/vtap/ingress-tunnel@tunnel-2.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/ingress-tunnel@tunnel-2.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "ingress-tunnel": [
        {
            "tunnel-name": "ingress-ipip",
            "host1-ip": "10.37.129.190",
            "host2-ip": "10.37.129.193",
            "destination-port": 4789,
            "tunnel-type": "vxlan",
            "vni": 123
        }
    ]
}
```

Configuring SMARTMatch policy

Method

POST

Resource URI

```
{http | https}://<host>:<port>/vtap
```

NOTE

If no rules are configured, the rule-id is set to 0.

Examples

URI

```
http://10.37.129.91:80/vtap
```

Request Body

```
{
  "smatch-policy": [
    {
      "policy-name": "sm1",
      "rules": [
        {
          "tunnel-name": "t1",
          "vlan": 400,
          "protocol": "tcp",
          "host1-ip": "10.37.129.95",
          "host2-ip": "10.37.131.165",
          "host1-port": 223,
          "host2-port": 221,
          "smatch-alias": "a1:egress-1,alias-2:egress-2",
          "egress": "egress-1",
          "header-strip": "802.1br_vntag",
          "packet-slice": 72
        },
   {
          "vlan": "any",
          "host1-ip": "10.10.10.0",
          "host2-ip": "20.20.0.0",
          "egress": "drop",
          "header-strip": "802.1br_vntag,vxlan,nvgre,mpls",
          "export-ipfix": "all"
        },
   {
          "vlan": "2048",
          "host1-ip": "2001:43:12::1",
          "host2-ip": "2600:32:11::1",
          "egress": "egress-2",
          "export-ipfix": "disable"
        },
   {
          "vlan": "1024",
          "host1-ip": "192.168.12.1",
          "host2-ip": "192.165.1.1",
          "egress": "egress-2",
          "export-ipfix": "disable"
        }
      ]
    }
  ]
}
```

cURL command

```
curl -v -X POST -H "Expect:" "https://10.37.129.91:443/vtap" -d @/root/smp1.json --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK{
    "Response": [
        {
            "detail": {
                "policy-name": "sm1",
                "protocol": "tcp",
                "egress": "egress-1",
                "rule-id": 1
            },
            "response": "Policy and rules configured successfully"
        },
        {
            "detail": {
                "policy-name": "sm1",
                "egress": "drop",
                "rule-id": 2
            },
            "response": "Policy and rules configured successfully"
        },
        {
            "detail": {
                "policy-name": "sm1",
                "egress": "egress-2",
                "rule-id": 3
            },
            "response": "Policy and rules configured successfully"
        },
        {
            "detail": {
                "policy-name": "sm1",
                "egress": "egress-2",
                "rule-id": 4
            },
            "response": "Policy and rules configured successfully"
        },
        {
            "detail": {
                "policy-name": "sm1",
                "rules": [
                    {
                        "tunnel-name": "t1",
                        "vlan": 400,
                        "protocol": "tcp",
                        "host1-ip": "10.37.129.95",
                        "host2-ip": "10.37.131.165",
                        "host1-port": 223,
                        "host2-port": 221,
                        "smatch-alias": "a1:egress-1,alias-2:egress-2",
                        "egress": "egress-1",
                        "header-strip": "802.1br_vntag",
                        "packet-slice": 72
                    },
                    {
                        "vlan": "any",
                        "host1-ip": "10.10.10.0",
                        "host2-ip": "20.20.0.0",
                        "egress": "drop",
                        "header-strip": "802.1br_vntag,vxlan,nvgre,mpls",
                        "export-ipfix": "all"
                    },
                    {
                        "vlan": "2048",
                        "host1-ip": "2001:43:12::1",
                        "host2-ip": "2600:32:11::1",
                        "egress": "egress-2",
```

```
                    "export-ipfix": "disable"
                },
                {
                    "vlan": "1024",
                    "host1-ip": "192.168.12.1",
                    "host2-ip": "192.165.1.1",
                    "egress": "egress-2",
                    "export-ipfix": "disable"
                }
            ]
        },
        "response": "Policy configured successfully"
    }
]
}
```

Error Response Body

```
{
    "Response": [
        {
            "detail": {
                "policy-name": "sm1",
                "rules": [
                    {
                        "tunnel-name": "t1",
                        "vlan": 400,
                        "protocol": "tcp",
                        "host1-ip": "10.37.129.95",
                        "host2-ip": "10.37.131.165",
                        "host1-port": 223,
                        "host2-port": 221,
                        "smatch-alias": "a1:egress-1,alias-2:egress-2",
                        "egress": "egress-1",
                        "header-strip": "802.1br_vntag",
                        "packet-slice": 72
                    }
                ]
            },
            "error-code": -2,
            "error-message": "Rule already exists for smatch Policy"
        }
    ]
}
```

Deleting SMARTMatch policy

Method

DELETE

Resource URI

```
{http | https}://<host>:<port>/vtap
```

Examples

URI

```
http://10.37.129.91:80/vtap
```

Request Body

```
{
  "smatch-policy": [
    {
      "policy-name": "smatch-1"
    }
  ]
}
```

cURL command

```
curl -v -X DELETE -H "Expect:" "https://10.37.129.91:443/vtap" -d @/root/delete_smartmatch1.json --
insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Response": [
        {
            "detail": {
                "policy-name": "smatch-1"
            },
            "response": "SMART match policy deleted successfully"
        }
    ]
}
```

Error Response Body

```
400 Bad Request
{
    "Response": [
        {
            "detail": {
                "policy-name": "smatch-1"
            },
            "error-code": -2,
            "error-message": "Policy smatch-1 not configured."
        }
    ]
}
```

Getting SMARTMatch policy configuration

Retrieve configuration information for all or specific SMARTMatch policies from vTAP.

Method

GET

All SMARTMatch policies

Resource URI

```
{http | https}://<host>:<port>/vtap/smatch-policy@all.json
```

Example

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get information about all SMARTMatch policies.

URI

```
http://10.37.129.91:80/vtap/smatch-policy@all.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/smatch-policy@all.json" --insecure
```

Success Response Body

```
204 No Content
```

or


```
{
    "Smatch-Policy": [
        {
            "Policy Name": "sp2",
            "State": "inactive",
            "Rule Info": [
                {
                    "rule id": 1,
                    "vlan id": "any",
                    "protocol": "udp",
                    "host1 ip": "any",
                    "host1 port": "any",
                    "host2 ip": "any",
                    "host2 port": "any",
                    "tunnel name": "--",
                    "egress": "--",
                    "sampling policy": "--",
                    "number of alias": 0,
                    "export IPFIX": "disable",
                    "header-strip": "-",
                    "packet-slice": "-"
                },
                {
                    "rule id": 2,
                    "vlan id": "any",
                    "protocol": "tcp",
                    "host1 ip": "any",
                    "host1 port": "any",
                    "host2 ip": "any",
                    "host2 port": "any",
                    "tunnel name": "--",
                    "egress": "--",
                    "sampling policy": "--",
                    "number of alias": 0,
                    "export IPFIX": "disable",
                    "header-strip": "-",
                    "packet-slice": "-"
                }
            ]
        }
    ]
}
```


Individual SMARTMatch policy

Resource URI

```
{http | https}://<host>:<port>/vtap/smatch-policy@<smatch_policy_name>.json
```

Example

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get information about SMARTMatch policy named `sm1`.

URI

```
http://10.37.129.91:80/vtap/smatch-policy@sm1.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/smatch-policy@sm1.json" --insecure
```

Success Response Body

```
204 No Content
```

or


```
{
    "Smatch-Policy": [
        {
            "Policy Name": "sp2",
            "State": "inactive",
            "Rule Info": [
                {
                    "rule id": 1,
                    "vlan id": "any",
                    "protocol": "udp",
                    "host1 ip": "any",
                    "host1 port": "any",
                    "host2 ip": "any",
                    "host2 port": "any",
                    "tunnel name": "--",
                    "egress": "--",
                    "sampling policy": "--",
                    "number of alias": 0,
                    "export IPFIX": "disable",
                    "header-strip": "-",
                    "packet-slice": "-"
                }
            ]
        }
    ]
}
```

Getting SMARTMatch policies and rules

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/smart-match@rule-detail
```

Example 1 - Rules, detail

URI

```
http://10.37.129.91:80/vtap/smart-match@rule-detail
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/smart-match@rule-detail" --insecure
```

Success Response Body

```
204 No Content
```

or


```
{
    "rule detail": [
        {
            "Policy Name": "sp2",
            "State": "inactive",
            "Rule Info": [
                {
                    "rule id": 1,
                    "vlan id": "any",
                    "protocol": "udp",
                    "host1 ip": "any",
                    "host1 port": "any",
                    "host2 ip": "any",
                    "host2 port": "any",
                    "tunnel name": "--",
                    "egress": "egr1",
                    "sampling policy": "--",
                    "number of alias": 0,
                    "export IPFIX": "disable",
                    "header-strip": "vxlan,mpls",
                    "packet-slice": "300"
                },
                {
                    "rule id": 2,
                    "vlan id": "any",
                    "protocol": "tcp",
                    "host1 ip": "any",
                    "host1 port": "any",
                    "host2 ip": "any",
                    "host2 port": "any",
                    "tunnel name": "t1",
                    "egress": "--",
                    "sampling policy": "--",
                    "number of alias": 0,
                    "export IPFIX": "disable",
                    "header-strip": "nvgre,mpls",
                    "packet-slice": "220"
                }
            ]
        }
    ]
}
```

IPFIX

Configuring IPFIX collector

Method

POST

Resource URI

```
{http | https}://<host>:<port>/vtap
```

Examples

URI

```
http://10.37.129.91:80/vtap
```

Request Body

```
{
  "flow-exporter": [
    {
      "name": "Name",
      "collector-ip": "10.9.9.186",
      "collector-port": 8008,
      "transport": "udp",
      "export-ip": "10.9.9.184",
      "export-port": 9009
    }
  ]
}
```

cURL command

```
curl -v -X POST -H "Expect:" "https://10.37.129.91:443/vtap" -d @/root/collector1.json --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
  "Response": [
    {
      "detail": {

      "name": "Name",
      "collector-ip": "10.9.9.186",
      "collector-port": 8008,
      "transport": "udp",
      "export-ip": "10.9.9.184",
      "export-port": 9009
      },
      "response": "Successfully configured"
    }
  ]
}
```

Error Response Body

```
400 Bad Request
```

Configuring flow templates

Method

POST

Resource URI

```
{http | https}://<host>:<port>/vtap
```

Examples

URI

```
http://10.37.129.91:80/vtap
```

Request Body

```
{
  "Template": [
    {
      "ID": 271,
      "SetID": 2,
      "ScopeFieldCount": 0,
      "FieldCount": 5,
      "IE": [
        {
          "ID": 148,
          "PEN": "None",
          "Length": 8
        },
        {
          "ID": 40,
          "PEN": "EXTREME",
          "Length": 65535
        },
        {
          "ID": 41,
          "PEN": "EXTREME",
          "Length": 65535
        }
      ]
    }
  ]
}
```

cURL command

```
curl -v -X POST -H "Expect:" "https://10.37.129.91:443/vtap" -d @/root/templates1.json --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
  "Response": [
    {
      "detail": {
        "ID": 271,
        "SetID": 2,
        "ScopeFieldCount": 0,
        "FieldCount": 5,
        "IE": [
          {
            "ID": 148,
            "PEN": "None",
            "Length": 8
          },
          {
            "ID": 40,
            "PEN": "EXTREME",
            "Length": 65535
          },
          {
            "ID": 41,
            "PEN": "EXTREME",
            "Length": 65535
          }
        ]
      },
      "response": "Successfully configured"
    }
  ]
}
```

Error Response Body

```
400 Bad Request
```

Setting/clearing flow parameters

Method

PUT

Resource URI

```
PUT {http | https}://<host>:<port>/vtap
```

Examples

URI

```
http://10.37.129.91:80/vtap
```

Request Body - Set

```
{
  "export-interval": [
    {
      "command": "set",
      "export-interval": 10
    }
  ]
}
```

Request Body - Clear

```
{
  "export-interval": [
    {
      "command": "clear",
      "export-interval": 10
    }
  ]
}
```

cURL command

```
curl -v -X PUT -H "Expect:" "https://10.37.129.91:443/vtap" -d @/root/put_flow1.json --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
  "Response": [
    {
      "detail": {
        "export-interval ": "10"
      },
      "response": "Flow-export collector configured successfully."
    }
  ]
}
```

Error Response Body

```
400 Bad Request
```

Getting flow information

Retrieve information for all the flows.

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/flow-info@all.json
```

Examples

URI

```
http://10.37.129.91:80/vtap/flow-info@all.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/flow-info@all.json" --insecure
```

Success Response Body

```
204 No Content
```

or


```
{
    "flow-info": [
        {
            "flow-id": 2,
            "parent id": 0,
            "parent-type": "-",
            "host1-ip": "172.11.11.202",
            "host2-ip": "172.11.11.201",
            "host1-port": 0,
            "host2-port": 0,
            "protocol": "icmp",
            "dir1 rx packets": 2,
            "dir1 rx bytes": 538,
            "dir2 rx packets": 0,
            "dir2 rx bytes": 0,
            "dir1 tx packets": 2,
            "dir1 tx bytes": 538,
            "dir2 tx packets": 0,
            "dir2 tx bytes": 0,
            "export-ipfix": "disable",
            "sampled-out": "no",
            "sampled out packets": 0,
            "dropped packets": 2,
            "smatch rule id": "-",
            "egress-action": "drop",
            "l7-type": "other"
        },
        {
            "flow-id": 1,
            "parent id": 0,
            "parent-type": "-",
            "host1-ip": "172.11.11.201",
            "host2-ip": "172.11.11.202",
            "host1-port": 9000,
            "host2-port": 8888,
            "protocol": "udp",
            "dir1 rx packets": 2,
            "dir1 rx bytes": 482,
            "dir2 rx packets": 0,
            "dir2 rx bytes": 0,
            "dir1 tx packets": 2,
            "dir1 tx bytes": 482,
            "dir2 tx packets": 0,
            "dir2 tx bytes": 0,
            "export-ipfix": "disable",
            "sampled-out": "no",
            "sampled out packets": 0,
            "dropped packets": 2,
            "smatch rule id": "-",
            "egress-action": "drop",
            "l7-type": "other"
        }
    ]
}
```

Getting flow-exporter export interval

Method

GET

Retrieve information about flow-exporter export interval.

Resource URI

```
{http | https}://<host>:<port>/vtap/export-interval
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get information about flow-exporter export interval.

URI

```
http://10.37.129.91:80/vtap/export-interval
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/export-interval" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    “export-interval” : 1
}
```

Getting flow-exporter collector statistics

Retrieve statistics for all or specific flow-exporter collector from vTAP.

Method

GET

All flow-exporter collectors

Resource URI

```
{http | https}://<host>:<port>/vtap/flow-exporter@stats/collector=all
```

Examples

URI

```
http://10.37.129.91:80/vtap/flow-exporter@stats/collector=all
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/flow-exporter@stats/collector=all" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Flow-Export": [
        {
            "Collector ID": 1,
            "name": "Name",
            "Number Message Sent": 20,
            "Max Record Sent": 2,
            "Number Template Sent": 4,
            "Number of Data Record Sent": 32
        }
    ]
}
```

Individual flow-exporter collector

Resource URI

```
{http | https}://<host>:<port>/vtap/flow-exporter@stats/collector=<collector_id>
```

Examples

URI

```
http://10.37.129.91:80/vtap/flow-exporter@stats/collector=1
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/flow-exporter@stats/collector=1" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Flow-Export": [
        {
            "Collector ID": 1,
            "name": "Name",
            "Number Message Sent": 20,
            "Max Record Sent": 2,
            "Number Template Sent": 4,
            "Number of Data Record Sent": 32
        }
    ]
}
```

Getting flow-exporter collector information

Retrieve configuration information for all or specific flow-exporter collector from vTAP.

Method

GET

All flow-exporter collectors

Resource URI

```
{http | https}://<host>:<port>/vtap/flow-exporter@config/collector=all
```

Examples

URI

```
http://10.37.129.91:80/vtap/flow-exporter@config/collector=all
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/flow-exporter@config/collector=all" --
insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
  "Flow-Export": [
    {
      "Collector ID": 1,
      "name": "Name",
      "Collector Address": "10.9.9.186",
      "Export Address": "10.9.9.184",
      "Transport Protocol": "UDP",
      "Collector Port": 8008,
      "Export Port": 9009
    },
    {
      "collector ID": 2,
      "name": "test",
      "Collector Address": "10.37.131.165",
      "Export Address": "10.37.131.140",
      "Collector Port": 5030,
      "Export Port": 5031,
      "Transport Protocol": "udp"
    }
  ]
}
```

Individual flow-exporter collector

Resource URI

{http | https}://<host>:<port>/vtap/flow-exporter@config/collector=*<collector_id>*

Examples

URI

```
http://10.37.129.91:80/vtap/flow-exporter@config/collector=1
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/flow-exporter@config/collector=1" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
  "Flow-Export": [
    {
      "Collector ID": 1,
      "name": "Name",
      "Collector Address": "10.9.9.186",
      "Export Address": "10.9.9.184",
      "Transport Protocol": "UDP",
      "Collector Port": 8008,
      "Export Port": 9009
    }
  ]
}
```

Getting flow-exporter template statistics

Retrieve statistics for all or specific flow-exporter template from vTAP.

Method

GET

All flow-exporter templates

Resource URI

```
{http | https}://<host>:<port>/vtap/flow-exporter@stats/template=all
```

Examples

URI

```
http://10.37.129.91:80/vtap/flow-exporter@stats/template=all
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/flow-exporter@stats/template=all" --insecure
```

Success Response Body

```
204 No Content
```

or

```
{
    "Flow-Export": [
        {
            "Template ID": 300,
            "Number of Template Sent": 0,
            "Number of DataRecord Sent": 6
        },
        {
            "Template ID": 257,
            "Number of Template Sent": 0,
            "Number of DataRecord Sent": 0
        },
        {
            "Template ID": 256,
            "Number of Template Sent": 0,
            "Number of DataRecord Sent": 0
        },
        {
            "Template ID": 258,
            "Number of Template Sent": 3,
            "Number of DataRecord Sent": 3
        },
        {
            "Template ID": 272,
            "Number of Template Sent": 0,
            "Number of DataRecord Sent": 0
        },
        {
            "Template ID": 271,
            "Number of Template Sent": 0,
            "Number of DataRecord Sent": 0
        },
        {
            "Template ID": 273,
            "Number of Template Sent": 0,
            "Number of DataRecord Sent": 0
        }
    ]
}
```

Individual flow-exporter template

Resource URI

{http | https}://<host>:<port>/vtap/flow-exporter@stats/template=<*template_id*>

Examples

URI

```
http://10.37.129.91:80/vtap/flow-exporter@stats/template=1
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/flow-exporter@stats/template=1" --insecure
```

Success Response Body

```
204 No Content
```

or

```
{
    "Flow-Export": [
        {
            "Template ID": 300,
            "Number of Template Sent": 0,
            "Number of DataRecord Sent": 6
        }
    ]
}
```

Getting flow-exporter template information

Retrieve configuration information for all or specific flow-exporter template from vTAP.

Method

GET

All flow-exporter templates

Resource URI

```
{http | https}://<host>:<port>/vtap/flow-exporter@config/template=all
```

Examples

URI

```
http://10.37.129.91:80/vtap/flow-exporter@config/template=all
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/flow-exporter@config/template=all" --insecure
```

Success Response Body

```
204 No Content
```

or


```
{
    "Flow-Export": [
        {
            "Template ID": 300,
            "Set ID": 3,
            "Number of Scope Fields": 2,
            "Number of Fields": 8,
            "Number of Dependent Templates": 0
        },
        {
            "Template ID": 257,
            "Set ID": 2,
            "Number of Fields": 24,
            "Number of Dependent Templates": 1,
            "Dependent Template ID": 300
        },
        {
            "Template ID": 256,
            "Set ID": 2,
            "Number of Fields": 20,
            "Number of Dependent Templates": 1,
            "Dependent Template ID": 300
        },
        {
            "Template ID": 258,
            "Set ID": 2,
            "Number of Fields": 8,
            "Number of Dependent Templates": 1,
            "Dependent Template ID": 300
        },
        {
            "Template ID": 272,
            "Set ID": 2,
            "Number of Fields": 4,
            "Number of Dependent Templates": 0
        },
        {
            "Template ID": 271,
            "Set ID": 2,
            "Number of Fields": 5,
            "Number of Dependent Templates": 0
        },
        {
            "Template ID": 273,
            "Set ID": 2,
            "Number of Fields": 11,
            "Number of Dependent Templates": 0
        }
    ]
}
```


Individual flow-exporter template

Resource URI

```
{http | https}://<host>:<port>/vtap/flow-exporter@config/template=<template_id>
```

Examples

URI

```
http://10.37.129.91:80/vtap/flow-exporter@config/template=1
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/flow-exporter@config/template=1" --insecure
```

Success Response Body

```
204 No Content
```

or

```
{
    "Flow-Export": [
        {
            "Template ID": 300,
            "Set ID": 3,
            "Number of Scope Fields": 2,
            "Number of Fields": 8,
            "Number of Dependent Templates": 0
        }
    ]
}
```

Interface

Setting/clearing interface parameters

Method

PUT

Resource URI

```
{http | https}://<host>:<port>/vtap
```

Examples

URI

```
http://10.37.129.91:80/vtap
```

Request Body - Set

Set:

```
{
  "interface-config": [
    {
      "command": "set",
      "interface-name": "ip",
      "sampling-policy": "sp1"
    }
  ]
}
```

Request Body - Clear

Clear:

```
{
  "interface-config": [
    {
      "command": "clear",
      "interface-name": "ip",
      "sampling-policy": "sp1",
    }
  ]
}
```

NOTE

When clearing property for interface, value of the property is ignored. As JSON expects the key-value pair, it is mandatory to provide the value.

Request Body - Change `mode` from `session` to `packet`

```
{
  "interface-config": [
    {
      "command": "set",
      "interface-name": "ip",
      "mode": "packet"
    }
  ]
}
```

cURL command

```
curl -v -X PUT -H "Expect:" "https://10.37.129.91:443/vtap" -d @/root/put_interface1.json --insecure
```

Success Response Body - Set

```
204 No Content
```

or

```
200 OK
{
    "Response": [
        {
            "detail": {
                "command": "set",
                "interface-name": "ip",
                "sampling-policy": "sp1"
            },
            "response": "Interface property updated successfully"
        }
    ]
}
```

Success Response Body - Clear

```
204 No Content
```

or

```
{
  "Response": [
    {
      "detail": {
        "command": "clear",
        "interface-name": "ip",
        "sampling-policy": "sp1"
      },
      "response": "Interface property cleared successfully"
    }
  ]
}
```

Success Response Body - Changing mode from `session` to `packet`

```
{
  "Response": [
    {
      "detail": {
        "command": "set",
        "interface-name": "ip",
        "mode": "packet"
      },
      "response": "Interface property updated successfully"
    }
  ]
}
```

Error Response Body

```
400 Bad Request
{
  "Response": [
    {
      "detail": {
        "command": "set",
        "interface-name": "s11-s1u-gngp",
        "port-group": "pg"
      },
      "error-code": -2,
      "error-message": "Port group already set to interface"
    }
  ]
}
```

Getting interface config

Retrieve interface configuration information from vTAP.

Method

GET

All interfaces

Resource URI

```
{http | https}://<host>:<port>/vtap/interface-config@all.json
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get information about all interfaces.

URI

```
http://10.37.129.91:80/vtap/interface-config@all.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/interface-config@all.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
{
    "Interface": [
        {
            "name": "ip",
            "traffic type": "ip",
            "egress": "drop",
            "sampling policy": "sp1",
            "smartmatch Policy": "smatch-1",
            "mode": "session",
            "export-ipfix": "disabled",
            "header-strip": "-",
            "packet-slice": "-"
        }
    ]
}
```

Logging

Setting/clearing logging module

Method

PUT

Resource URI

```
{http | https}://<host>:<port>/vtap
```

Examples

URI

```
http://10.37.129.91:80/vtap
```

Request Body - Set

```
{
  "logging": [
    {
      "command": "set",
      "module": "tunnel-flow",
      "level": "debug"
    }
  ]
}
```

Request Body - Clear

```
{
  "Logging": [
    {
      "command": "clear",
      "module": "tunnel-flow"
    }
  ]
}
```

cURL command

```
curl -v -X PUT -H "Expect:" "https://10.37.129.91:443/vtap" -d @/root/put_logging.json --insecure
```

Success Response Body - Set

```
204 No Content
```

or

```
200 OK
{
    "Response": [
        {
            "detail": {
                "command": "set",
                "module": "tunnel-flow",
                "level": "debug"
            },
            "response": "Successfully configured"
        }
    ]
}
```

Success Response Body - Clear

```
{
    "Response": [
        {
            "detail": {
                "command": "clear",
                "module": "tunnel-flow",
            },
            "response": "Successfully cleared"
        }
    ]
}
```

Error Response Body

```
400 Bad Request
```

Getting logging information

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/logging@all.json
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get all logging information.

URI

```
http://10.37.129.91:80/vtap/logging@all.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/logging@all.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "logging": [
        {
            "tunnel-flow": "debug",
            "smatch": "critical",
            "sampling": "critical",
            "ipfix": "critical",
            "mgmt": "critical",
            "L7app": "critical",
            "kni": "critical"
        }
    ]
}
```

vTAP services

Starting vTAP service

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/start@vtap
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to start vTAP service.

URI

```
http://10.37.129.91:80/vtap/start@vtap
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/start@vtap" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "start": [
        {
            "service started successfully": 0
        }
    ]
}
```

Starting SNMP service

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/start@vtap-snmp
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to start SNMP service.

URI

```
http://10.37.129.91:80/vtap/start@vtap-snmp
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/start@vtap-snmp" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "start": [
        {
            "snmp service started successfully": 0
        }
    ]
}
```

Starting all services

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/start@all
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to start vTAP, SNMP and NGINX services..

URI

```
http://10.37.129.91:80/vtap/start@all
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/start@all" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "start": [
        {
            "service started successfully": 0
        },
        {
            "snmp service started successfully": 0
        }
    ]
}
```

Getting status for NGINX service

Resource URI

```
{http | https}://<host>:<port>/vtap/status@vtap-nginx
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get the status for NGINX service.

URI

```
http://10.37.129.91:80/vtap/status@vtap-nginx
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/status@vtap-nginx" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "status": [
        {
            "vtap-ngnix": "Active",
         "Status": "   Active: active (running) since Tue 2019-01-22 15:51:22 EST; 18h ago\n"

        }
    ]
}
```

Getting status for SNMP service

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/status@vtap-snmp
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get the status for SNMP service.

URI

```
http://10.37.129.91:80/vtap/status@vtap-snmp
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/status@vtap-snmp" --insecure
```

Success Response Body

```
204 No Content
```

or

```
{
    "status": [
        {
            "vtap": "ACTIVE",
            "Physical contiguous memory": "initialized",
            "Mempool and hash": "initialized",
            "status": "   Active: active (running) since Wed 2019-01-23 08:25:29 EST; 2h 22min ago"
        },
        {
            "vtap-snmp": "Active",
            "Status": "   Active: active (running) since Tue 2019-01-22 15:51:53 EST; 18h ago\n"
        },
        {
            "vtap-ngnix": "Active",
            "Status": "   Active: active (running) since Tue 2019-01-22 15:51:22 EST; 18h ago\n"
        }
    ]
}
```

Getting status for all services

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/status@all
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get the status for vTAP, SNMP and NGINX services.

URI

```
http://10.37.129.91:80/vtap/status@all
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/status@all" --insecure
```

Success Response Body

```
204 No Content
```

or

```
{
    "status": [
        {
            "vtap": "ACTIVE",
            "Physical contiguous memory": "initialized",
            "Mempool and hash": "initialized",
            "status": "   Active: active (running) since Wed 2019-01-23 08:25:29 EST; 2h 22min ago"
        },
        {
            "vtap-snmp": "Active",
            "Status": "   Active: active (running) since Tue 2019-01-22 15:51:53 EST; 18h ago\n"
        },
        {
            "vtap-ngnix": "Active",
            "Status": "   Active: active (running) since Tue 2019-01-22 15:51:22 EST; 18h ago\n"
        }
    ]
}
```

Getting vTAP status

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/status@vtap
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to check the status of the vTAP.

URI

```
http://10.37.129.91:80/vtap/status@vtap
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/status@vtap" --insecure
```

Success Response Body

```
204 No Content
```

or

```
{

"status": [
        {
            "vtap": "ACTIVE",
            "Physical contiguous memory": "initialized",
            "Mempool and hash": "initialized",
            "status": "   Active: active (running) since Wed 2019-01-23 08:25:29 EST; 2h 17min ago"
        }
    ]
}
```

Restarting vTAP

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/restart@vtap
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to restart vTAP services.

URI

```
http://10.37.129.91:80/vtap/restart@vtap
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/restart@vtap" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "restart": [
        {
            "service restarted successfully": 0
        }
    ]
}
```

Restarting SNMP service

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/restart@vtap-snmp
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to restart SNMP service.

URI

```
http://10.37.129.91:80/vtap/restart@vtap-snmp
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/restart@vtap-snmp" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "restart": [
        {
            "snmp service restarted successfully": 0
        }
    ]
}
```

Restarting all services

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/restart@all
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to restart vTAP, SNMP and NGINX services.

URI

```
http://10.37.129.91:80/vtap/restart@all
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/restart@all" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "restart": [
        {
            "service restarted successfully": 0
        },
        {
            "snmp service restarted successfully": 0
        }
    ]
}
```

Stopping vTAP service

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/stop@vtap
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to stop vTAP service.

URI

```
http://10.37.129.91:80/vtap/stop@vtap
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/stop@vtap" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "stop": [
        {
            "service stopped successfully": 0
        }
    ]
}
```

Stopping SNMP service

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/stop@vtap-snmp
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to stop SNMP service.

URI

```
http://10.37.129.91:80/vtap/stop@vtap-snmp
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/stop@vtap-snmp" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "stop": [
        {
            "snmp service stopped successfully": 0
        }
    ]
}
```

Stopping all services

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/stop@all
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to stop vTAP, SNMP and NGINX services.

URI

```
http://10.37.129.91:80/vtap/stop@all
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/stop@all" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "stop": [
        {
            "service stopped successfully": 0
        },
        {
            "snmp service stopped successfully": 0
        }
    ]
}
```

Configuration management

Saving configuration

Resource URIs

```
{http | https}://<host>:<port>/vtap
```

Examples

The following is an example of a POST method request to a vTAP IP at 10.37.129.91 to save the current running configuration to the startup configuration file.

URI

```
http://10.37.129.91:80/vtap
```

Request Body

```
{
  "save-config": [
    {
      "path": "/etc/vtap/config/saved_config"
    }
  ]
}
```

cURL command

```
curl -v -X POST -H "Expect:" "https://10.37.129.91:443/vtap" -d @/etc/vtap/config/saved_config --
insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Response": [
        {
            "Configuration file saved successfully": 1
        }
    ]
}
```

Error Response Body

```
400 Bad Request
{
    "Response": [
        {
            "File Already Present.": 1
        }
    ]
}
```

Loading configuration

Resource URIs

```
{http | https}://<host>:<port>/vtap
```

Examples

The following is an example of a POST method request to a vTAP IP at 10.37.129.91 to load the configuration with the contents of a previously saved configuration file.

> NOTE
> If the saved configuration has 3k rules, vTAP takes up to 6 minutes to load all the rules in the file.

URI

```
http://10.37.129.91:80/vtap
```

Request Body

```
{
  "load-config": [
    {
      "path": "/etc/save.cnf"
    }
  ]
}
```

cURL command

```
<<<
```

Success Response Body

```
204 No Content
```

or


```
{
    "Response": [
        {
            "path": "/etc/minu.cnf"
        },
        {
            "Clearing interface properties for": "ip"
        },
        {
            "Info: mode is already set to session.": 0
        },
        {
            "Clearing egress": 0
        },
        {
            "Info: Egress action is not set": 0
        },
        {
            "Clearing egress alias": 0
        },
        {
            "Clearing Intunnel": 0
        },
        {
            "Clearing flexmatch alias": 0
        },
        {
            "match criteria deleted successfully.": 0
        },
        {
            "Clearing log modules": 0
        },
        {
            "Ingress tunnel configured successfully.": 0
        },
        {
            "Ingress tunnel configured successfully.": 0
        },
        {
            "Info: Egress action is not set": 0
        },
        {
            "Interface parameters updated successfully.": "egress drop-egress"
        },
        {
            "Sampling Policy and IPFIX settings on Interface are enabled with session mode": "egress
drop-egress"
        }
    ]
}
```

Error Response Body

```
400 Bad Request
{
    "Response": [
        {
            "detail": {
                "path": "/etc/vtap/min"
            },
            "error-code": -2,
            "error-message": "Info: mode is already set to session"
        },
        {
            "detail": {
                "path": "/etc/vtap/min"
            },
            "error-code": -2,
            "error-message": "Info: Egress action is not set"
        },
        {
            "detail": {
                "path": "/etc/vtap/min"
            },
            "error-code": -2,
            "error-message": "Ingress tunnel configured successfully."
        },
        {
            "detail": {
                "path": "/etc/vtap/min"
            },
            "error-code": -2,
            "error-message": "Ingress tunnel configured successfully."
        },
        {
            "detail": {
                "path": "/etc/vtap/min"
            },
            "error-code": -2,
            "error-message": "Match criteria updated successfully."
        },
        {
            "detail": {
                "path": "/etc/vtap/min"
            },
            "error-code": -2,
            "error-message": "Info: Egress action is not set"
        },
        {
            "detail": {
                "path": "/etc/vtap/min"
            },
            "error-code": -2,
            "error-message": "Invalid Egress action"
        },
        {
            "detail": {
                "path": "/etc/vtap/min"
            },
            "error-code": -2,
            "error-message": "Interface \"ip\"  parameters \"mode\" updated successfully."
        }
    ]
}
```

Clearing configuration

Resource URIs

```
{http | https}://<host>:<port>/vtap
```

Examples

The following is an example of a POST method request to a vTAP IP at 10.37.129.91 to clear the current running configuration.

URI

```
http://10.37.129.91:80/vtap
```

Request Body

```
{
  "clear-config": [
    {

    }
  ]
}
```

cURL command

```
<<<
```

Success Response Body

```
204 No Content
```

or

```
{
    "Response": [
        {
            "Clearing interface properties for": "ip"
        },
        {
            "Info: mode is already set to session.": 0
        },
        {
            "Clearing egress": 0
        },
        {
            "Info: Egress action is not set": 0
        },
        {
            "SMARTMatch policy deleted successfully.": ""
        },
        {
            "sampling policy deleted successfully": "sp-1"
        },
        {
            "Clearing egress alias": 0
        },
        {
            "Clearing Intunnel": 0
        },
        {
            "Clearing flexmatch alias": 0
        },
        {
            "match criteria deleted successfully.": 0
        },
        {
            "Clearing log modules": 0
        },
        {
            "Configuration cleared successfully": 0
        }
    ]
}
```

Error Response Body

```
400 Bad Request
{
    "Response": [
        {
            "detail": {},
            "error-code": -2,
            "error-message": "Info: mode is already set to session"
        },
        {
            "detail": {},
            "error-code": -2,
            "error-message": "Info: Egress action is not set"
        },
        {
            "Configuration cleared successfully": 0
        }
    ]
}
```

GET method

Clearing all statistics

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/clear-all
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to clear all statistics.

URI

```
http://10.37.129.91:80/vtap/clear-all
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/clear-all" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Clear All": [
        {
            "Cleared Tx Stats": 0,
            "Cleared Rx Stats": 0,
            "Cleared Egress Stats": 0,
            "Cleared Smatch Stats": 0
        }
    ]
}
```

Clearing packet statistics for all parameters

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/clear@packet-stats/all
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to clear all packet statistics.

URI

```
http://10.37.129.91:80/vtap/clear@packet-stats/all
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/clear@packet-stats/all" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Packet-Stats": [
        {
            "Cleared Tx Stats": 0,
            "Cleared Rx Stats": 0,
            "Cleared Egress Stats": 0,
            "Cleared Smatch Stats": 0
        }
    ]
}
```

Clearing packet statistics for egress

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/clear-packet-stats@egs
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to clear egress packet statistics.

URI

```
http://10.37.129.91:80/vtap/clear-packet-stats@egs
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/clear-packet-stats@egs" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Packet-Stats": [
        {
            "Cleared Egress Stats": 0
        }
    ]
}
```

Clearing packet statistics for Rx

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/clear-packet-stats@rx
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to clear Rx packet statistics.

URI

```
http://10.37.129.91:80/vtap/clear-packet-stats@rx
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/clear-packet-stats@rx" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Packet-Stats": [
        {
            "Cleared Rx Stats": 0
        }
    ]
}
```

Clearing packet statistics for SMARTMatch

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/clear-packet-stats@smatch
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to clear SMARTMatch packet statistics.

URI

```
http://10.37.129.91:80/vtap/clear-packet-stats@smatch
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/clear-packet-stats@smatch" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Packet-Stats": [
        {
            "Cleared Stats": 0
        }
    ]
}
```

Clearing SMARTMatch rule statistics

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/clear-smart-match@clear-rule-stats
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to clear SMARTMatch rule statistics.

URI

```
http://10.37.129.91:80/vtap/clear-smart-match@clear-rule-stats
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/clear-smart-match@clear-rule-stats" --
insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "clear rule stats": [
                {
                        "Cleared rule smartmatch stats successfully": 0
        }
    ]
}
```

Clearing all SMARTMatch statistics

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/clear-smart-match@clear-all
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to clear all SMARTMatch statistics.

URI

```
http://10.37.129.91:80/vtap/clear-smart-match@clear-all
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/clear-smart-match@clear-all" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "clear smartmatch stats": [
        {
            "Cleared all smartmatch stats successfully": 0
        }
    ]
}
```

Clearing packet statistics for Tx

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/clear-packet-stats@tx
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to clear Tx packet statistics.

URI

```
http://10.37.129.91:80/vtap/clear-packet-stats@tx
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/clear-packet-stats@tx" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Packet-Stats": [
        {
            "Cleared Tx Stats": 0
        }
    ]
}
```

Getting alias

Retrieve configuration information for all or specific aliases from vTAP.

Method

GET

All aliases

Resource URI

```
{http | https}://<host>:<port>/vtap/alias-config@all.json
```

Example 1

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get information about all aliases.

URI

```
http://10.37.129.91:80/vtap/alias-config@all.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/alias@all.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
  "Configured aliases": [
    {
      "alias name": "alias-2",
      "Number of flex matches": 4,
      "  Flex match1 byte offset": 32,
      "  Flex match1 length to search": 10,
      "  Flex match1 pattern": "cloud",
      "  Flex match2 byte offset": 0,
      "  Flex match2 length to search": 0,
      "  Flex match2 pattern": "0x3456",
      "  Flex match3 byte offset": 0,
      "  Flex match3 length to search": 0,
      "  Flex match3 pattern": "cl.*.d",
      "  Flex match4 byte offset": 10,
      "  Flex match4 length to search": 20,
      "  Flex match4 pattern": "0x5645"
    }
  ]
}
```

Individual alias

Resource URI

```
{http | https}://<host>:<port>/vtap/alias-config@<alias_name>.json
```

Example 1

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get information about alias named `alias1`.

URI

```
http://10.37.129.91:80/vtap/alias-config@alias1.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/alias-config@alias1.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
  "Configured aliases": [
    {
      "alias name": "alias-2",
      "Number of flex matches": 4,
      "  Flex match1 byte offset": 32,
      "  Flex match1 length to search": 10,
      "  Flex match1 pattern": "cloud",
      "  Flex match2 byte offset": 0,
      "  Flex match2 length to search": 0,
      "  Flex match2 pattern": "0x3456",
      "  Flex match3 byte offset": 0,
      "  Flex match3 length to search": 0,
      "  Flex match3 pattern": "cl.*.d",
      "  Flex match4 byte offset": 10,
      "  Flex match4 length to search": 20,
      "  Flex match4 pattern": "0x5645"
    }
  ]
}
```

Getting all configuration information

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/all
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get all vTAP-related configuration information in the running configuration.

URI

```
http://10.37.129.91:80/vtap/all
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/all" --insecure
```

Success Response Body

```
204 No Content
```

or


```
{
    "Sampling-Policy": [
        {
            "policy name": "sp1",
            "state": "active",
            "sample-rate": 100,
            "preserve-pkts": 0
        }
    ],
    "Smatch-Policy": [
        {
            "Policy Name": "sm1",
            "State": "inactive",
            "Rule Info": [
                {
                    "rule id": 1,
                    "vlan id": "any",
                    "protocol": "ip",
                    "host1 ip": "1.1.1.1",
                    "host1 port": "any",
                    "host2 ip": "2.2.2.2",
                    "host2 port": "any",
                    "tunnel name": "--",
                    "egress": "--",
                    "sampling policy": "--",
                    "number of alias": 0,
                    "export IPFIX": "all",
                    "header-strip": "-",
                    "packet-slice": "-"
                }
            ]
        }
    ],
    "Interface": [
        {
            "name": "ip",
            "traffic type": "ip",
            "egress": "drop",
            "sampling policy": "sp1",
            "smartmatch Policy": "-",
            "mode": "session",
            "export-ipfix": "sampled-out",
            "header-strip": "-",
            "packet-slice": "-"
        }
    ],
    "Ingress-Tunnel": [
        {
            "No ingress tunnel configured": 0
        }
    ],
    "Egress-Config": [
        {
            "No egress alias configured": 0
        }
    ],
    "Collector": [
        {
            "Collector ID": 1,
            "name": "test",
            "Collector Address": "172.10.10.102",
            "Export Address": "172.10.10.101",
            "Transport Protocol": "UDP",
            "Collector Port": 9999,
            "Export Port": 8888,
```

```
            "State": "ACTIVE"
        }
    ],
    "Template": [
        {
            "Template ID": 300,
            "Set ID": 3,
            "Number of Scope Fields": 2,
            "Number of Fields": 8,
            "Number of Dependent Templates": 0
        },
        {
            "Template ID": 257,
            "Set ID": 2,
            "Number of Fields": 24,
            "Number of Dependent Templates": 1,
            "Dependent Template ID": 300
        },
        {
            "Template ID": 256,
            "Set ID": 2,
            "Number of Fields": 20,
            "Number of Dependent Templates": 1,
            "Dependent Template ID": 300
        },
        {
            "Template ID": 258,
            "Set ID": 2,
            "Number of Fields": 8,
            "Number of Dependent Templates": 1,
            "Dependent Template ID": 300
        },
        {
            "Template ID": 272,
            "Set ID": 2,
            "Number of Fields": 4,
            "Number of Dependent Templates": 0
        },
        {
            "Template ID": 271,
            "Set ID": 2,
            "Number of Fields": 5,
            "Number of Dependent Templates": 0
        },
        {
            "Template ID": 273,
            "Set ID": 2,
            "Number of Fields": 11,
            "Number of Dependent Templates": 0
        }
    ],
    "Export-Interval": [
        {
            "export-interval:": 1
        }
    ],
    "logging": [
        {
            "tunnel-flow": "debug",
            "smatch": "debug",
            "sampling": "debug",
            "ipfix": "debug",
            "mgmt": "debug",
            "kni": "critical",
            "L7app": "critical"
        }
    ],
    "alias": [
        {
            "detail": "all",
            "response": "No alias configured"
        }
```

```
            ]
        }
```

Getting status for all services

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/status@all
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get the status for vTAP, SNMP and NGINX services.

URI

```
http://10.37.129.91:80/vtap/status@all
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/status@all" --insecure
```

Success Response Body

```
204 No Content
```

or

```
{
    "status": [
        {
            "vtap": "ACTIVE",
            "Physical contiguous memory": "initialized",
            "Mempool and hash": "initialized",
            "status": "   Active: active (running) since Wed 2019-01-23 08:25:29 EST; 2h 22min ago"
        },
        {
            "vtap-snmp": "Active",
            "Status": "   Active: active (running) since Tue 2019-01-22 15:51:53 EST; 18h ago\n"
        },
        {
            "vtap-ngnix": "Active",
            "Status": "   Active: active (running) since Tue 2019-01-22 15:51:22 EST; 18h ago\n"
        }
    ]
}
```

Getting device information

Retrieve device information for the VM that is running vTAP.

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/dev-info
```

Examples

URI

```
http://10.37.129.91:80/vtap/dev-info
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/dev-info" --insecure
```

Success Response Body

```
204 No Content
```

or

```
{
    "device-info": [
        {
            " NIC port   (  0)": "  0",
            "\t PCI Bus Location                                  (  0)": "     0000:13:00.00 ",
            "\t Vendor id                                         (  0)": "     15ad ",
            "\t Device id                                         (  0)": "     7b0 ",
            "\t Sub System Vendor id                              (  0)": "     15ad ",
            "\t Sub System Device id                              (  0)": "     7b0 ",
            "\t Driver Name                                       (  0)": "     net_vmxnet3 ",
            "\t if_index                                          (  0)": "     0 ",
            "\t Interface Name                                    (  0)": "      ",
            "\t Minimum RX Buffer size                            (  0)": "     1646 ",
            "\t Minimum configurable length of RX Packet          (  0)": "     16384 ",
            "\t Maximum number of RX Queues                       (  0)": "     16 ",
            "\t Maximum number of TX Queues                       (  0)": "     8 ",
            "\t Maximum number of VFs                             (  0)": "     0 ",
            "\t Maximum number of VMDq pools                      (  0)": "     0 ",
            "\t RX offload capabilities                           (  0)": "     29 ",
            "\t TX offload capabilities                           (  0)": "     45 ",
            "\t Redirection table size                            (  0)": "     0 ",
            "\t Hash key size                                     (  0)": "     0 ",
            "\t RSS offloads                                      (  0)": "     1300 ",
            "\t First queue ID for VMDQ pools                     (  0)": "     0 ",
            "\t Queue number for VMDQ pools                       (  0)": "     0 ",
            "\t First ID of VMDQ pools                            (  0)": "     0 ",
            "\t Maximum number of RX allowed descriptors          (  0)": "     4096 ",
            "\t Minimum number of RX allowed descriptors          (  0)": "     128 ",
            "\t Number of RX descriptors should be aligned        (  0)": "     1 ",
            "\t Maximum number of TX allowed descriptors          (  0)": "     4096",
            "\t Minimum number of TX allowed descriptors          (  0)": "     512 ",
            "\t Number of TX descriptors should be aligned        (  0)": "     1 ",
            " NIC port   (  1)": "  1",
            "\t PCI Bus Location                                  (  1)": "     0000:1b:00.00 ",
            "\t Vendor id                                         (  1)": "     15ad ",
            "\t Device id                                         (  1)": "     7b0 ",
            "\t Sub System Vendor id                              (  1)": "     15ad ",
            "\t Sub System Device id                              (  1)": "     7b0 ",
            "\t Driver Name                                       (  1)": "     net_vmxnet3 ",
            "\t if_index                                          (  1)": "     0 ",
            "\t Interface Name                                    (  1)": "      ",
            "\t Minimum RX Buffer size                            (  1)": "     1646 ",
            "\t Minimum configurable length of RX Packet          (  1)": "     16384 ",
            "\t Maximum number of RX Queues                       (  1)": "     16 ",
            "\t Maximum number of TX Queues                       (  1)": "     8 ",
            "\t Maximum number of VFs                             (  1)": "     0 ",
            "\t Maximum number of VMDq pools                      (  1)": "     0 ",
            "\t RX offload capabilities                           (  1)": "     29 ",
            "\t TX offload capabilities                           (  1)": "     45 ",
            "\t Redirection table size                            (  1)": "     0 ",
            "\t Hash key size                                     (  1)": "     0 ",
            "\t RSS offloads                                      (  1)": "     1300 ",
            "\t First queue ID for VMDQ pools                     (  1)": "     0 ",
            "\t Queue number for VMDQ pools                       (  1)": "     0 ",
            "\t First ID of VMDQ pools                            (  1)": "     0 ",
            "\t Maximum number of RX allowed descriptors          (  1)": "     4096 ",
            "\t Minimum number of RX allowed descriptors          (  1)": "     128 ",
            "\t Number of RX descriptors should be aligned        (  1)": "     1 ",
            "\t Maximum number of TX allowed descriptors          (  1)": "     4096",
            "\t Minimum number of TX allowed descriptors          (  1)": "     512 ",
            "\t Number of TX descriptors should be aligned        (  1)": "     1 "
        }
    ]
}
```

Getting egress alias configuration

Retrieve configuration information for all or specific egress alias from vTAP.

Method

GET

All egress aliases

Resource URI

```
{http | https}://<host>:<port>/vtap/egress-config@all.json
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get information about all egress aliases.

URI

```
http://10.37.129.91:80/vtap/egress-config@all.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/egress-config@all.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "egress-config": [
        {
            "egress alias": "egress-1",
            "state": "inactive",
            "local-ip": "1.1.1.1",
            "destination-ip": "2.2.2.2",
            "tunnel-type": "gre",
            "nexthop-mac": "11:12:13:14:15:16"
        },
        {
            "egress alias": "egress-2",
            "state": "inactive",
            "local-ip": "10.37.129.4",
            "destination-ip": "10.37.129.193",
            "tunnel-type": "gre",
            "nexthop-mac": "11:12:13:14:15:16"
        }
    ]
}
```

Individual egress alias

Resource URI

{http | https}://<host>:<port>/vtap/egress-config@*<egress_alias_name>*.json

Examples

URI

```
http://10.37.129.91:80/vtap/egress-config@egress-1.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/egress-config@egress-1.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "egress-config": [
        {
            "egress alias": "egress-1",
            "state": "inactive",
            "local-ip": "1.1.1.1",
            "destination-ip": "2.2.2.2",
            "tunnel-type": "gre",
            "nexthop-mac": "11:12:13:14:15:16"
        }
    ]
}
```

Error Response Body

```
400 Bad Request
{
  "Response": [
    {
      "egress-name": "egress-1"
    }
  ],
  "error code": 6,
  "error message": "not configured"
}
```

Getting flow information

Retrieve information for all the flows.

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/flow-info@all.json
```

Examples

URI

```
http://10.37.129.91:80/vtap/flow-info@all.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/flow-info@all.json" --insecure
```

Success Response Body

```
204 No Content
```

or


```
{
    "flow-info": [
        {
            "flow-id": 2,
            "parent id": 0,
            "parent-type": "-",
            "host1-ip": "172.11.11.202",
            "host2-ip": "172.11.11.201",
            "host1-port": 0,
            "host2-port": 0,
            "protocol": "icmp",
            "dir1 rx packets": 2,
            "dir1 rx bytes": 538,
            "dir2 rx packets": 0,
            "dir2 rx bytes": 0,
            "dir1 tx packets": 2,
            "dir1 tx bytes": 538,
            "dir2 tx packets": 0,
            "dir2 tx bytes": 0,
            "export-ipfix": "disable",
            "sampled-out": "no",
            "sampled out packets": 0,
            "dropped packets": 2,
            "smatch rule id": "-",
            "egress-action": "drop",
            "l7-type": "other"
        },
        {
            "flow-id": 1,
            "parent id": 0,
            "parent-type": "-",
            "host1-ip": "172.11.11.201",
            "host2-ip": "172.11.11.202",
            "host1-port": 9000,
            "host2-port": 8888,
            "protocol": "udp",
            "dir1 rx packets": 2,
            "dir1 rx bytes": 482,
            "dir2 rx packets": 0,
            "dir2 rx bytes": 0,
            "dir1 tx packets": 2,
            "dir1 tx bytes": 482,
            "dir2 tx packets": 0,
            "dir2 tx bytes": 0,
            "export-ipfix": "disable",
            "sampled-out": "no",
            "sampled out packets": 0,
            "dropped packets": 2,
            "smatch rule id": "-",
            "egress-action": "drop",
            "l7-type": "other"
        }
    ]
}
```

Getting flow-exporter collector information

Retrieve configuration information for all or specific flow-exporter collector from vTAP.

Method

GET

All flow-exporter collectors

Resource URI

```
{http | https}://<host>:<port>/vtap/flow-exporter@config/collector=all
```

Examples

URI

```
http://10.37.129.91:80/vtap/flow-exporter@config/collector=all
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/flow-exporter@config/collector=all" --
insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
  "Flow-Export": [
    {
      "Collector ID": 1,
      "name": "Name",
      "Collector Address": "10.9.9.186",
      "Export Address": "10.9.9.184",
      "Transport Protocol": "UDP",
      "Collector Port": 8008,
      "Export Port": 9009
    },
    {
      "collector ID": 2,
      "name": "test",
      "Collector Address": "10.37.131.165",
      "Export Address": "10.37.131.140",
      "Collector Port": 5030,
      "Export Port": 5031,
      "Transport Protocol": "udp"
    }
  ]
}
```

Individual flow-exporter collector

Resource URI

{http | https}://<host>:<port>/vtap/flow-exporter@config/collector=*<collector_id>*

Examples

URI

```
http://10.37.129.91:80/vtap/flow-exporter@config/collector=1
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/flow-exporter@config/collector=1" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
  "Flow-Export": [
    {
      "Collector ID": 1,
      "name": "Name",
      "Collector Address": "10.9.9.186",
      "Export Address": "10.9.9.184",
      "Transport Protocol": "UDP",
      "Collector Port": 8008,
      "Export Port": 9009
    }
  ]
}
```

Getting flow-exporter collector statistics

Retrieve statistics for all or specific flow-exporter collector from vTAP.

Method

GET

All flow-exporter collectors

Resource URI

```
{http | https}://<host>:<port>/vtap/flow-exporter@stats/collector=all
```

Examples

URI

```
http://10.37.129.91:80/vtap/flow-exporter@stats/collector=all
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/flow-exporter@stats/collector=all" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Flow-Export": [
        {
            "Collector ID": 1,
            "name": "Name",
            "Number Message Sent": 20,
            "Max Record Sent": 2,
            "Number Template Sent": 4,
            "Number of Data Record Sent": 32
        }
    ]
}
```

Individual flow-exporter collector

Resource URI

```
{http | https}://<host>:<port>/vtap/flow-exporter@stats/collector=<collector_id>
```

Examples

URI

```
http://10.37.129.91:80/vtap/flow-exporter@stats/collector=1
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/flow-exporter@stats/collector=1" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Flow-Export": [
        {
            "Collector ID": 1,
            "name": "Name",
            "Number Message Sent": 20,
            "Max Record Sent": 2,
            "Number Template Sent": 4,
            "Number of Data Record Sent": 32
        }
    ]
}
```

Getting flow-exporter export interval

Method

GET

Retrieve information about flow-exporter export interval.

Resource URI

```
{http | https}://<host>:<port>/vtap/export-interval
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get information about flow-exporter export interval.

URI

```
http://10.37.129.91:80/vtap/export-interval
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/export-interval" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    “export-interval” : 1
}
```

Getting flow-exporter template information

Retrieve configuration information for all or specific flow-exporter template from vTAP.

Method

GET

All flow-exporter templates

Resource URI

```
{http | https}://<host>:<port>/vtap/flow-exporter@config/template=all
```

Examples

URI

```
http://10.37.129.91:80/vtap/flow-exporter@config/template=all
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/flow-exporter@config/template=all" --insecure
```

Success Response Body

```
204 No Content
```

or

```
{
    "Flow-Export": [
        {
            "Template ID": 300,
            "Set ID": 3,
            "Number of Scope Fields": 2,
            "Number of Fields": 8,
            "Number of Dependent Templates": 0
        },
        {
            "Template ID": 257,
            "Set ID": 2,
            "Number of Fields": 24,
            "Number of Dependent Templates": 1,
            "Dependent Template ID": 300
        },
        {
            "Template ID": 256,
            "Set ID": 2,
            "Number of Fields": 20,
            "Number of Dependent Templates": 1,
            "Dependent Template ID": 300
        },
        {
            "Template ID": 258,
            "Set ID": 2,
            "Number of Fields": 8,
            "Number of Dependent Templates": 1,
            "Dependent Template ID": 300
        },
        {
            "Template ID": 272,
            "Set ID": 2,
            "Number of Fields": 4,
            "Number of Dependent Templates": 0
        },
        {
            "Template ID": 271,
            "Set ID": 2,
            "Number of Fields": 5,
            "Number of Dependent Templates": 0
        },
        {
            "Template ID": 273,
            "Set ID": 2,
            "Number of Fields": 11,
            "Number of Dependent Templates": 0
        }
    ]
}
```

Individual flow-exporter template

Resource URI

```
{http | https}://<host>:<port>/vtap/flow-exporter@config/template=<template_id>
```

Examples

URI

```
http://10.37.129.91:80/vtap/flow-exporter@config/template=1
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/flow-exporter@config/template=1" --insecure
```

Success Response Body

```
204 No Content
```

or

```
{
    "Flow-Export": [
        {
            "Template ID": 300,
            "Set ID": 3,
            "Number of Scope Fields": 2,
            "Number of Fields": 8,
            "Number of Dependent Templates": 0
        }
    ]
}
```

Getting flow-exporter template statistics

Retrieve statistics for all or specific flow-exporter template from vTAP.

Method

GET

All flow-exporter templates

Resource URI

```
{http | https}://<host>:<port>/vtap/flow-exporter@stats/template=all
```

Examples

URI

```
http://10.37.129.91:80/vtap/flow-exporter@stats/template=all
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/flow-exporter@stats/template=all" --insecure
```

Success Response Body

```
204 No Content
```

or

```
{
    "Flow-Export": [
        {
            "Template ID": 300,
            "Number of Template Sent": 0,
            "Number of DataRecord Sent": 6
        },
        {
            "Template ID": 257,
            "Number of Template Sent": 0,
            "Number of DataRecord Sent": 0
        },
        {
            "Template ID": 256,
            "Number of Template Sent": 0,
            "Number of DataRecord Sent": 0
        },
        {
            "Template ID": 258,
            "Number of Template Sent": 3,
            "Number of DataRecord Sent": 3
        },
        {
            "Template ID": 272,
            "Number of Template Sent": 0,
            "Number of DataRecord Sent": 0
        },
        {
            "Template ID": 271,
            "Number of Template Sent": 0,
            "Number of DataRecord Sent": 0
        },
        {
            "Template ID": 273,
            "Number of Template Sent": 0,
            "Number of DataRecord Sent": 0
        }
    ]
}
```

Individual flow-exporter template

Resource URI

{http | https}://<host>:<port>/vtap/flow-exporter@stats/template=<*template_id*>

Examples

URI

```
http://10.37.129.91:80/vtap/flow-exporter@stats/template=1
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/flow-exporter@stats/template=1" --insecure
```

Success Response Body

```
204 No Content
```

or

```
{
    "Flow-Export": [
        {
            "Template ID": 300,
            "Number of Template Sent": 0,
            "Number of DataRecord Sent": 6
        }
    ]
}
```

Getting ingress tunnel configuration

Retrieve configuration information for all or specific ingress tunnels from vTAP.

Method

GET

All ingress tunnels

Resource URI

```
{http | https}://<host>:<port>/vtap/ingress-tunnel@all.json
```

Examples

URI

```
http://10.37.129.91:80/vtap/ingress-tunnel@all.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/ingress-tunnel@all.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
  "ingress-tunnel": [
    {
      "tunnel-name": "ingress-ipip",
      "host1-ip": "10.37.129.190",
      "host2-ip": "10.37.129.193",
      "destination-port": 4789,
      "tunnel-type": "vxlan",
      "vni": 123
    },
    {
      "ingress-tunnel": [
        {
          "tunnel-name": "ingress-1",
          "host1-ip": "10.37.129.190",
          "host2-ip": "10.37.129.193",
          "tunnel-type": "gre"
        }
      ]
    }
  ]
}
```

Individual ingress tunnel

Resource URI

{http | https}://<host>:<port>/vtap/ingress-tunnel@<*ingress_tunnel_name*>.json

Examples

URI

```
http://10.37.129.91:80/vtap/ingress-tunnel@tunnel-2.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/ingress-tunnel@tunnel-2.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "ingress-tunnel": [
        {
            "tunnel-name": "ingress-ipip",
            "host1-ip": "10.37.129.190",
            "host2-ip": "10.37.129.193",
            "destination-port": 4789,
            "tunnel-type": "vxlan",
            "vni": 123
        }
    ]
}
```

Getting list of ingress ports

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/ingress-ports
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get the list of ingress ports.

URI

```
http://10.37.129.91:80/vtap/ingress-ports
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/ingress-ports" --insecure
```

Success Response Body

```
204 No Content
```

or

```
{
    "ingress port(s)": [
        {
            "Port Name": "IP-PORT-RX",
            "Interface name": "ingress",
            "Jumbo Frame": "enabled"
        },
        {
            "Port Name": "IP-PORT-TX",
            "Interface name": "egress",
            "Jumbo Frame": "enabled"
        }
    ]
}
```

Getting interface config

Retrieve interface configuration information from vTAP.

Method

GET

All interfaces

Resource URI

```
{http | https}://<host>:<port>/vtap/interface-config@all.json
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get information about all interfaces.

URI

```
http://10.37.129.91:80/vtap/interface-config@all.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/interface-config@all.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
{
    "Interface": [
        {
            "name": "ip",
            "traffic type": "ip",
            "egress": "drop",
            "sampling policy": "sp1",
            "smartmatch Policy": "smatch-1",
            "mode": "session",
            "export-ipfix": "disabled",
            "header-strip": "-",
            "packet-slice": "-"
        }
    ]
}
```

Getting interface statistics

Retrieve interface statistics.

Method

GET

All interfaces

Resource URI

```
{http | https}://<host>:<port>/vtap/interface-stats
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get interface statistics.

URI

```
http://10.37.129.91:80/vtap/interface-stats
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/interface-stats" --insecure
```

Success Response Body

```
204 No Content
```

or


```
{
    "interface stats": [
        {
            "Sampling": [
                {
                    "sampling policy-name": "-"
                }
            ],
            "Smartmatch": [
                {
                    "smatch policy-name": "-"
                }
            ],
            "Egress": [
                {
                    "egress alias name": "eg1",
                    "forwarded packets": 0
                }
            ],
            "Header-strip": [
                {
                    "header-strip": "nvgre,mpls",
                    "vxlan packets": 0,
                    "vxlan bytes": 0,
                    "nvgre packets": 0,
                    "nvgre bytes": 0,
                    "erspan2 packets": 0,
                    "erspan2 bytes": 0,
                    "gtpu packets": 0,
                    "gtpu bytes": 0,
                    "mpls packets": 0,
                    "mpls bytes": 0,
                    "vlan packets": 0,
                    "vlan bytes": 0,
                    "802.1br_vntag packets": 0,
                    "802.1br_vntag bytes": 0
                }
            ],
            "Packet-slice": [
                {
                    "packet-slice": 200,
                    "packets": 0,
                    "bytes": 0
                }
            ],
            "IPFIX": [
                {
                    "IPFIX": "Disabled"
                }
            ]
        }
    ]
}
```

Getting link status

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/link-stats
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get the link status.

URI

```
http://10.37.129.91:80/vtap/link-stats
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/link-stats" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "link-status": [
        {
            "NIC Port  ( 0)": " 0 (Rx-Port)",
            "Link Status                                          ( 0)": "     UP    ",
            "Link Speed                                           ( 0)": "     UP    ",
            "Link Duplex                                          ( 0)": "     FULL  ",
            "NIC Port  ( 1)": " 1 (Tx-Port)",
            "Link Status                                          ( 1)": "     UP    ",
            "Link Speed                                           ( 1)": "     UP    ",
            "Link Duplex                                          ( 1)": "     FULL  "
        }
    ]
}
```

Getting logging information

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/logging@all.json
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get all logging information.

URI

```
http://10.37.129.91:80/vtap/logging@all.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/logging@all.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "logging": [
        {
            "tunnel-flow": "debug",
            "smatch": "critical",
            "sampling": "critical",
            "ipfix": "critical",
            "mgmt": "critical",
            "L7app": "critical",
            "kni": "critical"
        }
    ]
}
```

Getting mempool usage statistics

Resource URI

```
{http | https}://<host>:<port>/vtap/mempool-usage
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get mempool usage statistics.

URI

```
http://10.37.129.91:80/vtap/mempool-usage
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/mempool-usage" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "mempool-usage": [
        {
            "Total Memory Allocated                  ": "    235520 MB ",
            "Total Memory Used                       ": "       804 MB ",
            "Total Memory Available                  ": "    234716 MB ",
            "Mempool ( 1 )": " 1 ",
            "Number of Used Entries              ( 1 )": "    4068",
            "Number of Free Entries              ( 1 )": "    127005",
            "Total Number of Entries             ( 1 )": "    131073",
            "Mempool ( 2 )": " 2 ",
            "Number of Used Entries              ( 2 )": "    2368",
            "Number of Free Entries              ( 2 )": "    128705",
            "Total Number of Entries             ( 2 )": "    131073",
            "Mempool ( 3 )": " 3 ",
            "Number of Used Entries              ( 3 )": "    8256",
            "Number of Free Entries              ( 3 )": "    28609",
            "Total Number of Entries             ( 3 )": "    36865",
            "Mempool ( 4 )": " 4 ",
            "Number of Used Entries              ( 4 )": "    4352",
            "Number of Free Entries              ( 4 )": "    28417",
            "Total Number of Entries             ( 4 )": "    32769"
        }
    ]
}
```

Getting NIC statistics

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/nic-stats
```

NOTE

The NIC statistics does not display byte count details for Rx and Tx for e1000 vNIC driver.

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get NIC statistics.

URI

```
http://10.37.129.91:80/vtap/nic-stats
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/nic-stats" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "nic-stats": [
        {
            "NIC Stats": "",
            "NIC Port  ( 0)": " 0",
            "Total number of received packets ( 0)": " 1055",
            "Total number of transmitted packets ( 0)": " 0",
            "Total number of received bytes ( 0)": " 0",
            "Total number of transmitted bytes ( 0)": " 0",
            "Total number of RX packets dropped by hardware ( 0)": " 0",
            "Total number of erroneous received packets ( 0)": " 0",
            "Total number of failed transmitted packets ( 0)": " 0",
            "Total number of RX mbuf allocation failures ( 0)": " 0",
            "NIC Port  ( 1)": " 1",
            "Total number of received packets ( 1)": " 6",
            "Total number of transmitted packets ( 1)": " 36",
            "Total number of received bytes ( 1)": " 0",
            "Total number of transmitted bytes ( 1)": " 0",
            "Total number of RX packets dropped by hardware ( 1)": " 0",
            "Total number of erroneous received packets ( 1)": " 0",
            "Total number of failed transmitted packets ( 1)": " 0",
            "Total number of RX mbuf allocation failures ( 1)": " 0"
        }
    ]
}
```

Getting packet statistics for all parameters

Resource URI

```
{http | https}://<host>:<port>/vtap/packet-stats@all
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get packet statistics for LTE SIP.

URI

```
http://10.37.129.91:80/vtap/packet-stats@all
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/packet-stats@all" --insecure
```

Success Response Body

```
204 No Content
```

or


```
{
    "Packet-Stats": [
        {
            "Rx": [
                {
                    "Packet Stats for Port ": " ingress",
                    "No Of Packets Received ": " 0",
                    "No Of Packets Filtered ": " 0"
                }
            ]
        },
        {
            "Tx": [
                {
                    "Packet Stats for Port ": " egress",
                    "No of Packets Transmitted ": " 9"
                }
            ]
        },
        {
            "Egress Statistics": [
                {
                    "egress name": "VXLAN_STAT2",
                    "egress type": "vxlan",
                    "packet count": 0,
                    "bytes count": 0,
                    "packet rate": 0,
                    "throughput": 0
                },
                {
                    "egress name": "GRE_STATNEW",
                    "egress type": "gre",
                    "packet count": 0,
                    "bytes count": 0,
                    "packet rate": 0,
                    "throughput": 0
                },
                {
                    "egress name": "VXLAN_STAT1",
                    "egress type": "vxlan",
                    "packet count": 0,
                    "bytes count": 0,
                    "packet rate": 0,
                    "throughput": 0
                },
                {
                    "egress name": "VXLAN_STAT3",
                    "egress type": "vxlan",
                    "packet count": 0,
                    "bytes count": 0,
                    "packet rate": 0,
                    "throughput": 0
                },
                {
                    "egress name": "VXLAN_STAT4",
                    "egress type": "vxlan",
                    "packet count": 0,
                    "bytes count": 0,
                    "packet rate": 0,
                    "throughput": 0
                }
            ]
        },
        {
            "Total Egress Statistics": [
```

```
                {
                    "dropped packets": 0,
                    "dropped rate": 0,
                    "dropped throughput": 0,
                    "total packets": 0,
                    "total rate": 0,
                    "total throughput": 0
                }
            ]
        },
        {
            "Smartmatch Statistics": [
                {
                    "packets received": 0,
                    "bytes received": 0,
                    "packets forwarded": 0,
                    "bytes forwarded": 0,
                    "packets dropped": 0,
                    "bytes dropped": 0,
                    "packets not matched smartmatch rule": 0,
                    "packets not matched flex match": 0,
                    "packets matched flex match": 0
                }
            ]
        }
    ]
}
```

Getting packet statistics for egress

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/packet-stats@egs
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get egress packet statistics.

URI

```
http://10.37.129.91:80/vtap/packet-stats@egs
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/packet-stats@egs" --insecure
```

Success Response Body

```
204 No Content
```

or


```
{
    "Packet-Stats": [
        {
            "Egress Statistics": [
                {
                    "egress name": "sm_egress",
                    "egress type": "vxlan",
                    "packet count": 0,
                    "bytes count": 0,
                    "packet rate": 0,
                    "throughput": 0
                },
                {
                    "egress name": "alias_egress",
                    "egress type": "gre",
                    "packet count": 0,
                    "bytes count": 0,
                    "packet rate": 0,
                    "throughput": 0
                }
            ]
        },
        {
            "Total Egress Statistics": [
                {
                    "dropped packets": 0,
                    "dropped rate": 0,
                    "dropped throughput": 0,
                    "total packets": 0,
                    "total rate": 0,
                    "total throughput": 0
                }
            ]
        }
    ]
}
```

Getting packet statistics for Rx

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/packet-stats@rx
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get Rx statistics.

URI

```
http://10.37.129.91:80/vtap/packet-stats@rx
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/packet-stats@rx" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Packet-Stats": [
        {
            "Rx": [
                {
                    "Packet Stats for Port ": " ingress",
                    "No Of Packets Received ": " 2754680",
                    "No Of Packets Filtered ": " 0"
                }
            ]
        }
    ]
}
```

Getting packet statistics for SMARTMatch

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/packet-stats@smatch
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get SMARTMatch packet statistics.

URI

```
http://10.37.129.91:80/vtap/packet-stats@smatch
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/packet-stats@smatch" --insecure
```

Success Response Body

```
204 No Content
```

or


```
{
    "Packet-Stats": [
        {
            "Smartmatch Statistics": [
                {
                    "packets received": 0,
                    "bytes received": 0,
                    "packets forwarded": 0,
                    "bytes forwarded": 0,
                    "packets dropped": 0,
                    "bytes dropped": 0,
                    "packets not matched smartmatch rule": 0,
                    "packets not matched flex match": 0,
                    "packets matched flex match": 0
                }
            ]
        }
    ]
}
```

Getting packet statistics for Tx

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/packet-stats@tx
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get Tx statistics.

URI

```
http://10.37.129.91:80/vtap/packet-stats@tx
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/packet-stats@tx" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Packet-Stats": [
        {
            "Tx": [
                {
                    "Packet Stats for Port ": " egress",
                    "No of Packets Transmitted ": " 0"
                }
            ]
        }
    ]
}
```

Getting sampling policy configuration

Retrieve configuration information for all or specific sampling policies from vTAP.

Method

GET

All sampling policies

Resource URI

```
{http | https}://<host>:<port>/vtap/sampling-policy@all.json
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get information about all sampling policies.

URI

```
http://10.37.129.91:80/vtap/sampling-policy@all.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/sampling-policy@all.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Sampling-Policy": [
        {
            "policy name": "sp100",
            "state": "inactive",
            "sample-rate": 100,
            "preserve-pkts": 15
        }
    ]
}
```

Individual sampling policy

Resource URI

```
{http | https}://<host>:<port>/vtap/sampling-policy@<sampling_policy_name>.json
```

Examples

URI

```
http://10.37.129.91:80/vtap/sampling-policy@sp1.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/sampling-policy@sp1.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Sampling-Policy": [
        {
            "policy name": "sp1",
            "state": "active",
            "sample-rate": 10,
            "preserve-pkts": 0
        }
    ]
}
```

Error Response Body

```
400 Bad Request
{
    "Sampling-Policy": [
        {
            "detail": "sp2",
            "response": "sampling policy not present"
        }
    ]
}
```

Getting sampling statistics

Retrieve sampling statistics from vTAP.

Method

GET

All sampling policies

Resource URI

```
{http | https}://<host>:<port>/vtap/sampling-stats@all.json
```

Examples

URI

```
http://10.37.129.91:80/vtap/sampling-stats@all.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/sampling-stats@all.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Sampling-Stats": [
        {
            "policy name": "sp1",
            "sample-rate": 10,
            "total-flow": 30,
            "drop-flow": 3,
            "drop-pkts-count": 2,
            "preserve-pkts-count": 0
        }
    ]
}
```

Individual sampling policy

Resource URI

```
{http | https}://<host>:<port>vtap/sampling-stats@<sampling_policy_name>.json
```

Examples

URI

```
http://10.37.129.91:80/vtap/sampling-stats@sp1.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/sampling-stats@sp1.json" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Sampling-Stats": [
        {
            "policy name": "sp1",
            "sample-rate": 10,
            "total-flow": 30,
            "drop-flow": 3,
            "drop-pkts-count": 2,
            "preserve-pkts-count": 0
        }
    ]
}
```

Getting SMARTMatch alias stats

Retrieve SMARTMatch alias statistics.

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/smart-match@alias-stats
```

Examples

URI

```
http://10.37.129.91:80/vtap/smart-match@alias-stats
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/smart-match@alias-stats" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "smatch-alias": [
        {
            "flexmatch alias ": "IPV4_outergre",
            "policy name ": "smatch-1",
            "rule id ": "2",
            "packets matched ": "0",
            "packets forwarded/dropped ": "-",
            "egress ": "-"
        },
        {
            "flexmatch alias ": "a2",
            "policy name ": "smatch-1",
            "rule id ": "1",
            "packets matched ": "0",
            "packets forwarded/dropped ": "-",
            "egress ": "-"
                  }
          ]
       }
```

Getting SMARTMatch policy configuration

Retrieve configuration information for all or specific SMARTMatch policies from vTAP.

Method

GET

All SMARTMatch policies

Resource URI

```
{http | https}://<host>:<port>/vtap/smatch-policy@all.json
```

Example

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get information about all SMARTMatch policies.

URI

```
http://10.37.129.91:80/vtap/smatch-policy@all.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/smatch-policy@all.json" --insecure
```

Success Response Body

```
204 No Content
```

or


```
{
    "Smatch-Policy": [
        {
            "Policy Name": "sp2",
            "State": "inactive",
            "Rule Info": [
                {
                    "rule id": 1,
                    "vlan id": "any",
                    "protocol": "udp",
                    "host1 ip": "any",
                    "host1 port": "any",
                    "host2 ip": "any",
                    "host2 port": "any",
                    "tunnel name": "--",
                    "egress": "--",
                    "sampling policy": "--",
                    "number of alias": 0,
                    "export IPFIX": "disable",
                    "header-strip": "-",
                    "packet-slice": "-"
                },
                {
                    "rule id": 2,
                    "vlan id": "any",
                    "protocol": "tcp",
                    "host1 ip": "any",
                    "host1 port": "any",
                    "host2 ip": "any",
                    "host2 port": "any",
                    "tunnel name": "--",
                    "egress": "--",
                    "sampling policy": "--",
                    "number of alias": 0,
                    "export IPFIX": "disable",
                    "header-strip": "-",
                    "packet-slice": "-"
                }
            ]
        }
    ]
}
```


Individual SMARTMatch policy

Resource URI

```
{http | https}://<host>:<port>/vtap/smatch-policy@<smatch_policy_name>.json
```

Example

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get information about SMARTMatch policy named `sm1`.

URI

```
http://10.37.129.91:80/vtap/smatch-policy@sm1.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/smatch-policy@sm1.json" --insecure
```

Success Response Body

```
204 No Content
```

or


```
{
    "Smatch-Policy": [
        {
            "Policy Name": "sp2",
            "State": "inactive",
            "Rule Info": [
                {
                    "rule id": 1,
                    "vlan id": "any",
                    "protocol": "udp",
                    "host1 ip": "any",
                    "host1 port": "any",
                    "host2 ip": "any",
                    "host2 port": "any",
                    "tunnel name": "--",
                    "egress": "--",
                    "sampling policy": "--",
                    "number of alias": 0,
                    "export IPFIX": "disable",
                    "header-strip": "-",
                    "packet-slice": "-"
                }
            ]
        }
    ]
}
```

Getting SMARTMatch rule statistics

Retrieves SMARTMatch rules statistics.

Resource URI

```
{http | https}://<host>:<port>/vtap/smart-match@rules-stats/ruleid=<rule_id>
```

Examples

URI

```
http://10.37.129.91:80/vtap/smart-match@rules-stats/rule-id=1
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/smart-match@rules-stats/rule-id=1" --insecure
```

Success Response Body

```
204 No Content
```

or


```
{
    "smatch-rule-stats": [
        {
            "smatch policy name": "smp1",
            "Rule Info": [
                {
                    "rule id": 1,
                    "rx packets": 0,
                    "rx bytes": 0,
                    "tx packets": 0,
                    "tx bytes": 0,
                    "dropped packets": 0,
                    "sampled packets": 0,
                    "preserve packets": 0,
                    "number of flows": 0,
                    "packet-slice packets": 0,
                    "packet-slice bytes": 0,
                    "Header-strip": [
                        {
                            "vxlan packets": 0,
                            "vxlan bytes": 0,
                            "nvgre packets": 0,
                            "nvgre bytes": 0,
                            "erspan2 packets": 0,
                            "erspan2 bytes": 0,
                            "gtpu packets": 0,
                            "gtpu bytes": 0,
                            "mpls packets": 0,
                            "mpls bytes": 0,
                            "vlan packets": 0,
                            "vlan bytes": 0,
                            "802.1br_vntag packets": 0,
                            "802.1br_vntag bytes": 0
                        }
                    ]
                }
            ]
        }
    ]
}
```

Getting SMARTMatch policies and rules

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/smart-match@rule-detail
```

Example 1 - Rules, detail

URI

```
http://10.37.129.91:80/vtap/smart-match@rule-detail
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/smart-match@rule-detail" --insecure
```

Success Response Body

```
204 No Content
```

or


```
{
    "rule detail": [
        {
            "Policy Name": "sp2",
            "State": "inactive",
            "Rule Info": [
                {
                    "rule id": 1,
                    "vlan id": "any",
                    "protocol": "udp",
                    "host1 ip": "any",
                    "host1 port": "any",
                    "host2 ip": "any",
                    "host2 port": "any",
                    "tunnel name": "--",
                    "egress": "egr1",
                    "sampling policy": "--",
                    "number of alias": 0,
                    "export IPFIX": "disable",
                    "header-strip": "vxlan,mpls",
                    "packet-slice": "300"
                },
                {
                    "rule id": 2,
                    "vlan id": "any",
                    "protocol": "tcp",
                    "host1 ip": "any",
                    "host1 port": "any",
                    "host2 ip": "any",
                    "host2 port": "any",
                    "tunnel name": "t1",
                    "egress": "--",
                    "sampling policy": "--",
                    "number of alias": 0,
                    "export IPFIX": "disable",
                    "header-strip": "nvgre,mpls",
                    "packet-slice": "220"
                }
            ]
        }
    ]
}
```

Getting vTAP status

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/status@vtap
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to check the status of the vTAP.

URI

```
http://10.37.129.91:80/vtap/status@vtap
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/status@vtap" --insecure
```

Success Response Body

```
204 No Content
```

or

```
{

"status": [
        {
            "vtap": "ACTIVE",
            "Physical contiguous memory": "initialized",
            "Mempool and hash": "initialized",
            "status": "   Active: active (running) since Wed 2019-01-23 08:25:29 EST; 2h 17min ago"
        }
    ]
}
```

Getting status for NGINX service

Resource URI

```
{http | https}://<host>:<port>/vtap/status@vtap-nginx
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get the status for NGINX service.

URI

```
http://10.37.129.91:80/vtap/status@vtap-nginx
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/status@vtap-nginx" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "status": [
        {
            "vtap-ngnix": "Active",
         "Status": "   Active: active (running) since Tue 2019-01-22 15:51:22 EST; 18h ago\n"

        }
    ]
}
```

Getting status for SNMP service

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/status@vtap-snmp
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get the status for SNMP service.

URI

```
http://10.37.129.91:80/vtap/status@vtap-snmp
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/status@vtap-snmp" --insecure
```

Success Response Body

```
204 No Content
```

or

```
{
    "status": [
        {
            "vtap": "ACTIVE",
            "Physical contiguous memory": "initialized",
            "Mempool and hash": "initialized",
            "status": "   Active: active (running) since Wed 2019-01-23 08:25:29 EST; 2h 22min ago"
        },
        {
            "vtap-snmp": "Active",
            "Status": "   Active: active (running) since Tue 2019-01-22 15:51:53 EST; 18h ago\n"
        },
        {
            "vtap-ngnix": "Active",
            "Status": "   Active: active (running) since Tue 2019-01-22 15:51:22 EST; 18h ago\n"
        }
    ]
}
```

Getting system statistics

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/system-stats
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get system statistics.

URI

```
http://10.37.129.91:80/vtap/system-stats
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/system-stats" --insecure
```

Success Response Body

```
204 No Content
```

or


```
{
    "system-stats": [
        {
            "Rx Packets": 91506,
            "Rx Bytes": 7753182,
            "Drop Packets": 91504,
            "Drop Bytes": 7753062,
            "Flows": 5,
            "Tunnels": 0,
            "Not Sent Flows": 0,
            "Tunnels Terminated": 0,
            "Kni Rx Packets": 4,
            "Kni Tx Packets": 16,
            "IP Fragmented Packets": 0,
            "IP Reassembly Packets": 0,
            "Total Deleted Tunnels": 0,
            "Total Deleted Flows": 100
        }
    ]
}
```

Getting egress tunnel information

Retrieve information about the tunnels received by vTAP.

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/tunnel-info@all.json
```

Examples

URI

```
http://10.37.129.91:80/vtap/tunnel-info@all.json
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/tunnel-info@all.json" --insecure
```

Success Response Body

```
204 No Content
```

or


```
{
    "tunnel-info": [
        {
            "tunnel-id": 2,
            "parent-tunnel-id": 0,
            "tunnel-type": "gre",
            "smatch rule id": "-",
            "host1-ip": "10.0.0.1",
            "host2-ip": "10.0.0.2",
            "child tunnels": 0,
            "flows": 1,
            "dir1 rx packets": 5,
            "dir1 rx bytes": 690,
            "dir2 rx packets": 5,
            "dir2 rx bytes": 690,
            "egress-action": "drop"
        }
    ]
}
```

Getting vTAP version

Resource URI

```
{http | https}://<host>:<port>/vtap/version
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to get the vTAP release version.

URI

```
http://10.37.129.91:80/vtap/version
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/version" --insecure
```

Success Response Body

```
204 No Content
```

or

```
200 OK
{
    "Version": [
        {
            "Version": "2.0.0"
        }
    ]
}
```

Starting test pcap

Method

GET

Resource URIs

```
{http | https}://<host>:<port>/vtap
```

Examples

Example 1

This is an example of a POST method request to a vTAP IP at 10.37.129.91 to start sending test pcaps.

URI

```
http://10.37.129.91:80/vtap/send-pcap
```

Request Body

None

cURL command

```
curl -v -X POST -H "Expect:" "https://10.37.129.91:443/vtap/send-pcap --insecure
```

Success Response Body

```
204 No Content
```

or

```
{
    "start-testconnection": [
        {
            "starting sample pcap": 0
        }
    ]
}
```

or

```
{
    "start-testconnection": [
        {
            "sample pcap is already running": 0
        }
    ]
}
```

Stopping test pcap

Method

GET

Resource URI

```
{http | https}://<host>:<port>/vtap/stop-pcap
```

Examples

The following is an example of a GET method request to a vTAP IP at 10.37.129.91 to stop sending test pcaps.

URI

```
http://10.37.129.91:80/vtap/stop-pcap
```

Request Body

None

cURL command

```
curl -v -X GET -H "Expect:" "https://10.37.129.91:443/vtap/stop-pcap" --insecure
```

Success Response Body

```
204 No Content
```

or

```
{
    "stop-testconnection": [
        {
            "stopping sample pcap": 0
        }
    ]
}
```

or

```
{
    "stop-testconnection": [
        {
            "sample pcap is already stopped": 0
        }
    ]
}
```