H3C VCF 控制器

REST API

杭州华三通信技术有限公司
http://www.h3c.com.cn

资料版本：5W100-20140611
产品版本：H3C VCF Controller E1124

前　言

H3C VCF 控制器 REST API 介绍了 VCF 控制器支持的 REST API 及其使用方法。

前言部分包含如下内容：

- 读者对象
- 本书约定
- 产品配套资料
- 资料获取方式
- 技术支持
- 资料意见反馈

读者对象

本手册主要适用于如下工程师：

- 网络规划人员
- 现场技术支持与维护人员
- 负责网络配置和维护的网络管理员

本书约定

1. 格式约定

格　式	意　义
粗体	REST API请求方法中的关键字（保持不变、必须照输的部分）采用**加粗**字体表示。
斜体	REST API请求方法中的参数（必须由实际值进行替代的部分）采用*斜体*表示。

1. 各类标志

本书还采用各种醒目标志来表示在操作过程中应该特别注意的地方，这些标志的意义如下：

警告	该标志后的注释需给予格外关注，不当的操作可能会对人身造成伤害。
注意	提醒操作中应注意的事项，不当的操作可能会导致数据丢失或者设备损坏。
提示	为确保设备配置成功或者正常工作而需要特别关注的操作或信息。
说明	对操作内容的描述进行必要的补充和说明。
窍门	配置、操作、或使用设备的技巧、小窍门。

产品配套资料

H3C VCF 控制器用户手册的配套资料包括如下部分：

大类	资料名称	内容介绍
安装指导	H3C VCF控制器安装指导	介绍H3C VCF控制器的安装和卸载方法
License注册和激活指南	H3C VCF控制器License注册和激活指南	帮助您了解H3C VCF控制器的注册方法

资料获取方式

您可以通过H3C网站（www.h3c.com.cn）获取最新的产品资料：

H3C 网站与产品资料相关的主要栏目介绍如下：

- [服务支持/文档中心]：可以获取软件升级类、配置类或维护类等产品资料。
- [产品技术]：可以获取产品介绍和技术介绍的文档，包括产品相关介绍、技术介绍、技术白皮书等。
- [解决方案]：可以获取解决方案类资料。
- [服务支持/软件下载]：可以获取与软件版本配套的资料。

技术支持

用户支持邮箱：service@h3c.com

技术支持热线电话：400-810-0504（手机、固话均可拨打）

网址：http://www.h3c.com.cn

资料意见反馈

如果您在使用过程中发现产品资料的任何问题，可以通过以下方式反馈：

E-mail：info@h3c.com

感谢您的反馈，让我们做得更好！

目 录

1 REST API简介

1.1 概述

REST API 是一种基于 HTTP 协议和 REST 架构策略的一种简单 web service。H3C VCF（Virtual Converged Framework，虚拟应用融合架构）控制器的 REST APIs 分成三个不同部分：核心(**/sdn/v2.0**)，OpenFlow (**/sdn/v2.0/of**)和网络服务(**/sdn/v2.0/net**)。每部分都有自己的 JSON 数据格式，并且都可以通过 URL（Uniform Resource Locator，统一资源定位符）格式直接访问。

- 核心 API 提供控制器的基础功能，如配置管理、状态监测、集群管理、告警管理、审计日志、帮助日志等。
- OpenFlow API 提供控制器的 OpenFlow 功能，包括只读（如端口统计）和修改（如流表下发）操作。OpenFlow API 可以同时应用在支持 OpenFlow 1.0 和 OpenFlow 1.3 协议版本的设备上，但是某些 API（如 Meter 表 API、组表 API）只能应用在支持 OpenFlow 1.3 协议版本的设备上。
- 网络服务 API 提供基本的网络信息，如拓扑信息和网络诊断信息。

除非另有说明，否则所有控制器都已通过认证。

说明

RSdoc 界面是 VCF 控制器提供的 API 操作界面，在 RSdoc 界面上暂时不支持删除（**DELETE**）请求。如有需求建议通过 cURL 命令进行操作。

1.2 请求方法

从客户端向服务器端发送请求消息时，须指明请求方法，本文中涉及的请求方法如下：

- **GET**：一般用于告诉服务器需要获取哪些内容；
- **POST**：一般用于创建服务器的资源信息；
- **PUT**：一般用于更新服务器的资源信息；
- **DELETE**：删除 URL 指定的资源信息。

说明

对于 **GET** 方法，请求的数据会附在 URL 之后，即把数据放置在 HTTP 协议头中，以“？”符号分割 URL 和传输的数据，多个参数用“&”符号连接（eg: **/sdn/v2.0/auditlog?start=2013-09-19T18%3A06%3A54.086Z&end=2013-09-19T18%3A06%3A54.108Z**），如果数据是英文字母或数据，原样发送；如果是空格，转换为%2A；如果是中文或其它字符，则直接把字符串用 BASE64 加密，得出如%3A，其中%XX 的 XX 为该符号以 16 进制表示的 ASCII。

1.3 返回码

本文中涉及 API 操作的返回码及其含义如下：

- 2xx：成功收到、理解和接受动作。
  - OK (200)：是指客户端的请求已经成功收到、解析、接受；
  - Created (201)：表示请求成功并且服务器创建了新的资源；
  - Accepted (202)：表示服务器已接受请求，但尚未处理；
  - No Content (204)：表示服务器已经接受请求并且没必要返回实体数据；
  - Partial Content (206)：表示服务器成功处理了部分 GET 请求；
- 4xx：表示客户端错误。
  - BadRequest (400)：表示因错误的语法导致服务器无法理解请求信息；
  - Unauthorized (401)：表示请求授权失败；
  - Forbidden (403)表示请求不允许；
  - Not Found (404)：表示服务器找不到任何匹配 Request-URL 的资源；
  - BadMethod (405)：表示用户定义的请求方法错误；
  - DuplicateData (409)：服务器在完成请求时发生冲突；
  - PageSizeExceeded（413）：服务器无法处理请求，因为请求实体过大，超出服务器处理能力。
- 5xx：服务器端错误。
  - Internal Server Error (500)：表示服务器遭遇异常阻止了当前请求的执行；
  - ServiceUnavailable (503)：表示因临时文件超载导致服务器不能处理当前请求。

2 /sdn/v2.0

2.1 用户认证/Auth

2.1.1 登录

【方法】

获取认证 Token：

POST /sdn/v2.0/auth

【请求举例】

```
{"login":
{"user":"sdn","password":"skyline","domain":"sdn"}
}
```

【响应举例】

```
{
  "record": {
    "token": "ab45bb0092e5416ebfb202c936655fdf",
    "expiration": 1399426067000,
    "expirationDate": "2014-05-07 09-27-47 +0800",
    "userId": "51208f0c56e04dcdbd64d47cc9304902",
    "userName": "sdn",
    "domainId": "8bb3ff18358e402ead36f14a9d2f349d",
    "domainName": "sdn"
  }
}
```

【返回码】

- 正确：OK (200)
- 错误：BadRequest (400)， Unauthorized (401)， Forbidden (403)， BadMethod (405)， ServiceUnavailable (503)

2.1.2 退出

【方法】

DELETE /sdn/v2.0/auth

【返回码】

- 正确：OK (200)
- 错误：BadRequest (400)， Forbidden (403)，BadMethod (405)， ServiceUnavailable (503)

2.2 支持报告/support

【方法】

获取完整的支持报告，并显示所有字段：

GET /sdn/v2.0/support

获取指定 ID 的支持报告，并显示所有字段：

GET /sdn/v2.0/support?id="*alert* "

获取全部 ID 的支持报告，但只显示“title”和“content”字段：

GET /sdn/v2.0/support?fields="*title*,*content*"

获取 ID 为“alert”的支持报告，但只显示“title”和“content”字段：

GET /sdn/v2.0/support?id="*alert*"&fields="*title*,*content*"

【参数】

id：可选，表示基础功能的标识 ID（如 alert、application-Management、audit_log 等）。

fields：可选，表示支持报告所包含的字段，取值可以是“title”、“content”或“id”字段，可任选其一也可两两组合或三者同时选定。

【响应举例】

如下响应示例是包含所有 ID 和字段的支持报告。

```
{
  "support_report": [
    {
      "title": "Alert Framework",
      "id": "alert",
      "content": [
        "Alert-Topics: licensing, of_controller, of_controller_link,
of_controller_pathdiag, teaming",
        "Alert-Count: 4",
        "Data Retention Age Out: 14 days",
        "Data Trim Interval: 24 hours",
        "Data Trim Enabled: true",
        "Last trim conducted at: Sat Nov 09 08:02:19 CST 2013"
      ]
    },
    {
      "title": "Alert Topic Listener",
      "id": "alert_listener",
      "content": [
        "No registered alert topic listeners"
      ]
    },
    {
      "title": "Application Manager",
      "id": "application-Management",
      "content": [
        "Installed Applications: 7",
        "Path Diagnostics, Version: 2.0.0, State: ACTIVE",
        "xinlisha, Version: 1.0.0.SNAPSHOT, State: ACTIVE",
        "Link Manager, Version: 2.0.0, State: ACTIVE",
        "Path Daemon, Version: 2.0.0, State: ACTIVE",
        "Topology Manager, Version: 2.0.0, State: ACTIVE",
        "Node Manager, Version: 2.0.0, State: ACTIVE",
        "Topology Viewer, Version: 2.0.0, State: ACTIVE"
      ]
    },
    {
      "title": "Audit Log Framework",
      "id": "audit_log",
      "content": [
        "Audit Log Count: 2",
        "Data Retention Age Out: 365 days",
        "Data Trim Interval: 24 hours",
        "Data Trim Enabled: true",
        "Last trim conducted at: Sat Nov 09 08:02:19 CST 2013"
      ]
    },
    {
      "title": "Server Environment",
      "id": "env",
      "content": [
        "OS architecture: amd64",
        "OS Name: Linux",
        "OS Version: 3.2.0-29-generic",
        "Java Vendor: Oracle Corporation",
        "Java Version: 23.7-b01",
        "Java Name: OpenJDK 64-Bit Server VM",
        "Available processors (cores): 1",
        "Max Heap: 4151836672  [3959Mb]",
        "Heap: 519634944  [495Mb]",
        "Heap used: 173583152  [165Mb]",
        "Start Date: Sat Nov 09 08:01:26 CST 2013",
        "UpTime: 10 Hours, 29 Minutes",
        "H3C VCF Controller: B1122"
      ]
    },
    {
      "title": "Licensing",
      "id": "licensing",
      "content": [
        "Number of licenses Found: None"
      ]
    },
    {
      "title": "OpenFlow Controller",
      "id": "of-ctrl",
      "content": [
        "bind interfaces: All Available",
        "listen port: 6633",
        "tls port: 6634",
        "udp port: 6635",
        "key store: ",
        "trust store: ",
        "suppress set config: false",
                "suppress set flow miss: false",
                "workers: 16",
                "confirm flow-mods: true",
                "max idle ms: 5000",
                "max echo ms: 5000",
                "max echo attempts: 5",
                "strict message parsing: false",
                "Sequenced Packet Listeners",
                "Advisers:",
                "com.h3c.sdn.ctl.diag.impl.PathDiagnosticManager$PDTAdvisor",
                "com.h3c.sdn.ctl.linkdisco.impl.LinkManager$PacketListener",
                "com.h3c.sdn.ctl.nodemgr.impl.NodeManager$PacketListener",
                "Directors:",
                "com.h3c.sdn.ctl.diag.impl.PathDiagnosticManager$PDTDirector",
                "com.h3c.sdn.ctl.path.impl.PathDaemon$DirectorCallback",
                "Observers: none registered"
              ]
            }
          ]
        }
```

【返回码】

- 正确：OK (200)
- 错误：BadRequest (400)， Unauthorized (401)， Forbidden (403)，BadMethod (405)，ServiceUnavailable (503)

2.3 配置/configs

2.3.1 获取所有组件的配置信息

【方法】

GET /sdn/v2.0/configs

【响应举例】

```
{
  "configs": [
    {
      "com.h3c.sdn.adm.alert.impl.AlertManager": {
        "trim.alert.age": {
          "val": "14",
          "def_val": "14",
          "desc": "Days an alert remains in storage (1 - 31)"
        },
        "trim.enabled": {
          "val": "true",
          "def_val": "true",
          "desc": "Allow trim operation (true/false)"
        },
        "trim.frequency": {
          "val": "24",
          "def_val": "24",
          "desc": "Frequency in hours of trim operations (8 - 168)"
        }
      }
    },
    {
      "com.h3c.sdn.adm.auditlog.impl.AuditLogManager": {
        "trim.auditlog.age": {
          "val": "365",
          "def_val": "365",
          "desc": "Days an audit log remains in storage (31 - 1870)"
        },
        "trim.enabled": {
          "val": "true",
          "def_val": "true",
          "desc": "Allow trim operation (true/false)"
        },
        "trim.frequency": {
          "val": "24",
          "def_val": "24",
          "desc": "Frequency in hours of trim operations (8 - 168)"
        }
      }
    },
    {
      "com.h3c.sdn.adm.auth.impl.AuthenticationManager": {
        "AdminToken": {
          "val": "ENC()",
          "def_val": "ENC()",
          "desc": "Keystone admin token"
        },
        "CachedTokenIdle": {
          "val": "300",
          "def_val": "300",
          "desc": "Time to live for a token from its last accessed or modified date (seconds).
Min: 1. Max: CachedTokenTTL"
        },
        "CachedTokenTTL": {
          "val": "86400",
          "def_val": "86400",
          "desc": "Time to live for a token from its creation date (seconds). Min: 1. Max:
Keystone's token expiration"
        },
        "ConnPoolEvictPeriod": {
          "val": "600000",
          "def_val": "600000",
          "desc": "Keystone idle connection clean-up cycle in milliseconds. Min: 1000.  Max:
1000*CachedTokenTTL"
        },
        "ConnPoolMaxActive": {
          "val": "4",
          "def_val": "4",
          "desc": "Keystone max active connections. Min: 1"
        },
        "ConnPoolMaxIdle": {
          "val": "1",
          "def_val": "1",
          "desc": "Keystone max idle connections. Min: 1"
```

```
},
"ConnPoolMinIdleTime": {
  "val": "300000",
  "def_val": "300000",
  "desc": "Keystone min idle connection time in milliseconds. Min: 1000"
},
"ConnSSLClientAuth": {
  "val": "false",
  "def_val": "false",
  "desc": "Keystone 2-way SSL: True or False"
},
"ConnTimeout": {
  "val": "2000",
  "def_val": "2000",
  "desc": "Keystone connection timeout in milliseconds. Min: 0 (never timeout)"
},
"Keystore": {
  "val": "",
  "def_val": "",
  "desc": "Keystone keystore location"
},
"KeystorePass": {
  "val": "ENC()",
  "def_val": "ENC()",
  "desc": "Keystone keystore passworde"
},
"MaxCachedTokens": {
  "val": "10000",
  "def_val": "10000",
  "desc": "Maximum number of cached tokens. Min: 0"
},
"ServerPort": {
  "val": "35357",
  "def_val": "35357",
  "desc": "Keystone server port"
},
"ServerVIP": {
  "val": "localhost",
  "def_val": "localhost",
  "desc": "Keystone server virtual IP"
},
"ServiceRole": {
  "val": "sdn-admin",
  "def_val": "sdn-admin",
  "desc": "Role for shared secret"
},
"ServiceTenant": {
  "val": "sdn",
  "def_val": "sdn",
  "desc": "Tenant (project) for shared secret"
},
"ServiceToken": {
  "val": "ENC()",
  "def_val": "ENC()",
  "desc": "Shared secret for internal requests"
},
    "ServiceTokenTimeout": {
      "val": "0",
      "def_val": "0",
      "desc": "Timeout for shared secret, 0 for never. Min: 0"
    },
    "ServiceUser": {
      "val": "sdn-service-client",
      "def_val": "sdn-service-client",
      "desc": "User for shared secret"
    },
    "Truststore": {
      "val": "",
      "def_val": "",
      "desc": "Keystone truststore location"
    },
    "TruststorePass": {
      "val": "ENC()",
      "def_val": "ENC()",
      "desc": "Keystone truststore password"
    }
  }
},
{
  "com.h3c.sdn.adm.ctlAddrManager.impl.CtlAddrManagerManager": {
    "trim.enabled": {
      "val": "true",
      "def_val": "true",
      "desc": "Allow trim operation (true/false)"
    },
    "trim.frequency": {
      "val": "24",
      "def_val": "24",
      "desc": "Frequency in hours of trim operations (8 - 168)"
    }
  }
},
{
  "com.h3c.sdn.adm.hostManager.impl.HostManagerManager": {
    "trim.enabled": {
      "val": "true",
      "def_val": "true",
      "desc": "Allow trim operation (true/false)"
    },
    "trim.frequency": {
      "val": "24",
      "def_val": "24",
      "desc": "Frequency in hours of trim operations (8 - 168)"
    }
  }
},
{
  "com.h3c.sdn.adm.log.impl.LogManager": {
    "max.display.rows": {
      "val": "100",
      "def_val": "100",
      "desc": "Maximum rows of log data to present (100 - 1500)"
```

```
        }
      }
    },
    {
      "com.h3c.sdn.adm.metric.impl.MetricManagerComponent": {
        "metric.interval": {
          "val": "1",
          "def_val": "1",
          "desc": "Interval in minutes [1, 5, 15] at which values for subsequently-created
metrics will be persisted unless specifically overridden for a metric"
        },
        "trim.metric.age": {
          "val": "14",
          "def_val": "14",
          "desc": "Days for which data will be saved, ranging from 1 to 31 days"
        },
        "trim.metric.hour": {
          "val": "2",
          "def_val": "2",
          "desc": "Hour of 24-hour clock (0 - 23) when old metric data is removed from
persistent storage"
        }
      }
    },
    {
      "com.h3c.sdn.adm.role.impl.RoleAssertManager": {
        "role.max.retries": {
          "val": "5",
          "def_val": "5",
          "desc": "Maximum number of retries to attempt for sending a role message to device"
        },
        "role.msg.timeout": {
          "val": "5000",
          "def_val": "5000",
          "desc": "Time in miliseconds to wait for a response to role message sent to device"
        }
      }
    },
    {
      "com.h3c.sdn.adm.system.impl.SystemWatchdogManager": {
        "watchdog.frequency": {
          "val": "60",
          "def_val": "60",
          "desc": "Interval (seconds) for heart beat between team members"
        }
      }
    },
    {
      "com.h3c.sdn.api.impl.AlertPostManager": {
        "clientauth": {
          "val": "false",
          "def_val": "false",
          "desc": "2-way SSL (true/false)"
        },
        "connTimeout": {
          "val": "15000",
          "def_val": "0",
          "desc": "Timeout (ms) until a connection is established. Zero is an infinite
timeout."
        },
        "keystore": {
          "val": "/opt/sdn/admin/keystore",
          "def_val": "/opt/sdn/admin/keystore",
          "desc": "Controller keystore location"
        },
        "keystore.password": {
          "val": "ENC()",
          "def_val": "ENC()",
          "desc": "Controller keystore password"
        },
        "maxperroute": {
          "val": "2",
          "def_val": "2",
          "desc": "Max connections per URL. Min: 1"
        },
        "maxtotal": {
          "val": "20",
          "def_val": "20",
          "desc": "Max total connections. Min: 1"
        },
        "port": {
          "val": "8443",
          "def_val": "8443",
          "desc": "Communication port"
        },
        "selfsigned": {
          "val": "true",
          "def_val": "false",
          "desc": "Trust self-signed certificates (true/false)"
        },
        "socketTimeout": {
          "val": "15000",
          "def_val": "120000",
          "desc": "Socket timeout (ms). Max inactivity between two consecutive data packets.
Zero is infinite timeout"
        },
        "ssl": {
          "val": "true",
          "def_val": "true",
          "desc": "SSL communication (true/false)"
        },
        "truststore": {
          "val": "/opt/sdn/admin/truststore",
          "def_val": "/opt/sdn/admin/truststore",
          "desc": "Controller truststore location"
        },
        "truststore.password": {
          "val": "ENC()",
          "def_val": "ENC()",
          "desc": "Controller truststore password"
        },
        "ttl": {
          "val": "5000",
          "def_val": "5000",
          "desc": "Connection Time-To-Live in milliseconds. Min: 1000"
        }
      }
    },
    {
      "com.h3c.sdn.ctl.diag.impl.PathDiagnosticManager": {
        "hard.timeout": {
          "val": "300",
          "def_val": "300",
          "desc": "Flow hard time out (in seconds). Min: 0"
        },
        "idle.timeout": {
          "val": "300",
          "def_val": "300",
          "desc": "Flow idle time out (in seconds). Min: 0"
        }
      }
    },
    {
      "com.h3c.sdn.ctl.linkdisco.impl.LinkManager": {
        "learn.multihop.links": {
          "val": "false",
          "def_val": "false",
          "desc": "Flag indicating whether BDDP packets should be sent to learn 'Multihop
Links'."
        },
        "lldp.frequency": {
          "val": "1",
          "def_val": "1",
          "desc": "Link Rediscovery Frequency (minutes).'0' disables link rediscovery. Min:
0"
        },
        "timeout.links": {
          "val": "true",
          "def_val": "true",
          "desc": "Flag indicating whether discovered links should be timed out. If disabled,
links will be removed only on a switch port down event or device disconnected event."
        },
        "transparent": {
          "val": "false",
          "def_val": "false",
          "desc": " When BDDP packets have sented to learn 'Multihop Links', Flag indicating
whether change the 'Multihop Links' to 'Indirect Links'."
        }
      }
    },
    {
      "com.h3c.sdn.ctl.nodemgr.impl.NodeManager": {
        "arp.age": {
          "val": "20",
          "def_val": "20",
          "desc": "ARP aging time out (in minutes).Min:0,Max:65535"
        },
        "bus.frequency": {
          "val": "1000000",
          "def_val": "1000000",
          "desc": "The frequency for posting message on the bus (in micro seconds). 
Min:100,Max:10000000"
        }
      }
    },
    {
      "com.h3c.sdn.ctl.of.impl.ControllerManager": {
        "addresses": {
          "val": "",
          "def_val": "",
          "desc": "A comma separated list of interface addresses to listen on"
        },
        "confirm.flowmod": {
          "val": "true",
          "def_val": "true",
          "desc": "Flag indicating whether flow-mods should be followed by a barrier request 
for positive acknowledgement from the data path"
        },
        "default.tableid": {
          "val": "0",
          "def_val": "0",
          "desc": "This value is applicable only if Suppress Pipeline Definition flag is set 
to truers"
        },
        "idle.check": {
          "val": "500",
          "def_val": "500",
          "desc": " Number of milliseconds between checks for idle connections"
        },
        "idle.echo": {
          "val": "5000",
          "def_val": "5000",
          "desc": "Number of milliseconds between sending echo requests on idle connections"
        },
        "idle.echo.attempts": {
          "val": "5",
          "def_val": "5",
          "desc": "Number of times echo requests will be sent on idle connections before 
disconnects"
        },
        "idle.max": {
          "val": "5000",
          "def_val": "5000",
          "desc": "Number of milliseconds before connection is considered idle"
        },
        "keystore": {
          "val": "",
          "def_val": "",
          "desc": "Keystore file name"
        },
        "keystore.password": {
          "val": "ENC()",
          "def_val": "ENC()",
          "desc": "Keystore password"
        },
        "msg.parse.strict": {
          "val": "false",
          "def_val": "false",
          "desc": "Flag indicating whether the message library should employ strict parsing
of OpenFlow messages"
        },
        "port.nonsecure": {
          "val": "6633",
          "def_val": "6633",
          "desc": "OpenFlow Controller non-secure listen port (0 to disable)"
        },
        "port.secure": {
          "val": "6634",
          "def_val": "6634",
          "desc": " OpenFlow Controller secure (TLS) listen port (0 to disable)"
        },
        "receive.buffer": {
          "val": "1048576",
          "def_val": "1048576",
          "desc": "TCP or TLS receive buffer size"
        },
        "suppress.flowmiss": {
          "val": "false",
          "def_val": "false",
          "desc": "Flag indicating whether the default behavior of installing a default
FlowMiss rule to tables on a newly connected data path should be suppressed"
        },
        "suppress.pipeline.definition": {
          "val": "false",
          "def_val": "false",
          "desc": "Flag indicating whether the default behavior of using Pipeline Definition
while installing OpenFlow 1.3 compliant flows should be suppressed"
        },
        "suppress.setconfig": {
          "val": "false",
          "def_val": "false",
          "desc": "Flag indicating whether the SetConfig behavior of the controller should
be suppressed"
        },
        "truststore": {
          "val": "",
          "def_val": "",
          "desc": "Truststore file name"
        },
        "truststore.password": {
          "val": "ENC()",
          "def_val": "ENC()",
          "desc": "Truststore password"
        },
        "workers": {
          "val": "16",
          "def_val": "16",
          "desc": "Number of I/O loop workers"
        }
      }
```

```
    },
    {
      "com.h3c.sdn.ctl.of.impl.TraceManager": {
        "record.duration": {
          "val": "10",
          "def_val": "10",
          "desc": "Duration in seconds for active trace recording (1 - 60)"
        }
      }
    },
    {
      "com.h3c.sdn.ctl.path.impl.PathDaemon": {
        "ecmp_supporting": {
          "val": "false",
          "def_val": "false",
          "desc": "Sets ECMP supporting"
        },
        "forward_mode": {
          "val": "0",
          "def_val": "0",
          "desc": "forward mode settings (0: L2_forward 1: L2_dmaconly_forward 2:
L3_forward)"
        },
        "hard.timeout": {
          "val": "0",
          "def_val": "0",
          "desc": "Flow hard time out (in seconds). Min: 0,Max: 65535"
        },
        "idle.timeout": {
          "val": "300",
          "def_val": "300",
          "desc": "Flow idle time out (in seconds). Min: 0,Max: 65535"
        }
      }
    },
    {
      "com.h3c.sdn.misc.AdminRestComponent": {
        "clientauth": {
          "val": "false",
          "def_val": "false",
          "desc": "2-way SSL (true/false)"
        },
        "connTimeout": {
          "val": "10000",
          "def_val": "0",
          "desc": "Timeout (ms) until a connection is established. Zero is an infinite
timeout."
        },
        "keystore": {
          "val": "/opt/sdn/admin/keystore",
          "def_val": "/opt/sdn/admin/keystore",
          "desc": "Controller keystore location"
        },
        "keystore.password": {
          "val": "ENC()",
          "def_val": "ENC()",
          "desc": "Controller keystore password"
        },
        "maxperroute": {
          "val": "2",
          "def_val": "2",
          "desc": "Max connections per URL. Min: 1"
        },
        "maxtotal": {
          "val": "20",
          "def_val": "20",
          "desc": "Max total connections. Min: 1"
        },
        "port": {
          "val": "8081",
          "def_val": "8443",
          "desc": "Communication port"
        },
        "selfsigned": {
          "val": "true",
          "def_val": "false",
          "desc": "Trust self-signed certificates (true/false)"
        },
        "socketTimeout": {
          "val": "10000",
          "def_val": "120000",
          "desc": "Socket timeout (ms). Max inactivity between two consecutive data packets.
Zero is infinite timeout"
        },
        "ssl": {
          "val": "true",
          "def_val": "true",
          "desc": "SSL communication (true/false)"
        },
        "truststore": {
          "val": "/opt/sdn/admin/truststore",
          "def_val": "/opt/sdn/admin/truststore",
          "desc": "Controller truststore location"
        },
        "truststore.password": {
          "val": "ENC()",
          "def_val": "ENC()",
          "desc": "Controller truststore password"
        },
        "ttl": {
          "val": "5000",
          "def_val": "5000",
          "desc": "Connection Time-To-Live in milliseconds. Min: 1000"
        }
      }
    },
    {
      "com.h3c.sdn.misc.ServiceRestComponent": {
        "clientauth": {
          "val": "false",
          "def_val": "false",
          "desc": "2-way SSL (true/false)"
        },
        "connTimeout": {
          "val": "20000",
          "def_val": "0",
          "desc": "Timeout (ms) until a connection is established. Zero is an infinite
timeout."
        },
        "keystore": {
          "val": "/opt/sdn/admin/keystore",
          "def_val": "/opt/sdn/admin/keystore",
          "desc": "Controller keystore location"
        },
        "keystore.password": {
          "val": "ENC()",
          "def_val": "ENC()",
          "desc": "Controller keystore password"
        },
        "maxperroute": {
          "val": "2",
          "def_val": "2",
          "desc": "Max connections per URL. Min: 1"
        },
        "maxtotal": {
          "val": "20",
          "def_val": "20",
          "desc": "Max total connections. Min: 1"
        },
        "port": {
          "val": "8443",
          "def_val": "8443",
          "desc": "Communication port"
        },
        "selfsigned": {
          "val": "true",
          "def_val": "false",
          "desc": "Trust self-signed certificates (true/false)"
        },
        "socketTimeout": {
          "val": "120000",
          "def_val": "120000",
          "desc": "Socket timeout (ms). Max inactivity between two consecutive data packets.
Zero is infinite timeout"
        },
        "ssl": {
          "val": "true",
          "def_val": "true",
          "desc": "SSL communication (true/false)"
        },
        "truststore": {
          "val": "/opt/sdn/admin/truststore",
          "def_val": "/opt/sdn/admin/truststore",
          "desc": "Controller truststore location"
        },
        "truststore.password": {
          "val": "ENC()",
          "def_val": "ENC()",
          "desc": "Controller truststore password"
        },
        "ttl": {
          "val": "5000",
          "def_val": "5000",
          "desc": "Connection Time-To-Live in milliseconds. Min: 1000"
        }
      }
    },
    {
      "com.h3c.sdn.rs.RestPerfProvider": {
        "perf.profile": {
          "val": "0",
          "def_val": "0",
          "desc": "REST instrumentation profile (0=NONE,1=PRODUCTION,3=DEV)"
        }
      }
    }
  ]
}
```

【返回码】

- 正确：OK (200)
- 错误：BadRequest (400)， Unauthorized (401)， Forbidden (403)， BadMethod (405)，ServiceUnavailable (503)

2.3.2 获取指定组件的配置信息

【方法】

GET sdn/v2.0/configs/ {*component*}

【参数】

component：必选，表示组件名。

【响应举例】

```
{
  "config": {
    "com.h3c.sdn.adm.alert.impl.AlertManager": {
      "trim.alert.age": {
        "val": "14",
        "def_val": "14",
        "desc": "Days an alert remains in storage (1 - 31)"
      },
      "trim.enabled": {
        "val": "true",
        "def_val": "true",
        "desc": "Allow trim operation (true/false)"
      },
      "trim.frequency": {
        "val": "24",
        "def_val": "24",
        "desc": " Frequency in hours of trim operations (8 - 168)"
      }
    }
    }
  }
```

【返回码】

- 正确：OK (200)
- 错误：BadRequest (400)， Unauthorized (401)， Forbidden (403)， BadMethod (405)，ServiceUnavailable (503)

2.3.3 更新指定组件的（部分）配置信息

【方法】

PUT /sdn/v2.0/configs/{*component*}

【参数】

component：必选，表示组件名。

【请求举例】

```
{
   "config":{
        "trim.frequency":"12"
            }
}
```

【响应举例】

```
{
     "config": {
          "com.h3c.sdn.adm.alert.impl.AlertManager": {
               "trim.alert.age":
                       {
                    "def_val": "14",
                    "desc": "Days an alert remains in storage (1 - 31)",
                    "val": "14"
               },
               "trim.enabled": {
                    "def_val": "true",
                    "desc": "Allow trim operation (true/false)",
                    "val": "true"
               },
               "trim.frequency": {
                    "def_val": "24",
                    "desc": "Frequency in hours of trim operations (8 - 168)",
                    "val": "12"
               }
          }
     }
}
```

【返回码】

- 正确：OK (200)
- 错误：BadRequest (400)， Unauthorized (401)， Forbidden (403)， BadMethod (405)，ServiceUnavailable (503)

2.3.4 恢复指定组件到默认配置

【方法】

可以通过删除指定组件的配置项信息将其恢复到默认值。

DELETE /sdn/v2.0/configs/{*component*}

如果没有通过请求 JSON 码指定删除的配置项，该组件的所有配置项都将恢复到默认值。

【参数】

component：必选，表示组件名。

【请求举例】

```
{
   "config":["trim.frequency"]
}
```

【响应举例】

```
    {
         "config": {
              "com.h3c.sdn.adm.alert.impl.AlertManager": {
                   "trim.alert.age":
                            {
                        "def_val": "14",
                        "desc": "Days an alert remains in storage (1 - 31)",
                        "val": "14"
                   },
                   "trim.enabled": {

                        "def_val": "true",

                        "desc": "Allow trim operation (true/false)",
                        "val": "true"
                   },
                 "trim.frequency": {
                        "def_val": "24",
                        "desc": "Frequency in hours of trim operations (8 - 168)",
                        "val": "24"
                   }
              }
         }
    }
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Not Found (404)，ServiceUnavailable (503)

2.4 应用程序/apps

2.4.1 获取所有应用程序信息

【方法】

GET /sdn/v2.0/apps

【响应举例】

```
{
  "apps": [
    {
      "deployed": "2013-11-05T01:50:49.454Z",
      "desc": "Path Diagnostic Utility",
      "name": "Path Diagnostics",
      "state": "ACTIVE",
      "uid": "com.h3c.sdn.ctl.diag",
      "vendor": "H3C",
      "version": "2.0.0"
    },
    {
      "deployed": "2013-11-05T01:50:53.495Z",
      "desc": "Link Management",
      "name": "Link Manager",
      "state": "ACTIVE",
      "uid": "com.h3c.sdn.ctl.linkdisco",
      "vendor": "H3C",
      "version": "2.0.0"
    },
    {
      "deployed": "2013-11-05T01:50:56.736Z",
      "desc": "Path analysis",
      "name": "Path Daemon",
      "state": "ACTIVE",
      "uid": "com.h3c.sdn.ctl.path",
      "vendor": "H3C",
      "version": "2.0.0"
    },
    {
      "deployed": "2013-11-05T01:50:54.758Z",
      "desc": "Topology Management",
      "name": "Topology Manager",
      "state": "ACTIVE",
      "uid": "com.h3c.sdn.ctl.topo",
      "vendor": "H3C",
      "version": "2.0.0"
    },
    {
      "deployed": "2013-11-05T01:50:55.710Z",
      "desc": "Node Management",
      "name": "Node Manager",
      "state": "ACTIVE",
      "uid": "com.h3c.sdn.ctl.nodemgr",
      "vendor": "H3C",
      "version": "2.0.0"
    },
    {
      "deployed": "2013-11-05T01:50:51.614Z",
      "desc": "Topology Viewer",
      "name": "Topology Viewer",
      "state": "ACTIVE",
      "uid": "com.h3c.sdn.tvue",
      "vendor": "H3C",
      "version": "2.0.0"
    }
  ]
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Forbidden (403)， Internal Server Error (500)，ServiceUnavailable (503)

2.4.2 获取指定应用程序信息

【方法】

GET /sdn/v2.0/apps/{*app_uid*}

【参数】

app_uid：必选，表示应用程序标识符。

【响应举例】

```
{
  "app": {
    "uid": "com.h3c.sdn.ctl.diag",
    "name": "Path Diagnostics",
    "version": "2.0.0",
    "vendor": "H3C",
    "desc": "Path Diagnostic Utility",
    "state": "ACTIVE",
    "deployed": "2013-11-05T01:50:49.454Z"
  }
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Forbidden (403)，Internal Server Error (500)，ServiceUnavailable (503)

2.4.3 加载应用程序

【方法】

POST /sdn/v2.0/apps

请求主体的数据需要包括应用程序的版本、名称、厂商名和可选描述等信息。

【响应举例】

```
{"app":
{
    "uid": "com.h3c.cloud",
     "version": "01.11.00.2342",
     "vendor": "H3C",
     "name": "Cloud Controller",
     "desc": "Cloud Network Controller",
     "state": "INSTALLED",
        "deployed": "2013-05-23T10:09:08.000Z"
      }
    }
```

【返回码】

- 正确：Created (201)
- 错误： Unauthorized (401)，Not Found (404)，ServiceUnavailable (503)

2.4.4 安装已加载的应用程序

【方法】

POST /sdn/v2.0/apps/{*app_uid*}/action

【参数】

app_uid：必选，表示应用程序标识符。

action：必选，取值为 install。

【请求举例】

```
install
```

【响应举例】

```
{
  "app": {
    "uid": "com.h3c.sdn.ctl.diag",
    "name": "Path Diagnostics",
    "version": "2.0.0",
    "vendor": "H3C",
    "desc": "Path Diagnostic Utility",
    "state": "ACTIVE",
    "deployed": "2013-11-05T01:50:49.454Z"
   }
}
```

【返回码】

- 正确：OK (200)
- 错误：BadRequest (400)，Unauthorized (401)，Forbidden (403)，Not Found (404)，Internal Server Error (500)，ServiceUnavailable (503)

2.4.5 启动已暂停的应用程序

【方法】

POST /sdn/v2.0/apps/{*app_uid*}/action

【参数】

app_uid：必选，表示应用程序标识符。

action：必选，取值为 start。

【请求举例】

```
start
```

【响应举例】

```
{
  "app": {
    "uid": "com.h3c.sdn.ctl.diag",
    "name": "Path Diagnostics",
    "version": "2.0.0",
    "vendor": "H3C",
    "desc": "Path Diagnostic Utility",
    "state": "ACTIVE",
    "deployed": "2013-11-05T01:50:49.454Z"
  }
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Not Found (404)，ServiceUnavailable (503)

2.4.6 暂停已运行的应用程序

【方法】

POST /sdn/v2.0/apps/{*app_uid*}/action

【参数】

app_uid：必选，表示应用程序标识符。

action：必选，取值为 stop。

【请求举例】

```
stop
```

【响应举例】

```
{
  "app": {
    "uid": "com.h3c.sdn.ctl.diag",
    "name": "Path Diagnostics",
    "version": "2.0.0",
    "vendor": "H3C",
    "desc": "Path Diagnostic Utility",
    "state": "RESOLVED",
    "deployed": "2013-11-05T01:50:49.454Z"
  }
}
```

【返回码】

- 正确：OK (200)
- 错误：Bad Request (400)， Unauthorized (401)，Forbidden (403)，Not Found (404)，Internal Server Error (500)，ServiceUnavailable (503)

2.4.7 卸载应用程序

【方法】

DELETE /sdn/v2.0/apps/ {*app_uid*}

【参数】

app_uid：必选，表示应用程序标识符。

【返回码】

- 正确：No Content (204)
- 错误： Unauthorized (401)，Not Found (404)，Service Unavailable (503)

2.4.8 获取应用程序状态诊断信息

【方法】

GET /sdn/v2.0/apps/{*app_uid*}/health

【参数】

app_uid：必选，表示应用程序标识符。

【响应举例】

```
{
  "app": {
    "uid": "com.h3c.sdn.ctl.diag",
    "name": "Path Diagnostics",
    "state": "ACTIVE",
    "status": "WARN",
    "deployed": "2013-11-05T01:50:49.454Z"
  }
}
```

有效状态可以反映出 OSGi 的状态。

正常的状态是 OK 和 WARN，非正常的状态是 NG。

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Not Found (404)，ServiceUnavailable (503)

2.5 审计日志/auditlog

【方法】

获取所有审计日志：

GET /sdn/v2.0/auditlog

获取指定用户的审计日志：

GET /sdn/v2.0/auditlog?user="*john.doe*"

获取指定行为的审计日志：

GET /sdn/v2.0/auditlog?activity="*suspicious*"

获取指定时间范围内的审计日志：

GET /sdn/v2.0/auditlog?start="*2013-11-10*T*08:34:15:590*Z"&end="*2013-11-10*T*08:34:15.590*Z"

【参数】

user：可选，用户名。指定该关键字， 将获取指定用户的审计日志。

activity：可选，生成审计日志的触发行为（如 Uninstall、Upload 等）。指定该关键字， 将获取指定行为的审计日志。

start：可选，起始时间，格式遵循 RFC 822 标准 （例如：YYYY-MM-DDTHH:MM:SS.000Z）。

end：可选，结束时间，格式遵循 RFC 822 标准（例如：YYYY-MM-DDTHH:MM:SS.000Z）。

【响应举例】

获取所有审计日志：

```
{
  "audit_log_entries": [
    {
      "uid": "884eeb06-96a4-497c-91be-585be772ac2a",
      "system_uid": "b5a7eb22-d2aa-43b1-8efb-8616eed6e4da",
      "user": "sdn",
      "origin": "Application Management",
      "ts": "2013-11-10T08:34:06.883Z",
      "activity": "Uninstall",
      "description": "Path Diagnostics has been removed"
    },
    {
      "uid": "80d9631e-32e2-4c90-9073-c3e48680917d",
      "system_uid": "b5a7eb22-d2aa-43b1-8efb-8616eed6e4da",
      "user": "sdn",
      "origin": "Application Management",
      "ts": "2013-11-10T08:34:15.590Z",
      "activity": "Upload",
      "description": "sdn-ctl-diag_2.0.0.zip has been staged"
    },
    {
      "uid": "953aa67e-d88a-40c7-a956-bda4b226db57",
      "system_uid": "b5a7eb22-d2aa-43b1-8efb-8616eed6e4da",
      "user": "sdn",
      "origin": "Application Management",
      "ts": "2013-11-10T08:37:08.327Z",
      "activity": "Install",
      "description": "Path Diagnostics has been installed"
    },
    {
      "uid": "ffd26967-0cce-49c4-8367-26d27161e2db",
      "system_uid": "b5a7eb22-d2aa-43b1-8efb-8616eed6e4da",
      "user": "sdn",
      "origin": "Application Management",
      "ts": "2013-11-11T10:42:17.814Z",
      "activity": "Upload",
      "description": "hm-1.0.0-SNAPSHOT.zip has been staged"
    },
    {
      "uid": "b5d8a550-6799-4ab9-bb01-b445af19db47",
      "system_uid": "b5a7eb22-d2aa-43b1-8efb-8616eed6e4da",
      "user": "sdn",
      "origin": "Application Management",
      "ts": "2013-11-11T10:42:22.519Z",
      "activity": "Install",
      "description": "hm has been installed"
    },
    {
      "uid": "65565166-2ca8-42bb-949b-3a11343b2e2e",
      "system_uid": "b5a7eb22-d2aa-43b1-8efb-8616eed6e4da",
      "user": "sdn",
      "origin": "Application Management",
      "ts": "2013-11-12T01:26:57.495Z",
      "activity": "Uninstall",
      "description": "hm has been removed"
    },
    {
      "uid": "a5bbb7a8-e93e-47de-809c-bb1668337f57",
      "system_uid": "b5a7eb22-d2aa-43b1-8efb-8616eed6e4da",
      "user": "sdn",
      "origin": "Application Management",
      "ts": "2013-11-12T01:27:16.311Z",
      "activity": "Upload",
      "description": "hm-1.0.0-SNAPSHOT.zip has been staged"
    },
    {
      "uid": "a0c21197-211d-4ff4-a1b4-cee34a2d3e41",
      "system_uid": "b5a7eb22-d2aa-43b1-8efb-8616eed6e4da",
      "user": "sdn",
      "origin": "Application Management",
      "ts": "2013-11-12T01:27:21.934Z",
      "activity": "Install",
      "description": "hm has been installed"
    }
  ]
}
```

【返回码】

- 正确：OK (200)，Partial Content (206)
- 错误：Bad Request (400)， Unauthorized (401)， Forbidden (403)，Bad Method (405)，Internal Server Error (500)，Service Unavailable (503)

2.6 控制器管理/systems

2.6.1 获取所有控制器信息

【方法】

获取所有控制器信息：

GET /sdn/v2.0/systems

获取指定 IP 地址的控制器信息：

GET /sdn/v2.0/systems?ip=*"ip_address"*

【参数】

ip_address：可选，控制器的 IP 地址。

【响应举例】

```
{
  "systems": [
    {
      "uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
      "version": "2.0.0.0000",
      "ip": "192.168.56.168",
      "role": "leader",
      "core_data_version": 43,
      "core_data_version_timestamp": "2013-11-05T08:43:01.268Z",
      "time": "2013-11-05T07:46:57.210Z",
      "status": "active",
      "self": true
    }
  ]
}
```

【返回码】

- 正确：OK (200)
- 错误：Bad Request (400)， Unauthorized (401)， Forbidden (403)，Bad Method (405)，PageSizeExceeded (413)，Service Unavailable (503)， Item NotFound (404)

2.6.2 获取指定控制器信息

【方法】

GET /sdn/v2.0/systems/{*systems_uid*}

【参数】

systems_uid：必选，表示控制器的唯一标识 ID。

【响应举例】

```
{
  "system": {
    "uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
    "version": "2.0.0.0000",
    "ip": "192.168.56.168",
    "role": "leader",
    "core_data_version": 43,
    "core_data_version_timestamp": "2013-11-05T08:43:01.268Z",
    "time": "2013-11-05T07:46:57.210Z",
    "status": "active",
    "self": true
  }
}
```

【返回码】

- 正确：OK (200)
- 错误：Bad Request (400)， Unauthorized (401)， Forbidden (403)，Bad Method (405)，PageSizeExceeded (413)，Service Unavailable (503)， Item NotFound (404)

2.6.3 更新控制器的IP地址

更新 IP 地址仅适用于独立运行模式的控制器，如果更新 IP 地址是在集群模式的控制器下运行，则将返回错误报告。

【方法】

PUT /sdn/v2.0/systems/{*systems_uid*}

【参数】

*systems*_uid：必选，表示控制器的唯一标识 ID。

【请求举例】

```
{
  "system": {
  "ip": "192.168.1.200"
  }
}
```

【响应举例】

JSON 码返回的是更新后的控制器信息，例如：

```
{
  "system": {
  "uid": "adc5e492-957c-4f8c-aa0a-97fa2dac5f23",
  "version": "01.14.00.0000",
  "ip": "192.168.1.200",
  "role": "leader",
  "core_data_version": 8,
  "core_data_version_timestamp": "2013-08-21T18:17:33.187Z",
  "time": "2013-08-21T18:17:23.899Z",
  "status": "active",
  "self": true
  }
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Not Found (404)，Service Unavailable (503)

2.6.4 备份控制器

【方法】

POST /sdn/v2.0/systems/{*system_uid*}/action

【参数】

*systems*_uid：必选，表示控制器的唯一标识 ID。

action：必选，取值为 backup。

【请求举例】

```
backup
```

【响应举例】

```
{
  "session_ids": {
    "session_id": "Azho8odIMS",
    "nodetokens": []
  },
  "uri": "https://192.168.56.168:8443/sdn/v2.0/backups/Azho8odIMS"
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Not Found (404)，Service Unavailable (503)

2.6.5 恢复控制器

【方法】

POST /sdn/v2.0/systems/{*system_uid*}/action

【参数】

*systems*_uid：必选，表示控制器的唯一标识 ID。

action：必选，取值为 restore。

【请求举例】

```
restore
```

【响应举例】

```
{
  "session_ids": {
    "session_id": "1ul1RKQxQZ",
    "nodetokens": []
  },
  "uri": "file:///opt/sdn/backup/restore.log"
}
```

说明

没有 API 接口用于确认恢复状态。

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Not Found (404)，Service Unavailable (503)

2.6.6 上传备份文件

【方法】

POST /sdn/v2.0/systems/{*system uid*}/backup

请求数据是一个包含待上传备份文件的字节流。

【参数】

*systems*_uid：表示控制器的唯一标识 ID。

【返回码】

- 正确：Created (201)
- 错误： Unauthorized (401)，Not Found (404)，Service Unavailable (503)

2.7 备份组配置/regions

2.7.1 获取所有备份组信息

【方法】

GET /sdn/v2.0/regions

【响应举例】

```
{
  "regions": [
    {
      "uid": "f8d325d5-951d-4da8-ad64-cc716156d07b",
      "master": {
        "ip": "192.168.56.167",
        "name": "Controller_1"
      },
      "slaves": [
        {
          "ip": "192.168.56.169",
          "name": "Controller_3"
        },
        {
          "ip": "192.168.56.168",
          "name": "Controller_2"
        }
      ],
      "devices": [
        {
          "ip": "192.168.56.161"
        },
        {
          "ip": "192.168.56.162"
        }
      ]
    }
  ]
}
```

【返回码】

- 正确：OK (200)
- 错误：Bad Request (400)，Unauthorized (401)， Forbidden (403)，Not Found (404)， Service Unavailable (503)

2.7.2 创建备份组

【方法】

POST /sdn/v2.0/regions

【请求举例】

```
{
  "region": {
    "master": {
      "ip": "192.168.56.167",
      "name": "Controller_1"
    },
    "slaves": [
      {
        "ip": "192.168.56.168",
        "name": "Controller_2"
      },
      {
      "ip": "192.168.56.169",
        "name": "Controller_3"
      }
    ],
    "devices": [
      {
        "ip": "192.168.56.161"
      },
      {
        "ip": "192.168.56.162"
      }
    ]
  }
}
```

【响应举例】

```
{
  "region": {
    "uid": "f8d325d5-951d-4da8-ad64-cc716156d07b",
    "master": {
      "ip": "192.168.56.167",
      "name": "Controller_1"
    },
    "slaves": [
      {
        "ip": "192.168.56.169",
        "name": "Controller_3"
      },
      {
        "ip": "192.168.56.168",
        "name": "Controller_2"
      }
    ],
    "devices": [
      {
        "ip": "192.168.56.161"
      },
      {
        "ip": "192.168.56.162"
      }
    ]
  }
}
```

【返回码】

- 正确：Created (201)
- 错误：Bad Request (400)， Unauthorized (401)，Forbidden (403)，Not Found (404)，DuplicateData (409)， InternalError (500)，Service Unavailable (503)

2.7.3 获取指定备份组信息

【方法】

GET /sdn/v2.0/regions/{*region_uid*}

【参数】

region_uid：必选，表示备份组标识符。

【响应举例】

```
{
  "region": {
    "uid": "f8d325d5-951d-4da8-ad64-cc716156d07b",
    "master": {
      "ip": "192.168.56.167",
      "name": "Controller_1"
    },
    "slaves": [
      {
        "ip": "192.168.56.169",
        "name": "Controller_3"
      },
      {
        "ip": "192.168.56.168",
        "name": "Controller_2"
      }
    ],
    "devices": [
      {
        "ip": "192.168.56.161"
      },
      {
        "ip": "192.168.56.162"
      }
    ]
  }
}
```

【返回码】

- 正确：OK (200)
- 错误：Bad Request (400)，Unauthorized (401)， Forbidden (403)，Not Found (404)， Service Unavailable (503)

2.7.4 更新备份组

【方法】

PUT /sdn/v2.0/regions/{*region_uid*}

【参数】

region_uid：必选，表示备份组标识符。

【请求举例】

```
{
  "region" : {
    "master" : {
      "ip" : "125.200.104.101",
      "name" : "Controller_1"
    },
    "slaves" : [ {
      "ip" : "125.200.104.102",
      "name" : "Controller_2"
    } ],
    "devices" : [ {
      "ip" : "125.200.104.200"
    } ]
  }
}
```

【响应举例】

```
{
  "region" : {
    "uid" : "adc5e492-957c-4f8c-aa0a-97fa2dac5f01",
    "master" : {
      "ip" : "125.200.104.101",
      "name" : "Controller_1"
    },
    "slaves" : [ {
      "ip" : "125.200.104.102",
      "name" : "Controller_2"
    } ],
    "devices" : [ {
      "ip" : "125.200.104.200"
    } ]
  }
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)， Not Found (404)， Service Unavailable (503)

2.7.5 刷新备份组

【方法】

POST /sdn/v2.0/regions/{*region_uid*}/refresh

【参数】

region_uid：必选，表示备份组标识符。

【请求举例】

```
{
  "region_refresh": {
    "master": {
      "ip": "192.168.56.167",
      "name": "Controller_1"
    },
    "slaves": [
      {
        "ip": "192.168.56.168",
        "name": "Controller_2"
      } ,
      {
        "ip": "192.168.56.169",
        "name": "Controller_3"
      }
    ],
    "devices": [
      {
        "ip": "192.168.56.193"
      }
    ]
  }
}
```

【响应举例】

```
{
  "Result": "Successfully refreshed roles for the device(s): [192.168.56.193]"
}
```

【返回码】

- 正确：OK (200)
- 错误：BadRequest (400)， Unauthorized (401)，Forbidden (403)， Not Found (404)，InternalError (500)， Service Unavailable (503)

2.7.6 删除备份组

【方法】

DELETE /sdn/v2.0/regions/{*region_uid*}

【参数】

region_uid：必选，表示备份组标识符。

【返回码】

- 正确：No Content (204)
- 错误：BadRequest (400)， Unauthorized (401)， Not Found (404)，InternalError (500)，Service Unavailable (503)

2.8 集群配置/team

2.8.1 获取集群配置信息

【方法】

GET /sdn/v2.0/team

【响应举例】

```
{
  "team": {
    "name": "Test Cluster",
    "ip": "192.168.56.101",
    "version": "13755950952",
    "systems": [
      {
        "name": "member 1",
        "ip": "192.168.56.167",
        "priority": 30
      },
      {
        "name": "member 2",
        "ip": "192.168.56.168",
        "priority": 20
      },
      {
        "name": "member 3",
        "ip": "192.168.56.169",
        "priority": 10
      }
    ]
  }
}
```

【返回码】

- 正确：OK (200)
- 错误：BadRequest (400)， Unauthorized (401)，Forbidden (403)， Not Found (404)，Bad Method (405)， Service Unavailable (503)

2.8.2 创建集群

【方法】

POST /sdn/v2.0/team

【请求举例】

```
{
  "team": {
    "name": "Test Cluster",
    "ip": "192.168.56.101",
    "version": "13755950952",
    "systems": [
      {
        "name": "member 1",
        "ip": "192.168.56.167",
        "priority": 30
      },
      {
        "name": "member 2",
        "ip": "192.168.56.168",
        "priority": 20
      },
      {
        "name": "member 3",
        "ip": "192.168.56.169",
        "priority": 10
      }
    ]
  }
}
```

如果“forward_request”属性值为 true(默认值为 true)，那么集群配置信息将会同步到指定列表中的所有控制器上。

如果“forward_request”属性值为 false，那么集群配置信息将不会同步到指定列表中的其它控制器上。

【返回码】

- 正确：OK (200)
- 错误：multi-status (207)，BadRequest (400)， Unauthorized (401)，Forbidden (403)，Bad Method (405)， Service Unavailable (503)

2.8.3 删除集群

【方法】

DELETE /sdn/v2.0/team

【返回码】

- 正确：No Content (204)
- 错误： Unauthorized (401)， Not Found (404)， Service Unavailable (503)

2.8.4 触发集群Leader重新选举

【方法】

POST /sdn/v2.0/team/action

【参数】

action：必选，取值为 election。

【请求举例】

```
election
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)， Not Found (404)， Service Unavailable (503)

2.9 备份/backups

【方法】

查询控制器的备份状态：

GET /sdn/v2.0/backups/{*session_uid*}

【参数】

session _uid：必选，用于标识本次备份操作的标识码。

【响应举例】

```
{
  "statusCode": 3,
  "Description": "Operation complete."
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)， Not Found (404)， Service Unavailable (503)

2.10 告警/alerts

2.10.1 获取告警信息

【方法】

GET /sdn/v2.0/alerts

【响应举例】

```
{
  "alerts": [
    {
      "uid": "4d17963a-1c0f-448d-b1af-c70d57fac1fc",
    "system_uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
    "topic": "teaming",
    "org": "TeamingManager",
    "ts": "2013-11-05T01:51:04.539Z",
    "sev": "INFO",
    "state": true,
    "desc": "BECOME_MEMBER, ID: Id[value=6ecf003c-96f5-4dc8-935a-e7cad0f3d44f]"
  },
  {
    "uid": "d7ea7e6e-e43b-46af-a8ca-f63d6f2b0241",
    "system_uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
    "topic": "teaming",
    "org": "TeamingManager",
    "ts": "2013-11-05T01:51:04.586Z",
    "sev": "INFO",
    "state": true,
    "desc": "BECOME_LEADER, ID: Id[value=6ecf003c-96f5-4dc8-935a-e7cad0f3d44f]"
  },
  {
    "uid": "95a93f3f-0bec-4df5-b3ae-e73d1a9507b9",
    "system_uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
    "topic": "of_controller",
    "org": "Core Controller",
    "ts": "2013-11-05T01:51:04.672Z",
    "sev": "INFO",
    "state": true,
    "desc": "OpenFlow Controller active on port 6633"
  },
  {
    "uid": "ad08983c-61ff-43de-8487-6c7d72cffeeb",
    "system_uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
    "topic": "licensing",
    "org": "compliance-manager",
    "ts": "2013-11-05T01:52:37.057Z",
    "sev": "CRITICAL",
    "state": true,
    "desc": "No active base product licenses are found, license compliance failed!"
  },
  {
    "uid": "c463060f-509b-42fe-886a-4871a9986324",
    "system_uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
    "topic": "teaming",
    "org": "TeamingManager",
    "ts": "2013-11-05T05:18:32.480Z",
    "sev": "INFO",
    "state": true,
    "desc": "BECOME_SUSPENDED, ID: Id[value=6ecf003c-96f5-4dc8-935a-e7cad0f3d44f]"
  },
  {
    "uid": "bed2c9b8-b047-4b30-820a-2eae8a632173",
    "system_uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
    "topic": "teaming",
    "org": "TeamingManager",
    "ts": "2013-11-05T05:18:56.706Z",
    "sev": "INFO",
    "state": true,
    "desc": "BECOME_MEMBER, ID: Id[value=6ecf003c-96f5-4dc8-935a-e7cad0f3d44f]"
  },
  {
    "uid": "95cbb1cc-d803-44ce-bfcf-27152fe8fcee",
    "system_uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
    "topic": "teaming",
    "org": "TeamingManager",
    "ts": "2013-11-05T05:18:56.760Z",
    "sev": "INFO",
    "state": true,
    "desc": "BECOME_LEADER, ID: Id[value=6ecf003c-96f5-4dc8-935a-e7cad0f3d44f]"
  },
  {
    "uid": "7d8be274-09f4-4789-82d2-aab20c662bff",
    "system_uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
    "topic": "of_controller",
    "org": "Core Controller",
    "ts": "2013-11-05T05:18:56.813Z",
    "sev": "INFO",
    "state": true,
    "desc": "OpenFlow Controller port disabled"
  },
  {
    "uid": "d9997937-778b-4f21-bb53-7777f1e0467a",
    "system_uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
    "topic": "of_controller",
    "org": "Core Controller",
    "ts": "2013-11-05T05:18:57.249Z",
    "sev": "INFO",
    "state": true,
    "desc": "OpenFlow Controller active on port 6633"
  },
  {
    "uid": "b8221181-fcdb-45bb-9755-c3b15a9167fc",
    "system_uid": "0e34e269-519d-4b98-ba2c-d453c8d863be",
    "topic": "teaming",
    "org": "TeamingManager",
    "ts": "2013-11-05T05:18:57.288Z",
    "sev": "INFO",
    "state": true,
    "desc": "NEW_LEADER, ID: Id[value=6ecf003c-96f5-4dc8-935a-e7cad0f3d44f]"
  },
  {
    "uid": "99b73f4d-5e73-4ad5-ae96-47629a47e5e7",
    "system_uid": "0e34e269-519d-4b98-ba2c-d453c8d863be",
    "topic": "teaming",
    "org": "TeamingManager",
    "ts": "2013-11-05T05:18:57.315Z",
    "sev": "INFO",
    "state": true,
    "desc": "BECOME_MEMBER, ID: Id[value=0e34e269-519d-4b98-ba2c-d453c8d863be]"
  },
  {
    "uid": "7e1cb84b-286f-4033-bcea-d51c4983e34a",
    "system_uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
    "topic": "teaming",
    "org": "TeamingManager",
    "ts": "2013-11-05T05:18:57.345Z",
    "sev": "INFO",
    "state": true,
    "desc": "MEMBER_JOIN, ID: Id[value=0e34e269-519d-4b98-ba2c-d453c8d863be]"
  },
  {
    "uid": "8f235a7b-02fa-4dda-be3b-79417c30cd20",
    "system_uid": "0e34e269-519d-4b98-ba2c-d453c8d863be",
    "topic": "teaming",
    "org": "TeamingManager",
    "ts": "2013-11-05T05:18:58.942Z",
    "sev": "INFO",
    "state": true,
    "desc": "MEMBER_JOIN, ID: Id[value=4cf34c21-b2e8-4a38-8a35-6db3e75dd9bf]"
  },
  {
    "uid": "fe6515ee-7b5b-4435-8f3e-ada94afb85c5",
    "system_uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
    "topic": "teaming",
    "org": "TeamingManager",
    "ts": "2013-11-05T05:18:59.062Z",
    "sev": "INFO",
    "state": true,
    "desc": "MEMBER_JOIN, ID: Id[value=4cf34c21-b2e8-4a38-8a35-6db3e75dd9bf]"
  },
  {
    "uid": "29e18796-85f0-4e04-ab5b-8706344e84a5",
    "system_uid": "4cf34c21-b2e8-4a38-8a35-6db3e75dd9bf",
    "topic": "teaming",
    "org": "TeamingManager",
    "ts": "2013-11-05T05:18:59.719Z",
    "sev": "INFO",
    "state": true,
    "desc": "NEW_LEADER, ID: Id[value=6ecf003c-96f5-4dc8-935a-e7cad0f3d44f]"
  },
  {
    "uid": "3ee92d25-66ef-42fb-8739-c37feeb1bb05",
    "system_uid": "4cf34c21-b2e8-4a38-8a35-6db3e75dd9bf",
    "topic": "teaming",
    "org": "TeamingManager",
    "ts": "2013-11-05T05:18:59.742Z",
    "sev": "INFO",
    "state": true,
    "desc": "BECOME_MEMBER, ID: Id[value=4cf34c21-b2e8-4a38-8a35-6db3e75dd9bf]"
  },
  {
    "uid": "c917c358-2e0b-408d-83f4-56977a8899d8",
    "system_uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
    "topic": "teaming",
    "org": "TeamingManager",
    "ts": "2013-11-05T05:21:01.415Z",
    "sev": "INFO",
    "state": true,
    "desc": "MEMBER_LEAVE, ID: Id[value=0e34e269-519d-4b98-ba2c-d453c8d863be]"
  },
{
  "uid": "9aed7dfe-b4db-4830-baaf-74a910cf950a",
  "system_uid": "4cf34c21-b2e8-4a38-8a35-6db3e75dd9bf",
  "topic": "teaming",
  "org": "TeamingManager",
  "ts": "2013-11-05T05:21:02.008Z",
  "sev": "INFO",
  "state": true,
  "desc": "MEMBER_LEAVE, ID: Id[value=0e34e269-519d-4b98-ba2c-d453c8d863be]"
},
{
  "uid": "e7bc506d-a8ef-4196-ad44-ea1d8e2e6dc4",
  "system_uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
  "topic": "teaming",
  "org": "TeamingManager",
  "ts": "2013-11-05T05:23:35.302Z",
  "sev": "INFO",
  "state": true,
  "desc": "BECOME_SUSPENDED, ID: Id[value=6ecf003c-96f5-4dc8-935a-e7cad0f3d44f]"
},
{
  "uid": "56373bac-1b5f-4cf9-9937-c529847a5007",
  "system_uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
  "topic": "of_controller",
  "org": "Core Controller",
  "ts": "2013-11-05T05:25:13.481Z",
  "sev": "INFO",
  "state": true,
  "desc": "OpenFlow Controller port disabled"
},
{
  "uid": "7b1bb572-ca40-48dc-9d3f-325933803a68",
  "system_uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
  "topic": "teaming",
  "org": "TeamingManager",
  "ts": "2013-11-05T05:25:13.606Z",
  "sev": "INFO",
  "state": true,
  "desc": "BECOME_MEMBER, ID: Id[value=6ecf003c-96f5-4dc8-935a-e7cad0f3d44f]"
},
{
  "uid": "2f46eef7-6e70-4365-9aab-33a766cb37e5",
  "system_uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
  "topic": "teaming",
  "org": "TeamingManager",
  "ts": "2013-11-05T05:25:13.652Z",
  "sev": "INFO",
  "state": true,
  "desc": "BECOME_LEADER, ID: Id[value=6ecf003c-96f5-4dc8-935a-e7cad0f3d44f]"
},
{
  "uid": "36e88f8e-874e-4217-93e6-1d22f451e6a5",
  "system_uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
  "topic": "of_controller",
  "org": "Core Controller",
  "ts": "2013-11-05T05:25:13.662Z",
  "sev": "INFO",
  "state": true,
  "desc": "OpenFlow Controller port disabled"
},
{
  "uid": "4b2db847-6040-4e0b-93eb-368130b7e2f4",
  "system_uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
  "topic": "teaming",
  "org": "TeamingManager",
  "ts": "2013-11-05T05:25:14.887Z",
  "sev": "INFO",
  "state": true,
  "desc": "MEMBER_LEAVE, ID: Id[value=4cf34c21-b2e8-4a38-8a35-6db3e75dd9bf]"
},
{
  "uid": "9a0dca22-31e6-4b33-9bf8-44044f258f94",
  "system_uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
  "topic": "teaming",
  "org": "TeamingManager",
  "ts": "2013-11-05T05:25:14.934Z",
  "sev": "INFO",
  "state": true,
  "desc": "MEMBER_JOIN, ID: Id[value=4cf34c21-b2e8-4a38-8a35-6db3e75dd9bf]"
},
{
  "uid": "0bf64fa5-18e4-41f5-98a2-c3be66ed57bf",
  "system_uid": "4cf34c21-b2e8-4a38-8a35-6db3e75dd9bf",
  "topic": "teaming",
  "org": "TeamingManager",
  "ts": "2013-11-05T05:25:15.595Z",
  "sev": "INFO",
  "state": true,
  "desc": "NEW_LEADER, ID: Id[value=6ecf003c-96f5-4dc8-935a-e7cad0f3d44f]"
},
{
  "uid": "ef539acf-e574-46e4-99e8-b83c332f57b0",
  "system_uid": "4cf34c21-b2e8-4a38-8a35-6db3e75dd9bf",
  "topic": "teaming",
  "org": "TeamingManager",
  "ts": "2013-11-05T05:25:15.634Z",
  "sev": "INFO",
  "state": true,
  "desc": "BECOME_MEMBER, ID: Id[value=4cf34c21-b2e8-4a38-8a35-6db3e75dd9bf]"
},
{
  "uid": "1881de51-7984-4332-a26c-8da369c7e0b6",
  "system_uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
  "topic": "of_controller",
  "org": "Core Controller",
  "ts": "2013-11-05T05:25:23.639Z",
  "sev": "INFO",
  "state": true,
  "desc": "OpenFlow Controller active on port 6633"
},
{
  "uid": "89de9d51-7c0a-4566-904f-852c8c5ad9aa",
"system_uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
"topic": "teaming",
"org": "TeamingManager",
"ts": "2013-11-05T05:25:59.107Z",
"sev": "INFO",
"state": true,
"desc": "MEMBER_JOIN, ID: Id[value=0e34e269-519d-4b98-ba2c-d453c8d863be]"
},
{
"uid": "eb2a9215-eb29-47b1-835d-6ceccb2c7227",
"system_uid": "0e34e269-519d-4b98-ba2c-d453c8d863be",
"topic": "teaming",
"org": "TeamingManager",
"ts": "2013-11-05T05:25:59.130Z",
"sev": "INFO",
"state": true,
"desc": "NEW_LEADER, ID: Id[value=6ecf003c-96f5-4dc8-935a-e7cad0f3d44f]"
},
{
"uid": "cc978adf-b00a-4867-be98-0ea4499abf7f",
"system_uid": "0e34e269-519d-4b98-ba2c-d453c8d863be",
"topic": "teaming",
"org": "TeamingManager",
"ts": "2013-11-05T05:25:59.163Z",
"sev": "INFO",
"state": true,
"desc": "BECOME_MEMBER, ID: Id[value=0e34e269-519d-4b98-ba2c-d453c8d863be]"
},
{
"uid": "48d5b074-1819-4475-9b18-f27618a28b95",
"system_uid": "4cf34c21-b2e8-4a38-8a35-6db3e75dd9bf",
"topic": "teaming",
"org": "TeamingManager",
"ts": "2013-11-05T05:25:59.694Z",
"sev": "INFO",
"state": true,
"desc": "MEMBER_JOIN, ID: Id[value=0e34e269-519d-4b98-ba2c-d453c8d863be]"
},
{
"uid": "606d3dff-1fe4-4887-8817-9085abc026d5",
"system_uid": "4cf34c21-b2e8-4a38-8a35-6db3e75dd9bf",
"topic": "teaming",
"org": "TeamingManager",
"ts": "2013-11-05T05:52:57.330Z",
"sev": "INFO",
"state": true,
"desc": "BECOME_SUSPENDED, ID: Id[value=4cf34c21-b2e8-4a38-8a35-6db3e75dd9bf]"
},
{
"uid": "3cb8a812-9420-4c8f-9665-243db5c51d05",
"system_uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f",
"topic": "teaming",
"org": "TeamingManager",
"ts": "2013-11-05T05:54:15.171Z",
"sev": "INFO",
"state": true,
    "desc": "BECOME_SUSPENDED, ID: Id[value=6ecf003c-96f5-4dc8-935a-e7cad0f3d44f]"
  },
  {
    "uid": "daa6fda9-2a61-4ce4-b17a-ce7e0c667555",
    "system_uid": "4cf34c21-b2e8-4a38-8a35-6db3e75dd9bf",
    "topic": "of_controller",
    "org": "Core Controller",
    "ts": "2013-11-05T05:56:33.162Z",
    "sev": "INFO",
    "state": true,
    "desc": "OpenFlow Controller port disabled"
  },
  {
    "uid": "b3195ea0-03b4-45d0-8360-f116e709435b",
    "system_uid": "4cf34c21-b2e8-4a38-8a35-6db3e75dd9bf",
    "topic": "teaming",
    "org": "TeamingManager",
    "ts": "2013-11-05T05:56:54.976Z",
    "sev": "INFO",
    "state": true,
    "desc": "NEW_LEADER, ID: Id[value=6ecf003c-96f5-4dc8-935a-e7cad0f3d44f]"
  },
  {
    "uid": "931795cf-4d04-46dd-93d2-22afdf5abbae",
    "system_uid": "4cf34c21-b2e8-4a38-8a35-6db3e75dd9bf",
    "topic": "teaming",
    "org": "TeamingManager",
    "ts": "2013-11-05T05:56:55.018Z",
    "sev": "INFO",
    "state": true,
    "desc": "BECOME_MEMBER, ID: Id[value=4cf34c21-b2e8-4a38-8a35-6db3e75dd9bf]"
  },
  {
    "uid": "25427a76-475b-4756-a752-4d6dc082e214",
    "system_uid": "4cf34c21-b2e8-4a38-8a35-6db3e75dd9bf",
    "topic": "of_controller",
    "org": "Core Controller",
    "ts": "2013-11-05T05:56:55.072Z",
    "sev": "INFO",
    "state": true,
    "desc": "OpenFlow Controller port disabled"
  },
  {
    "uid": "381377d5-4370-4e1b-846f-83b816c4abeb",
    "system_uid": "4cf34c21-b2e8-4a38-8a35-6db3e75dd9bf",
    "topic": "of_controller",
    "org": "Core Controller",
    "ts": "2013-11-05T05:56:55.270Z",
    "sev": "INFO",
    "state": true,
    "desc": "OpenFlow Controller active on port 6633"
  }
}
```

【返回码】

- 正确：OK (200)，PartialContent (206)

- 错误：Bad Request (400)， Unauthorized (401)， Forbidden (403)， Bad Method (405)， Internal Server Error (500)， Service Unavailable (503)

2.10.2 获取告警主题

【方法】

获取所有告警主题：

GET /sdn/v2.0/alerts/topics

获取指定来源（origin）的告警主题：

GET /sdn/v2.0/alerts/topics?org="*origin*"

【参数】

origin：可选，表示日志来源模块。

【响应举例】

获取所有告警主题：

```
{
  "alert_topics": [
    {
      "topic": "connection_point",
      "org": "OF-Controller",
      "desc": "Alerts associated with links"
    },
    {
      "topic": "datapath",
      "org": "OF-Controller",
      "desc": "Alerts associated with links"
    },
    {
      "topic": "of_controller",
      "org": "OF-Controller",
      "desc": "Alerts from the Controller"
    },
    {
      "topic": "of_controller_link",
      "org": "OF-Controller",
      "desc": "Alerts associated with links"
    },
    {
      "topic": "of_controller_pathdiag",
      "org": "OF-Controller",
      "desc": "Alerts associated with path diagnostic"
    }
  ]
}
```

【返回码】

- 正确：OK (200)
- 错误：Bad Request (400)， Unauthorized (401)，Forbidden (403)，Bad Method (405)，Service Unavailable (503)

2.10.3 获取告警监听器

【方法】

获取所有告警监听器：

GET /sdn/v2.0/alerts/ listeners

【响应举例】

```
{
  "alert_topic_listeners": [
    {
      "uid": "0c46665b-a27b-4536-8fd8-d5dcbb8c79d1",
      "app_id": "imc",
      "name": "IMC OpenFLow Listener",
      "callbacks": [
        {
          "topics": [
            "of_controller_link",
            "of_controller"
          ],
          "uri": "http://imc.h3c.com/sdn"
        }
      ]
    }
  ]
}
```

【返回码】

- 正确：OK (200)
- 错误：Bad Request (400)， Unauthorized (401)，Forbidden (403)，Bad Method (405)，Service Unavailable (503)

2.10.4 创建一个告警监听器

【方法】

创建一个告警监听器（包括指定监听的主题）：

POST /sdn/v2.0/alerts/listeners

【请求举例】

```
        {
          "alert_topic_listener" : {
          "app_id" : "imc",
          "name" : "IMC OpenFLow Listener",
          "callbacks" : [ {
          "topics" : [ "of_controller", “of_controller_link” ],
          "uri" : "http://imc.h3c.com/sdn"
            } ]
          }
      }
```

【响应举例】

```
{
  "alert_topic_listener": {
    "uid": "0c46665b-a27b-4536-8fd8-d5dcbb8c79d1",
    "app_id": "imc",
    "name": "IMC OpenFLow Listener",
    "callbacks": [
      {
        "topics": [
          "of_controller_link",
          "of_controller"
        ],
        "uri": "http://imc.h3c.com/sdn"
      }
    ]
  }
}
```

【返回码】

- 正确：OK (200)
- 错误：Bad Request (400)， Unauthorized (401)，Forbidden (403)，Bad Method (405)，Internal Server Error (500)， Service Unavailable (503)

2.10.5 获取告警监听器信息

【方法】

获取指定告警监听器的详细信息：

GET /sdn/v2.0/alerts/listeners/{*listener_uid*}

【参数】

*listener*_uid：必选，表示监听标识符。

【响应举例】

```
{
  "alert_topic_listener": {
    "uid": "0c46665b-a27b-4536-8fd8-d5dcbb8c79d1",
    "app_id": "imc",
    "name": "IMC OpenFLow Listener",
    "callbacks": [
      {
        "topics": [
          "of_controller_link",
          "of_controller"
        ],
        "uri": "http://imc.h3c.com/sdn"
      }
    ]
  }
}
```

【返回码】

- 正确：OK (200)
- 错误：Bad Request (400)，Unauthorized (401)， Forbidden (403)，Bad Method (405)，Service Unavailable (503)

2.10.6 更新告警监听器信息

【方法】

更新告警监听器信息（包括更改监听的主题）：

PUT /sdn/v2.0/alerts/listeners/{*listener_uid*}

【参数】

*listener*_uid：必选，表示监听标识符。

【请求举例】

```
{
  "alert_topic_listener" : {
  "uid" : "0c46665b-a27b-4536-8fd8-d5dcbb8c79d1",
  "app_id" : "imc",
  "name" : "IMC OpenFLow Listener1",
  "callbacks" : [ {
  "topics" : [ "of_controller ", "of_controller_link" ],
  "uri" : "http://imc.h3c.com/sdn"
  } ]
  }
}
```

【响应举例】

```
{
  "alert_topic_listener": {
    "uid": "0c46665b-a27b-4536-8fd8-d5dcbb8c79d1",
    "app_id": "imc",
    "name": "IMC OpenFLow Listener1",
    "callbacks": [
      {
        "topics": [
          "of_controller ",
          "of_controller_link"
        ],
        "uri": "http://imc.h3c.com/sdn"
      }
    ]
  }
}
```

【返回码】

- 正确：OK (200)
- 错误：Bad Request (400)，Unauthorized (401)， Forbidden (403)，Bad Method (405)，Internal Server Error (500)， Service Unavailable (503)

2.10.7 删除告警监听器

【方法】

DELETE /sdn/v2.0/alerts/listeners/{*listener_uid*}

【参数】

*listener*_uid：必选，表示监听标识符。

【返回码】

- 正确：No Content (204)
- 错误：Bad Request (400)，Unauthorized (401)， Forbidden (403)，Bad Method (405)，Internal Server Error (500)， Service Unavailable (503)

2.11 度量/metrics

2.11.1 获取注册度量的应用程序信息

【方法】

获取注册度量的应用程序名称和 ID：

GET /sdn/v2.0/metrics/apps

【响应举例】

```
{
  "apps": [
    {
      "app_id": "com.h3c.sdn",
      "app_name": "H3C VCF Controller"
    }
  ]
}
```

【返回码】

- 正确：OK (200)
- 错误：Bad Request (400)， Unauthorized (401)，Forbidden (403)，Bad Method (405)，Service Unavailable (503)

2.11.2 获取指定应用程序注册的所有度量

【方法】

GET /sdn/v2.0/metrics/apps/{*app_id***}?primary_tag="***primary_tag***"&secondary_tag="** *secondary_tag* **"&name="***name***"**

【参数】

app_id：必选，应用程序 ID。如果没有指定，则返回错误。

primary_tag：可选，一级标签。如果没有指定，则系统自动使用通配符，如果指定，则返回指定一级标签的度量。

secondary_tag：可选，二级标签。如果没有指定，则系统自动使用通配符，如果指定，则返回指定二级标签的度量。

name：可选，度量名称。如果没有指定，则系统自动使用通配符，如果指定，则返回度量名的度量。

【响应举例】

```
{
  "metrics": [
    {
      "app_id": "com.h3c.sdn",
      "type": "COUNTER",
      "name": "number_of_nodes",
      "description": "Counting the number of nodes with",
      "primary_tag": "nodes",
      "secondary_tag": "node",
      "jmx": true,
      "persistence": true,
      "summary_interval": "FIVE",
      "uid": "6bafa6fb-7c00-49bf-84c9-6bffa63b4e66"
        }
      ]
    }
```

【返回码】

- 正确：OK (200)
- 错误：Bad Request (400)， Unauthorized (401)，Forbidden (403)，Bad Method (405)，Service Unavailable (503)

2.11.3 获取指定度量的详细信息

【方法】

GET /sdn/v2.0/metrics/{*app_id*}

【参数】

app_id：必选，表示应用程序 ID。

【响应举例】

```
{
  "metric": {
   "app_id": "com.h3c.sdn",
      "type": "COUNTER",
      "name": "number_of_nodes",
      "description": "Counting the number of nodes with",
      "primary_tag": "nodes",
      "secondary_tag": "node",
      "jmx": true,
      "persistence": true,
      "summary_interval": "FIVE",
      "uid": "6bafa6fb-7c00-49bf-84c9-6bffa63b4e66"
  }
}
```

【返回码】

- 正确：OK (200)
- 错误：Bad Request (400)， Unauthorized (401)，Forbidden (403)，Bad Method (405)，Service Unavailable (503)

2.11.4 获取度量值

【方法】

GET /sdn/v2.0/metrics/{*metric_uid*}/values

【参数】

metric_uid：必选，表示度量标识符。

start：可选，请求时间周期的开始时间，可选项，格式为“YYYY-MM-dd-hh:mm”。如果开始和结束时间都没有指定，则会返回最后记录的度量值；如果只设定结束时间而不设定开始时间，开始时间则是未老化的第一个度量值的记录时间。

end：可选，请求时间周期的结束时间，可选项，格式为“YYYY-MM-dd-hh:mm”。如果结束时间没有指定，则结束时间为本次请求的时间。

Interval：可选，度量值的统计时间间隔，当开始和结束时间都未指定时为可选，当开始或结束时间选定时为必选。取值范围为：1，5，15，30，60，“day”，“all”。单位为分钟，取值为“day”时表示 24 小时，取值为“all”时表示统计的是生命周期内的所有度量值。

【响应举例】

```
{
  "metric_values": {
    "uid": "6bafa6fb-7c00-49bf-84c9-6bffa63b4e66"

  }
}
```

【返回码】

- 正确：OK (200)
- 错误：Bad Request (400)， Unauthorized (401)，Forbidden (403)，Bad Method (405)，Service Unavailable (503)

2.11.5 获取一级度量值

【方法】

获取指定应用程序注册的一级度量值：

GET /sdn/v2.0/metrics apps/{*app_id*}/primaries

【参数】

app_id：必选，表示应用程序 ID。

【响应举例】

```
{
  "primaries": [
    "nodes"
  ]
}
```

【返回码】

- 正确：OK (200)

- 错误：Bad Request (400)， Unauthorized (401)，Forbidden (403)，Bad Method (405)，Service Unavailable (503)

2.11.6 获取二级度量值

【方法】

获取指定应用程序注册的二级度量值：

GET /sdn/v2.0/metrics apps/{*app_id*}/ secondaries

【参数】

app_id：必选，表示应用程序 ID。

【响应举例】

```
{
  "secondaries": [
    "node"
  ]
}
```

2.11.7 获取度量名

【方法】

获取指定应用程序注册的度量名：

GET /sdn/v2.0/metrics apps/{*app_id*}/names

【参数】

app_id：必选，表示应用程序 ID。

【响应举例】

```
{
  "names": [
    "number_of_nodes"
  ]
}
```

3 /sdn/v2.0/of

3.1 统计信息/stats

3.1.1 获取控制器统计信息

【方法】

获取本控制器所属集群内所有控制器成员的统计信息：

GET /sdn/v2.0/of/stats

【响应举例】

```
{
  "controller_stats": [
    {
      "duration_ms": 20075292,
      "packet_in": {
        "packets": 0,
        "bytes": 0
      },
      "packet_out": {
        "packets": 1,
        "bytes": 70
      },
      "lost": {
        "packets": 0,
        "bytes": 0
      },
      "msg_in": 2091,
      "msg_out": 2083,
      "uid": "78ed6ea9-5bfc-46b4-a371-8dacccb02ce4"
    },
    {
      "duration_ms": 273008789,
      "packet_in": {
        "packets": 0,
        "bytes": 0
      },
      "packet_out": {
        "packets": 1,
        "bytes": 70
      },
      "lost": {
        "packets": 0,
        "bytes": 0
      },
      "msg_in": 2081,
      "msg_out": 2078,
      "uid": "6ecf003c-96f5-4dc8-935a-e7cad0f3d44f"
    },
    {
      "duration_ms": 270136743,
      "packet_in": {
        "packets": 0,
        "bytes": 0
      },
      "packet_out": {
        "packets": 2,
        "bytes": 142
      },
      "lost": {
        "packets": 0,
        "bytes": 0
      },
      "msg_in": 2086,
      "msg_out": 2080,
      "uid": "0e34e269-519d-4b98-ba2c-d453c8d863be"
    }
  ]
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Not Found (404)， Service Unavailable (503)

3.1.2 获取端口的统计信息

【方法】

获取所有端口的统计信息：

GET /sdn/v2.0/of/stats/ports

获取指定 Datapath ID 所有端口的统计信息：

GET /sdn/v2.0/of/stats/ports?dpid=" *dpid* **"**

获取指定端口的统计信息：

GET /sdn/v2.0/of/stats/ports?dpid=" *dpid***"&port="***port_id***"**

【参数】

dpid：可选，用于标识一台 OpenFlow 设备。取值范围为 0~FFFFFFFFFFFFFFFF，输入格式：HH:HH:HH:HH:HH:HH:HH:HH（十六进制），如果 DataPath ID 取值不足 16 位，用 0 补足位数，例如：DataPath ID 为 123456，输入格式为 00:00:00:00:00:12:34:56。

port_id：可选，表示端口 ID。

【响应举例】

```
{
  "stats": [
    {
      "dpid": "00:00:00:00:00:53:21:67",
      "version": "1.3.0",
      "port_stats": [
        {
          "port_id": 3,
          "rx_packets": 0,
          "tx_packets": 0,
          "rx_bytes": 0,
          "tx_bytes": 0,
          "rx_dropped": -1,
          "tx_dropped": -1,
          "rx_errors": 0,
          "tx_errors": 0,
          "collisions": 0,
          "duration_sec": 90,
          "duration_nsec": 4294967295,
          "rx_crc_err": 0,
          "rx_frame_err": 0,
          "rx_over_err": -1
        }
      ]
    }
  ]
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Not Found (404)， Service Unavailable (503)

3.1.3 获取组表项的统计信息

【方法】

获取所有组表项的统计信息：

GET / sdn/v2.0/stats/groups

获取指定 Datapath ID 所有组表项的统计信息：

GET / sdn/v2.0/stats/groups?dpid="*dpid* **"**

获取指定组表项的统计信息：

GET / sdn/v2.0/stats/groups?dpid="*dpid* **"&group_id="** *group_id***"**

【参数】

dpid：可选，用于标识一台 OpenFlow 设备。取值范围为 0~FFFFFFFFFFFFFFFF，输入格式：HH:HH:HH:HH:HH:HH:HH:HH（十六进制），如果 DataPath ID 取值不足 16 位，用 0 补足位数，例如：DataPath ID 为 123456，输入格式为 00:00:00:00:00:12:34:56。

group_id：可选，表示组表项 ID。

【响应举例】

```
{
  "version": "1.3.0",
  "group_stats": {
    "id": 1,
    "ref_count": 0,
    "packet_count": 0,
    "byte_count": 0,
    "duration_sec": 317,
    "duration_nsec": 4294967295,
    "bucket_stats": [
      {
        "packet_count": 0,
        "byte_count": 0
      },
      {
        "packet_count": 0,
        "byte_count": 0
      }
    ]
  }
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Not Found (404)， Service Unavailable (503)

3.1.4 获取Meter表的统计信息

【方法】

获取所有 Meter 表的统计信息：

GET / sdn/v2.0/of/stats/meters

获取指定 Datapath ID 的 Meter 表的统计信息：

GET / sdn/v2.0/of/stats/meters?dpid="*dpid* **"**

获取指定 Meter 表的统计信息：

GET/sdn/v2.0/of/stats/meters?dpid=" *dpid***" &meterid="** *metric_id***"**

【参数】

dpid：可选，用于标识一台 OpenFlow 设备。取值范围为 0~FFFFFFFFFFFFFFFF，输入格式：HH:HH:HH:HH:HH:HH:HH:HH（十六进制），如果 DataPath ID 取值不足 16 位，用 0 补足位数，例如：DataPath ID 为 123456，输入格式为 00:00:00:00:00:12:34:56。

metric_id：可选，表示度量 ID。

【响应举例】

获取所有 Meter 表的统计信息：

```
{
  "version": "1.3.0",
  "meter_stats": {
    "id": 7,
    "flow_count": 0,
    "packet_count": -1,
    "byte_count": -1,
    "duration_sec": 99,
    "duration_nsec": 0,
    "band_stats": [
      {
        "packet_count": -1,
        "byte_count": -1
        }
      ]
    }
  }
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Not Found (404)， Service Unavailable (503)

3.2 OpenFlow设备管理/datapaths

【方法】

获取所有 OpenFlow 设备信息：

GET / sdn/v2.0/of/datapaths

【响应举例】

```
{
  "datapaths": [
    {
      "dpid": "01:4d:74:25:8a:c4:e4:64",
      "negotiated_version": "1.3.0",
      "ready": "2013-11-08T05:47:26.564Z",
      "last_message": "2013-11-08T06:29:58.063Z",
      "num_buffers": 1024,
      "num_tables": 1,
      "device_ip": "192.168.56.161",
      "device_port": 59364,
      "description": "",
      "capabilities": [
        "flow_stats",
        "port_blocked",
        "queue_stats",
        "table_stats",
        "port_stats"
      ]
    }
  ]
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Not Found (404)， Service Unavailable (503)

3.2.1 获取指定OpenFlow设备信息

【方法】

获取指定 Datapath ID 的 OpenFlow 设备信息：

GET / sdn/v2.0/of/datapaths/{ *dpid* }

【参数】

dpid：必选，用于标识一台 OpenFlow 设备。取值范围为 0~FFFFFFFFFFFFFFFF，输入格式：HH:HH:HH:HH:HH:HH:HH:HH（十六进制），如果 DataPath ID 取值不足 16 位，用 0 补足位数，例如：DataPath ID 为 123456，输入格式为 00:00:00:00:00:12:34:56。

【响应举例】

```
{
  "datapath": {
    "dpid": "01:4d:74:25:8a:c4:e4:64",
    "negotiated_version": "1.3.0",
    "ready": "2013-11-08T05:47:04.742Z",
    "last_message": "2013-11-08T06:35:56.375Z",
    "num_buffers": 1024,
    "num_tables": 1,
    "device_ip": "192.168.56.161",
    "device_port": 59357,
    "description": "",
    "capabilities": [
      "flow_stats",
      "port_blocked",
      "queue_stats",
      "table_stats",
      "port_stats"
    ]
  }
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Not Found (404)， Service Unavailable (503)

3.2.2 获取指定OpenFlow设备连接的控制器信息

【方法】

获取指定 OpenFlow 设备连接的控制器 IP 地址和角色：

GET / sdn/v2.0/of/datapaths/{ *dpid* }/controllers

【参数】

dpid：必选，用于标识一台 OpenFlow 设备。

【响应举例】

```
{
  "controllers": {
    "master": "192.168.56.167",
    "slaves": [
      "192.168.56.169",
      "192.168.56.168"
    ]
  }
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Not Found (404)， Service Unavailable (503)

3.2.3 获取指定OpenFlow设备的Meter表能力集

【方法】

GET / sdn/v2.0/of/datapaths/{ *dpid* }/features/meter

【参数】

dpid：必选，用于标识一台 OpenFlow 设备。

【响应举例】

```
{
  "version": "1.3.0",
  "meter_features": {
    "max_meters": 512,
    "types": [
      "drop"
    ],
    "flags": [
      "kbps",
      "burst"
    ],
    "max_bands_per_meter": 1,
    "max_color_value": 2
  }
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Not Found (404)， Service Unavailable (503)

3.2.4 获取指定OpenFlow设备的组表能力集

【方法】

GET / sdn/v2.0/of/datapaths/{ *dpid* }/features/group

【参数】

dpid：必选，用于标识一台 OpenFlow 设备。

【响应举例】

```
{
  "version": "1.3.0",
  "group_features": {
    "capabilities": [],
    "types": [
      "all"
    ],
    "max_groups": [
      {
        "all": 1000
      }
    ],
    "actions": [
      {
        "all": [
          "output"
        ]
      }
    ]
  }
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Not Found (404)， Service Unavailable (503)

3.2.5 获取指定OpenFlow设备的所有端口信息

【方法】

GET / sdn/v2.0/of/datapaths/{ *dpid* }/ports。

【参数】

dpid：必选，用于标识一台 OpenFlow 设备。

【响应举例】

```
{
  "version": "1.3.0",
  "ports": [
    {
      "id": 3,
      "name": "XGE1/0/3",
      "mac": "74:25:8a:c4:e4:8f",
      "current_speed": 10000000,
      "max_speed": 10000000,
      "config": [],
      "state": [
        "link_down"
      ],
      "current_features": [
        "rate_10gb_fd"
      ],
      "advertised_features": [
        "rate_10gb_fd"
      ],
      "supported_features": [
        "rate_10gb_fd",
        "rate_other"
      ],
      "peer_features": []
    },
    {
      "id": 5,
      "name": "XGE1/0/5",
      "mac": "74:25:8a:c4:e4:91",
      "current_speed": 10000000,
      "max_speed": 10000000,
      "config": [],
      "state": [
        "link_down"
      ],
      "current_features": [
        "rate_10gb_fd"
      ],
      "advertised_features": [
        "rate_10gb_fd"
      ],
      "supported_features": [
        "rate_10gb_fd",
        "rate_other"
      ],
      "peer_features": []
    },
    {
      "id": 1746,
      "name": "Vlan333",
      "mac": "74:25:8a:c4:e4:7a",
      "config": [],
      "state": [
        "link_down"
      ],
      "current_features": [],
      "advertised_features": [],
      "supported_features": [],
      "peer_features": []
    },
    {
      "id": 4294967294,
      "name": "OFPP_LOCAL",
      "mac": "74:25:8a:c4:e4:64",
      "config": [],
      "state": [
        "live"
      ],
      "current_features": [],
      "advertised_features": [],
      "supported_features": [],
      "peer_features": []
    }
  ]
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)， Forbidden (403)，Not Found (404)， BadMethod (405)，Service Unavailable (503)

3.2.6 获取指定OpenFlow设备的指定端口信息

【方法】

GET / sdn/v2.0/datapaths/{ *dpid* }/ports/{*port id*}

【参数】

dpid：必选，用于标识一台 OpenFlow 设备。

port id：必选，表示端口 ID。

【响应举例】

```
{
  "version": "1.3.0",
  "port": {
    "id": 3,
    "name": "XGE1/0/3",
    "mac": "74:25:8a:c4:e4:8f",
    "current_speed": 10000000,
    "max_speed": 10000000,
    "config": [],
    "state": [
      "link_down"
    ],
    "current_features": [
      "rate_10gb_fd"
    ],
    "advertised_features": [
      "rate_10gb_fd"
    ],
    "supported_features": [
      "rate_10gb_fd",
      "rate_other"
    ],
    "peer_features": []
  }
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)， Forbidden (403)，Not Found (404)， Bad Method (405)，Service Unavailable (503)

3.2.7 开启指定OpenFlow设备的指定端口

【方法】

POST /sdn/v2.0/datapaths/{ *dpid* }/ports/{*port_id*}/action

【参数】

dpid：必选，用于标识一台 OpenFlow 设备。

port id：必选，表示端口 ID。

action：必选，取值为 enable。

【请求举例】

```
enable
```

【返回码】

- 正确：Accepted (202)
- 错误： Unauthorized (401)， Not Found (404)， Service Unavailable (503)

3.2.8 开启或关闭指定OpenFlow设备的指定端口

【方法】

POST /sdn/v2.0/datapaths/{ *dpid* }/ports/{*port_id*}/action

【参数】

dpid：必选，用于标识一台 OpenFlow 设备。

port id：必选，表示端口 ID。

action：必选，取值为 disable。

【请求举例】

```
disable
```

【返回码】

- 正确：Accepted (202)
- 错误： Unauthorized (401)， Not Found (404)， Service Unavailable (503)

3.2.9 获取指定OpenFlow设备的所有Meter表项信息

【方法】

GET /sdn/v2.0/of/datapaths/{ *dpid* }/meters

【参数】

dpid：必选，用于标识一台 OpenFlow 设备。

【响应举例】

```
{
  "version": "1.3.0",
  "meters": [
    {
      "id": 6,
      "flags": [
        "kbps"
      ],
      "bands": [
        {
          "burst_size": 400,
          "rate": 800,
          "mtype": "drop"
        }
      ]
    },
    {
      "id": 7,
      "flags": [
        "kbps"
      ],
      "bands": [
        {
          "burst_size": 400,
          "rate": 800,
          "mtype": "drop"
        }
      ]
    }
  ]
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Forbidden (403)， Not Found (404)，Bad Method (405)，Service Unavailable (503)

3.2.10 下发Meter表项

【方法】

POST /sdn/v2.0/of/datapaths/{ *dpid* }/meters

【参数】

dpid：必选，用于标识一台 OpenFlow 设备。

【请求举例】

```
{
  "version": "1.3.0",
  "meter":
  {
      "id": 7,
      "command": "add",
      "flags": [
      "kbps"
      ],
      "bands": [
      {
          "burst_size": 1000,
          "rate": 800,
          "mtype": "drop"
      }
      ]
  }
}
```

【响应举例】

```
{
  "version": "1.3.0",
  "meter": {
    "id": 7,
    "flags": [
      "kbps"
    ],
    "bands": [
      {
        "burst_size": 1000,
        "rate": 800,
        "mtype": "drop"
      }
    ]
  }
}
```

【返回码】

- 正确：Created (201)
- 错误： Unauthorized (401)，Forbidden (403)， Bad Method (405)， Service Unavailable (503)

3.2.11 获取指定OpenFlow设备的指定Meter表项信息

【方法】

GET /sdn/v2.0/of/datapaths/{ *dpid* }/meters/{*meter_id*}

【参数】

dpid：必选，用于标识一台 OpenFlow 设备。

meter_id：必选，表示 Meter 表 ID。

【响应举例】

```
{
  "version": "1.3.0",
  "meter": {
    "id": 6,
    "flags": [
      "kbps"
    ],
    "bands": [
      {
        "burst_size": 400,
        "rate": 800,
        "mtype": "drop"
      }
    ]
  }
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Forbidden (403)， Not Found (404)，Bad Method (405)，Service Unavailable (503)

3.2.12 更新Meter表项

【方法】

PUT /sdn/v2.0/of/datapaths/{ *dpid* }/meters/{*meter_id*}

【参数】

dpid：必选，用于标识一台 OpenFlow 设备。

meter_id：必选，表示 Meter 表 ID。

【请求举例】

```
{
  "version": "1.3.0",
  "meter":
  {

      "id": 6,
      "command": "modify",
      "flags": [
      "kbps"
      ],
      "bands": [
      {
          "burst_size": 1500,
          "rate": 1000,
          "mtype": "drop"
      }
      ]
  }
}
```

【响应举例】

```
{
  "version": "1.3.0",
  "meter": {
    "id": 6,
    "flags": [
      "kbps"
    ],
    "bands": [
      {
        "burst_size": 1500,
        "rate": 1000,
        "mtype": "drop"
      }
    ]
  }
}
```

【返回码】

- 正确：No Content (204)

- 错误： Unauthorized (401)，Forbidden (403)， Not Found (404)，Bad Method (405)，Service Unavailable (503)

3.2.13 删除Meter表项

【方法】

DELETE /sdn/v2.0/of/datapaths/{ *dpid* }/meters/{*meter_id*}

【参数】

dpid：必选，用于标识一台 OpenFlow 设备。

meter_id：必选，表示 Meter 表 ID。

【返回码】

- 正确：No Content (204)
- 错误： Unauthorized (401)，Forbidden (403)， Not Found (404)，Bad Method (405)，Service Unavailable (503)

3.2.14 获取流表项信息

【方法】

获取指定 OpenFlow 设备的所有流表项信息：

GET /sdn/v2.0/of/datapaths/{ *dpid* }/flows

获取指定流表的流表项信息：

GET /sdn/v2.0/of/datapaths/{ *dpid* }/flows?table_id =" *table_id*"

【参数】

dpid：必选，用于标识一台 OpenFlow 设备。

table_id：可选，表示流表 ID。

【响应举例】

```
{
  "version": "1.3.0",
  "flows": [
    {
      "table_id": 0,
      "duration_sec": 16,
      "duration_nsec": 4294967295,
      "priority": 35,
      "idle_timeout": 0,
      "hard_timeout": 0,
      "cookie": "0x1234",
      "packet_count": -1,
      "byte_count": -1,
      "match": [
        {
          "eth_type": "ipv4"
        },
        {
          "ip_proto": "tcp"
        },
        {
          "ipv4_src": "192.168.56.167",
          "mask": "255.255.255.0"
        },
        {
          "ipv4_dst": "192.168.56.168",
          "mask": "255.255.255.255"
        },
        {
          "tcp_dst": 81
        }
      ],
      "flow_mod_flags": [
        "send_flow_rem",
        "no_packet_counts",
        "no_byte_counts"
      ],
      "instructions": [
        {
          "write_actions": [
            {
              "output": 3
            }
          ]
        }
      ]
    },
    {
      "table_id": 0,
      "duration_sec": 8075,
      "duration_nsec": 4294967295,
      "priority": 0,
      "idle_timeout": 0,
      "hard_timeout": 0,
      "cookie": "0x0",
      "packet_count": 0,
      "byte_count": -1,
      "match": [],
      "flow_mod_flags": [
        "send_flow_rem"
      ],
      "instructions": [
        {
          "apply_actions": [
            {
              "output": 4294967293
            }
          ]
        }
      ]
    }
  ]
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Forbidden (403)， Not Found (404)，Bad Method (405)，Service Unavailable (503)

3.2.15 下发流表项

【方法】

POST /sdn/v2.0/of/datapaths/{ *dpid* }/flows

【参数】

dpid：必选，用于标识一台 OpenFlow 设备。

【请求举例】

```
{

    "version": "1.3.0",
    "flow": {
          "table_id": 0,
          "priority": 35,
          "idle_timeout": 0,
          "hard_timeout": 0,
          "flow_mod_cmd": "add",
          "cookie": "0x1234",
          "cookie_mask": "0xffff",
          "buffer_id":4294967295,
          "out_port": 3,
          "flow_mod_flags": [
              "send_flow_rem",
              "no_packet_counts",
              "no_byte_counts"
                                  ],
          "match": [
      {
          "eth_type":"ipv4"
      },
      {
          "ipv4_src": "192.168.56.167",
          "mask": "255.255.255.0"
      },
      {
          "ipv4_dst": "192.168.56.168",
          "mask": "255.255.255.255"
      },
      {
          "ip_proto":"tcp"
      },
      {
```

```
                "tcp_dst":81
            }
            ],
             "instructions":
             [
                        {
                            "write_actions": [
                                {
                                    "output":3
                                }
                            ]
                        }
             ]
          }
      }
```

【返回码】

- 正确：Created (201)
- 错误： Unauthorized (401)，Forbidden (403)，Bad Method (405)，Service Unavailable (503)

3.2.16 更新流表项

【方法】

PUT /sdn/v2.0/of/datapaths/{ *dpid* }/flow

【参数】

dpid：必选，用于标识一台 OpenFlow 设备。

【请求举例】

JSON 码中的 priority 字段、match 字段二者唯一标识一条流表项，如果没有匹配到指定流表项，则会返回一个 404 报错的消息。

```
{

    "version": "1.3.0",
    "flow": {
          "table_id": 0,
          "priority": 30,
          "idle_timeout": 0,
          "hard_timeout": 0,
          "flow_mod_cmd": "modify",
          "cookie": "0x1234",
          "cookie_mask": "0xffff",
          "buffer_id":4294967295,
          "out_port": 3,
          "flow_mod_flags": [
              "send_flow_rem",
              "no_packet_counts",
              "no_byte_counts"
```

```
                                ],
            "match": [
        {
            "eth_type":"ipv4"
        },
        {
            "ipv4_src": "192.168.56.167",
            "mask": "255.255.255.0"
        },
        {
            "ipv4_dst": "192.168.56.169",
            "mask": "255.255.255.255"
        },
        {
            "ip_proto":"tcp"
        },
        {
            "tcp_dst":81
        }
        ],
         "instructions":
         [
                    {
                        "write_actions": [
                            {
                                "output": 3
                            }
                        ]
                    }
         ]
      }
  }
```

【返回码】

- 正确：No Content (204)
- 错误： Unauthorized (401)， Forbidden (403)，Not Found (404)，Bad Method (405)，Service Unavailable (503)

3.2.17 删除流表项

【方法】

DELETE /sdn/v2.0/of/datapaths/{ *dpid* }/flows

【参数】

dpid：必选，用于标识一台 OpenFlow 设备。

【请求举例】

```
{"flow": { "priority": 30000,
"table_id": 200, "match": [
{"ipv4_src": "10.0.0.1"},
{"ipv4_dst": "10.0.0.22"},
{"ip_proto": "tcp"},
{"eth_type": "ipv4"},
69
{"tcp_dst": "80"}
]
}}
```

【返回码】

- 正确：No Content (204)
- 错误：Bad Request (400)， Unauthorized (401)， Not Found (404)， Service Unavailable (503)

3.2.18 获取指定OpenFlow设备的所有组表项信息

【方法】

GET /sdn/v2.0/of/datapaths/{ *dpid* }/groups

【参数】

dpid：必选，用于标识一台 OpenFlow 设备。

【响应举例】

```
{{
  "version": "1.3.0",
  "groups": [
    {
      "id": 1,
      "type": "all",
      "buckets": [
        {
          "weight": 0,
          "watch_group": 4294967295,
          "watch_port": 4294967295,
          "actions": [
            {
              "output": 3
            }
          ]
        },
        {
          "weight": 0,
          "watch_group": 4294967295,
          "watch_port": 4294967295,
          "actions": [
            {
              "output": 5
            }
          ]
        }
      ]
    }
  ]
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Forbidden (403)， Not Found (404)，Bad Method (405)，Service Unavailable (503)

3.2.19 下发组表项

【方法】

为指定 OpenFlow 设备下发组表项：

POST /sdn/v2.0/of/datapaths/{ *dpid* }/groups

【参数】

dpid：必选，用于标识一台 OpenFlow 设备。

【请求举例】

```
{
  "version" : "1.3.0",
  "group" :
  {
      "id" : 1,
      "type" : "all",
      "command" : "add",
      "buckets" :
      [{
          "weight" : 0,
          "watch_group" : 4294967295,
          "watch_port" : 4294967295,
          "actions" : [{
          "output" : 3
          }]
      },
      {
          "weight" : 0,
          "watch_group" : 4294967295,
          "watch_port" : 4294967295,
          "actions" : [{
          "output" : 5
          }]
      }]
  }
}
```

【响应举例】

```
{
  "version": "1.3.0",
  "group": {
    "id": 1,
    "type": "all",
    "buckets": [
      {
        "weight": 0,
        "watch_group": 4294967295,
        "watch_port": 4294967295,
        "actions": [
          {
            "output": 3
          }
        ]
      },
      {
        "weight": 0,
        "watch_group": 4294967295,
        "watch_port": 4294967295,
        "actions": [
          {
            "output": 5
          }
        ]
      }
    ]
  }
}
```

【返回码】

- 正确：Created (201)
- 错误： Unauthorized (401)， Forbidden (403)，Bad Method (405)，Service Unavailable (503)

3.2.20 获取组表项信息

【方法】

获取指定 OpenFlow 设备的指定组表项信息：

GET /sdn/v2.0/of/datapaths/{ *dpid* }/groups/{*group_id*}

【参数】

dpid：必选，用于标识一台 OpenFlow 设备。

group_id：必选，表示组表项 ID。

【响应举例】

```
{
  "version": "1.3.0",
  "group": {
    "id": 1,
    "type": "all",
    "buckets": [
      {
        "weight": 0,
        "watch_group": 4294967295,
        "watch_port": 4294967295,
        "actions": [
          {
            "output": 3
          }
        ]
      },
      {
        "weight": 0,
        "watch_group": 4294967295,
        "watch_port": 4294967295,
        "actions": [
          {
            "output": 5
          }
        ]
      }
    ]
  }
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Not Found (404)， Service Unavailable (503)

3.2.21 更新组表项

【方法】

PUT /sdn/v2.0/of/datapaths/{ *dpid* }/groups/{*group_id*}

【参数】

dpid：必选，用于标识一台 OpenFlow 设备。

group_id：必选，表示组表项 ID。

【请求举例】

```
{
  "version" : "1.3.0",
  "group" :
  {
      "id" : 1,
      "type" : "all",
      "command" : "modify",
      "buckets" :
      [{
          "weight" : 1,
          "watch_group" : 4294967295,
          "watch_port" : 4294967295,
          "actions" : [{
          "output" : 3
          }]
      },
      {
```

```
                "weight" : 1,
                "watch_group" : 4294967295,
                "watch_port" : 4294967295,
                "actions" : [{
                "output" : 5
                }]
            }]
        }
    }
```

【返回码】

- 正确：No content (204)
- 错误： Unauthorized (401)，Forbidden (403)，Not Found (404)， Bad Method (405)，Service Unavailable (503)

3.2.22 删除组表项

【方法】

删除指定 OpenFlow 设备的指定组表项：

DELETE /sdn/v2.0/of/datapaths/{ *dpid* **}/groups/{***group_id***}**

【参数】

dpid：必选，用于标识一台 OpenFlow 设备。

group_id：必选，表示组表项 ID。

【返回码】

- 正确：No Content (204)
- 错误： Unauthorized (401)，Not Found (404)， Service Unavailable (503)

4 /sdn/v2.0/net

4.1 OpenFlow网络域/Clusters

4.1.1 获取所有OpenFlow网络域信息

【方法】

GET /sdn/v2.0/net/clusters

【响应举例】

```
{
  "clusters": [
    {
      "uid": "1651313",
      "links": [
        {
          "src_dpid": "00:00:00:00:00:19:62:71",
          "src_port": 30,
          "dst_dpid": "00:00:00:00:00:19:32:71",
          "dst_port": 30
        },
        {
          "src_dpid": "00:00:00:00:00:19:62:71",
          "src_port": 48,
          "dst_dpid": "00:00:00:00:01:94:02:71",
          "dst_port": 48
        },
        {
          "src_dpid": "00:00:00:00:00:19:32:71",
          "src_port": 30,
          "dst_dpid": "00:00:00:00:00:19:62:71",
          "dst_port": 30
        },
        {
          "src_dpid": "00:00:00:00:01:94:02:71",
          "src_port": 48,
          "dst_dpid": "00:00:00:00:00:19:62:71",
          "dst_port": 48
        }
      ]
    }
  ]
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Forbidden (403)，Not Found (404)，Bad Method (405)， Service Unavailable (503)

4.1.2 获取指定OpenFlow 网络域信息

【方法】

GET /sdn/v2.0/net/clusters/{*cluster_uid*} /tree

【参数】

cluster_uid：必选，表示网络域标识符。

【响应举例】

```
{
  "cluster": {
    "uid": "1651313",
    "links": [
      {
        "src_dpid": "00:00:00:00:00:19:62:71",
        "src_port": 30,
        "dst_dpid": "00:00:00:00:00:19:32:71",
        "dst_port": 30
      },
      {
        "src_dpid": "00:00:00:00:00:19:62:71",
        "src_port": 48,
        "dst_dpid": "00:00:00:00:01:94:02:71",
        "dst_port": 48
      },
      {
        "src_dpid": "00:00:00:00:00:19:32:71",
        "src_port": 30,
        "dst_dpid": "00:00:00:00:00:19:62:71",
        "dst_port": 30
      },
      {
        "src_dpid": "00:00:00:00:01:94:02:71",
        "src_port": 48,
        "dst_dpid": "00:00:00:00:00:19:62:71",
        "dst_port": 48
      }
    ]
  }
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Forbidden (403)，Not Found (404)，Bad Method (405)， Service Unavailable (503)

4.2 链路/links

【方法】

获取所有链路信息：

GET /sdn/v2.0/net/links

获取与指定 OpenFlow 设备相连的所有链路信息（包括该 OpenFlow 设备作为源或目的的所有链路信息）：

GET /sdn/v2.0/net/links?dpid="*datapath_id*"

【参数】

datapath_id：可选，用于标识一台 OpenFlow 设备。

【响应举例】

```
{
  "links": [
    {
      "src_dpid": "44:44:44:44:11:11:11:11",
      "src_port": 224,
      "dst_dpid": "44:44:44:44:88:88:88:88",
      "dst_port": 345,
      "info": {
        "link_type": "directLink",
        "src_port_state": [
          "live"
        ],
        "dst_port_state": [
          "blocked",
          "live"
        ]
      }
    },
    {
      "src_dpid": "44:44:44:44:11:11:11:11",
      "src_port": 211,
      "dst_dpid": "00:00:00:00:00:00:00:40",
      "dst_port": 253,
      "info": {
        "link_type": "directLink",
        "src_port_state": [
          "live"
        ],
        "dst_port_state": [
          "live"
        ]
      }
    },
    {
      "src_dpid": "00:00:00:00:00:00:00:40",
      "src_port": 251,
      "dst_dpid": "44:44:44:44:11:11:11:11",
      "dst_port": 213,
      "info": {
        "link_type": "directLink",
        "src_port_state": [
          "live"
        ],
        "dst_port_state": [
          "live"
        ]
      }
    },
    {
      "src_dpid": "00:00:00:00:00:00:00:40",
      "src_port": 253,
      "dst_dpid": "44:44:44:44:11:11:11:11",
      "dst_port": 211,
      "info": {
        "link_type": "directLink",
        "src_port_state": [
          "live"
        ],
        "dst_port_state": [
          "live"
        ]
      }
    },
    {
      "src_dpid": "44:44:44:44:11:11:11:11",
      "src_port": 213,
      "dst_dpid": "00:00:00:00:00:00:00:40",
      "dst_port": 251,
      "info": {
        "link_type": "directLink",
        "src_port_state": [
          "live"
        ],
        "dst_port_state": [
          "live"
        ]
      }
    },
    {
      "src_dpid": "44:44:44:44:11:11:11:11",
      "src_port": 210,
      "dst_dpid": "44:44:44:44:22:22:22:22",
      "dst_port": 313,
      "info": {
        "link_type": "directLink",
        "src_port_state": [
          "live"
        ],
        "dst_port_state": [
          "blocked",
          "live"
        ]
      }
    }
  ]
}
```

4.3 转发路径/forward_path

【方法】

获取指定 OpenFlow 设备间的转发路径：

GET/sdn/v2.0/net/paths/forward?src_dpid="*src_dpid***"&dst_dpid="** *dst_dpid***"**

【参数】

src_dpid：必选，表示为源交换机的 datapath ID。

dst_dpid：必选，表示为目的交换机的 datapath ID。

【响应举例】

```
{
  "path": {
    "cost": 1,
    "links": [
      {
        "src_dpid": "44:44:44:44:11:11:11:11",
        "src_port": 211,
        "dst_dpid": "00:00:00:00:00:00:00:40",
        "dst_port": 253
      }
    ]
  }
}
```

【返回码】

- 正确：OK (200)
- 错误：illigalArgument (400)， Unauthorized (401)，Forbidden (403)，Not Found (404)，Service Unavailable (503)

4.4 ARP信息/arps

【方法】

获取学习到的所有主机的 ARP 信息：

GET /sdn/v2.0/net/arps

获取指定 VLAN 内学习到的所有主机的 ARP 信息：

GET /sdn/v2.0/net/arps?vid="*vlan-id***"**

获取学习到的关于指定主机 IP 地址的所有 ARP 信息：

GET /sdn/v2.0/net/arps?ip="*ip-address***"&vid="***vlan-id***"**

【参数】

vlan-id：可选，指定 VLAN 的 ID。

ip-address：可选，表示 IP 地址标识。

【响应举例】

```
{
  "arps": [
    {
      "ip": "10.0.0.3",
      "mac": "a2:c0:98:8e:ec:4a",
      "vid": 3
    }
  ]
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)， Not Found (404)， Service Unavailable (503)

4.5 节点/nodes

【方法】

获取所有学习到的主机信息：

GET /sdn/v2.0/net/nodes

获取指定 VLAN 内学习到的所有主机信息：

GET /sdn/v2.0/net/nodes?vid="*vlan-id***"**

获取指定 VLAN 内指定 IP 地址的主机信息：

GET /sdn/v2.0/net/nodes?ip="*ip_adress***"&vid="***vlan-id***"**

获取学习到的所有与指定 OpenFlow 设备相连的主机信息：

GET /sdn/v2.0/net/nodes?dpid="*dpid***"**

获取学习到的所有与指定端口相连的主机信息：

GET /sdn/v2.0/net/nodes?dpid="*dpid***"&port="***pord-id***"**

获取指定 VLAN 内指定 MAC 地址的主机信息：

GET /sdn/v2.0/net/nodes?vid="*vlan-id***"&mac="***MAC-id***"**

【参数】

dpid：可选，用于标识一台 OpenFlow 设备。

vlan-id：可选，指定 VLAN 的 ID。

pord-id：可选，表示端口 ID。

ip_adress：可选，表示 IP 地址标识。

MAC-id：可选，表示 MAC 地址标识。

【响应举例】

```
{
  "nodes": [
    {
      "ip": "20.20.20.20",
      "mac": "20:20:20:20:20:20",
      "vid": 300,
      "dpid": "01:2c:74:25:8a:d8:12:b1",
      "port": 28
    }
  ]
}
```

【返回码】

- 正确：OK (200)
- 错误：Bad Request (400)， Unauthorized (401)， Not Found (404)， Service Unavailable (503)

4.6 LLDP管理/lldp

4.6.1 获取所有LLDP抑制端口

【方法】

GET /sdn/v2.0/net/lldp

【响应举例】

```
{
  "lldp_suppressed": [
    {
      "dpid": "01:4d:74:25:8a:c4:e4:64",
      "ports": [
        3,
        5
      ]
    }
  ]
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)， Not Found (404)， Service Unavailable (503)

4.6.2 添加端口到LLDP抑制端口列表

【方法】

在 LLDP 抑制端口列表中添加端口（批量添加）：

POST /sdn/v2.0/net/lldp

【请求举例】

```
{"lldp_suppressed": [{
  "dpid": "01:4d:74:25:8a:c4:e4:64",
  "ports": [ 3, 5]
```

```
 }]}
```

【响应举例】

```
{
  "lldp_suppressed": [
    {
      "dpid": "01:4d:74:25:8a:c4:e4:64",
      "ports": [
        3,
        5
      ]
    }
  ]
}
```

【返回码】

- 正确：Created (201)
- 错误：Bad Request (400)，Unauthorized (401)，Item NotFound (404)，Service Unavailable (503)

4.6.3 在LLDP禁止端口列表中删除端口

【方法】

在 LLDP 禁止端口列表中删除端口（可批量删除）：

DELETE /sdn/v2.0/net/lldp

【响应举例】

```
{"lldp_suppressed": [{
   "dpid": "00:00:00:00:00:00:00:02",
   "ports": [ 3, 5, 7 ]
 }]}
```

【返回码】

- 正确：No Content(204)
- 错误： Unauthorized (401)，Item NotFound (404)， Service Unavailable (503)

4.7 诊断/diag

4.7.1 获取监测站信息

【方法】

获取所有监测站信息：

GET /sdn/v2.0/diag/observations

获取指定报文标识符的所有监测站信息：

GET /sdn/v2.0/diag/observations?packet_uid="*packet_uid***"**

获取指定报文类型的监测站信息：

GET /sdn/v2.0/diag/observations?packet_type=" *packet_type* **"**

【参数】

packet_uid：可选，表示报文标识符。

packet_type：可选，表示报文类型。

【响应举例】

获取所有监测站信息：

```
{
  "observations": [
    {
      "packet_uid": "1440680226",
      "type": "UDP",
      "dpid": "00:00:00:00:00:19:32:71"
    }
  ]
}
```

【返回码】

- 正确：OK (200)
- 错误：Bad Request (400)， Unauthorized (401)，Forbidden (403)， Bad Method (405)，Service Unavailable (503)

4.7.2 创建一个监测站

【方法】

POST /sdn/v2.0/diag/observations

【请求举例】

```
{"observation": {
  "dpid": "00:00:00:00:00:19:32:71",
  "packet_uid": "1440680226"
  }
}
```

【响应举例】

```
{
  "observation": {
    "dpid": "00:00:00:00:00:19:32:71",
    "packet_uid": "1440680226"
  }
}
```

【返回码】

- 正确：Created (201)
- 错误：Bad Request (400)， Unauthorized (401)， Forbidden (403)， Bad Method (405)，Service Unavailable (503)， Item NotFound (404)

4.7.3 删除一个监测站

【方法】

删除一个指定 datapath ID 和报文标识符的监测站：

DELETE /sdn/v2.0/diag/observations

【请求举例】

```
{"observation": {
       "dpid": "00:00:00:00:00:00:00:01",
         "packet_uid": "1"
    }
}
```

【返回码】

- 正确：No Content (204)
- 错误： Unauthorized (401)，Service Unavailable (503)

4.7.4 获取报文信息

【方法】

获取所有报文信息：

GET /sdn/v2.0/diag/packets

获取指定类型的报文信息：

GET /sdn/v2.0/diag/packets?type=" *packet_type* **"**

【参数】

packet_type：可选，表示报文类型。

【响应举例】

```
{
  "packets": [
    {
    {
      "uid": "1440680226",
      "eth": {
        "eth_type": "0x0800(IPv4)",
        "eth_src": "40:40:40:40:40:40",
        "eth_dst": "50:50:50:50:50:50",
        "vlan_vid": "271",
        "vlan_pcp": "PRIORITY_1"
      },
      "ip": {
        "ip_proto": "UDP",
        "ipv4_src": "80.40.40.40",
        "ipv4_dst": "90.50.50.50",
        "ip_ident": 0,
        "ip_dscp": "CS0",
        "ip_ecn": "NOT_ECT"
      },
      "udp": {
        "udp_src": 12345,
        "udp_dst": 152
      }
    }
  ]
}
```

【返回码】

- 正确：OK (200)
- 错误：Bad Request (400)， Unauthorized (401)，Forbidden (403)， Bad Method (405)，Service Unavailable (503)

4.7.5 创建报文

【方法】

POST /sdn/v2.0/diag/packets

【请求举例】

```
{"packet": {
  "type": "UDP",
  "eth": {

  "eth_src" : "40:40:40:40:40:40",
  "eth_dst" : "50:50:50:50:50:50",
  "eth_type": "IPv4",
  "vlan_vid" : "271",
  "vlan_priority" : "PRIORITY_5"
        },
        "ip":  {
      "ipv4_dst":  "90.50.50.50",
      "ipv4_src":  "80.40.40.40",
      "ip_proto":  "UDP",
          "ip_dscn": "CS0",
          "ip_scn": "NOT_ECT"
  },
  "udp": {
  "udp_dst": 152,
  "udp_src": 12345
  }
}
```
```
  }
```

【响应举例】

```
{
  "packet": {
    "uid": "1491536671",
    "eth": {
      "eth_type": "0x0800(IPv4)",
      "eth_src": "40:40:40:40:40:40",
      "eth_dst": "50:50:50:50:50:50",
      "vlan_vid": "271",
      "vlan_pcp": "PRIORITY_1"
    },
    "ip": {
      "ip_proto": "UDP",
      "ipv4_src": "80.40.40.40",
      "ipv4_dst": "90.50.50.50",
      "ip_ident": 0,
      "ip_dscp": "CS0",
      "ip_ecn": "NOT_ECT"
    },
    "tcp": {
      "tcp_src": 12345,
      "tcp_dst": 152
    }
  }
}
```

【返回码】

- 正确：Created (201)
- 错误：Bad Request (400)， Unauthorized (401)，Forbidden (403)， Bad Method (405)，Service Unavailable (503)，Item NotFound (404)

4.7.6 获取指定报文标识符的报文

【方法】

GET /sdn/v2.0/diag/packets/{*packet_uid*}

【参数】

packet_uid：必选，表示报文标识符。

【响应举例】

```
{
  "packet": {
    "uid": "1440680226",
    "eth": {
      "eth_type": "0x0800(IPv4)",
      "eth_src": "40:40:40:40:40:40",
      "eth_dst": "50:50:50:50:50:50",
      "vlan_vid": "271",
      "vlan_pcp": "PRIORITY_1"
    },
    "ip": {
      "ip_proto": "UDP",
      "ipv4_src": "80.40.40.40",
      "ipv4_dst": "90.50.50.50",
      "ip_ident": 0,
      "ip_dscp": "CS0",
      "ip_ecn": "NOT_ECT"
    },
    "udp": {
      "udp_src": 12345,
      "udp_dst": 152
    }
  }
}
```

【返回码】

- 正确：OK (200)
- 错误：Bad Request (400)， Unauthorized (401)，Forbidden (403)， Bad Method (405)，Service Unavailable (503)，Item NotFound (404)

4.7.7 删除指定报文标识符的报文

【方法】

DELETE /sdn/v2.0/diag/packets/ {*packets_id*}

【参数】

packets_id：必选，表示报文 ID。

【返回码】

- 正确：No Content (204)
- 错误： Unauthorized (401)， Service Unavailable (503)

4.7.8 获取报文传输时所依次经过的链路信息

【方法】

GET /sdn/v2.0/diag/packets/ {*packets_id*} /path

【参数】

packets_id：必选，表示报文 ID。

【响应举例】

起始设备通过报文的源 MAC 地址标识。

终点设备通过报文的目的 MAC 地址标识。

```
{
  "paths": [
    {
      "src_dpid": "00:00:00:00:00:19:32:71",
      "dst_dpid": "00:00:00:00:00:19:62:71",
      "src_port": "0x1e",
      "dst_port": "0x1e"
    },
    {
      "src_dpid": "00:00:00:00:00:19:62:71",
      "dst_dpid": "00:00:00:00:01:94:02:71",
      "src_port": "0x30",
      "dst_port": "0x30"
    }
  ]
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Not Found (404)， Service Unavailable (503)

4.7.9 获取下一跳交换机信息

【方法】

GET /sdn/v2.0/diag/packets/1/nexthop?src_dpid=" *src_ dpid* **"**

【参数】

src_dpid：必选，表示为源交换机的 datapath ID。

【响应举例】

```
{
  "nexthops": [
    {
      "dpid": "00:00:00:00:01:94:02:71",
      "port": "0x30"
    }
  ]
}
```

【返回码】

- 正确：OK (200)
- 错误： Unauthorized (401)，Not Found (404)， Service Unavailable (503)

4.7.10 模拟报文

【方法】

在网络上模拟发送已创建的报文，模拟时可以指定报文传输的起始设备：

POST /sdn/v2.0/diag/packets/ {*packets_id***}/action**

【参数】

packets_id：必选，已创建报文的 ID。

【响应举例】

缺省无需输入 JSON 码，默认在与主机节点直接相连的交换机上模拟发送报文。

通过如下 JSON 码可以指定模拟发送报文的起始设备。

```
{"simulation": {
  "dpid": "00:00:00:00:00:19:32:71",
  "out_port": "30"
  }
}
```

【返回码】

- 正确：OK (200)
- 错误：Bad Request (400)， Unauthorized (401)，Forbidden (403)，Ite mNotFound (404)，Bad Method (405)， Service Unavailable (503)