INTERNET-DRAFT
Intended Status: Standards Track

Mingui Zhang
Yuan Gao
Haibo Wang
Bing Liu
Huawei
Fu Qiao
China Mobile

Expires: September 9, 2019
March 8, 2019

Base YANG Data Model for NVO3 Protocols
draft-zhang-nvo3-yang-cfg-05.txt

Abstract

This document describes the base YANG data model that can be used by operators to configure and manage Network Virtualization Overlay protocols. The model is focused on the common configuration requirement of various encapsulation options, such as VXLAN, NVGRE, GENEVE and VXLAN-GPE. Using this model as a starting point, incremental work can be done to satisfy the requirement of a specific encapsulation.

Table of Contents

1. Introduction

Network Virtualization Overlays (NVO3), such as VXLAN, NVGRE, GENEVE and VXLAN-GPE, enable network virtualization for data center networks environment that assumes an IP-based underlay.

YANG [RFC6020] is a data definition language that was introduced to define the contents of a conceptual data store that allows networked devices to be managed using NETCONF [RFC6241]. This document specifies a YANG data model that can be used to configure and manage NVO3 protocols. The model covers the configuration of NVO3 instances as well as their operation states, which are the basic common requirements of the different tunnel encapsulations. Thus it is called "the base model for NVO3" in this document.

As the Network Virtualization Overlay evolves, newly defined tunnel encapsulation may require extra configuration. For example, GENEVE may require configuration of TLVs at the NVE. The base model can be augmented to accommodate these new solutions.

2. Acronyms and Terminology

2.1. Acronyms

NVO3: Network Virtualization Overlays
VNI: Virtual Network Instance
BUM: Broadcast, Unknown Unicast, Multicast traffic

BUM: Broadcast, Unknown Unicast, Multicast traffic

2.2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Familiarity with [RFC7348], [RFC7364], [RFC7365] and [RFC8014] is assumed in this document.

3. The YANG Data Model for NVO3

The NVO3 base YANG model is divided in three containers. The first container contains writable parameters. The second container contains the writable enablers per VNI for the statistical operational states as well as the status of these enablers. The third container contains the statistical operational states.

3.1. The Configuration Parameters

3.1.1. NVE ID

The 'nves' list contains NVEs under the management. The NVE is identified using the 'srcAddr', which is the underlay IP address of the NVE.

3.1.2. Virtual Network Instance

A Virtual Network Instance ('VNI') is a specific VN instance on an NVE [RFC7365]. At each NVE, a Tenant System is connect to VNIs through Virtual Access Points (VAP). VAPs can be physical ports or virtual ports identified by the bridge domain Identifier ('bdId'). The mapping between VNI and bdId is managed by the operator.

3.1.3. Flags in the Header

Flags in the NVO3 header are to be configured. For VxLAN, the I bit of 'flag' MUST be set to 1 while other 7 bits are reserved and MUST be set to zero on transmission and ignored on receipt. For NVGRE, bits in position 0 and 3 MUST be set to zero. The bit in position 2 MUST be set to 1. Bits from position 4 through 12 are reserved and MUST be set to zero on transmission and ignored on receipt.

3.1.4. BUM Mode

An NVE SHOULD support either ingress replication, or multicast proxy, or point to multipoint tunnels on a per-VNI basis. It is possible that both modes be used simultaneously in one NVO3 network by different NVEs.

If ingress replication is used, the receiver addresses are listed in 'peers'. If multicast proxy is used, the proxy's address is given in "flood-proxy". If the choice is point to multipoint tunnels, the multicast address is given as 'multiAddr'.

3.2. Statistics

Operators can determine whether a NVE should gather statistic values on a per-VNI base. The enablers are contained in the 'nvo3Info' list as 'statisticsEnable' leaf.

If the gathering for a VNI is enabled, the statistical information about the local NVEs, the remote NVEs, the flows and the MAC addresses will be collected by the NVEs in this VNI.

3.3. Model Structure

```
module: ietf-nvo3-base
+--rw nov3
   +--rw common
   |  +--rw nvo3-enable?   boolean
   +--rw nves
   |  +--rw nve* [if-name source-interface]
   |     +--rw if-name                       ifName
   |     +--rw vtep-ip?                      inet:ipv4-address-no-zone
   |     +--rw ipv6-vtep-ip?                 inet:ipv6-address-no-zone
   |     +--rw source-interface              ifName
   |     +--rw mac-address?                  yang:mac-address
   |     +--rw bypass-vtep-ip?               inet:ipv4-address-no-zone
   |     +--rw vni-members
   |     |  +--rw vni-member* [vni-id]
   |     |     +--rw vni-id          uint32
   |     |     +--rw protocol-bgp?   protocolType
   |     |     +--rw peers
   |     |     |  +--rw peer* [peer-ip]
   |     |     |     +--rw peer-ip          inet:ipv4-address-no-zone
   |     |     |     +--rw out-vni-id?            uint32
   |     |     |     +--rw split-horizon-group?   string
   |     |     +--rw ipv6-peers
   |     |     |  +--rw ipv6-peer* [ipv6-peer-ip]
   |     |     |     +--rw ipv6-peer-ip     inet:ipv6-address-no-zone
   |     |     +--rw flood-proxys
   |     |     |  +--rw flood-proxy* [peer-ip]
   |     |     |     +--rw peer-ip     inet:ipv4-address-no-zone
   |     |     +--rw mcast-group* [mcast-group]
   |     |        +--rw mcast-group    inet:ipv4-address-no-zone
   |     +--rw vni-map-l2vpns
   |     |  +--rw vni-map-l2vpn* [vni-id]
   |     |     +--rw l2vpn-id               uint32
   |     |     +--rw l2vpn-name?            string
   |     |     +--rw vni-id                 uint32
   |     |     +--rw split-horizon-group?   string
   |     +--rw vni-map-l3vpns
   |     |  +--rw vni-map-l3vpn* [l3vpn-name]
   |     |     +--rw l3vpn-name    vrfName
   |     |     +--rw vni-id?       uint32
   |     +--ro vni-statistics-infos
   |     |  +--ro vni-statistic-info* [vni-id]
   |     |     +--ro vni-id            uint32
   |     |     +--ro vni-statistics
   |     |        +--ro rx-bits-persec?        uint64
   |     |        +--ro rx-pkts-persec?        uint64
   |     |        +--ro tx-bits-persec?        uint64
   |     |        +--ro tx-pkts-persec?        uint64
   |     |        +--ro rx-pkts?               uint64
```

```
       |      |        +--ro rx-bytes?              uint64
       |      |        +--ro tx-pkts?               uint64
       |      |        +--ro tx-bytes?              uint64
       |      |        +--ro rx-unicast-pkts?       uint64
       |      |        +--ro rx-multicast-pkts?     uint64
       |      |        +--ro rx-broadcast-pkts?     uint64
       |      |        +--ro drop-unicast-pkts?     uint64
       |      |        +--ro drop-multicast-pkts?   uint64
       |      |        +--ro drop-broadcast-pkts?   uint64
       |      |        +--ro tx-unicast-pkts?       uint64
       |      |        +--ro tx-multicast-pkts?     uint64
       |      |        +--ro tx-broadcast-pkts?     uint64
       |      +--rw vnipeer-statistics-cfgs
       |      |  +--rw vnipeer-satistics-cfg* [vni-id peer-ip]
       |      |     +--rw vni-id     uint32
       |      |     +--rw peer-ip    inet:ipv4-address-no-zone
       |      +--ro vnipeer-statistics-infos
       |      |  +--ro vnipeer-satistics-info* [vni-id source-ip peer-ip]
       |      |     +--ro vni-id                uint32
       |      |     +--ro source-ip             inet:ipv4-address-no-zone
       |      |     +--ro peer-ip               inet:ipv4-address-no-zone
       |      |     +--ro vnipeer_statistics
       |      |        +--ro rx-bits-persec?        uint64
       |      |        +--ro rx-pkts-persec?        uint64
       |      |        +--ro tx-bits-persec?        uint64
       |      |        +--ro tx-pkts-persec?        uint64
       |      |        +--ro rx-pkts?               uint64
       |      |        +--ro rx-bytes?              uint64
       |      |        +--ro tx-pkts?               uint64
       |      |        +--ro tx-bytes?              uint64
       |      |        +--ro rx-unicast-pkts?       uint64
       |      |        +--ro rx-multicast-pkts?     uint64
       |      |        +--ro rx-broadcast-pkts?     uint64
       |      |        +--ro drop-unicast-pkts?     uint64
       |      |        +--ro drop-multicast-pkts?   uint64
       |      |        +--ro drop-broadcast-pkts?   uint64
       |      |        +--ro tx-unicast-pkts?       uint64
       |      |        +--ro tx-multicast-pkts?     uint64
       |      |        +--ro tx-broadcast-pkts?     uint64
       |      +--rw ipv6-vnipeer-statistics-cfgs
       |      |  +--rw ipv6-vnipeer-statistics-cfg* [vni-id ipv6-source-ip ipv6-pe
er-ip]
       |      |     +--rw vni-id            uint32
       |      |     +--rw ipv6-source-ip    inet:ipv6-address-no-zone
       |      |     +--rw ipv6-peer-ip      inet:ipv6-address-no-zone
       |      +--ro ipv6-vnipeer-statistics-infos
       |         +--ro ipv6-vnipeer-statistics-info* [vniId ipv6-source-ip ipv6-pe
er-ip]
       |            +--ro vniId                      uint32
       |            +--ro ipv6-source-ip            inet:ipv6-address-no-zone
```

```
|           +--ro ipv6-peer-ip               inet:ipv6-address-no-zone
|           +--ro ipv6_vnipeer_statistics
|              +--ro rx-bits-persec?        uint64
|              +--ro rx-pkts-persec?        uint64
|              +--ro tx-bits-persec?        uint64
|              +--ro tx-pkts-persec?        uint64
|              +--ro rx-pkts?               uint64
|              +--ro rx-bytes?              uint64
|              +--ro tx-pkts?               uint64
|              +--ro tx-bytes?              uint64
|              +--ro rx-unicast-pkts?       uint64
|              +--ro rx-multicast-pkts?     uint64
|              +--ro rx-broadcast-pkts?     uint64
|              +--ro drop-unicast-pkts?     uint64
|              +--ro drop-multicast-pkts?   uint64
|              +--ro drop-broadcast-pkts?   uint64
|              +--ro tx-unicast-pkts?       uint64
|              +--ro tx-multicast-pkts?     uint64
|              +--ro tx-broadcast-pkts?     uint64
|        +--ro vnipeer-infos
|        |  +--ro vnipeer-info* [vni-id source-ip peer-ip]
|        |     +--ro vni-id        uint32
|        |     +--ro source-ip     inet:ip-address-no-zone
|        |     +--ro peer-ip       inet:ip-address-no-zone
|        |     +--ro type?         peerType
|        |     +--ro out-vni-id?   uint32
|        +--rw vni-infos
|        |  +--rw vni_info* [vni-id]
|        |     +--rw vni-id               uint32
|        |     +--rw statistics-enable?   vniStatisticsEnable
|        |     +--ro status?              vniStatus
|        +--ro tunnel-infos
|           +--ro tunnel-infos* [tunnel-id]
|              +--ro tunnel-id    uint32
|              +--ro source-ip?   inet:ip-address-no-zone
|              +--ro peer-ip?     inet:ip-address-no-zone
|              +--ro status?      tunnelStatus
|              +--ro type?        tunnelType
|              +--ro up-time?     string
|              +--ro vrf-name?    vrfName
```

Figure 3.1. The tree structure of YANG module for NVO3 configuration

3.4. YANG Module

<CODE BEGINS> file "ietf-nvo3-base@2019-03-01.yang"

```
   module ietf-nvo3-base {
     namespace "urn:ietf:params:xml:ns:yang:ietf-nvo3-base";
     //namespace need to be assigned by IANA
     prefix "nvo3";
     import ietf-inet-types {
       prefix "inet";
     }
     import ietf-yang-types {
       prefix yang;
     }

     organization "IETF NVO3 Working Group";
     contact "zhangmingui@huawei.com
       sean.gao@huawei.com";
     description "nvo3 yang module";
     revision "2019-03-01" {
       description
         "Initial version";
       reference "RFC 7348 RFC 7637";
     }
     typedef vrfName {
       type string {
         length "1..31";
       }
       description
         "vrfName is a string type with length constraints";
     }
     typedef ifName {
       type string {
         length "1..63";
       }
       description
         "ifName is a string type with length constraints";
     }
     typedef vniStatus {
       type enumeration {
         enum up {
           description
                 "The VNI is up";
         }
         enum down {
           description
             "The VNI is down";
         }
       }
       description
         "The status of a VNI can be either up or down";
```

```
    }
    typedef protocolType {
      type enumeration {
        enum null {
          description
            "No specific protocol is used";
        }
        enum evpn {
          description
                "EVPN is used as the control plane protocol";
        }
      }
      description
        "The protocol type being used as control plane";
    }
    typedef peerType {
      type enumeration {
        enum static {
          description
                "Static peer NVE";
        }
        enum dynamic {
          description
                "Dynamic peer NVE";
        }
      }
      description
        "The type of the peer NVE";
    }
    typedef tunnelStatus {
      type enumeration {
        enum up {
          description
            "The tunnel is up";
         }
         enum down {
          description
            "The tunnel is down";
        }
      }
      description
        "The status of a tunnel can be either up or down";
    }
    typedef vniStatisticsEnable {
      type enumeration {
        enum enable {
          description
            "The statistics is enabled";
```

```
          }
          enum disable {
           description
             "The statistics is disabled";
          }
        }
        description
              "The statistics of a VNI can be enabled or disabled";
      }
      typedef tunnelType {
        type enumeration {
          enum dynamic {
            description
              "The tunnel is built in a dynamic way";
          }
          enum static {
           description
             "The tunnel is built by static configuration";
          }
        }
        description
          "The type of a tunnel";
      }
      container nvo3{
        description
          "NVO3 base YANG main part";
        container common {
          description
            "Common part";
          leaf nvo3-enable {
            type boolean;
            default "false";
            description
              "Enable/Disable NVO3 featrues";
        }
      }
        container nves {
          description
            "Parameters of NVEs";
          list nve {
            key "if-name source-interface";
            description
                  "The list of NVEs";
            leaf if-name {
              type ifName;
              description
                "nve interface name";
            }
```

```
            leaf vtep-ip {
              type inet:ipv4-address-no-zone;
              description
                "NVO3 tunnel source address";
            }
            leaf ipv6-vtep-ip {
              type inet:ipv6-address-no-zone;
              description
                "ipv6 NVO3 tunnel source address";
            }
            leaf source-interface {
              type ifName;
              description
                "source interface";
            }
            leaf mac-address {
              type yang:mac-address;
              description
                "mac address";
            }
            leaf bypass-vtep-ip {
              type inet:ipv4-address-no-zone;
              description
                "bypass NVO3 tunnel source address";
            }
            container vni-members {
                  description
                    "The VNI members on the NVE";
              list vni-member {
                key "vni-id";
                description
                  "nve vni member";
                leaf vni-id {
                  type uint32 {
                    range "1..16777215";
                  }
                  description
                    "Associate NVO3 VNIs (Virtual Network Identifiers) with the N
VE interface.";
                }
                leaf protocol-bgp {
                  type protocolType;
                  default "null";
                  description
                    "Enables BGP EVPN with ingress replication for the VNI.";
                }
                container peers {
                  description
                        "The peers in this VNI";
```

```
      list peer {
        key "peer-ip";
        description
                  "A peer NVE in this VNI";
        leaf peer-ip {
          type inet:ipv4-address-no-zone;
          description
            "peer ip address";
        }
        leaf out-vni-id {
          type uint32 {
            range "1..16777215";
          }
          description
            "out vni id";
        }
        leaf split-horizon-group {
          type string {
            length "1..31";
          }
          description
            "split group name";
        }
      }
    }
    container ipv6-peers {
      description
        "The IPv6 peers in this VNI";
      list ipv6-peer {
        key "ipv6-peer-ip";
          description
                "An IPv6 peer NVE in this VNI";
        leaf ipv6-peer-ip {
          type inet:ipv6-address-no-zone;
          description
            "peer ipv6 address";
        }
      }
    }
    container flood-proxys {
          description
            "The flood proxys for this VNI";
      list flood-proxy {
        key "peer-ip";
        leaf peer-ip {
          type inet:ipv4-address-no-zone;
          description
            "peer ip address";
```

```
                }
                        description
                          "List of the flood proxys";
              }
            }
            list mcast-group {
              key "mcast-group";
                  description
                    "The multicast groups in this VNI";
              leaf mcast-group {
                type inet:ipv4-address-no-zone;
                description
                  "mcast ip address";
              }
            }
          }
        }
           container vni-map-l2vpns {
         description
           "Mapping VNIs to L2VPNs";
           list vni-map-l2vpn {
             key "vni-id";
             description
               "L2VPN";
             leaf l2vpn-id {
               type uint32 {
                 range "1..16777215";
               }
               mandatory true;
               description
                 "l2vpn id";
             }
             leaf l2vpn-name {
               type string {
                 length "1..31";
               }
               description
                 "l2vpn name";
             }
             leaf vni-id {
               type uint32 {
                 range "1..16777215";
               }
               description
                 "vni id";
             }
             leaf split-horizon-group {
               type string {
```

```
                  length "1..31";
                }
                description
                  "split group name";
              }
            }
          }
          container vni-map-l3vpns {
            description
          "Mapping VNIs to L3VPNs";
            list vni-map-l3vpn {
              key "l3vpn-name";
              description
            "L3VPN";
              leaf l3vpn-name {
                type vrfName;
                description
                  "l3vpn name";
              }
              leaf vni-id {
                type uint32 {
                  range "1..16777215";
                }
                description
                  "vni id";
              }
            }
          }
        container vni-statistics-infos {
            config false;
            description
              "The statistics information for VNIs";
            list vni-statistic-info {
              key "vni-id";
              config false;
              description
                "The statistics information for a VNI";
              leaf vni-id {
                type uint32 {
                  range "1..16777215";
                }
                config false;
                description
                  "vni id";
              }
              container vni-statistics {
                config false;
                description
```

```
      "The statistics information items for a VNI";
    leaf rx-bits-persec {
      type uint64;
      config false;
          description
        "Received bits per second";
    }
    leaf rx-pkts-persec {
      type uint64;
      config false;
          description
        "Received packets per second";
    }
    leaf tx-bits-persec {
      type uint64;
      config false;
          description
        "Transmitted bits per second";
    }
    leaf tx-pkts-persec {
      type uint64;
      config false;
          description
        "Transmitted packets per second";
    }
    leaf rx-pkts {
      type uint64;
      config false;
          description
        "Received packets";
    }
    leaf rx-bytes {
      type uint64;
      config false;
          description
        "Received bytes";
    }
    leaf tx-pkts {
      type uint64;
      config false;
                     description
        "Transmitted packets";
    }
    leaf tx-bytes {
      type uint64;
      config false;
                     description
        "Transmitted bytes";
```

```
                }
                leaf rx-unicast-pkts {
                  type uint64;
                  config false;
                                  description
                    "Received unicast packets";
                }
                leaf rx-multicast-pkts {
                  type uint64;
                  config false;
                                  description
                    "Received multicast packets";
                }
                leaf rx-broadcast-pkts {
                  type uint64;
                  config false;
                                  description
                    "Received broadcast packets";
                }
                leaf drop-unicast-pkts {
                  type uint64;
                  config false;
                                  description
                    "Dropped unicast packets";
                }
                leaf drop-multicast-pkts {
                  type uint64;
                  config false;
                                  description
                    "Dropped multicast packets";
                }
                leaf drop-broadcast-pkts {
                  type uint64;
                  config false;
                                  description
                    "Dropped broadcast packets";
                }
                leaf tx-unicast-pkts {
                  type uint64;
                  config false;
                                  description
                    "Transmitted unicast packets";
                }
                leaf tx-multicast-pkts {
                  type uint64;
                  config false;
                                  description
                    "Transmitted multicast packets";
```

```
            }
            leaf tx-broadcast-pkts {
              type uint64;
              config false;
                              description
                "Transmitted broadcast packets";
            }
          }
        }
      }
      container vnipeer-statistics-cfgs {
        description
          "Statistics configuration of peers in the VNI.";
                    list vnipeer-satistics-cfg {
          key "vni-id peer-ip";
                     description
            "Statistics configuration of a peer in the VNI.";
          leaf vni-id {
            type uint32 {
              range "1..16777215";
            }
            description
              "vni id";
          }
          leaf peer-ip {
            type inet:ipv4-address-no-zone;
                        description
                    "The Ipv4 address of the peer";
          }
        }
      }
      container vnipeer-statistics-infos {
        config false;
                description
          "Statistics configuration of the peers in a VNI";
        list vnipeer-satistics-info {
          key "vni-id source-ip peer-ip";
          config false;
                      description
            "Statistics information of a peer in the VNI.";
          leaf vni-id {
            type uint32 {
              range "0..16777215";
            }
            config false;
            description
              "vni id";
          }
```

```
                leaf source-ip {
                  type inet:ipv4-address-no-zone;
                  config false;
                          description
                    "Source address";
                }
                leaf peer-ip {
                  type inet:ipv4-address-no-zone;
                  config false;
                          description
                    "Address of the peer";
                }
                container vnipeer_statistics {
                  config false;
                              description
                                    "Statistics information items of a peer in th
e VNI";
                  leaf rx-bits-persec {
                    type uint64;
                    config false;
                                 description
                      "Received bits per second";
                    }
                    leaf rx-pkts-persec {
                      type uint64;
                      config false;
                                     description
                        "Received packets per second";
                    }
                    leaf tx-bits-persec {
                      type uint64;
                      config false;
                                     description
                        "Transmitted bits per second";
                    }
                    leaf tx-pkts-persec {
                      type uint64;
                      config false;
                                     description
                        "Transmitted packets per second";
                    }
                    leaf rx-pkts {
                      type uint64;
                      config false;
                                     description
                        "Received packets";
                    }
                    leaf rx-bytes {
                      type uint64;
```

```
        config false;
                        description
          "Received bytes";
      }
      leaf tx-pkts {
        type uint64;
        config false;
                        description
          "Transmitted packets";
      }
      leaf tx-bytes {
        type uint64;
        config false;
                        description
          "Transmitted bytes";
      }
      leaf rx-unicast-pkts {
        type uint64;
        config false;
                        description
          "Received unicast packets";
      }
      leaf rx-multicast-pkts {
        type uint64;
        config false;
                        description
          "Received multicast packets";
      }
      leaf rx-broadcast-pkts {
        type uint64;
        config false;
                        description
          "Received broadcast packets";
      }
      leaf drop-unicast-pkts {
        type uint64;
        config false;
                        description
          "Dropped unicast packets";
      }
      leaf drop-multicast-pkts {
        type uint64;
        config false;
                        description
          "Dropped multicast packets";
      }
      leaf drop-broadcast-pkts {
        type uint64;
```

```
                config false;
                                description
                  "Dropped broadcast packets";
                }
                leaf tx-unicast-pkts {
                  type uint64;
                  config false;
                                description
                    "Transmitted unicast packets";
                }
                leaf tx-multicast-pkts {
                  type uint64;
                  config false;
                                description
                    "Transmitted multicast packets";
                }
                leaf tx-broadcast-pkts {
                  type uint64;
                  config false;
                                description
                    "Transmitted broadcast packets";
                }
              }
            }
          }
          container ipv6-vnipeer-statistics-cfgs {
                    description
              "Statistics configuration of IPv6 peers in the VNI.";
            list ipv6-vnipeer-statistics-cfg {
              key "vni-id ipv6-source-ip ipv6-peer-ip";
                          description
                "Statistics configuration of an IPv6 peer in the VNI.";
              leaf vni-id {
                type uint32 {
                  range "1..16777215";
                }
                description
                  "vni id";
              }
              leaf ipv6-source-ip {
                type inet:ipv6-address-no-zone;
                            description
                  "Source IPv6 address";
              }
              leaf ipv6-peer-ip {
                type inet:ipv6-address-no-zone;
                        description
                  "IPv6 address of the peer";
```

```
        }
      }
    }
    container ipv6-vnipeer-statistics-infos {
      config false;
      description
        "Statistics information of IPv6 peers in the VNI.";
      list ipv6-vnipeer-statistics-info {
        key "vniId ipv6-source-ip ipv6-peer-ip";
        config false;
                    description
          "Statistics information of an IPv6 peer in the VNI.";
        leaf vniId {
          type uint32 {
            range "1..16777215";
          }
          config false;
          description
            "vni id";
        }
        leaf ipv6-source-ip {
          type inet:ipv6-address-no-zone;
          config false;
                      description
            "Source IPv6 address";
        }
        leaf ipv6-peer-ip {
          type inet:ipv6-address-no-zone;
          config false;
                      description
            "IPv6 address of the peer";
        }
        container ipv6_vnipeer_statistics {
          config false;
                      description
            "Statistics information items of an IPv6 peer in the VNI.";
          leaf rx-bits-persec {
            type uint64;
            config false;
                        description
              "Received bits per second";
          }
          leaf rx-pkts-persec {
            type uint64;
            config false;
                           description
              "Received packets per second";
          }
```

```
                leaf tx-bits-persec {
                  type uint64;
                  config false;
                                  description
                    "Transmitted bits per second";
                }
                leaf tx-pkts-persec {
                  type uint64;
                  config false;
                                  description
                    "Transmitted packets per second";
                }
                leaf rx-pkts {
                  type uint64;
                  config false;
                                  description
                    "Received packets";
                }
                leaf rx-bytes {
                  type uint64;
                  config false;
                                  description
                    "Received bytes";
                }
                leaf tx-pkts {
                  type uint64;
                  config false;
                                  description
                    "Transmitted packets";
                }
                leaf tx-bytes {
                  type uint64;
                  config false;
                                  description
                    "Transmitted bytes";
                }
                leaf rx-unicast-pkts {
                  type uint64;
                  config false;
                                  description
                    "Received unicast packets";
                }
                leaf rx-multicast-pkts {
                  type uint64;
                  config false;
                                  description
                    "Received multicast packets";
                }
```

```
                leaf rx-broadcast-pkts {
                  type uint64;
                  config false;
                                  description
                    "Received broadcast packets";
                }
                leaf drop-unicast-pkts {
                  type uint64;
                  config false;
                                  description
                    "Dropped unicast packets";
                }
                leaf drop-multicast-pkts {
                  type uint64;
                  config false;
                                  description
                    "Dropped multicast packets";
                }
                leaf drop-broadcast-pkts {
                  type uint64;
                  config false;
                                  description
                    "Dropped broadcast packets";
                }
                leaf tx-unicast-pkts {
                  type uint64;
                  config false;
                                  description
                    "Transmitted unicast packets";
                }
                leaf tx-multicast-pkts {
                  type uint64;
                  config false;
                                  description
                    "Transmitted multicast packets";
                }
                leaf tx-broadcast-pkts {
                  type uint64;
                  config false;
                                  description
                    "Transmitted broadcast packets";
                }
              }
            }
          }
        }
      }
      container vnipeer-infos {
```

```
        config false;
            description
          "Information of the peers";
        list vnipeer-info {
          key "vni-id source-ip peer-ip";
          config false;
                  description
          "Information of a peer";
          leaf vni-id {
            type uint32 {
              range "1..16777215";
            }
            config false;
            description
              "vni-id";
          }
          leaf source-ip {
            type inet:ip-address-no-zone;
            config false;
            description
              "source ip";
          }
          leaf peer-ip {
            type inet:ip-address-no-zone;
            config false;
            description
              "peer ip";
          }
          leaf type {
            type peerType;
            config false;
            description
              "peer type";
          }
          leaf out-vni-id {
            type uint32 {
              range "1..16777215";
            }
            config false;
            description
              "out vni id";
          }
        }
      }
      container vni-infos {
            description
          "Information of the VNIs";
        list vni_info {
```

```
          key "vni-id";
                  description
                    "Information of a VNI";
          leaf vni-id {
            type uint32 {
              range "1..16777215";
            }
            description
              "vni id";
          }
          leaf statistics-enable {
            type vniStatisticsEnable;
            default "disable";
                    description
                          "Statistics is enabled or not";
          }
          leaf status {
            type vniStatus;
            config false;
                    description
                          "The status of the VNI";
          }
        }
      }
      container tunnel-infos {
        config false;
            description
                  "Information of tunnels";
        list tunnel-infos {
          key "tunnel-id";
          config false;
                  description
                    "Information of a tunnel";
          leaf tunnel-id {
            type uint32 {
              range "1..4294967295";
            }
            config false;
            description
              "tunnel id";
          }
          leaf source-ip {
            type inet:ip-address-no-zone;
            config false;
            description
              "source";
          }
          leaf peer-ip {
```

```
            type inet:ip-address-no-zone;
            config false;
            description
              "peer ip";
          }
          leaf status {
            type tunnelStatus;
            config false;
            description
              "tunnel status";
          }
          leaf type {
            type tunnelType;
            config false;
            description
              "tunnel type";
          }
          leaf up-time {
            type string {
              length "1..10";
            }
            config false;
            description
              "tunnel up time";
          }
          leaf vrf-name {
            type vrfName;
            default "_public_";
            config false;
            description
              "vrf";
          }
        }
      }
    }
  }
  <CODE ENDS>
```

4. Security Considerations

This document raises no new security issues.

5. IANA Considerations

The namespace URI defined in Section 3.3 need be registered in the IETF XML registry [RFC3688].

This document need to register the 'ietf-nvo3-base' YANG module in

the YANG Module Names registry [RFC6020].

Acknowledgements

Authors would like to thank the comments and suggestions from Tao Han, Weilian Jiang.

6. References

6.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC7364] T. Narten, E. Gray, et al, "Problem Statement: Overlays for Network Virtualization", draft-ietf-nvo3-overlay-problem-statement, working in progress.

[RFC7365] Marc Lasserre, Florin Balus, et al, "Framework for DC Network Virtualization", draft-ietf-nvo3-framework, working in progress.

[RFC7348] Mahalingam, M., Dutt, D., Duda, K., Agarwal, P., Kreeger, L., Sridhar, T., Bursell, M., and C. Wright, "Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", RFC 7348, August 2014.

[I-D.ietf-nvo3-geneve] Gross, J., Ganga, I., and T. Sridhar, "Geneve: Generic Network Virtualization Encapsulation", draft-ietf-nvo3-geneve-10 (work in progress), March 2019.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

6.2. Informative References

[RFC7637] M. Sridharan, A. Greenberg, et al, "NVGRE: Network Virtualization using Generic Routing Encapsulation", RFC7637, September 2015.

[I-D.ietf-nvo3-vxlan-gpe] Maino, F., Kreeger, L., and U. Elzur,

"Generic Protocol Extension for VXLAN", draft-ietf-nvo3-vxlan-gpe-06 (work in progress), April 2018.

[RFC8014] D. Black, J. Hudson, L. Kreeger, M. Lasserre, T. Narten, An Architecture for Data-Center Network Virtualization over Layer 3 (NVO3), RFC8014, December 2016.

[I-D.ietf-nvo3-encap-02] Boutros, S., "NVO3 Encapsulation Considerations", draft-ietf-nvo3-encap-02 (work in progress), September, 2018.

Author's Addresses

Mingui Zhang
Huawei Technologies
No. 156 Beiqing Rd. Haidian District,
Beijing 100095
P.R. China

EMail: zhangmingui@huawei.com

Yuan Gao
Huawei Technologies
No. 101 Nanjing Rd. Yuhua District,
Nanjing 210012
P.R. China
EMail: sean.gao@huawei.com

Haibo Wang
Huawei Technologies
No. 156 Beiqing Rd. Haidian District,
Beijing 100095
P.R. China
EMail: rainsword.wang@huawei.com

Bing Liu
Huawei Technologies
No. 156 Beiqing Rd. Haidian District,
Beijing 100095
P.R. China
EMail: remy.liubing@huawei.com

Fu Qiao
China Mobile

EMail: fuqiao@chinamobile.com