# SD-AVC REST API

## REST API Overview

The REST API provides numerous system functions, including:

- Displaying information about devices in the SD-AVC network
- Controlling external sources
- Displaying information about generic traffic
- Creating user-defined applications

**Note** Using the REST API requires authentication. See Authentication from SD-AVC Network Service, on page 4.

***Table 1: Authentication***

| | |
|---|---|
| **POST**<br>https://*SD-AVC-network-service-address*:8443/avc-sd-service/external-api/login | Acquires an authentication token, enabling use of the REST API.<br>Authentication from SD-AVC Network Service, on page 4 |

*Table 2: System*

| | |
|---|---|
| **GET** /avc-sd-service/external-api/system-info | Displays the SD-AVC version and system times.<br>Display System Information, on page 7 |
| **GET** /avc-sd-service/external-api/devices | Displays devices in the SD-AVC network.<br>Display Devices, on page 7 |
| **POST** /avc-sd-service/external-api/remove-devices | Removes a device from the SD-AVC network.<br>Delete Devices from SD-AVC, on page 8 |
| **GET** /avc-sd-service/external-api/visibility?period=*period*<br>-<br>**GET** /avc-sd-service/external-api/visibility/*segmentName*<br>?period=*period*<br>-<br>**GET** /avc-sd-service/external-api/visibility/*segmentName*/<br>*deviceName*?period=*period* | Display traffic analytics (applications and bandwidth) for the complete SD-AVC network, a specific segement, or a specific device.<br>Display Traffic Analytics, on page 9 |

*Table 3: Cloud Connector*

| | |
|---|---|
| **POST** /avc-sd-service/external-api/cloud/authorize<br>**Note** Deprecated in SD-AVC 4.0.0. Use: /avc-sd-service/external-api/cloud/configure | Connect to Cloud Connector.<br>Connect to Cloud Connector, on page 10 |
| **POST** /avc-sd-service/external-api/cloud/configure | Enter Cloud Connector credentials.<br>Configure Cloud Connector Credentials, on page 5 |
| **POST**<br>/avc-sd-service/external-api/cloud/configureSegment?segment=*segmentName* | Specify the location for storing Cloud Connector telemetry data for a specific segment.<br>Configure Cloud Connector Telemetry Data Location, on page 6 |
| **POST** /avc-sd-service/external-api/cloud/disable | Disable Cloud Connector.<br>Disable Cloud Connector, on page 11 |
| **GET** /avc-sd-service/external-api/cloud/status | Get Cloud Connector status.<br>Display Cloud Connector Status, on page 13 |

| | |
|---|---|
| **POST** /avc-sd-service/external-api/cloud/removeCredentials | Clears the credentials that have been entered for Cloud Connector.<br>Remove Cloud Connector Credentials, on page 11 |
| **GET** /avc-sd-service/external-api/cloud/getCurrentCloudConfig | Get current configuration for Cloud Connector.<br>Display Current Cloud Configuration, on page 12 |
| **GET** /avc-sd-service/external-api/app-rules?detailed=true&sourceId=PP_Extension | Get currently collected cloud data (rules, server names, sockets, and so on).<br>Display Cloud Data, on page 12 |

*Table 4: External Sources*

| | |
|---|---|
| **POST** /avc-sd-service/external-api/external-sources/*externalSourceName* | Enables or disables receiving data from an external source.<br>Enable/Disable External Sources, on page 14 |
| **GET** /avc-sd-service/external-api/external-sources | Displays status of external sources.<br>Display Status of External Sources, on page 15 |

*Table 5: User-defined Applications*

| | |
|---|---|
| **POST** /avc-sd-service/external-api/app-rules<br>-<br>**POST** /avc-sd-service/external-api/app-rules?segment=*segmentName* | Create one or more user-defined applications.<br>Create User-defined Application Rules, on page 16 |
| **PUT** /avc-sd-service/external-api/app-rules<br>-<br>**PUT** /avc-sd-service/external-api/app-rules?segment=*segmentName* | Add a user-defined application to an existing set.<br>Add a User-defined Application Rule, on page 23 |

| | |
|---|---|
| **GET** /avc-sd-service /external-api/app-rules<br>-<br>**GET** /avc-sd-service /external-api/app-rules?sourceId=*sourceId*<br>-<br>**GET** /avc-sd-service /external-api/app-rules?segment=*segmentName*<br>-<br>**GET** /avc-sd-service /external-api/app-rules?segment=*segmentName*&app=*applicationName* | Displays user-defined applications defined by REST API.<br>Display User-defined Application Rules, on page 24 |
| **GET** /avc-sd-service/external-api/app-rules/status<br>-<br>**GET** /avc-sd-service /external-api/app-rules/status[?sourceId=*sourceId*] | Displays activation status of user-defined applications, per device.<br>Display User-defined Application Status, on page 25 |
| **DELETE** /avc-sd-service /external-api/app-rules<br>-<br>**DELETE** /avc-sd-service /external-api/app-rules?sourceId=*sourceId*<br>-<br>**DELETE** /avc-sd-service /external-api/app-rules?segment=*segmentName*<br>-<br>**DELETE** /avc-sd-service /external-api/app-rules?segment=*segmentName*&app=*applicationName* | Deletes user-defined applications.<br>Delete User-defined Applications, on page 26 |

***Table 6: Generic Applications***

| | |
|---|---|
| **GET** /avc-sd-service/external-api/apps/generics | Displays the list of traffic types that contribute to "generic" traffic.<br>Display Generic Application Traffic Types, on page 27 |

# Authentication from SD-AVC Network Service

Using the REST API requires a token-based authentication from the SD-AVC network service. To acquire an authentication token:

1. Send the following HTTP request to the API:

   **POST** https://*SD-AVC-network-service-address*:8443/avc-sd-service/external-api/login

   **Example**:

```
POST https://192.168.0.1:8443/avc-sd-service/external-api/login
```

- In the request header, include the following key:

  **Content-Type**: application/x-www-form-urlencoded

- In the request body, include the following keys, providing login credentials:

  **username**: *username*
  **password**: *password*

2. The API response body provides an authentication token. Use the token to authorize REST API calls to the SD-AVC network service.

**Note** The token expires after 12 hours.

**Example**:

```
{"token":"Bearer eyJhbGciOiJIUzUxMiJ9.eyJqdGkiOiJhYjZkGGUxOS0zMmU3LTRlY2ItYWQ5OC
1kYmVmZTdjZjE5YzYiLCJzdWIiOiJsYWIiLCJleHAiOjE1MzAwMqk1MzJ9.EfP3wd4fZbWrOQ6Skh-I0
bbPffF4NaruB-o_OV0EQ7fwMwfmkUUNP00R58fRGKkYWR3tQu8HjoVDp37EPtDl5Q"}
```

3. Use this token in the "Authorization" request header field of each HTTP request.

# Configure Cloud Connector Credentials

### API

POST /avc-sd-service/external-api/cloud/configure

### Description

Configures the credentials for Cloud Connector. Use this when specifying the location for storing Cloud Connector telemetry data for one or more network segments.

After using this API to configure the credentials, you can specify the location for storing Cloud Connector telemetry data individually for each segment. See Configure Cloud Connector Telemetry Data Location, on page 6.

**Note** This API is an alternative to the following API. Do not use them together.

POST /avc-sd-service/external-api/cloud/authorize

### Body

```
{
    "clientID": ”clientId”,
    "clientSecret": "secret",
    "orgName": "organizationName",
}
```

*Table 7: Properties*

| Property | Description |
|---|---|
| *clientId* | Client ID. |
| *secret* | Secret for authentication. |
| *organizationName* | Organization name. May include spaces. |

# Configure Cloud Connector Telemetry Data Location

### API

POST /avc-sd-service/external-api/cloud/configureSegment?segment=*segmentName*

### Description

Enables or disables transmitting and receiving Cloud Connector telemetry data, and specifies the location for storing the telemetry data. When using this API, specify the Cloud Connector credentials using the following API:

POST /avc-sd-service/external-api/cloud/configure

See Configure Cloud Connector Credentials, on page 5.

### Body

```
{
    "txConfig": {
         "isEnabled": [true | false],
         "location": "locationId"
    },
    “rxConfig”: {
         “isEnabled”: [true | false]
    }
}
```

*Table 8: Properties*

| Property | Description |
|---|---|
| segment=*segmentName* | Specifies a network segment. |
| txConfig | isEnabled: Enable or disable transmitting Cloud Connector telemetry data for the specified segment.<br>location: Specify the location for storing Cloud Connector telemetry data. See *locationId* below. |
| rxConfig | isEnabled: Enable or disable receiving Cloud Connector telemetry data for the specified segment. |
| *locationId* | Location for storing Cloud Connector telemetry data for the specified segment.<br>Values: ASIA, CANADA, EU, US |

### Example

In the following example, transmitting and receiving telemetry data are enabled, and the location is specified as "US".

```
{
    "txConfig": {
         "isEnabled": true,
         "location": "US"
    },
    “rxConfig”: {
         “isEnabled”: true
    }
}
```

# System

## System Overview

The REST API can display information about the SD-AVC system, and change the configuration.

## Display System Information

### API

GET /avc-sd-service/external-api/system-info

### Description

Displays:

- Current time: Time in UNIX format.
- System uptime: SD-AVC uptime in milliseconds.
- SD-AVC version
- Office 365 Connector status (enabled/disabled, errors, warnings)

### Example Response

```
{
    "systemTime": "2019-06-26T12:19:02Z",
    "systemUpTimeSec": "13490106",
    "version": "4.0.0",
    "o365Connector": {
        "isEnabled": true,
        "error": [],
        "warning": []
    },
    "cloudStage": "test"
}
```

## Display Devices

### API

GET /avc-sd-service/external-api/devices

**Description**

Displays the devices in the SD-AVC network, organized by segment, in JSON format. The response includes errors and warnings, and additional information per device.

**Response**

The output shows errors and warnings for:

- total network
- each segment
- each device

**Example Response**

The example represents a network with one segment (datacenter-01) and one device (asr-device-100) within that segment.

```
{
   "total":{
      "connection":{
         "error":[],
         "warn":[]
      },
   },
   "segments":[
      {
         "name":"datacenter-01",
         "connection":{
            "error":[],
            "warn":[]
         },
         "devices":[
            {
               "name":"asr-device-100",
               "ip":"192.168.1.0",
               "connection":{
                  "error":[],
                  "warn":[]
               }
            }
         ]
      }
   ]
}
```

# Delete Devices from SD-AVC

**API**

POST /avc-sd-service/external-api/remove-devices

**Description**

Removes a device from the SD-AVC network. Specify the device and segment in the body.

**Body**

```
{
   "devices":[
      {
         "name":"device-name-1",
         "ip":"address-1"
      },
      {
         "name":"device-name-2",
         "ip":"address-2"
      }
   ],
   "segment":"segment-name"
}
```

### Example Body

```
{
   "devices":[
      {
         "name":"dev1",
         "ip":"10.10.10.10"
      },
      {
         "name":"dev2",
         "ip":"10.10.10.11"
      }
   ],
   "segment":"dnac"
}
```

### Example Response

```
{"success":true,"message":"2 devices from segment dnac were deleted successfully"}
```

# Display Traffic Analytics

### API

GET /avc-sd-service/external-api/visibility?period=*period*

-

GET /avc-sd-service/external-api/visibility/*segmentName*?period=*period*

-

GET /avc-sd-service/external-api/visibility/*segmentName*/[*deviceName* | *deviceAddress*]?period=*Period*

### Description

Displays traffic analytics (applications and bandwidth) for the complete SD-AVC network, a specific segment, or a specific device. Optionally, specify a period for the analytics. The response includes:

- Application name and bandwidth (bytes) used by the application
- Total bandwidth (bytes) used

*Table 9: Properties*

| Property | Description |
|---|---|
| *segmentName* | (Optional)<br>Specifies a segment. Response includes only analytics from this segment. |
| *deviceName* | (Optional)<br>Specifies a device by name. Response includes only analytics from this device. |
| *deviceAddress* | (Optional)<br>Specifies a device by IP address. Response includes only analytics from this device. |
| period | Use ?period=*period* to specify the period to include in the analytics.<br>Possible values for *period*: 120, 720, 1440, 2880 minutes (These correspond to 2, 12, 24, and 48 hours.) |

**Example:**

In this example, the period is set to 24 hours (1440 minutes).

```
GET /avc-sd-service/external-api/visibility/datacenter01/device-300?period=1440

{
"apps": [{
"name": "vmwarevsphere",
"bandwidth": 226331127989634
}, {
"name": "telepresencecontrol",
"bandwidth ": 146787859067274
}, {
"name": "unknown",
"bandwidth": 132586088501412
}],
"totalBandwidth": 505705075558320
}
```

# Cloud Connector

## Connect to Cloud Connector

**API:**

POST /avc-sd-service/external-api/cloud/authorize

**Description:**

Connect to the Cloud Connector, using credentials. See Cloud Connector.

Note

- Deprecated in SD-AVC 4.0.0. Use: POST /avc-sd-service/external-api/cloud/configure
- If continuing to use this deprecated API, do not use it together with the POST /avc-sd-service/external-api/cloud/configure API.

**Example:**

```
cisco_client_id=YOUR_CLIENT_ID&cisco_client_secret=YOUR_CLIENT_SECRET&cloud_organization_name=
ORAGANIZATION_NAME&cloud_data_affinity=usa&telemetry_enabled=1
```

**Example Responses:**

```
{"success":"AUTH_SUCCESS"}
```

or

```
{"error":  "INVALID_CREDENTIALS"}
```

# Disable Cloud Connector

**API:**

POST /avc-sd-service/external-api/cloud/disable

**Description:**

Disables the Cloud Connector. See Cloud Connector.

Note

Use this API when you have configured Cloud Connector credentials with the "authorize" API:

POST /avc-sd-service/external-api/cloud/authorize

If you have configured Cloud Connector credentials using the "configure" API...

POST /avc-sd-service/external-api/cloud/configure

...then do not use this API to disable Cloud Connector. Instead, use the following "configureSegment" API, specifying "false" for the transmit and receive options:

POST /avc-sd-service/external-api/cloud/configureSegment?segment=name

See Configure Cloud Connector Credentials, on page 5.

**Example Response:**

```
{ "success": true }
```

# Remove Cloud Connector Credentials

**API:**

POST /avc-sd-service/external-api/cloud/removeCredentials

**Description:**

Clears the credentials that have been entered for Cloud Connector.

**Example Response:**

```
{ “success”: true }
```

# Display Current Cloud Configuration

**API:**

GET /avc-sd-service/external-api/cloud/getCurrentCloudConfig

**Description:**

Displays the current configuration for Cloud Connector.

**Example Response:**

```
{"cisco_client_id":{"key":"cisco_client_id","currentValue":"MY_CLIENT_ID","defaultValue":""},
"cloud_credentials_renew_threshold":{"key":"cloud_credentials_renew_threshold","currentValue":"90"},
"cloud_credentials_renew_time":{"key":"cloud_credentials_renew_time","currentValue":"1561547912794","defaultValue":"0"},
"cloud_data_affinity":{"key":"cloud_data_affinity","currentValue":"usa"},"cloud_enabled":{"key":"cloud_enabled",
"currentValue":"1","defaultValue":"0"},"cloud_organization_name":{"key":"cloud_organization_name","currentValue":"CSCO",
"defaultValue":""},"telemetry_enabled":{"key":"telemetry_enabled","currentValue":"1"}}
```

# Display Cloud Data

**API:**

GET /avc-sd-service/external-api/app-rules?sourceId=PP_Extension

**Description:**

Displays the currently collected cloud data (rules, server names, sockets, and so on).

**Example Response:**

```
[{
    "sourceId": "PP_Extension",
    "rules": [{
        "allSegments": true,
        "rules": [{
                "appName": "slack",
                "serverNames": ["slack-redir.net",
                    "www.slack-redir.net"
                ],
                "L3L4": [{
                    "ipAddresses": ["34.204.245.22"],
                    "ports": [443],
                    "l4Protocol": "TCP"
                }]
            },
            {
                "appName": "facebook",
                "L3L4": [{
                    "ipAddresses": [
                        "31.13.24.0/21",
                        "31.13.64.0/19"
                    ]
                }]
            }
        ]
    }]
}]
```

# Display Cloud Connector Status

**API:**

GET https://*SD-AVC-network-service-address*:8443/avc-sd-service/external-api/cloud/status

GET https://*SD-AVC-network-service-address*:8443/avc-sd-service/external-api/cloud/status?segment=*segmentName*

**Description:**

Display status of the Cloud Connector for the specified segment.

**Body:**

```
{
    "cloudConnector": {
        "isEnabled": true,
        "error": []
    }
}
```

- **isEnabled**: Cloud status for the segement. Values: true, false
- **error**: Detected errors.
  - CONNECTIVITY: SD-AVC cannot reach the cloud server. Connectivity problems may include DNS issues, and so on.
  - CREDENTIALS: Credentials for connecting to the cloud server are invalid. For example, the client secret may have expired.
  - INTERNAL_CLOUD: SD-AVC cannot retrieve the cloud rules. The output provides a reason for the error. The stated reason may be helpful when troubleshooting with Cisco technical assistance.

*Table 10: Properties*

| Property | Description |
|---|---|
| segment=*segmentName* | (Optional)<br>Specifies a segment. Response includes only analytics from this segment.<br>When no segment is specified, the output includes all segments.<br>**Note** If you have configured SD-AVC to store telemetry data for different segments in different locations, then you must include this parameter when using this API. |

**Example Responses:**

Cloud Connector enabled, no errors:

```
{
    "cloudConnector": {
        "isEnabled": true,
        "error": []
    }
}
```

Cloud Connector enabled, INTERNAL_CLOUD error, with reason:

```
{
    "cloudConnector": {
        "isEnabled": true,
        "error": [
        {
            id: "INTERNAL_CLOUD",
            reason: ["MINOR_PP"]
        }]
    }
}
```

# External Sources

## External Sources Overview

External sources provide additional application information that SD-AVC uses for classifying network traffic. They are managed by Cloud Connector. To use external sources, ensure that Cloud Connector is enabled. See Cloud Connector.

## Enable/Disable External Sources

**API**

POST /avc-sd-service/external-api/external-sources/*externalSourceName*

**Description**

Enables or disables receiving data from an external source.

*Table 11: Properties*

| Property | Description |
|---|---|
| *externalSourceName* | (Mandatory)<br>Name of the external source.<br>**Note** In this release, the only external source to specify is "ms-office-365". To use this external source, Cloud Connector must be enabled. See Cloud Connector. |
| start<br>stop | In the JSON body of the command, enables or disables the external source. |

**Examples**

```
POST /avc-sd-service/external-api/external-sources/ms-office-365

{
   "action":"start"
}
```

```
POST /avc-sd-service/external-api/external-sources/ms-office-365

{
   "action":"stop"
}
```

## Display Status of External Sources

**API**

GET /avc-sd-service/external-api/eternal-sources

**Description**

Displays external sources and their status: true = enabled, false = disabled.

**Example**

```
GET /avc-sd-service/external-api/external-sources
```

**Example Response**

In this example, the MS Office 365 Web Service, an external source, is enabled.

```
{
  "sources": [{
    "ms-office-365":true
  }]
}
```

# User-defined Applications

## User-defined Applications Overview

Network devices operating with SD-AVC use Cisco NBAR2 and other tools to identify network traffic. The composite of information that NBAR2 uses to identify a network applications is called an "application" (or a "protocol" in the Protocol Packs released periodically by Cisco). User-defined applications may be specified on individual devices by CLI, or network-wide using SD-AVC.

Each application includes:

- **Signature**: Details that identify the network application
- **Attributes**: Assigned characteristics of the application, such as business-relevance, used for visibility and QoS policy.

### SD-AVC User-defined Applications

SD-AVC can provision user-defined applications at the network level, available for all participating devices in the network. In effect, this is similar to adding user-defined applications manually on each device.

### Terminology of Applications and Protocols

The protocols provided in a Protocol Pack and the user-defined applications configured in SD-AVC or in other ways function similarly, but the terminology varies. The table below describes the terminology.

*Table 12: Application Types*

| Application Type | Description |
|---|---|
| Protocol Pack applications | Standard applications provided by Cisco in a Protocol Pack. |
| User-defined applications on individual devices | Defined by CLI on individual devices, sometimes called custom protocols. |
| Network-wide user-defined applications | Defined by SD-AVC REST API.<br>These appear on the **SD-AVC Dashboard** > **Connectors** page. |
| Custom applications defined in Cisco SD-WAN | Defined by Cisco SD-WAN, operating together with Cisco SD-AVC. These are equivalent to SD-AVC user-defined applications. |

# Create User-defined Application Rules

### API

POST /avc-sd-service/external-api/app-rules

POST /avc-sd-service/external-api/app-rules?segment=*segmentName*

### Description

Defines one or more user-defined applications.

*Table 13: Properties*

| Property | Description |
|---|---|
| segment=*segmentName* | Defines the full set of user-defined applications for the specified segment. |

### Body

Body must include the full set of user-defined applications. Executing the API overwrites any currently defined user-defined applications for the specified source (sourceId).

```
{
    "sourceId": string,
    "rules": [{
        "allSegments": boolean,
        "segment": string,
        "rules": [{
            "appName": string,
            "serverNames": [string],
            "L3L4": [{
                "ipAddresses": [string],
                "ports": [integer(s) or range],
                "l4Protocol": string,
                "vrf": string
            }],
            "attributes": {
                "category": string,
                "sub-category": string,
                "application-group": string,
                "business-relevance": string,
                "traffic-class": string,
                "application-set": string
                }
        }]
    }]
}
```

*Table 14: Top-level Properties*

| Property | Description |
|---|---|
| sourceId | (Mandatory)<br>ID of the external source.<br>**Note** In the initial release of the REST API, only one source is supported.<br>**Note** If you have configured custom applications using the SD-AVC Dashboard, then you must use the following sourceID: "sdavc_ui_custom_source" |
| rules | (Mandatory)<br>Contains complete list of the user-defined application rules.<br>**Note** This property contains a sub-property also called rules. |

*Table 15: Sub-properties of rules*

| Property | Description |
|---|---|
| allSegments | (Must include either allSegments or segment.)<br>Set to **true** to apply the user-defined applications to all segments, not only one segment.<br>**Possible values**: true, false (default) |
| segment | (Must include either allSegments or segment.)<br>List of user-defined application rules for a specific SD-AVC segment. |
| rules | (Mandatory)<br>List of segment rules. |

*Table 16: Sub-properties of rules > rules*

| Property | Description |
|---|---|
| appName | (Mandatory)<br>Name of user-defined application, reflecting name of the network application.<br>**Note** Do not use a name that conflicts with an existing application, such as one defined in the Protocol Pack. |

| Property | Description |
|---|---|
| serverNames | (Must include at least one of serverNames, L3L4, and attributes.)<br>List of all server names (FQDNs, SNIs, …) for the network application.<br>**Note** Server names are case-sensitive. |
| L3L4 | (Must include at least one of serverNames, L3L4, and attributes.)<br>List of all IP-based rules.<br>(See sub-properties below.) |
| attributes | (Must include at least one of serverNames, L3L4, and attributes.)<br>Attributes to assign to the application.<br>(See sub-properties below.) |

***Table 17: Sub-properties of rules > rules > L3L4***

| Property | Description |
|---|---|
| IpAddresses | (Mandatory)<br>List of IPs. Can be both normal IP and subnet (using CIDR notation). |
| ports | Port(s) or port range.<br>If this property is defined, you must also include **l4protocol**.<br>**Examples**:<br>`"ports": [23]`<br>`"ports": [23,24]`<br>`"ports": [23, "25-30"]` |
| l4Protcol | Transport layer protocol.<br>If this property is defined, you must also include **ports**.<br>**Possible values**: TCP, UDP, TCP-UDP |
| vrf | VRF name. |

***Table 18: Sub-properties of rules > rules > attributes***

| Property | Description |
|---|---|
| application-set | (Must include at least one of serverNames, L3L4, and attributes.)<br>Attributes to assign to the application. |

| Property | Description |
|---|---|
| application-group | (Defining a partial list of attributes is supported. If **attributes** is included, must include at least one of these properties.) |
| category | |
| sub-category | |
| traffic-class | |
| business-relevance | |

**Response**

Response code 200 indicates success.

In case of failure, the response body provides information about the reason for failure.

## Example 1: Single domain name

This example shows:

- 1 network segment: datacenter01
- 1 user-defined application: myDocs
- 1 server name
- No attributes specified

```
{
  "sourceId": "mySource",
  "rules": [{
    "segment": "datacenter01",
    "rules": [{
      "appName": "myDocs",
      "serverNames": [
        "www.myApp.com"
      ]
    }]
  }]
}
```

## Example 2: Three IP addresses and ports

This example shows:

- 1 network segment: datacenter01
- 1 user-defined application: myDocs
- 3 IP addresses and 3 ports
- No attributes specified

```
{
    "sourceId": "mySource",
    "rules": [{
        "segment": "datacenter01",
        "rules": [{
            "appName": "myDocs",
            "L3L4": [{
                    "ipAddresses": ["2.2.2.2"],
                    "ports": [20],
                    "l4Protocol":"TCP"
                },
                {
                    "ipAddresses": ["3.3.3.3"],
                    "ports": [30],
                    "l4Protocol":"TCP"
                },
                {
                    "ipAddresses": ["4.4.4.4"],
                    "ports": [40],
                    "l4Protocol":"TCP"
                }
            ]
        }]
    }]
}
```

## Example 3: Two user-defined applications in one network segment

This example shows:

- 1 network segment: datacenter01
- 2 user-defined applications: myDocs and myTelepresence
- No attributes specified for the myDocs user-defined application
- business-relevance attribute specified for the myTelepresence user-defined application
- IP address with subnet specified
- Individual ports and a range of ports

```
{
  "sourceId": "mySource",
  "rules": [{
    "segment": "datacenter01",
    "rules": [{
        "appName": "myDocs",
        "serverNames": [
          "www.myApp.com"
        ],
        "L3L4": [{
          "ipAddresses": ["10.1.1.0/24", "2.2.2.2"],
          "ports": [23, 34, "37-42"],
          "l4Protocol": "TCP",
          "vrf": "vrf1"
        }]
      },
      {
        "appName": "myTelepresence",
        "L3L4": [{
          "ipAddresses": ["2.2.2.2"],
          "ports": [35],
          "l4Protocol": "TCP"
        }],
        "attributes": {
```

```
     "business-relevance": "business-relevant"
    }
   }
  ]
 }]
}
```

## Example 4: User-defined applications in two network segments

This example shows:

- 2 network segments: datacenter01, datacenter02
- 3 user-defined applications: myDocs, myTelepresence, myEnterpriseIM
- No attributes specified for: myDocs, myEnterpriseIM
- business-relevance attribute specified for myTelepresence
- IP address with subnet specified
- Individual ports and a range of ports

```
{
 "sourceId": "mySource",
 "rules": [{
   "segment": "datacenter01",
   "rules": [{
     "appName": "myDocs",
     "serverNames": [
      "www.myDocs.com"
     ],
     "L3L4": [{
      "ipAddresses": ["10.1.1.0/24", "2.2.2.2"],
      "ports": [23, 34, "37-42"],
      "l4Protocol": "TCP",
      "vrf": "vrf1"
     }]
    },
    {
     "appName": "myTelepresence",
     "L3L4": [{
      "ipAddresses": ["2.2.2.2"],
      "ports": [35],
      "l4Protocol": "TCP"
     }],
     "attributes": {
      "business-relevance": "business-relevant"
     }
    }
   ]
  },
  {
   "segment": "datacenter02",
   "rules": [{
    "appName": "myEnterpriseIM",
    "serverNames": [
     "www.myEnterpriseIM.com"
    ],
    "L3L4": [{
     "ipAddresses": ["2.2.2.10"],
     "ports": [23],
          "l4Protocol": "TCP"
        }]
      }]
    }
  ]
}
```

## Example 5: Using allSegments and specific network segments

This example shows:

- 2 user-defined applications (myDocs, myTelepresence) for all network segments, using allSegments
- User-defined application (myEnterpriseIM) only for 1 network segment: datacenter02
- 3 user-defined applications: myDocs, myTelepresence, myEnterpriseIM
- No attributes specified for: myDocs, myEnterpriseIM
- business-relevance attribute specified for myTelepresence
- IP address with subnet specified
- Individual ports and a range of ports

```
{
  "sourceId": "mySource",
  "rules": [{
      "allSegments": true,
      "rules": [{
          "appName": "myDocs",
          "serverNames": [
            "www.myApp.com"
          ],
          "L3L4": [{
            "ipAddresses": ["10.1.1.0/24", "2.2.2.2"],
            "ports": [23, 34, "37 - 42"],
            "l4Protocol": "TCP",
            "vrf": "vrf1"
          }]
        },
        {
          "appName": "myTelepresence",
          "L3L4": [{
            "ipAddresses": ["2.2.2.2"],
            "ports": [35],
            "l4Protocol": "TCP"
          }],
          "attributes": {
            "business-relevance": "business-relevant"
          }
        }
      ]
    },
    {
      "segment": "datacenter02",
      "rules": [{
        "appName": "myEnterpriseIM",
        "serverNames": [
          "www.myEnterpriseIM.com"
        ],
        "L3L4": [{
                "ipAddresses": ["2.2.2.10"],
                "ports": [23],
                "l4Protocol": "TCP"
              }]
            }]
        }
    ]
}
```

# Add a User-defined Application Rule

### API

PUT /avc-sd-service/external-api/app-rules

PUT /avc-sd-service/external-api/app-rules?segment=*segmentName*

### Description

Add a user-defined application to an existing set of applications in a particular segment.

### Body

The body must include a single user-defined application. See Create User-defined Application Rules, on page 16 for descriptions of the properties to use for user-defined applications. Executing the API overwrites any currently defined user-defined applications for the specified source (sourceId).

If you use the name (appName field) as an existing user-defined application, this API overwrites the existing application.

**Note** If you have configured custom applications using the SD-AVC Dashboard, then you must use the following sourceID: "sdavc_ui_custom_source"

*Table 19: Properties*

| Property | Description |
|---|---|
| segment=*segmentName* | In the current release, this parameter is optional. If used, it must match the segment specified in the API body.<br>Example:<br>PUT /avc-sd-service/external-api/app-rules?segment=segment1<br>**Note** The segment must be specified in the API body, even if it is added as a parameter in the URI. |

```
{
    "sourceId": string,
    "rules": [{
        "allSegments": boolean,
        "segment": string,
        "rules": [{
            "appName": string,
            "serverNames": [string],
            "L3L4": [{
                "ipAddresses": [string],
                "ports": [integer(s) or range],
                "l4Protocol": string,
                "vrf": string
            }],
            "attributes": {
                "category": string,
                "sub-category": string,
                "application-group": string,
                "business-relevance": string,
                "traffic-class": string,
                "application-set": string
                }
        }]
    }]
}
```

### Response

Response code 200 indicates success.

In case of failure, the response body provides information about the reason for failure.

## Example

This example shows:

- 1 network segment: datacenter01
- 1 user-defined application: segment1_myDocs
- 1 server name
- No attributes specified

```
{
  "sourceId": "mySource",
  "rules": [{
    "segment": "segment1",
    "rules": [{
      "appName": "segment1_myDocs",
      "serverNames": [
        "www.myApp.com"
      ]
    }]
  }]
}
```

# Display User-defined Application Rules

### API

GET /avc-sd-service /external-api/app-rules

GET /avc-sd-service /external-api/app-rules?sourceId=*sourceId*

GET /avc-sd-service /external-api/app-rules?segment=*segmentName*

GET /avc-sd-service /external-api/app-rules?segment=*segmentName*&app=*applicationName*

### Description

Displays the user-defined applications.

*Table 20: Properties*

| Property | Description |
|---|---|
| sourceId=*sourceId* | If no *sourceId* is specified, the response lists the user-defined applications for all sources.<br>If *sourceId* is specified, the response lists the user-defined applications for the specified source. The *sourceId* is user-defined by POST when defining user-defined applications.<br>**Note** In the initial release of the REST API, only one source is supported. |
| segment=*segmentName* | Lists the full set of user-defined applications in the specified segment. |
| app=*applicationName* | (Must also specify the segment)<br>Lists the specified application.<br>Example:<br>GET /avc-sd-service /external-api/app-rules?segment=datacenter01&app=segment1_myDocs |

### Response

The response lists the user-defined applications defined for a single source or all sources. The response body uses the same JSON structure as POST.

# Display User-defined Application Status

### API

GET /avc-sd-service/external-api/app-rules/status

GET /avc-sd-service /external-api/app-rules/status[?sourceId=*sourceId*]

### Description

The SD-AVC network service sends the user-defined applications defined by REST API to the devices in the network. This API displays the activation status of the applications, per device.

If *sourceId* is specified, the output is limited to that source.The *sourceId* is user-defined by POST when defining user-defined applications.

**Note** In the initial release of the REST API, only one source is supported.

### Response

The response lists each network device, arranged by segment. For each device:

- ID/version of application rules currently loaded on the device
- Status: SUCCESS, FAILED, IN-PROGRESS

# Delete User-defined Applications

### API

DELETE /avc-sd-service /external-api/app-rules

DELETE /avc-sd-service /external-api/app-rules?sourceId=*sourceId*

DELETE /avc-sd-service /external-api/app-rules?segment=*segmentName*

DELETE /avc-sd-service /external-api/app-rules?segment=*segmentName*&app=*applicationName*

### Description

Deletes a set of user-defined applications.

If no source, segment, or application are specified, the API deletes the full set of user-defined applications.

*Table 21: Properties*

| Property | Description |
|---|---|
| sourceId=*sourceId* | **DEPRECATED in SD-AVC 4.0.0: This option is not recommended.**<br>Deletes the full set of user-defined applications for the specified source. The *sourceId* is user-defined by POST when defining user-defined applications.<br>**Note** In the initial release of the REST API, only one source is supported. |
| segment=*segmentName* | Deletes the full set of user-defined applications in the specified segment. |
| app=*applicationName* | (Must also specify the segment)<br>Deletes the specified application.<br>Example:<br>DELETE /avc-sd-service /external-api/app-rules?segment=datacenter01&app=segment1_myDocs |

### Response

Response code 200 indicates success.

# Generic Applications

## Generic Applications Overview

"Generic" network traffic is not attributed to a specific network application. This portion of network traffic reduces the classification index, which is shown in the SD-AVC Dashboard.

## Display Generic Application Traffic Types

**API**

GET /avc-sd-service/external-api/apps/generics

**Description**

Displays the list of traffic types that contribute to generic traffic. The response is preconfigured - it does not depend on current traffic.

**Response**

```
["statistical-conf-audio","rtp-audio","spdy","statistical-p2p","rtp-video","http","statistical-conf-video",
"quic","statistical-download","ssl","unknown","rtp"]
```

# REST API Notes and Limitations

See SD-AVC Notes and Limitations.