NVO3 WG
Internet-Draft
Intended status: Standards Track
Expires: March 1, 2019

Fangwei Hu
Ran Chen
ZTE Corporation
Mallik Mahalingam
Springpath
Qiang Zu
Ericsson
S. Davari
yahoo
Xufeng Liu
Volta Networks

August 28, 2018

YANG Data Model for VxLAN Protocol
draft-chen-nvo3-vxlan-yang-07.txt

Abstract

This document defines a YANG data model for VxLAN protocol.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on March 1, 2019.

Table of Contents

1. Introduction 2
2. Terminology 2
3. Requirements Language 2
4. YANG Data Model for VxLAN Configuration 2
4.1. VxLAN Multicast IP Address 2
4.2. VxLAN Access Type 3
4.3. Inner VLAN Tag Handling Mode 3
5. Design Tree of VxLAN YANG Data Model 3
6. VxLAN YANG Model 5
7. Security Considerations 17
8. Acknowledgements 18
9. IANA Considerations 18
10. Normative References 19
Authors' Addresses 20

1. Introduction

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 defines a YANG data model for the configuration of VxLAN protocol [RFC7348].

2. Terminology

3. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

4. YANG Data Model for VxLAN Configuration

4.1. VxLAN Multicast IP Address

The vxlan-multicast-ip is used to configure the IP multicast group, which the VxLAN VNI of the VTEP is mapping to. Both the IPv4 and IPv6 address family are supported in this document.

4.2. VxLAN Access Type

There are several access types supported for VxLAN:

- vlan-1:1: the vxlan access type is VLAN, and each VxLAN is only mapping to one VLAN.
- vlan- n:1: the vxlan access type is VLAN, and each VxLAN is mapped to several VLANs.
- L3-interface: the VxLAN access type is layer 3 interface.
- mac: the VxLAN access type is MAC address.
- vlan-l2-interface: the VxLAN access type is VLAN plus Layer 2 interface.

4.3. Inner VLAN Tag Handling Mode

There are two handling modes for the inner VLAN tag: discard-inner-vlan mode and no-discard-inner-vlan mode. If the VTEP interface works in the discard-inner-vlan mode, the VxLAN is only mapped to one VLAN. The inner VLAN tag will be stripped when encapsulating the VxLAN frame. On the decapsulation side, if VTEP receives the VxLAN frame with inner VLAN tag, it will discard the frame in this work mode. If the VTEP receives the VxLAN frame without VLAN tag, it will fill in the VLAN tag based on the VxLAN and VLAN mapping entry.

If the VTEP interface works in the no-discard-inner-vlan mode, the VxLAN could be mapped to several VLANs. The inner VLAN tag will not stripped when encapsulating the VxLAN frame in the VxLAN encapsulation side. On the decapsulation side, if VTEP receives the VxLAN frame, it will strip the VxLAN header, and keep the VLAN frame.

5. Design Tree of VxLAN YANG Data Model

```
module: ietf-vxlan
    +--rw vxlan
    |  +--rw global-enable?                empty
    |  +--rw vxlan-instance* [vxlan-id]
    |     +--rw vxlan-id                   vxlan-id
    |     +--rw description?               string
    |     +--rw unknow-unicast-drop?       enumeration
    |     +--rw filter-vrrp?               enumeration
    |     +--rw (vxlan-access-types)? {vxlan-access-types}?
    |     |  +--:(access-type-vlan)
    |     |  |  +--rw access-type-vlan?          access-type-vlan
    |     |  |  +--rw access-vlan-list* [vlan-id]
```

```
|     |  |     +--rw vlan-id    vlan
|     |  +--:(access-type-mac)
|     |  |  +--rw access-type-mac?           empty
|     |  |  +--rw mac                        yang:mac-address
|     |  +--:(access-type-l2interface)
|     |  |  +--rw access-type-l2interface?   empty
|     |  |  +--rw vlan-id                    vlan
|     |  |  +--rw interface-name             if:interface-ref
|     |  +--:(access-type-l3interface)
|     |     +--rw access-type-l3interface?   empty
|     |     +--rw map-l3interface* [interface-name]
|     |        +--rw interface-name    if:interface-ref
|     +--rw vtep-instances* [vtep-id]
|     |  +--rw vtep-id                    uint32
|     |  +--rw vtep-name?                 string
|     |  +--rw source-interface?          if:interface-ref
|     |  +--rw multicast-ip               inet:ip-address
|     |  +--rw mtu?                       uint32 {mtu}?
|     |  +--rw inner-vlan-handling-mode?   inner-vlan-handling-mode
|     |  +--rw bind-vxlan-id* [vxlan-id]
|     |     +--rw vxlan-id    vxlan-id
|     +--rw static-vxlan-tunnel* [vxlan-tunnel-id]
|     |  +--rw vxlan-tunnel-id      uint32
|     |  +--rw vxlan-tunnel-name?   string
|     |  +--rw address-family* [af]
|     |     +--rw af                      address-family-type
|     |     +--rw tunnel-source-ip?       inet:ip-address
|     |     +--rw tunnel-destination-ip?   inet:ip-address
|     |     +--rw bind-vxlan-id* [vxlan-id]
|     |        +--rw vxlan-id    vxlan-id
|     +--rw redundancy-group-binds
|        +--rw redundancy-group-bind* [vxlan-id redundancy-group]
|           +--rw vxlan-id            uint32
|           +--rw redundancy-group    uint32
+--ro vxlan-state
   +--ro vxlan
      +--ro vxlan-tunnels
         +--ro vxlan-tunnel* [local-ip remote-ip]
            +--ro local-ip            inet:ip-address
            +--ro remote-ip           inet:ip-address
            +--ro static-tunnel-id?   uint32
            +--ro evpn-tunnel-id?     uint32
            +--ro statistics
               +--ro tunnel-statistics
               |  +--ro in-bytes?      string
               |  +--ro out-bytes?     string
               |  +--ro in-packets?    string
               |  +--ro out-packets?   string
```

```
                  +--ro tunnel-vni-statistics
                     +--ro tunnel-vni-statistic* [vxlan-id]
                        +--ro vxlan-id       uint32
                        +--ro in-bytes?      string
                        +--ro out-bytes?     string
                        +--ro in-packets?    string
                        +--ro out-packets?   string

  augment /evpn:evpn/evpn:evpn-instances/evpn:evpn-instance/evpn:bgp-
parameters/evpn:common:
    +--rw bgp-parameters
       +--rw common
          +--rw rd-rt* [route-distinguisher]
             +--rw route-distinguisher    string
             +--rw vpn-target* [rt-value]
                +--rw rt-value    string
                +--rw rt-type     bgp-rt-type
```

6. VxLAN YANG Model

```
<CODE BEGINS> file "ietf-vxlan@2018-08-29.yang"
module ietf-vxlan {
    namespace "urn:ietf:params:xml:ns:yang:ietf-vxlan";
    prefix "vxlan";

     import ietf-evpn {
       prefix "evpn";
     }

    import ietf-interfaces {
      prefix "if";
     }

    import ietf-inet-types {
       prefix "inet";
     }

    import ietf-yang-types {
       prefix yang;
     }

    organization
      "IETF NVO3(Network Virtualization Overlays) Working Group";

      contact
       "
```

```
         WG List:  <mailto:nvo3@ietf.org>

         WG Chair: Matthew Bocci
                  <mailto:matthew.bocci@alcatel-lucent.com>

         WG Chair: Benson Schliesser
                  <mailto:bensons@queuefull.net>

         Editor:   Fangwei Hu
                   <mailto:hu.fangwei@zte.com.cn>

         Editor:   Ran Chen
                   <mailto:chen.ran@zte.com.cn>

         Editor:   Mallik Mahalingam
                   <mailto:mallik_mahalingam@yahoo.com>

         Editor:   Zu Qiang
                   <mailto:Zu.Qiang@Ericsson.com>
        ";

    description
       "The YANG module defines a generic configuration
        model for VxLAN protocol";

    revision 2018-08-29 {
      description "Fixs some type error.";
      reference
         "draft-chen-nvo3-vxlan-yang-07";
    }

    revision 2018-01-03 {
      description "Changes the yang data model according to the NMDA style.";
      reference
         "draft-chen-nvo3-vxlan-yang-06";
    }

    revision 2017-06-29 {
      description "no changes.";
      reference
         "draft-chen-nvo3-vxlan-yang-05";
    }

    revision 2016-12-08 {
      description "updated the vxlan yang model based on the comments from IETF
97th meeting,"
       +"augmenting EVPN data model, adding access type configuration and MTU
configuration.";
      reference
```

"draft-chen-nvo3-vxlan-yang-04";

```
    }

    revision 2016-06-02 {
       description
         "03 revision. Update the YANG data model based on thec comments of
IETF 95th meeting.";
       reference
         "draft-chen-nvo3-vxlan-yang-03";
     }

    revision 2015-12-01 {
       description
         "02 revision.";
       reference
         "draft-chen-nvo3-vxlan-yang-02";
     }

    revision 2015-10-12 {
       description
         "01 revision.";
       reference
         "draft-chen-nvo3-vxlan-yang-01";
     }

    revision 2015-05-05 {
      description "Initial revision";
      reference
         "draft-chen-nvo3-vxlan-yang-00";
    }

    /* Feature  */

    feature vxlan-access-types {
       description
         "Support configuration vxlan access types.";
     }

    feature mtu {
        description
         "Support configuration vxlan MTU value.";
    }

    feature evpn-bgp-params {
       description "Support EVPN BGP parameter.";
      }

    /* Typedefs  */

    typedef vlan {
```

```
      type uint16 {
        range 1..4094;
      }
      description
      "Typedef for VLAN";
    }

    typedef vxlan-id {
      type uint32;
      description
        "Typedef for VxLAN ID.";
    }
      typedef access-type-vlan {
      type enumeration {
        enum access-type-vlan1to1 {
          description
            "Access type is VLAN 1:1.";
        }
        enum access-type-vlan1ton {
          description
            "Access type is VLAN 1:n.";
        }

      }
      default access-type-vlan1to1 ;
      description
        "VxLAN access type is VLAN.";
    }

    typedef access-type-mac {
       type empty ;
       description
         "VxLAN access type is MAC.";
    }


    typedef inner-vlan-handling-mode {
       type enumeration {
         enum discard-inner-vlan {
              description
                "Discard inner-VLAN.";
         }
         enum no-discard-inner-vlan {
              description
                "No discard inner-VLAN.";
         }
       }
       default discard-inner-vlan ;
```

```
       description
        "Typedef for inner-vlan-handling-mode";
    }

    typedef address-family-type {
       type enumeration {
         enum ipv4 {
           description
             "IPv4";
         }
         enum ipv6 {
           description
             "IPv6";
         }
       }
       description
         "Typedef for address family type.";
     }

    /* Configuration Data */

      container vxlan{
        leaf global-enable {
          type empty ;
          description 'VXLAN global enble.';
        }

        list vxlan-instance {
           key vxlan-id ;
           leaf vxlan-id {
             type vxlan-id;
             description "VxLAN ID.";
           }

           leaf description {
            type string {
              length 0..64 {
                description 'VXLAN instance description information.';
              }
            }
            description 'The description information of VXLAN instance.';
           }

           leaf unknow-unicast-drop {
            type enumeration {
              enum enable {
                value 1 ;
                description 'Unknown unicast drop enable.';
```

```
            }
            enum disable {
              value 2 ;
              description 'Unknown unicast drop disable.';
            }
          }
          default enable ;
          description 'Unknow unicast drop configuration of VXLAN instance.';
        }

        leaf filter-vrrp {
          type enumeration {
            enum enable {
              value 1 ;
              description 'VRRP packets filter.';
            }
            enum disable {
              value 2 ;
              description 'VRRP packets not filter.';
            }
          }
          default enable ;
          description 'VRRP packets filter configuration of VXLAN instance.';
        }

        choice vxlan-access-types {
          if-feature  vxlan-access-types;
           case access-type-vlan {

             leaf access-type-vlan {
               type access-type-vlan;

               description
                  "Access type is VLAN.";
             }

             list access-vlan-list {
               key vlan-id ;
               leaf vlan-id {
                 type vlan;
                 description
                   "VLAN ID.";
               }
               description
                 "VLAN ID list." ;
             }
               description
                 "VxLAN access type choice is VLAN.";
```

```
            }

            case access-type-mac {
              leaf access-type-mac {
                type empty ;
                description
                  "Access type is MAC.";
              }

              leaf mac {
                  type yang:mac-address ;
                  mandatory true ;
                  description
                    "MAC Address.";
                }
              description
                "VxLAN access type choice is MAC Address.";
            }

            case access-type-l2interface {
              leaf access-type-l2interface {
                type empty ;
                description
                  "VXLAN map layer two interface.";
              }

              leaf vlan-id {
                  type vlan;
                  mandatory true ;
                  description
                    "VLAN ID.";
               }

              leaf interface-name {
                  type if:interface-ref;
                  mandatory true ;
                  description
                    "Layer two interface name.";
              }
              description
                "VxLAN access type choice is layer two interface.";
            }

            case access-type-l3interface {
              leaf access-type-l3interface {
                type empty ;
                description
                  "Access type of VxLAN is layer three interface.";
```

```
            }

            list map-l3interface {
              key interface-name ;
              leaf interface-name {
                type if:interface-ref;
                description
                  "Layer three interface name.";
              }
              description
                 "Layer three interface list.";
            }
            description
              "VxLAN access type choice is layer three interface.";
          }
          description
             "VxLAN access type choice.";
        }

      list vtep-instances {
          key vtep-id ;
          leaf vtep-id {
             type uint32;
             description
             "VTEP ID.";
          }

          leaf vtep-name{
             type string;
             description
              "VTEP instance name.";
          }

          leaf source-interface {
             type if:interface-ref;
             description
                   "Source interface name.";
          }

          leaf multicast-ip {
              type inet:ip-address;
              mandatory true ;
              description
                "VxLAN multicast IP address.";
          }

          leaf mtu {
            if-feature mtu;
```

```
              type uint32;
              description "vxlan mtu";
            }

            leaf inner-vlan-handling-mode {
                type inner-vlan-handling-mode;
                description
                   "The inner vlan tag handling mode.";
            }

            list bind-vxlan-id {
                key vxlan-id;
                leaf vxlan-id {
                  type vxlan-id;
                  description
                       "VxLAN ID.";
                }
                description
                  "VxLAN ID list for the VTEP.";
            }
            description
                "VTEP instance.";
        }

        list static-vxlan-tunnel{
            key vxlan-tunnel-id;
            leaf vxlan-tunnel-id {
               type uint32;
               description
                     "Static VxLAN tunnel ID.";
            }

            leaf vxlan-tunnel-name {
               type string;
               description
               "Name of the static VxLAN tunnel.";
            }

            list address-family {
               key "af";
               leaf af {
                 type address-family-type;
                 description
                 "Address family type value.";
               }

               leaf tunnel-source-ip {
                 type inet:ip-address;
```

```
                description
                "Source IP address for the static VxLAN tunnel";
              }

              leaf tunnel-destination-ip {
                 type inet:ip-address;
                 description
                  "Destination IP address for the static VxLAN tunnel";
              }

              list bind-vxlan-id {
                 key vxlan-id;
                 leaf vxlan-id {
                   type vxlan-id;
                   description
                      "VxLAN ID.";
                 }
                 description
                    "VxLAN ID list for the VTEP.";
              }

               description
                  "Per-af params.";
            }
           description
             "Configure the static VxLAN tunnel";
        }

        container redundancy-group-binds {
           list redundancy-group-bind {
              key 'vxlan-id redundancy-group';
              leaf vxlan-id {
                type uint32 {
                  range 1..16777215 {
                    description 'The value of VXLAN,it must between 1 to
16777215.';
                  }
                }
                description 'VXLAN ID binding by redundancy group.';
              }

              leaf redundancy-group {
                type uint32 {
                   range 1..4294967293 {
                      description 'The value of redundancy group,it must
between 1 to'
                           + ' 4294967293.';
                    }
```

```
                }
                description 'Redundancy group ID.';
```

```
            }
            description 'Redundancy group bind table.';
          }
          description 'Redundancy group bind table.';
        }
        description "vxlan instance list";
     }
     description
       "VxLAN configure model.";
    }


    augment "/evpn:evpn/evpn:evpn-instances/evpn:evpn-instance"
         +"/evpn:bgp-parameters/evpn:common" {

     uses evpn:bgp-parameters-grp {
       if-feature evpn-bgp-params;
      }
      description "EVPN configuration";
     }

      /* Operational  data */
    container vxlan-state{
      config false;
      container vxlan {
        container vxlan-tunnels {
          list vxlan-tunnel {
             key 'local-ip remote-ip';
             leaf local-ip {
               type inet:ip-address;
               description 'Local IP of tunnel.';
             }

             leaf remote-ip {
               type inet:ip-address;
               description 'Remote IP of tunnel.';
             }

             leaf static-tunnel-id {
               type uint32 ;
               description 'Static tunnel ID.';
             }

             leaf evpn-tunnel-id {
               type uint32 ;
               description 'EVPN tunnel ID.';
             }
```

```
            container statistics {
               container tunnel-statistics {
                 leaf in-bytes {
                    type string {
                      length 0..24 ;
                    }
                    description 'Total bytes received.';
                 }

                 leaf out-bytes {
                   type string {
                     length 0..24 ;
                   }
                   description 'Total bytes sent.';
                 }

                 leaf in-packets {
                   type string {
                      length 0..24;
                   }
                   description 'Total packets received.';
                 }

                 leaf out-packets {
                   type string {
                     length 0..24 ;
                   }
                   description 'Total packets sent.';
                 }
                 description 'Total tunnel statistics.';
               }

               container tunnel-vni-statistics {
                list tunnel-vni-statistic {
                 key vxlan-id ;
                 leaf vxlan-id {
                   type uint32 ;
                   description 'The VXLAN in tunnel.';
                 }

                 leaf in-bytes {
                   type string {
                     length 1..24 ;
                   }
                   description 'Total bytes received.';
                 }

                 leaf out-bytes {
```

```
                  type string {
                    length 1..24 ;
                  }
                  description 'Total bytes sent.';
                }

                leaf in-packets {
                  type string {
                    length 1..24 ;
                  }
                  description 'Total packets received.';
                }

                leaf out-packets {

                  type string {
                    length 1..24 ;
                  }
                  description 'Total packets sent.';
                }
                description 'Statistics in VXLAN tunnel.';
              }
              description 'Statistics in VXLAN tunnel.';
            }
             description 'Tunnel statistics.' ;
          }
            description 'VXLAN tunnel info.';
        }
          description 'VXLAN tunnel Info.';
      }
      description 'Information of VXLAN state.';
    }
    description 'Information of VXLAN state.';
  }
 }
<CODE ENDS>
```

7. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH)[RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC5246].

The NETCONF access control model [RFC6536] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

There are a number of data nodes defined in this YANG module that are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations.

The vulnerable "config true" parameters and subtree are the following:

ietf-vxlan/global-enable: this subtree specifies VxLAN enable switch. Modify the configuration can cause the VxLAN disable.

ietf-vxlan/vxlan-instance/static-vxlan-tunnel: this subtree specifies static VxLAN tunnel configuration. Modify the configuration can cause static VxLAN tunnel disconnection.

Unauthorized access to any of these lists can adversely affect the security of both the local device and the network. This may lead to network malfunctions, delivery of packets to inappropriate destinations, and other problems.

8. Acknowledgements

9. IANA Considerations

This document registers three URI in the IETF XML registry [RFC3688]. Following the format in [RFC3688], the following registrations are requested to be made.

urn:ietf:params:xml:ns:yang:ietf-vxlan.

Registrant Contact: The IESG.

XML: N/A, the requested URI is an XML namespace.

This document registers three YANG modules in the YANG Module Names registry [RFC6020].

```
name:       ietf-vxlan
namespace:  urn:ietf:params:xml:ns:yang:ietf-vxlan
prefix:     vxlan
reference:  RFC XXXX
```

10. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <https://www.rfc-editor.org/info/rfc5246>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <https://www.rfc-editor.org/info/rfc6242>.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <https://www.rfc-editor.org/info/rfc6536>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[RFC7223] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 7223, DOI 10.17487/RFC7223, May 2014, <https://www.rfc-editor.org/info/rfc7223>.

[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, DOI 10.17487/RFC7348, August 2014, <https://www.rfc-editor.org/info/rfc7348>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

Authors' Addresses

Fangwei Hu
ZTE Corporation
No.889 Bibo Rd
Shanghai 201203
China

Phone: +86 21 68896273
Email: hu.fangwei@zte.com.cn

Ran Chen
ZTE Corporation
No.50 Software Avenue,Yuhuatai District
Nanjing, Jiangsu Province 210012
China

Phone: +86 025 88014636
Email: chen.ran@zte.com.cn

Mallik Mahalingam
Springpath
640 W. California Ave, Suite #110
Sunnyvale, CA 94086
USA

Email: mallik_mahalingam@yahoo.com

Zu Qiang
Ericsson
8400, boul. Decarie
Ville Mont-Royal, QC
Canada

Email: Zu.Qiang@Ericsson.com

Davari Shahram
yahoo

Email: davarish@yahoo.com

Xufeng Liu
Volta Networks
USA

Email: xufeng.liu.ietf@gmail.com