Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: June 17, 2016

H. Shen
B. Liu, Ed.
Huawei Technologies
D. Bannister
M. Abrahamsson
T-Systems

December 15, 2015

A YANG Data Model for L2TPv3 Tunnel
draft-shen-l2tpext-l2tpv3-yang-model-01

Abstract

This document defines a YANG data model for managing L2TPv3 tunnels.

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 http://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 June 17, 2016.

Table of Contents

1. Introduction

This document defines a YANG [RFC6020] [RFC6021] data model for L2TPv3 tunnels.

2. Requirements Language and Terminology

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 [RFC2119] when they appear in ALL CAPS. When these words are not in ALL CAPS (such as "should" or "Should"), they have their usual English meanings, and are not to be interpreted as [RFC2119] key words.

Terminology:

o L2TPv3: Layer Two Tunneling Protocol - Version 3 [RFC3931]

3. L2TPv3 YANG Model Overview

The L2TPv3 YANG model mainly includes two objects. One (l2tpv3CtrlInstances) is for the L2TPv3 control plane configuration. The other one (l2tpv3TunnelInstances) is for managing the tunnels.

The overall structure of the model is dicpicted as the following.

```
   module: ietf-l2tpv3

      +--rw l2tpv3CtrlInstances
      |  +--rw l2tpv3CtrlInstance* [ctrlName]
      |     +-- rw ctrlName                      string
      |     +-- rw hostName                      string
      |     +-- rw routerID                      uint16
      |     +-- rw rcvWinSize?                   uint16
      |     +-- rw helloInterval?                uint16
      |     +-- rw digestType?                   enum
      |     +-- rw authenNonce?                  password
      +--rw l2tpv3TunnelInstances
         +--rw l2tpv3TunnelInstance* [tunnelName]
            +-- rw tunnelName                 string
            +-- rw sourceIfName               if:interface-ref
            +-- rw sourceIP                   inet:ip-address
            +-- rw destIP                     inet:ip-address
            +-- rw tunnelType                            enum
            |     +-- rw static:
            |     |   +-- rw localSessionId?               uint32
            |     |   +-- rw remoteSessionId?              uint32
            |     |   +-- rw localCookieAutoMode?          enum
            |     |   |   +-- rw authNone:
            |     |   |   +-- rw authPlain:
            |     |   |   +-- rw localCookieLength   enum
            |     |   |   +-- rw localHighCookie     hexBinary
            |     |   |   +-- rw localLowCookie      hexBinary
            |     |   |   +-- rw authCipher:
            |     |   |       +--rw localCookieCipher   password
            |     |   +-- rw remoteCookieAutoMode?         enum
            |     |       +-- rw authNone:
            |     |       +-- rw authPlain:
            |     |       +--rw remoteCookieLength enum
            |     |       +--rw remoteHighCookie   hexBinary
            |     |       +--rw remoteLowCookie    hexBinary
            |     |       +-- rw authCipher:
```

```
|     |           +--rw remoteCookieCipher password
|     +-- rw auto:
|           +-- rw ctrlName            string
|           +-- rw encapType           enum
+-- ro sendPacket                  uint64
+-- ro sendByte                    uint64
+-- ro rcvPacket                   uint64
+-- ro receiveByte                 uint64
+-- ro recvDropPacket              uint64
+-- ro cookieMisDropPacket         uint64
+-- ro state                       enum
```


3.1. l2tpv3CtrlInstance

The l2tpv3CtrlInstance container is a template used for configuring the control plane of L2TPv3 tunnels. The leaves under the container are the parameters of the control signaling datagram processing.

One l2tpv3CtrlInstance could be binding to a specific l2tpv3TunnelInstances through the key "ctrlName" defined in auto mode of the tunnel. One l2tpv3CtrlInstance could also be shared among multiple l2tpv3TunnelInstances.

3.2. l2tpv3TunnelInstances

This container is to manage the L2TPv3 tunnels. Two tunnel modes are supported: one is static tunnel, the other is automatic tunnel.

The basic information of a tunnle contains following elements:

- tunnelName: the identifier of the tunnel
- sourceIfName: the identifier of the loopback interface which is corresponding with the Persudo-Wire interface of the tunnel
- sourceIP: the IPv4/IPv6 address of the tunnel starting point
- destIP: the IPv4/IPv6 address of the tunnel ending point

The tunnelType node is to distinguish statically configured tunnels and dynamically configured tunnels. For static tunnels, the relevant

session and cookie information is included. For dynamic tunnels, only the corresponding control instance is referenced as a key there.

At the end, some stastic elements were defined to represent the running state of the tunnels.

4. L2TPv3 YANG Module

<CODE BEGINS>

```
<CODE BEGINS> file "ietf-l2tpv3@2015-12-15.yang"
module ietf-l2tpv3 {
        namespace "urn:ietf:params:xml:ns:yang:ietf-l2tpv3";
        prefix "l2tpv3";

    import ietf-interfaces {
        prefix if;
    }
/*
```



```
    import ietf-yang-types {
        prefix yang;
    }
*/
    import ietf-inet-types {
        prefix inet;
    }
        organization "ietf l2tpv3 working group";
        contact "shenhaoxing@huawei.com
                 leo.liubing@huawei.com";
        description "The module for implementing l2tpv3 protocol";
        revision 2015-12-15 {description "version-01, minor grammar revision to

        typedef hexBinary {
                type string {
                        length "1..127";
                        pattern "0[xX][0-9a-fA-F]+";
                }
        }

        typedef password {
                type string {
```

```
                length "1..127";
            }
        }

        container l2tpv3CtrlInstances {

            list l2tpv3CtrlInstance {

                key "ctrlName";
                min-elements "0";


                 leaf ctrlName {
                        config "true";
                        type "string"{
                                length "1..19";
                        }
                }
                leaf hostName {
                        config "true";
                        type "string";
                        mandatory "true";
                }
                leaf routerID {
                        config "true";
                        type "uint16";
```



```
                        mandatory "true";
                }
                leaf rcvWinSize {
                        config "true";
                        type "uint16";
                }
                leaf helloInterval {
                        config "true";
                        type "uint16";
                }
                leaf digestType{
                        config "true";
                                        type enumeration {
                                                enum "HMAC_MD5";
                                                enum "HMAC_SHA_1";
```

```
                                        }
                        }
                        leaf authenNonce{
                                config "true";
                                type password {
                                        length "1..16";
                                }
                        }
                }

        }
        container l2tpv3TunnelInstance {

                list l2tpv3TunnelInstance {

                        key "tunnelName";
                        min-elements "0";


                         leaf tunnelName {
                                config "true";
                                type "string"{
                                        length "1..19";
                                }
                        }
                        leaf sourceIfName {
                                config "true";
                type if:interface-ref;
                description
                "Interface name as defined by ietf-interfaces";

                        }
                        leaf sourceIP {
```



```
                                config "true";
                                mandatory "true";
                                type inet:ip-address;
                        }
                        leaf destIP {
                                config "true";
                                mandatory "true";
                                type inet:ip-address;
```

```
                    }
                    leaf tnlType {
                                    config "true";
                                    mandatory "true";
                                    type enumeration {
                                            enum "static";
                                            enum "auto";
                                    }
                    }
                choice tunnelType {
                            mandatory "true";

                    case static{
                                when "tnlType = 'static'";
                                    leaf localSessionId {
                                            config "true";
                                            default "4294967295";
                                            type uint32 {
                                                    range "1..4294967295";
                                            }
                                    }
                                    leaf remoteSessionId {
                                            config "true";
                                            default "4294967295";
                                            type uint32 {
                                                    range "1..4294967295";
                                            }
                                    }
                                    leaf localCookieAutoMode {
                                            config "true";
                                            mandatory "true";
                                            type enumeration {
                                                    enum "authNone";
                                                    enum "authPlain";
                                                    enum "authCipher";
                                            }
                                    }

                                    choice localCookieMode {
                                default authNone;
```



```
                                config true;
```

```
                case authNone {
                when "localCookieAutoMode = 'au

                }
                case authPlain {
                when "localCookieAutoMode = 'au
                        leaf localCookieLength
                                config "true";
                                default "4";
                                type enumeratio
                                        enum "4
                                        enum "8
                                }
                        }
                        leaf localHighCookie {
                                config "true";
                                type "hexBinary
                                    length "3..
                                }
                        }
                        leaf localLowCookie {
                                config "true";
                                type "hexBinary
                                    length "3..
                                }
                        }
                }
                case authCipher {
                when "localCookieAutoMode = 'au
                        leaf localCookieCipher
                                config "true";
                                type password {
                                        length
                                }
                        }
                }
        }
        leaf remoteCookieAutoMode {
                config "true";
                mandatory "true";
                type enumeration {
                        enum "authNone";
                        enum "authPlain";
                        enum "authCipher";
                }
        }
        choice remoteCookieMode {
```

```
                                        default authNone;
                                        config true;
                                        case authNone {
                    when "remoteCookieAutoMode = 'authNone'";
                                        }
                                        case authPlain {
                                        when "remoteCookieAutoMode = 'a
                                                leaf remoteCookieLength
                                                        config "true";
                                                        default "4";
                                                        type enumeratio
                                                                enum "4
                                                                enum "8
                                                        }
                                                }
                                                leaf remoteHighCookie {
                                                        config "true";
                                                        type "hexBinary
                                                            length "3..
                                                        }
                                                }
                                                leaf remoteLowCookie {
                                                        config "true";
                                                        type "hexBinary
                                                            length "3..
                                                        }
                                                }
                                        }
                                        case authCipher {
                                        when "remoteCookieAutoMode = 'a
                                                leaf remoteCookieCipher
                                                        config "true";
                                                        type password {
                                                                length
                                                        }
                                                }
                                        }
                                }
                }
                        case auto{
                    when "tnlType = 'auto'";
                                leaf ctrlName {
                                        config "true";
                                        type string{
                                                length "1..19";
                                        }
```

```
                                        mandatory "true";
                                    }
```



```
                                    leaf encapType {
                                        config "true";
                                        mandatory "true";
                                        type enumeration
                                        {
                                            enum "HDLC";
                                            enum "Ethernet";
                                            enum "VLAN";
                                            enum "ATM";
                                        }
                                    }


                            }
                    }


                        leaf sendPacket {
                            config "false";
                            type "uint64";
                        }
                        leaf sendByte {
                            config "false";
                            type "uint64";
                        }
                        leaf rcvPacket {
                            config "false";
                            type "uint64";
                        }
                        leaf receiveByte {
                            config "false";
                            type "uint64";
                        }
                        leaf recvDropPacket {
                            config "false";
                            type "uint64";
                        }
                        leaf cookieMisDropPacket {
                            config "false";
```

```
                        type "uint64";
                    }
                    leaf state {
                        config "false";
                        type enumeration {
                            enum "down" {
                                value "0";
                                description "down:";
                            }
```



```
                            enum "up" {
                                value "1";
                                description "up:";
                            }
                        }
                    }
                }
            }
}
```

<CODE ENDS>

5. Security Considerations

TBD.

6. IANA Considerations

This draft does not request any IANA action.

7. Acknowledgements

Gang Yan made significant contribution to design the YANG model. Valuable comment was received from Xianping Zhang to improve the draft.

This document was produced using the xml2rfc tool [RFC2629].

8. 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,
<http://www.rfc-editor.org/info/rfc2119>.

[RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629,
DOI 10.17487/RFC2629, June 1999,
<http://www.rfc-editor.org/info/rfc2629>.

[RFC3931] Lau, J., Ed., Townsley, M., Ed., and I. Goyret, Ed.,
"Layer Two Tunneling Protocol - Version 3 (L2TPv3)",
RFC 3931, DOI 10.17487/RFC3931, March 2005,
<http://www.rfc-editor.org/info/rfc3931>.

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

[RFC6021] Schoenwaelder, J., Ed., "Common YANG Data Types",
RFC 6021, DOI 10.17487/RFC6021, October 2010,
<http://www.rfc-editor.org/info/rfc6021>.

Authors' Addresses

Haoxing Shen
Huawei Technologies
Huawei Nanjing R&D Center
101 Software Avenue, Yuhua District, Nanjing, Jiangsu, 210012
P.R. China

Email: shenhaoxing@huawei.com

Bing Liu
Huawei Technologies
Q14, Huawei Campus, No.156 Beiqing Road
Hai-Dian District, Beijing, 100095
P.R. China

Email: leo.liubing@huawei.com

David Bannister

T-Systems

Email: David.Bannister@t-systems.com

Mikael Abrahamsson
T-Systems
Stockholm
Sweden

Email: mikael.abrahamsson@t-systems.se