Network Working Group
Internet-Draft
Intended status: Informational
Expires: February 25, 2015

S. Zhuang
H. Wang
Z. Li
Huawei Technologies

August 24, 2014

Yang Model for L2VPN
draft-zhuang-l2vpn-yang-cfg-00

Abstract

This document defines a YANG data model that can be used to configure and manage L2VPN. Both VPWS and VPLS are supported.

Requirements Language

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].

Status of This Memo

Copyright Notice

Table of Contents

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]. YANG is proving relevant beyond its initial confines, as bindings to other interfaces(e.g. ReST) and encoding other than XML (e.g. JSON) are being defined. Furthermore, YANG data models can be used as the basis of implementation for other interface, such as CLI and programmatic APIs.

This document defines a YANG data model that can be used to configure and manage L2VPN. Both VPWS and VPLS are supported.

2. Terminology

L2VPN: Layer 2 Virtual Private Network

VPLS: Virtual Private LAN Service

VPWS: Virtual Private Wire Service

3. Design of Data Model

3.1. Overview

The L2VPN Yang module is divided in following containers :

o l2vpncommon : that contains common writable configuration and readable objects for VPWS and VPLS.

o l2vpnvpws : that contains writable configuration and readable objects for VPWS.

o l2vpnvpls: that contains writable configuration and readable objects for VPLS.

The figure below describe the overall structure of the L2VPN Yang module :

```
module: l2vpn
   +--rw l2vpncommon
   ...
   +--rw l2vpnvpws
   ...
   +--rw l2vpnvpls
   ...
      ...
```

3.2. L2VPN Common Configuration

L2VPN common configuration container includes the global parameters for L2VPN, PW template configuration, etc. These parameters can be used by both VPWS and VPLS.

PW template configuration includes peer address, control word, MTU, sequence number, tunnel policy, parameters of AC, etc.

```
   +--rw l2vpncommon
   |  +--rw l2vpnGlobal
   |  |  +--rw l2vpnEnable             boolean
   |  |  +--rw vplsLoopDetectEnable?   boolean
   |  +--rw pwTemplates
   |  |  +--rw pwTemplate* [pwTemplateName]
   |  |     +--rw pwTemplateName          string
   |  |     +--rw peerAddr?               inet:ip-address
   |  |     +--rw mtu?                    uint16
   |  |     +--rw ctrlWord?               enumeration
   |  |     +--rw tunnelPolicy?           string
   |  |     +--rw tdmEncapsulateNumber?   uint8
   |  |     +--rw jitterBuffer?           uint16
   |  |     +--rw rtpHeader?              boolean
   |  |     +--rw idleCode?               string
   |  |     +--rw tdmSequenceNumber?      boolean
   |  |     +--rw payloadCompression?     boolean
   |  |     +--rw timeSlot?               uint8
   |  |     +--rw maxAtmCells?            uint8
   |  |     +--rw atmPackOvertime?        uint16
   |  |     +--rw atmTransmitCell?        uint8
   |  |     +--rw sequenceNumber?         boolean
   ...
```

3.3. VPWS Configuration

The VPWS configuration container includes VPWS instance configuration, VPWS switch instance configuration and VPWS statistics information.

```
   +--rw l2vpnvpws
   |  +--rw vpwsStatisticInfo
   |  ...
   |  +--rw vpwsInstances
   |  ...
   |  +--rw vpwsSwitchInstances
   |  ...
```

3.3.1. VPWS Instances Configuration

The VPWS instance configuration includes per-instance parameters, AC configuration, PW configuration, TDM parameters, ATM parameters, reliability (including PW redundancy) configuration etc.

```
   +--rw vpwsInstances
   |  +--rw vpwsInstance* [instanceName instanceType]
   |     +--rw instanceName       leafref
   |     +--rw instanceType       instanceType
   |     +--rw encapsulateType?   pw-encapsulation
   |     +--rw description?       string
   |     +--ro instanceState?     enumeration
   |     +--ro lastUpTime?        yang:date-and-time
   |     +--ro totalUpTime?       string
   |     +--rw tdmParameters
   |     |  +--rw tdmEncapsulateNumber?   uint8
   |     |  ...
   |     +--rw atmParameters
   |     |  ...
   |     +--rw l2vpnAcs
   |     |  ...
   |     +--rw vpwsPws
   |     |  ...
   |     +--rw reliabilitys
   |        +--rw reliability* [pwRedundancyMode]
   ...
```

3.3.2. VPWS Switch Instances Configuration

VPWS switch instance configuration includes the configuration for multi-segment PW such as per-instance parameters, PW configuration, ATM parameters, TDM parameters etc.

```
   +--rw vpwsSwitchInstances
      +--rw vpwsSwitchInstance* [instanceName instanceType]
         +--rw instanceName       string
         +--rw instanceType       instanceType
         +--rw encapsulateType?   pw-encapsulation
         +--rw switchType?        enumeration
         +--rw ctrlWordTrans?     boolean
         +--rw controlWord?       enumeration
         +--ro instanceState?     enumeration
         +--ro createTime?        string
         +--ro upTime?            string
         +--ro lastChgTime?       string
         +--ro lastUpTime?        yang:date-and-time
         +--ro totalUpTime?       string
         +--rw vpwsPws
            +--rw vpwsPw* [pwRole pwId]
               +--rw pwRole            pw-role
               +--rw pwId              uint32
               +--rw peerIp?           inet:ip-address
               +--rw transmitLabel?    uint32
               +--rw receiveLabel?     uint32
               +--rw ctrlWord?         enumeration
               +--rw vccvAbility?      boolean
               +--rw tnlPolicyName?    string
               +--rw pwTemplateName?   string
               +--rw requestVlanId?    uint16
               +--rw vlanTpId?         string
               +--rw pwTtl?            uint8
               +--rw tdmParameters
               ...
               +--rw atmParameters
               ...
               +--rw vpwsLdpPwInfo
   ...
```

3.3.3. VPWS Statistics Information

The VPWS statistics information container includes statistics information of VPWS.

```
   +--rw vpwsStatisticInfo
   |  +--rw vpwsLdpAcStatInfo
   |  |  +--ro totalLdpAcNum?   uint32
   |  |  +--ro upLdpAcNum?      uint32
   |  |  +--ro downLdpAcNum?    uint32
   |  +--rw vpwsLdpPwStatInfo
   |  |  +--ro totalLdpPwNum?   uint32
   |  |  +--ro upLdpPwNum?      uint32
   |  |  +--ro downLdpPwNum?    uint32
   |  +--rw vpwsLdpPwRemoteStatInfo
   |  |  +--ro remoteVcNum?   uint32
   |  +--rw vpwsSwitchInstanceStatInfo
   |     +--ro totalSwitchInstanceNum?   uint32
   |     +--ro upSwitchInstanceNum?      uint32
   |     +--ro downSwitchInstanceNum?    uint32
```

3.4. VPLS Configuration

The L2VPN VPLS configuration includes VPLS instance configuration, VPLS statistics information.

```
   +--rw l2vpnvpls
      +--rw vplsStatisticInfo
      |  +--rw vplsInstStatisticsInfo
      ...
      |  +--rw vplsPwStatisticsInfo
      ...
      |  +--rw vplsAcStatisticsInfo
      ...
      |  +--ro vplsTnlRefInfos
      ...
      |  +--rw vplsLoopDetectStaticInfo
      |     +--ro totalVplsLoopDetectNum?   uint32
      +--rw vplsInstances
         +--rw vplsInstance* [instanceName]
            +--rw instanceName            string
            +--rw description?            string
            +--rw memberDiscoveryMode?    enumeration
            +--rw encapsulateType?        pw-encapsulation
            +--rw mtuValue?               uint16
            ...
```

3.4.1. VPLS Instance Configuration

The VPLS instance configuration includes member discovery mode, encapsulate type, VPLS LDP instance configuration, VPLS BGP AD instance configuration, VPLS BGP instance configuration and VPLS ACs configuration etc.

-- VPLS LDP instance configuration: This configuration describes how to configure LDP-based VPLS, with the signaling type being LDP.

-- VPLS BGP AD instance configuration: This configuration describes how to configure BGP AD VPLS to exchange extended BGP packets to automatically discover member VSIs in a VPLS domain and then use LDP FEC 129 to negotiate PW establishment to achieve automatic VPLS PW deployment.

-- VPLS BGP instance configuration: This configuration describes how to configure BGP VPLS. Detailed operations include configuring BGP as the signaling protocol, and configuring VPN targets to implement automatic discovery of VPLS PEs.

-- VPLS ACs configuration: This configuration describes configuration parameters of ACs.

```
+--rw vplsInstances
   +--rw vplsInstance* [instanceName]
      +--rw instanceName            string
      +--rw description?            string
      +--rw memberDiscoveryMode?    enumeration
      +--rw encapsulateType?        pw-encapsulation
      +--rw mtuValue?               uint16
      ...
      +--rw vsiPipe
      ...
      +--rw vplsLdpInst
      |  +--rw vsiId?               uint32
      ...
      +--rw vplsBgpAdInst
      |  +--rw vplsId?           string
      |  +--ro bgpAdRd?          string
      |  +--ro vsiId?            inet:ip-address
      |  +--rw vpnTargets
      ...
      +--rw vplsBgpInst
      |  +--rw bgpRd?          string
      |  +--rw ignoreMtu?      boolean
      ...
      +--rw vplsAcs
      |  +--rw vplsAc* [interfaceName]
      ...
      +--rw vplsLoopDetectInfo
         ...
```

3.4.2. VPLS Statistics Information

The VPLS statistics information container includes VPLS instance statistics information, VPLS PW statistics information, VPLS AC statistics information etc.

```
+--rw vplsStatisticInfo
|  +--rw vplsInstStatisticsInfo
...
|  +--rw vplsPwStatisticsInfo
...
|  +--rw vplsAcStatisticsInfo
...
|  +--ro vplsTnlRefInfos
...
|  +--rw vplsLoopDetectStaticInfo
|     +--ro totalVplsLoopDetectNum?   uint32
...
```

4. L2VPN Yang Module

```
<CODE BEGINS> file "l2vpn-yang@2014-08-21.yang"

module l2vpn {
    namespace "urn:huawei:params:xml:ns:yang:l2vpn";
    prefix "l2vpn";

    /* import */
    import ietf-inet-types {
        prefix inet;
    }
    import ietf-interfaces {
        prefix if;
    }
    import ietf-yang-types {
        prefix yang;
    }
    description
        "This YANG module defines the generic configuration data for
         L2VPN services.

         Terms and Acronyms
         L2VPN: Layer 2 Virtual Private Network
         VPLS: Virtual Private LAN Service
         VPWS: Virtual Private Wire Service
         ...
         ";
    revision 2014-08-21 {
        description
            "Initial revision.";
    }

    /* Typedef */
    typedef pw-encapsulation {
        description "PW encapsulation type.";
        type enumeration {
            enum fr {
                value "0";
                description "fr:";
            }
            enum atm-aal5-sdu {
                value "1";
                description "atm-aal5-sdu:";
            }
            enum atm-trans-cell {
                value "2";
                description "atm-trans-cell:";
            }
            enum vlan {
                value "3";
                description "vlan:";
            }
            enum ethernet {
                value "4";
                description "ethernet:";
            }
            enum hdlc {
                value "5";
                description "hdlc:";
            }
            enum ppp {
                value "6";
                description "ppp:";
            }
            enum cem {
                value "7";
                description "cem:";
            }
            enum atm-nto1-vcc {
                value "8";
                description "atm-nto1-vcc:";
            }
            enum atm-nto1-vpc {
                value "9";
                description "atm-nto1-vpc:";
            }
            enum ip-layer2 {
                value "10";
                description "ip-layer2:";
            }
            enum atm-1to1-vcc {
                value "11";
                description "atm-1to1-vcc:";
            }
            enum atm-1to1-vpc {
                value "12";
                description "atm-1to1-vpc:";
            }
            enum atm-aal5-pdu {
                value "13";
                description "atm-aal5-pdu:";
            }
            enum fr-port-mode {
                value "14";
                description "fr-port-mode:";
            }
            enum cesop {
                value "15";
                description "cesop:";
            }
            enum satop-e1 {
                value "16";
                description "satop-e1:";
            }
            enum satop-t1 {
                value "17";
                description "satop-t1:";
            }
            enum satop-e3 {
                value "18";
                description "satop-e3:";
            }
            enum satop-t3 {
                value "19";
                description "satop-t3:";
            }
            enum cesopsn-basic {
                value "20";
                description "cesopsn-basic:";
            }
            enum tdmoip_aal1 {
                value "21";
                description "tdmoip_aal1:";
            }
            enum cesopsn_tdm {
                value "22";
                description "cesopsn_tdm:";
            }
            enum tdmoip_aal2 {
                value "23";
                description "tdmoip_aal2:";
            }
            enum fr_dlci {
                value "24";
                description "fr_dlci:";
            }
            enum ip-interworking {
                value "25";
                description "ip-interworking:";
            }
            enum unsupport {
                value "26";
                description "unsupport:";
            }
        }
    }

    typedef pw-role {
        description "pw role.";
        type enumeration {
            enum primary {
                value "0";
                description "primary:";
            }
            enum backup {
                value "1";
                description "backup:";
            }
            enum bypass {
                value "2";
                description "bypass:";
            }
            enum multiHopOneSidePrimary {
                value "3";
                description "multiHopOneSidePrimary:";
            }
            enum multiHopOtherSidePrimary {
                value "4";
                description "multiHopOtherSidePrimary:";
            }
            enum multiHopOtherSideBackup {
                value "5";
                description "multiHopOtherSideBackup:";
            }
        }
    }

    typedef tunnelType {
        description "Indicates the type of tunnel used by the PW.";
        type enumeration {
            enum invalid {
                value "0";
                description "invalid tunnel type";
            }
            enum ldp {
                value "1";
                description "LDP LSP";
            }
            enum bgp {
                value "2";
                description "BGP LSP";
            }
            enum te {
                value "3";
                description "TE tunnel";
            }
            enum static_lsp {
                value "4";
                description "static lsp";
            }
            enum gre {
                value "5";
                description "GRE tunnel";
            }
            enum uni {
                value "6";
                description "uni tunnel";
            }
            enum tnl_group {
                value "7";
                description "tnl-group";
            }
            enum sub_te {
                value "8";
                description "TE sub tunnel";
            }
            enum sub_group {
                value "9";
                description "sub tunnel group";
            }
            enum 6over4 {
                value "10";
                description "manual IPv6 tunnel carry IPv4 traffic";
            }
            enum 6to4 {
                value "11";
                description "automatic IPv6 tunnel carry IPv4 traffic";
            }
            enum bgp_local_ifnet {
                value "12";
                description "BGP created mpls localifnet tunnel";
            }
            enum ldp6 {
                value "13";
                description "IPv6 LDP LSP";
            }
        }
    }

    typedef ifState {
        description "Interface state.";
        type enumeration {
            enum down {
                value "0";
                description "down:";
            }
            enum up {
                value "1";
                description "up:";
            }
            enum plugOut {
                value "2";
                description "plugOut:";
            }
            enum notifyDown {
                value "3";
                description "notifyDown:";
            }
            enum downNotify {
                value "4";
                description "downNotify:";
            }
        }
    }

    typedef pwState {
        description "Indicates the status of the PW.";
        type enumeration {
            enum down {
                value "0";
                description "down:";
            }
            enum up {
                value "1";
                description "up:";
            }
            enum backup {
                value "2";
                description "backup:";
            }
        }
    }

    typedef instanceType {
        description "Instance type. ";

        type enumeration {
            enum vpwsLocalccc {
                value "0";
                description "vpwsLocalccc:";
            }
            enum vpwsRemoteccc {
                value "1";
                description "vpwsRemoteccc:";
            }
            enum vpwsSvc {
                value "2";
                description "vpwsSvc:";
            }
            enum vpwsLdp {
                value "3";
                description "vpwsLdp:";
            }
            enum vpwsSwitch {
                value "4";
                description "vpwsSwitch:";
            }
            enum vpls {
                value "5";
                description "vpls:";
            }
        }
    }

    /* Grouping */
    grouping tdmParameter {
        description
            "Configure TDM parameter.";
        leaf tdmEncapsulateNumber {
            description "Number of encapsulated TDM frames.";
            config "true";
            type uint8 {
                range "1..40";
            }
        }
        leaf jitterBuffer {
            description "Depth of the TDM jitter buffer.";
            config "true";
            type uint16 {
                range "1000..64000";
            }
        }
        leaf rtpHeader {
            description
                "Whether or not the RTP header is added into the
                 transparently transported TDM frame.";
            config "true";
            type boolean;
        }
        leaf idleCode {
            description
                "Specifies the value of the idle code that is filled
                 manually when the jitter buffer underflow occurs.";
            config "true";
            type string {
                length "1..2";
                pattern "^((([1-9]|[a-f]|[A-F])([0-9]|[a-f]|[A-F])?)|
                         (0([0-9]|[a-f]|[A-F])?))$";
            }
        }
        leaf tdmSequenceNumber {
            description "Enable the seq-number option.";
            config "true";
            type boolean;
        }
        leaf payloadCompression {
            description
                "Specifies the dynamic bandwidth allocation for payload
                 compression.";
            config "true";
            type boolean;
        }
        leaf timeSlot {
            description
                "Specifies the time slot of the serial interface.";
            config "true";
            type uint8 {
                range "1..32";
            }
        }
    }

    grouping atmParameter {

        description "Configure ATM parameter.";

        leaf maxAtmCells {
            description "Maximum number of transmitted ATM cells.";
            config "true";
            type uint8 {
                range "1..28";
            }
        }
        leaf atmPackOvertime {
            description "Delay in packing ATM cells.";
            config "true";
            type uint16 {
                range "100..10000";
            }
        }
        leaf atmTransmitCell {
            description "ATM transmit cell.";
            config "true";
            type uint8 {
                range "1..28";
            }
        }
        leaf sequenceNumber {
            description
                "Enable the seq-number option.";
            config "true";
            type boolean;
        }
    }

    grouping speInfos {

        leaf speCount {
            description "Number of Spe.";
            config "false";
            type uint32;
        }

        list speInfo {

            config "false";
            key "spePwId spePeerIp";

            leaf spePwId {
                description
                    "Indicates the identifier of the PW.";
                type uint32;
            }
            leaf spePeerIp {
                description
                    "Specifies the LSR ID of the peer PE.";
                type inet:ip-address;
            }
        }
    }

    grouping vpwsPws {

        list vpwsPw {

            key "pwRole pwId";
            description "L2vpn vpws pw class.";

            leaf pwRole {
                description
                    "VPWS pw role:primary, backup,bypass.";
                config "true";
                type pw-role;

            }
            leaf pwId {
                description
                    "Indicates the identifier of the PW.";
                config "true";
                type uint32 {
                    range "1..4294967295";
                }
            }
            leaf peerIp {
                description
                    "Specifies the LSR ID of the peer PE.";
                config "true";
                type inet:ip-address;
            }
            leaf transmitLabel {
                description
                    "Indicates the label value of sent packets.";
                config "true";
                type uint32 {
                    range "0..1048575";
                }
            }
            leaf receiveLabel {
                description
                    "Indicates the label value of received packets.";
                config "true";
                type uint32 {
                    range "16..32767";
                }
            }
            leaf ctrlWord {
                description
                    "Enables the control word function. The control word
                    function is usually enabled on PWs with encapsulation
                    types being TDM, ATM or FR.

                    By default:
                    The control word function is enabled for TDM-, ATM-, or
                    Frame Relay-encapsulated PWs if PW profiles are not used.
                    If a PW profile is used, the control word function can
                    be enabled only after the control word is explicitly
                    specified. The control word function can be enabled for
                    PWs that use other types of encapsulation only after
                    the control word is explicitly specified.";
                config "true";
                type enumeration {
                    enum default {
                        value "0";
                        description "default:";
                    }
                    enum disable {
                        value "1";
                        description "disable:";
                    }
                    enum enable {
                        value "2";
                        description "enable:";
                    }
                }
            }
            leaf vccvAbility {
                description
```

```
                    "Configures VC connectivity detection. VC connectivity
                    detection supports two modes: control word mode and
                    label alert mode.";

                config "true";
                default "true";
                type boolean;
            }
            leaf tnlPolicyName {
                description
                    "Specifies a tunnel policy name for the L2VC. If no name
                    is specified for a tunnel policy, the default tunnel
                    policy is adopted. The LSP tunnel is preferred and only
                    one LSP is used for load balancing in the default tunnel
                    policy. If the name of the tunnel policy is specified but
                    no tunnel policy is configured, the default tunnel policy
                    is still adopted.";

                config "true";
                type string {
                    length "1..39";
                }
            }
            leaf pwTemplateName {
                description
                    "Specifies the name of a PW template. You can set
                     attributes for a PW template, including the remote
                     peer, tunnel policy, and control word. When configuring
                     an LDP PW, you can directly apply the PW template rather
                     than specifying attributes for the PW.";

                config "true";
                type string {
                    length "1..19";
                }
            }
            leaf requestVlanId {
                description
                    "Indicates the requested VLAN ID.";
                config "true";
                type uint16 {
                    range "1..4094";
                }
            }
            leaf vlanTpId {
                description "Indicates the TPID of requested VLAN ID.";
                config "true";
                type string {
                    length "1..4";
                    pattern "^([0-9]|[a-f]|[A-F]){0,4}$";
                }
            }
            leaf pwTtl {
                description "Specify the TTL for PW.";
                config "true";
                type uint8 {
                    range "1..255";
                }
            }
            container tdmParameters {
                uses tdmParameter;
            }

            container atmParameters {
                uses atmParameter;
            }

            container vpwsLdpPwInfo {

                leaf interfaceName {
                    description
                        "Indicates the type and number of the AC
                         interface.";
                    config "false";
                    type leafref {
                        path "/if:interfaces/if:interface/if:name";
                    }
                }
                leaf ifState {
                    description "Indicates  status of the AC interface.";
                    config "false";
                    type ifState;
                }
                leaf sessionState {
                    description
                        "Indicates the status of the LDP session established
                         between both ends of the VC.";
                    config "false";
                    type enumeration {
                        enum default {
                            value "0";
                            description "default:";
                        }
                        enum down {
                            value "1";
                            description "down:";
                        }
                        enum up {
                            value "2";
                            description "up:";
                        }
                    }
                }
                leaf integrativeAcState {
                    description
                        "The integrative status of the AC.";
                    config "false";
                    type ifState;
                }
                leaf acState {
                    description
                        "Indicates the status of the AC.";
                    config "false";
                    type ifState;
                }
                leaf pwState {
                    description
                        "Indicates the status of the PW.";
                    config "false";
                    type pwState;
                }
                leaf pwId {
                    description
                        "Indicates the identifier of the PW.";
                    config "false";
                    type uint32;
                }
                leaf encapType {
                    description
                        "Indicates the encapsulation type of the PW.";
                    config "false";
                    type pw-encapsulation ;

                }
                leaf destination {
                    description
                        "Indicates the LSR ID of the VC peer device.";
                    config "false";
                    type inet:ip-address;
                }
                leaf localGroupId {
                    description "Indicates the local group ID.";
                    config "false";
                    type uint32;
                }
                leaf remoteGroupId {
                    description "Indicates the remote group ID.";
                    config "false";
                    type uint32;
                }
                leaf localVcLabel {
                    description "Indicates the local VC label.";
                    config "false";
                    type uint32;
                }
                leaf remoteVcLabel {
                    description "Indicates the remote VC label.";
                    config "false";
                    type uint32;
                }
                container tdmInfo {

                    description "TDM info";
                    config "false";

                    leaf localTdmEncapsulateNum {
                        description "Number of encapsulated TDM frames.";
                        config "false";
                        type uint8;
                    }
                    leaf remoteTdmEncapsulateNum {
                        description "Number of encapsulated TDM frames.";
                        config "false";
                        type uint8;
                    }
                    leaf jitterBuffer {
                        description "Depth of the TDM jitter buffer.";
                        config "false";
                        type uint8;
                    }
                    leaf idleCode {
                        description
                             "Indicates the idle code that is filled manually
                              when the jitter buffer underflow occurs.";
                        config "false";
                        type string {
                            length "0..3";
                        }
                    }
                    leaf localRtpHeaderEnable {
                        description
                             "Whether or not the RTP header is added into
```

```
                              the transparently transported TDM frame.";
                        config "false";
                        type boolean;
                    }
                    leaf remoteRtpHeaderEnable {
                        description
                             "Whether or not the RTP header is added into the
                              transparently transported TDM frame.";
                        config "false";
                        type boolean;
                    }
                    leaf localBitRate {
                        description "Indicates the bit-rate of the local VC.";
                        config "false";
                        type uint16;
                    }
                    leaf remoteBitRate {
                        description "Indicates the bit-rate of the remoteVC.";
                        config "false";
                        type uint16;
                    }
                }
                container atmInfo {

                    description "TDM info";
                    config "false";

                    leaf maxAtmCells {
                        description "Maximum number of transmitted ATM cells.";
                        config "false";
                        type uint8 {
                            range "1..28";
                        }
                    }
                    leaf remoteMaxAtmCells {
                        description "Maximum number of transmitted ATM cells.";
                        config "false";
                        type uint8 {
                            range "0..28";
                        }
                    }
                    leaf atmPackOvertime {
                        description "Delay in packing ATM cells.";
                        config "false";
                        type uint16 {
                            range "100..10000";
                        }
                    }
                    leaf atmTransmitCell {
                        description "ATM transmit cell.";
                        config "false";
                        type uint16 {
                            range "1..28";
                        }
                    }
                    leaf sequenceNumber {
                        description
                             "Enable the seq-number option.By default, the
                              seq-number option is disabled.";
                        config "false";
                        type boolean;
                    }
                }
                leaf localFwdState {
                    description
                        "Indicates the status of the local forwarding table.";
                    config "false";
                    type enumeration {
                        enum notForwarding {
                            value "0";
                            description "notForwarding:";
                        }
                        enum forwarding {
                            value "1";
                            description "forwarding:";
                        }
                    }
                }
                leaf localStateCode {
                    description
                        "Indicates the status code of the local PW:
                        0x0: indicates that the local PW functioning as the
                             master PW is in the Up state.
                        0x20: indicates that the local PW functioning as the
                              backup PW is in the Up state.
                        0x1: indicates that the PW functioning as the master
                             PW and is in the Down state.
                        0x21: indicates that the PW functioning as the backup
                             PW and is in the Down state.";
                    config "false";
                    type uint32;
                }
                leaf remoteFwdState {
                    description
                        "Indicates the status of the remote forwarding
                         table.";
                    config "false";
                    type enumeration {
                        enum notForwarding {
                            value "0";
                            description "notForwarding:";
                        }
                        enum forwarding {
                            value "1";
                            description "forwarding:";
                        }
                    }
                }
                leaf remoteStateCode {
                    description
                        " Indicates the status code of the remote PW:
                        0x0: indicates that the remote PW functioning as the
                             master PW is in the Up state.
                        0x20: indicates that the remote PW functioning as the
                              backup PW is in the Up state.
                        0x1: indicates that the PW functioning as the master
                             PW and is in the Down state.
                        0x21: indicates that the PW functioning as the backup
                              PW and is in the Down state.";
                    config "false";
                    type uint32;
                }
                leaf isActive {
                    description
                        "Indicates whether or not the PW is in active state
                        (if so, user packets can be forwarded).";
                    config "false";
                    type boolean;
                }
                leaf isForwardExist {
                    description
                        "Indicates whether or not forwarding entries exist.";
                    config "false";
                    type boolean;
                }
                leaf linkState {
                    description
                        "Indicates the link status of the AC interface:
                        Up: indicates that the physical layer status of
                            the interface is functional.
                        Down: indicates that the physical layer of the
                              interface fails.";
                    config "false";
                    type enumeration {
                        enum default {
                            value "0";
                            description "default:";
                        }
                        enum down {
                            value "1";
                            description "down:";
                        }
                        enum up {
                            value "2";
                            description "up:";
                        }
                    }
                }
                leaf localVcMtu {
                    description "Indicates the MTU of the local VC.";
                    config "false";
                    type uint16;
                }
                leaf remoteVcMtu {
                    description "Indicates the MTU of the remote VC.";
                    config "false";
                    type uint16;
                }
                leaf localVCCV {
                    description
                        "Indicates the type of VCCV that is supported
                         locally. By default, the control word function
                         is not enabled, and the supported VCCV type is
                         alert lsp-ping bfd, indicating that LSP ping
                         and BFD are supported for the alert channel.
                         If the control word function is enabled, the
                         VCCV type is cw alert lsp-ping bfd, indicating
                         that LSP ping and BFD are supported both for the
                         control word channel and the alert channel.";
                    config "false";
                    type string {
                        length "0..40";
                    }
                }
                leaf remoteVCCV {
                    description
                        "Indicates the type of VCCV that is supported
                         remotely. By default, the control word function
                         is not enabled, and the supported VCCV type is
                         alert lsp-ping bfd, indicating that LSP ping and
                         BFD are supported for the alert channel. If the
                         control word function is enabled, the VCCV type
                         is cw alert lsp-ping bfd, indicating that LSP ping
                         and BFD are supported both for the control word
                         channel and the alert channel.";

                    config "false";
                    type string {
                        length "0..40";
                    }
                }
                leaf localCtrlWord {
                    description
                        "Indicates whether or not the control word is enabled
                        on the local end.";
                    config "false";
                    type enumeration {
                        enum default {
                            value "0";
                            description "default:";
                        }
                        enum disable {
                            value "1";
                            description "disable:";
                        }
                        enum enable {
                            value "2";
                            description "enable:";
                        }
                    }
                }
                leaf remoteCtrlWord {
                    description
                        "Indicates whether or not the control word is enabled
                         on the remote end.";
                    config "false";
                    type enumeration {
                        enum default {
                            value "0";
                            description "default:";
                        }
                        enum disable {
                            value "1";
                            description "disable:";
                        }
                        enum enable {
                            value "2";
                            description "enable:";
                        }
                    }
                }
                leaf tnlPolicyName {
                    description "Indicates the name of the tunnel policy.";
                    config "false";
                    type string {
                        length "1..39";
                    }
                }
                leaf pwTemplateName {
                    description "Indicates the name of the PW template.";
                    config "false";
                    type string {
                        length "1..19";
                    }
                }
                leaf priOrSec {
                    description
                        "Indicates whether the local status of the VC is
                         primary or secondary.";
                    config "false";
                    type enumeration {
                        enum primary {
                            value "0";
                            description "primary:";
                        }
                        enum secondary {
                            value "1";
                            description "secondary:";
                        }
                        enum bypass {
                            value "2";
                            description "bypass:";
                        }
                    }
                }
                leaf tunnelCount {
                    description
                        "Indicates that the PW uses one tunnel or token";
                    config "false";
                    type uint8;
                }
                container tunnelInfos {

                    config "false";
                    list tunnelInfo {

                        key "tunnelKey";
                        leaf tunnelKey {
                            description
                                "Indicates the ID of the tunnel used by the
                                 PW.";

                            type string {
                                length "0..21";
                            }
                        }
                        leaf tunnelType {
                            description
                                "Indicates the type of tunnel used by the
                                 PW.";
                            config "false";
                            type tunnelType;
                        }
                        leaf tunnelName {
                            description
                                "Indicates the name of the tunnel used by the
                                 PW.";
                            config "false";
                            type string {
                                length "0..64";
                            }
                        }
                        leaf publicNextHop {
                            description
                                "Indicates public next hop of a tunnel.";
                            config "false";
                            type inet:ip-address;
                        }
                    }

                }

                container speInfos {
                    config "false";
                    uses speInfos;
                }

                leaf createTime {
                    description
                        "Indicates how long the VC has been created for.";
                    config "false";
                    type string {
                        length "1..80";
                    }
                }
                leaf upTime {
                    description
                        "Indicates how long the VC keeps the Up state.
                         If the PW is currently in the Down state, the
                         value is 0.";
                    config "false";
                    type string {
                        length "1..80";
                    }
                }
                leaf lastChgTime {
                    description
                        "Indicates how long the VC status has remained
                         unchanged.";
                    config "false";
                    type string {
                        length "1..80";
                    }
                }
                leaf pwLastUpTime {
                    description
                        "Indicates the last time the VC was Up.";
                    config "false";
                    type yang:date-and-time;
                }
                leaf pwTotalUpTime {
                    description
                        "Indicates the total duration the VC is Up.";
                    config "false";
                    type string {
                        length "1..80";
                    }
                }
                leaf supportNotification {
                    description
                        "Indicates whether or not the Notification
                         message is supported.";
                    config "false";
                    type boolean;
                }
            }

        }

    }

    grouping vplsPwInfo {
        leaf peerIp {
            description
                "Indicates the LSR ID of the VC peer device.";
            config "false";
            type inet:ip-address;
        }
        leaf pwId {
            description
                "Indicates the identifier of the PW.";
            config "false";
            type uint32;
        }
        leaf pwType {
            description "Type of the PW.label, QinQ,MEHVPLS";
            config "false";
            type enumeration {
                enum label {
                    value "0";
                    description "label:";
                }
                enum QinQ {
                    value "1";
                    description "QinQ:";
                }
                enum MEHVPLS {
                    value "2";
                    description "MEHVPLS:";
                }
            }
        }
        leaf sessionState {
            description
                "Indicates the status of the LDP session established
                 between both ends of the VC.";
            config "false";
            type enumeration {
                enum default {
                    value "0";
                    description "default:";
                }
                enum down {
                    value "1";
                    description "down:";
                }
                enum up {
                    value "2";
                    description "up:";
                }
            }
        }
        leaf pwState {
            description "Indicates the status of the PW.";
            config "false";
            type enumeration {
                enum down {
                    value "0";
                    description "down:";
                }
                enum up {
                    value "1";
                    description "up:";
                }
                enum backup {
                    value "2";
                    description "backup:";
                }
            }
        }
        leaf localVcLabel {
            description "Indicates the local VC label.";
            config "false";
            type uint32;
        }
        leaf remoteVcLabel {
            description "Indicates the remote VC label.";
            config "false";
            type uint32;
        }
        leaf tnlPolicyName {
            description
                "Indicates the name of the tunnel policy.";
            config "false";
            type string {
                length "1..39";
            }
        }
        leaf pwLastUpTime {
            description
                "Indicates the last time the VC was Up.";
            config "false";
            type yang:date-and-time;
        }
        leaf pwTotalUpTime {
            description
                "Indicates the total duration the VC is Up.";
            config "false";
            type string {
                length "1..80";
            }
        }
        container tunnelInfos {

            config "false";

            list tunnelInfo {

                key "tunnelKey";

                leaf tunnelKey {
                    description
                        "Indicates the ID of the tunnel used by the PW.";
                    type string {
                        length "0..21";
                    }
                }
                leaf tunnelType {
                    description
                        "Indicates the type of tunnel used by the PW.";
                    config "false";
                    type tunnelType;
                }
                leaf outIntf {
                    description
                        "Outbound interface.";
                    config "false";
                    type string {
                        length "0..256";
                    }
                }
                leaf tunnelName {
                    description
                        "Indicates the name of the tunnel used by the PW.";
                    config "false";
                    type string {
                        length "0..64";
                    }
                }
                leaf publicNextHop {
                    description "Assign public next hop of a tunnel.";
                    config "false";
                    type inet:ip-address;
                }
            }
        }
        container speInfos {
            config "false";
            uses speInfos;
        }
        leaf remoteGroupId {
            description "ID of the remote group.";
            config "false";
            type uint32;
        }
        leaf remoteMtu {
            description "Indicates the MTU of a remote VC.";
            config "false";
            type uint16;
        }
        leaf remoteVCCVcode {
            description "Indicates the VCCV of a remote VC.";
            config "false";
            type string {
                length "0..40";
            }
        }
        leaf remoteStateCode {
            description
                "Indicates the status of a remote VC, which can be:
                FORWARD: The remote VC is in the forwarding state.
                STANDBY: The remote VC is in the standby state.
                AC FAULT: The remote AC interface is faulty.
                PSN FAULT: The remote VC is faulty.
                NO FORWRD: The remote VC interface cannot forward packets
                           owing to other reasons. ";
            config "false";
            type enumeration {
                enum forward {
                    value "0";
                    description "forward:";
                }
                enum not-forward {
                    value "1";
                    description "not forward:";
                }
                enum standby {
                    value "2";
                    description "standby:";
                }
                enum ac-fault {
                    value "3";
                    description "ac fault:";
                }
                enum psn-fault {
                    value "4";
                    description "psn fault:";
                }
            }
        }
    }

    /* container */
    container l2vpncommon {

        container l2vpnGlobal {

            description "L2VPN golbal attribute.";

            leaf l2vpnEnable {
                description
                    "L2vpn enable flag.";
                type boolean;
                config "true";
                mandatory "true";
            }
            leaf vplsLoopDetectEnable {
                description
                    "Vpls mac withdraw loop detect enable flag.";
                type boolean;
                config "true";
            }
        }

        container pwTemplates {

            list pwTemplate {

                key "pwTemplateName";
                description "L2VPN pw template class.";

                leaf pwTemplateName {
                    description
                        "Specifies the PW template name. The value is a
                         case-sensitive string of 1 to 19 characters without
                         blank space.";
                    config "true";
                    type string {
                        length "1..19";
                    }
                }
                leaf peerAddr {
                    description
                        "Assign a peer IP address to a PW template.";
                    config "true";
                    type inet:ip-address;
                }
                leaf mtu {
                    description
                        "Configure the mtu value for PW template, 46 to 9600.";
                    config "true";
                    default "1500";
                    type uint16 {
                        range "46..9600";
                    }
                }
                leaf ctrlWord {
                    description
                        "Enable the control word in a PW template.";
                    config "true";
                    type enumeration {
                        enum default {
                            value "0";
                            description "default:";
                        }
                        enum disable {
                            value "1";
                            description "disable:";
                        }
                        enum enable {
                            value "2";
                            description "enable:";
                        }
                    }
                }
                leaf tunnelPolicy {
                    description
                        "Configure a tunnel policy for a PW template.";
                    config "true";
                    type string {
                        length "1..39";
                    }
                }
                uses tdmParameter;

                uses atmParameter;

            }
        }
        container notMatchRemoteLdpInfos {

            config "false";
            list notMatchRemoteLdpInfo {

                key "pwId peerIp encapsulateType";

                leaf pwId {
                    description
                        "After an ID is set for a VC, it cannot be changed.
                         Different VCs have different IDs.";
                    type uint32;
                }
                leaf peerIp {
                    description "Indicates the peer ip of the VC peer device.";
                    type inet:ip-address;
                }
                leaf encapsulateType{
                    description "Indicates the encapsualtion VC peer device.";
                    type pw-encapsulation;
                }
                leaf remoteLabel {
                    description "Indicates the remote VC label.";
                    config "false";
                    type uint32 ;
                }
                leaf remoteGroupId {
                    description "ID of the remote group.";
                    config "false";
                    type uint32;
                }
                leaf remoteMtu {
                    description "Indicates the MTU of a remote VC.";
                    config "false";
                    type uint16;
                }
                leaf remoteStateCode {
                    description
                        "Indicates the status of a remote VC, which can be:
                            FORWARD: The remote VC is in the forwarding state.
                            STANDBY: The remote VC is in the standby state.
                            AC FAULT: The remote AC interface is faulty.
                            PSN FAULT: The remote VC is faulty.
                            NO FORWRD: The remote VC interface cannot forward
                                       packets owing to other reasons.";
                    config "false";
                    type enumeration {
                        enum forward {
                            value "0";
                            description "forward:";
                        }
                        enum not-forward {
                            value "1";
                            description "not forward:";
                        }
                        enum standby {
                            value "2";
                            description "standby:";
                        }
                        enum ac-fault {
                            value "3";
                            description "ac fault:";
                        }
                        enum psn-fault {
                            value "4";
                            description "psn fault:";
                        }
                    }
                }
            }

        }
    }

    container l2vpnvpws {

        container vpwsStatisticInfo {

            container vpwsLdpAcStatInfo {

                leaf totalLdpAcNum {
                    description "Total number of L2VPN VPWS LDP AC.";
                    config "false";
                    type uint32;
                }
                leaf upLdpAcNum {
                    description "Up number of L2VPN VPWS LDP AC.";
                    config "false";
                    type uint32;
                }
                leaf downLdpAcNum {
                    description "Down number of L2VPN VPWS LDP AC.";
                    config "false";
                    type uint32;
                }
            }
            container vpwsLdpPwStatInfo {

                leaf totalLdpPwNum {
                    description
                        "Indicates the total number of established LDP PWs";
                    config "false";
                    type uint32;
                }
                leaf upLdpPwNum {
                    description "Number of LDP PWs in the up state.";
                    config "false";
                    type uint32;
                }
                leaf downLdpPwNum {
                    description "Number of LDP PWs in the down state.";
                    config "false";
                    type uint32;
                }
            }

            container vpwsLdpPwRemoteStatInfo {

                leaf remoteVcNum {
                    description
                        "Indicates the total number of created remote LDP
                         PWs.";
                    config "false";
                    type uint32;
                }
            }

            container vpwsSwitchInstanceStatInfo {

                leaf totalSwitchInstanceNum {
                    description
                        "Indicates the total number of established switch-vc";
                    config "false";
                    type uint32;
                }
                leaf upSwitchInstanceNum {
                    description "Number of switch-vc in the up state.";
                    config "false";
                    type uint32;
                }
                leaf downSwitchInstanceNum {
                    description "Number of switch-vc in the down state.";
                    config "false";
                    type uint32;
                }
            }

        }

        container vpwsInstances {

            list vpwsInstance {

                key "instanceName instanceType";

                description "L2vpn vpws instance class.";

                leaf instanceName {

                    description "Specifies VPWS instance name.";
                    config "true";

                    type leafref {
                        path "/if:interfaces/if:interface/if:name";
                    }
                }
                leaf instanceType {
                    description "VPWS instance type:ldp,localccc.";
                    config "true";
                    type instanceType;
                }
                leaf encapsulateType {
                    type pw-encapsulation;
                }
                leaf description {
                    description "Specifies a description for the VC.";
                    config "true";
                    type string {
                        length "1..64";
                    }
                }
                leaf instanceState {
                    description "VPWS instance state.";
                    config "false";
                    type enumeration {
                        enum down {
                            value "0";
                            description "down:";
                        }
                        enum up {
                            value "1";
                            description "up:";
                        }
                        enum adminDown {
                            value "2";
                            description "adminDown:";
                        }
                    }
                }
                leaf lastUpTime {
                    description
                        "Indicates how long the instance keeps the Up state.
                         If the PW is currently in the Down state, the value
                         is 0.";
                    config "false";
                    type yang:date-and-time;
                }
                leaf totalUpTime {
                    description
                        "Indicates the total duration the instance is Up.";
                    config "false";
                    type string {
                        length "1..80";
                    }
                }

                container tdmParameters {
                    uses tdmParameter;
                }

                container atmParameters {
                    uses atmParameter;
                }

                container l2vpnAcs {

                    list l2vpnAc {

                        key "interfaceName";
                        description "L2VPN ac class.";

                        leaf interfaceName {
                            description "Specifies the AC interface name.";
                            config "true";
                            type leafref {
                                path "/if:interfaces/if:interface/if:name";
                            }
                        }
                        leaf state {
                            description "Indicates the status of the AC.";
```

```
                       config "false";
                       type enumeration {
                           enum default {
                               value "0";
                               description "default:";
                           }
                           enum down {
                               value "1";
                               description "down:";
                           }
                           enum up {
                               value "2";
                               description "up:";
                           }
                       }
                   }

                   container l2vpnPipe {

                       description "L2VPN pipe mode.";

                       leaf pipeMode {
                           description "Pipe mode.";
                           default "uniform";
                           type enumeration {
                               enum pipe {
                                   value "0";
                                   description "pipe:";
                               }
                               enum shortPipe {
                                   value "1";
                                   description "shortPipe:";
                               }
                               enum uniform {
                                   value "2";
                                   description "uniform:";
                               }
                           }
                       }
                   }

                   container ifLinkProtocolTran {

                       description "lacp status";

                       leaf protocolLacp {
                           description "lacp status";
                           config "true";
```

```
                                type enumeration {
                                    enum enable {
                                        value "0";
                                        description "enable:";
                                    }
                                    enum disable {
                                        value "1";
                                        description "disable:";
                                    }
                                }
                            }
                            leaf protocolLldp {
                                description "lldp status";
                                config "true";
                                type enumeration {
                                    enum enable {
                                        value "0";
                                        description "enable:";
                                    }
                                    enum disable {
                                        value "1";
                                        description "disable:";
                                    }
                                }
                            }
                            leaf protocolBpdu {
                                description "bpdu status";
                                config "true";
                                type enumeration {
                                    enum enable {
                                        value "0";
                                        description "enable:";
                                    }
                                    enum disable {
                                        value "1";
                                        description "disable:";
                                    }
                                }
                            }
                            leaf protocolCdp {
                                description "cdp status";
                                config "true";
                                type enumeration {
                                    enum enable {
                                        value "0";
                                        description "enable:";
                                    }
                                    enum disable {
                                        value "1";
                                        description "disable:";
                                    }
                                }
                            }
                            leaf protocolUdld {
                                description "udld status";
                                config "true";
                                type enumeration {
                                    enum enable {
                                        value "0";
                                        description "enable:";
                                    }
                                    enum disable {
                                        value "1";
                                        description "disable:";
                                    }
                                }
                            }
                        }

                    }

                }

                container vpwsPws {
                    uses vpwsPws;
                }

                container reliabilitys {

                    list reliability {

                        key "pwRedundancyMode";

                        leaf pwRedundancyMode {
                            description
                                "Specifies the redundancy mode of VPWS
                                 instance.";

                            config "true";
                            type enumeration {
                                enum frr {
                                    value "0";
                                    description "frr:";
                                }
                                enum masterSlave {
                                    value "1";
                                    description "masterSlave:";
                                }
                                enum independent {
                                    value "2";
                                    description "independent:";
                                }
                            }
                        }
                        leaf switchover {
                            description
                                "Specifies switches traffic from the primary
                                 PW to the secondary PW.";
                            config "true";
                            type boolean;
                        }
                        leaf dualReceive {
                            description
                                "Specifies enables a UPE interface to receive
                                 packets from both the primary and secondary
                                 PWs.";
                            config "true";
                            type boolean;
                        }
                        container reRoute {

                            description "L2vpn vpws pw reroute class.";

                            leaf reRoutePolicy {
                                description "Specifies the Policy of Reroute.";
                                config "true";
                                type enumeration {
                                    enum delay {
                                        value "0";
                                        description "delay:";
                                    }
                                    enum immediately {
                                        value "1";
                                        description "immediately:";
                                    }
                                    enum never {
                                        value "2";
                                        description "never:";
                                    }
                                }
                            }
                            leaf delayTime {
                                description
                                    "Specifies the delay for switching traffic
```

```
                             back to the primary PW. ";
                        config "true";
                        type uint16 {
                            range "10..600";
                        }
                    }
                    leaf resumeTime {
                        description
                            "Specifies the time after which the peer PE
                             on the secondary PW is notified that the
                             local PE is recovered from the fault. ";
                        config "true";
                        type uint16 {
                            range "0..600";
                        }
                    }
                    leaf lastReRouteReason {
                        description
                            "Specifies the reason of Last Reroute.";
                        config "false";
                        type string {
                            length "0..100";
                        }
                    }
                    leaf lastReRouteTime {
                        description
                            "Specifies the time of Last Reroute.";
                        config "false";
                        type string {
                            length "1..60";
                        }
                    }
                    leaf delayResidual {
                        description
                            "Specifies the residual delay time for
                             switching traffic back to the primary PW.
                            ";
                        config "false";
                        type uint32;
                    }
                    leaf resumeResidual {
                        description
                            "Specifies the residual time after which
                             the peer PE on the secondary PW is
                             notified that the local PE is recovered
                             from the fault. ";
                        config "false";
                        type uint32;
```

```
                            }
                        }

                    }

                }

            }

        }

        container vpwsSwitchInstances {

            list vpwsSwitchInstance {

                key "instanceName instanceType";

                description "L2vpn vpws instance class.";

                leaf instanceName {
                    description "Specifies VPWS instance name.";
                    config "true";
                    type string {
                        length "1..31";
                    }
                }
                leaf instanceType {
                    description "VPWS instance type:vpwsSwitch.";
                    config "true";
                    type instanceType;
                }
                leaf encapsulateType {
                    description "VPWS instance encapsulation type.";
                    config "true";
                    default "ethernet";
                    type pw-encapsulation;
                }
                leaf switchType {
                    description
                        "VPWS switch instance type:ldp2ldp,ldp2svc.";
                    config "true";
                    default "ldp2ldp";
                    type enumeration {
                        enum svc2svc {
                            value "0";
                            description "svc2svc:";
                        }
                        enum ldp2svc {
```

```
                        value "1";
                        description "ldp2svc:";
                    }
                    enum ldp2ldp {
                        value "2";
                        description "ldp2ldp:";
                    }
                    enum upe {
                        value "3";
                        description "upe:";
                    }
                }
            }
            leaf ctrlWordTrans {
                description
                    "Transparent transmission of control word If BFD
                     is enabled to monitor dynamic multi-hop PWs,
                     transparent transmission of control word needs
                     to be configured on the SPE. Otherwise, the BFD
                     negotiation fails. By default, transparent
                     transmission of control word is disabled.";
                config "true";
                type boolean;
                default "false";
            }
            leaf controlWord {
                description
                    "Enables the control word function. The control
                     word function is usually enabled on PWs with
                     encapsulation types being TDM, ATM or FR.
                     By default:
                     The control word function is enabled for TDM-,
                     ATM-, or Frame Relay-encapsulated PWs if PW
                     profiles are not used. If a PW profile
                     is used, the control word function can be
                     enabled only after the control word is explicitly
                     specified. The control word function can be enabled
                     for PWs that use other types of encapsulation only
                     after the control word is explicitly specified.";
                config "true";
                type enumeration {
                    enum default {
                        value "0";
                        description "default:";
                    }
                    enum disable {
                        value "1";
                        description "disable:";
                    }
                    enum enable {
                        value "2";
                        description "enable:";
                    }
                }
            }
            leaf instanceState {
                description "VPWS instance state.";
                config "false";
                type enumeration {
                    enum down {
                        value "0";
                        description "down:";
                    }
                    enum up {
                        value "1";
                        description "up:";
                    }
                    enum adminDown {
                        value "2";
                        description "adminDown:";
                    }
                }
            }
            leaf createTime {
                description
                    "Indicates how long the VC has been created for.";
                config "false";
                type string {
                    length "1..80";
                }
            }
            leaf upTime {
                description
                    "Indicates how long the VC keeps the Up state. If the
                     PW is currently in the Down state, the value is 0.";
                config "false";
                type string {
                    length "1..80";
                }
            }
            leaf lastChgTime {
                description
                    "Indicates how long the VC status has remained
                     unchanged.";
                config "false";
                type string {
                        length "1..80";
                    }
            }
            leaf lastUpTime {
                description
                    "Indicates how long the instance keeps the Up state.
                     If the PW is currently in the Down state, the value
                     is 0.";
                config "false";
                type yang:date-and-time;
            }
            leaf totalUpTime {
                description
                    "Indicates the total duration the instance is Up.";
                config "false";
                type string {
                    length "1..80";
                }
            }

            container vpwsPws {
                uses vpwsPws;
            }

        }

    }
}

container l2vpnvpls {

    container vplsStatisticInfo {

        container vplsInstStatisticsInfo {

            leaf totalVsiNum {
                description "None";
                config "false";
                type uint32;
            }
            leaf vsiUpNum {
                config "false";
                type uint32;
            }
            leaf vsiDownNum {
                config "false";
                type uint32;
            }
            leaf ldpModeNum {
                config "false";
                type uint32;
            }
            leaf bgpVsiNum {
                config "false";
                type uint32;
            }
            leaf bgpAdVsiNum {
                config "false";
                type uint32;
            }
            leaf unspecifiedNum {
                config "false";
                type uint32;
            }
        }

        container vplsPwStatisticsInfo {

            leaf totalPwNum {
                description "None";
                config "false";
                type uint32;
            }
            leaf upPwNum {
                config "false";
                type uint32;
            }
            leaf downPwNum {
                config "false";
                type uint32;
            }
            leaf ldpPwNum {
                config "false";
                type uint32;
            }
            leaf bgpPwNum {
                config "false";
                type uint32;
            }
            leaf bgpAdPwNum {
                config "false";
                type uint32;
            }
        }

        container vplsAcStatisticsInfo {
            leaf totalVplsAcNum {
                config "false";
                type uint32;
            }
            leaf upVplsAcNum {
                config "false";
                type uint32;
            }
            leaf downVplsAcNum {
                config "false";
                type uint32;
            }
        }

        container vplsTnlRefInfos {

            config "false";
            list vplsTnlRefInfo {

                key "tnlPolicyName";

                leaf tnlPolicyName {
                    description "None";
                    config "false";
                    type string {
                        length "1..39";
                    }
                }
                container tnlVsiRefInfos {

                    list tnlVsiRefInfo {

                         key "instanceName";

                         leaf instanceName {
                            config "false";
                            type string {
                                length "1..31";
                            }
                        }
                    }

                }

            }

        }
        container vplsLoopDetectStaticInfo {

             leaf totalVplsLoopDetectNum {
                config "false";
                type uint32;
            }
        }

    }

    container vplsInstances {

        list vplsInstance {

            key "instanceName";

            leaf instanceName {
                description
                    "Specifies VPLS instance name.";
                config "true";
                type string {
                    length "1..31";
                }
            }
            leaf description {
                description
                    "Specify the VPLS instance description.";
                config "true";
                type string {
                    length "1..64";
                }
            }
            leaf memberDiscoveryMode {
                description
                    "The VPLS member discovery mode for a created VSI.";
                config "true";
                default "default";
                type enumeration {
                    enum default {
                        value "0";
                        description "default:";
                    }
                    enum auto {
                        value "1";
                        description "auto:";
                    }
                    enum static {
                        value "2";
                        description "static:";
                    }
                    enum bdmode {
                        value "3";
                        description "bd mode:";
                    }
                }
            }
            leaf encapsulateType {
                description "VPWS instance encapsulation type.";
                config "true";
                default "vlan";
                type pw-encapsulation;
            }
            leaf mtuValue {
                description
                    "MTU specified in dynamic PW signaling negotiation.";
                config "true";
                default "1500";
                type uint16 {
                    range "328..65535";
                }
            }
            leaf tnlPolicyName {
                description
                    "Specifies a tunnel policy name for the VSI. If no
                     name is specified for a tunnel policy, the default
                     tunnel policy is adopted. The LSP tunnel is
                     preferred and only one LSP is used for load balancing
                     in the default tunnel policy. If the name of the
                     tunnel policy is specified but no tunnel policy is
                     configured, the default tunnel policy is still adopted.
                     ";
                config "true";
                type string {
                    length "1..39";
                }
            }
            leaf shutdown {
                description
                    "Sometimes, because of service debugging or service
                     halt, a VSI can be disabled for function
                     modification.";
                config "true";
                default "false";
                type boolean;
            }
            leaf isolateSpoken {
                description
                    "The isolate spoken command enables forwarding
                     isolation between AC interfaces, between UPE
                     PWs, and between ACs and UPE PWs on a VSI. The
                     undo isolate spoken command disables forwarding
                     isolation";
                config "true";
                default "false";
                type boolean;
            }
            leaf unknownUnicastAction {
                description
                    "Specifies the processing mode for received unknown
                     unicast frames.";
                config "true";
                type enumeration {
                    enum broadcast {
                        value "0";
                        description "broadcast:";
                    }
                    enum drop {
                        value "1";
                        description "drop:";
                    }
                    enum local-handle {
                        value "2";
                        description "local-handle:";
                    }
                    enum drop-learn {
                        value "3";
                        description "drop-learn:";
                    }
                }
            }
            leaf macLearnEnable {
                description
                    "Enables MAC address learning on a VSI.";
                config "true";
                default "true";
                type boolean;
            }
            leaf macLearnStyle {
                description
                    "Sets the MAC address learning style of a VSI.By
                     default, MAC address learning style is unqualify.
                     Currently, the VRP supports only the unqualified
                     mode.";
                config "true";
                default "unqualify";
                type enumeration {
                    enum qualify {
                        value "0";
                        description "qualify:";
                    }
                    enum unqualify {
                        value "1";
                        description "unqualify:";
                    }
                }
            }
            leaf macAgeTimer {
                description
                    "Sets the aging time of MAC address entries in a VSI.
                     By default, the aging time of MAC address entries in
                     a VSI is the global aging time. You can set the global
                     aging time by the command mac-address aging-time
                     (system view).";
                config "true";
                type uint32 {
                    range "0..1000000";
                }
            }
            leaf totalAcService {
                description
                    "Total number of interface in the instance.";
                config "false";
                type uint32;
            }
            leaf createTime {
                description
                    "Indicates how long the VSI has been created for.";
                config "false";
                type string {
                    length "1..60";
                }
            }
            leaf vsiState {
                description "VPLS instance state.";
                config "false";
                type enumeration {
                    enum down {
                        value "0";
                        description "down:";
                    }
                    enum up {
                        value "1";
                        description "up:";
                    }
                    enum adminDown {
                        value "2";
                        description "adminDown:";
                    }
                }
            }
            leaf ignoreAcStateEffect {
                description
                    "After the ignore-ac-state command is configured,
                     the VSI status is not subject to changes in the
                     status of the AC. That is, a VSI can go Up even
                     if no AC is connected to the VSI.";

                config "false";
                default "false";
                type boolean;
            }

            container vsiPipe {

                leaf pipeMode {
                    description "Pipe mode";
                    config "true";
                    default "uniform";
                    type enumeration {
                        enum pipe {
                            value "0";
                            description "pipe:";
                        }
                        enum shortPipe {
                            value "1";
                            description "shortPipe:";
                        }
                        enum uniform {
                            value "2";
                            description "uniform:";
                        }
                    }
                }
                leaf serviceClass {
                    description "service class";
                    config "true";
                    default "be";
                    type enumeration {
                        enum be {
                            value "0";
                            description "be:";
                        }
                        enum af1 {
                            value "1";
                            description "af1:";
                        }
                        enum af2 {
                            value "2";
                            description "af2:";
                        }
                        enum af3 {
                            value "3";
                            description "af3:";
                        }
                        enum af4 {
                            value "4";
                            description "af4:";
                        }
                        enum ef {
                            value "5";
                            description "ef:";
                        }
                        enum cs6 {
                            value "6";
                            description "cs6:";
                        }
                        enum cs7 {
                            value "7";
                            description "cs7:";
                        }
                    }
                }
                leaf color {
                    description "service color";
                    config "true";
                    default "green";
                    type enumeration {
                        enum green {
                            value "0";
                            description "green:";
                        }
                        enum yellow {
                            value "1";
                            description "yellow:";
                        }
                        enum red {
                            value "2";
```

```
                                   description "red:";
                               }
                           }
                       }
                       leaf dsName {
                           description "domain name";
                           config "true";
                           type string {
                               length "1..31";
                           }
                       }
                   }

                   container vplsLdpInst {

                       leaf vsiId {
                           description
                               "After an ID is set for a VSI, it cannot be
                                changed. Different VSIs have different IDs.";
                           config "true";
                           type uint32 {
                               range "1..4294967295";
                           }
                       }
                       leaf macWithdraw {
                           description
                               "Configures a VSI to delete the local MAC
                                addresses and informs all the remote peers
                                of the deletion when an AC fault or a UPE
                                fault occurs and the VSI remains Up.";
                           config "true";
                           default "false";
                           type boolean;
                       }
                       leaf ifChgWithdraw {
                           description
                               "Configures PEs to send LDP MAC Withdraw
                                messages to all peers when the status of
                                the AC interface bound to the VSI changes.";
                           config "true";
                           default "false";
                           type boolean;
                       }
                       leaf upeUpeMacWithdraw {
                           description
                               "Configures an NPE to forward the LDP MAC
                                Withdraw messages received from a UPE to
                                other UPEs.";
```

```
                        config "true";
                        default "false";
                        type boolean;
                    }
                    leaf upeNpeMacWithdraw {
                        description
                            "Configures an NPE to forward the LDP MAC
                             Withdraw messages received from UPEs to
                             other NPEs.";
                        config "true";
                        default "false";
                        type boolean;
                    }
                    leaf npeUpeMacWithdraw {
                        description
                            "Configures an NPE to forward the LDP MAC
                             Withdraw messages received from other NPEs
                             to UPEs.";
                        config "true";
                        type boolean;
                    }
                    container vplsLdpPws {

                        list vplsLdpPw {

                            key "peerIp pwId pwEncapType";

                            leaf peerIp {
                                description "Specifies the LSR ID of the peer PE
.";

                                config "true";
                                type inet:ip-address;
                            }
                            leaf pwId {
                                description
                                    "Indicates the identifier of the PW. Default
,

                                     we may use vsiId as the pwId. Sometimes we
may

                                     create pw to different pw that the pwId not
                                     match our vsiId, so it must specify the pwI
d.";

                                config "true";
                                type uint32 {
                                    range "1..4294967295";
                                }
                            }
                            leaf pwEncapType {
                                description
                                    "Indicates the encapsulation of the PW.
                                     Default, we may use encapsulateType as
```

```
                                   the pwEncapType. Sometimes we may create
                                   pw that the pwEncapType not match our
                                   encapsulateType, so it must specify the
                                   pwEncapType.";
                              config "true";
                              type pw-encapsulation;
                          }
                          leaf pwRole {
                              description
                                  "VPLS pw role:primary, secondary.";
                              config "true";
                              default "primary";
                              type pw-role;
                          }
                          leaf pwName {
                              description
                                  "Specifies the name of a PW, which is used
                                   to distinguish the PW from other PWs. The
                                   PW name must be unique in the same VSI,
                                   but can be the same as the PW names in
                                   other VSIs. ";
                              config "true";
                              type string {
                                  length "1..15";
                              }
                          }
                          leaf ifParaVCCV {
                              description
                                  "Deletes the VCCV byte (an interface
                                   parameter) in the Mapping packet.";
                              config "true";
                              default "true";
                              type boolean;
                          }
                          leaf isUpe {
                              description "set VPLS PW as upe.";
                              config "true";
                              default "false";
                              type boolean;
                          }
                          leaf tnlPolicyName {
                              description
                                  "Specifies a tunnel policy name for the
                                   VPLS PW. If no name is specified for a
                                   tunnel policy, the default tunnel policy
                                   is adopted. The LSP tunnel is preferred
                                   and only one LSP is used for load
                                   balancing in the default tunnel policy.
```

```
                                     If the name of the tunnel policy is
                                     specified but no tunnel policy is
                                     configured, the default tunnel policy is
                                     still adopted.";
                                config "true";
                                type string {
                                    length "1..39";
                                }
                            }

                            container vplsLdpPwInfo {
                                config "false";
                                uses vplsPwInfo;
                            }
                        }

                    }
                }

                container vplsBgpAdInst {

                    leaf vplsId {
                        description
                            "Specifies the vpls id. The value is a
                             case-sensitive string of 3 to 21 characters
                             without blank space.";
                        config "true";
                        type string {
                            length "1..21";
                        }
                    }

                    leaf bgpAdRd {
                        description
                            "Specifies the Route Distinguisher. The value is
                             a case-sensitive string of 3 to 21 characters
                             without blank space.";
                        config "false";
                        type string {
                            length "1..21";
                        }
                    }

                    leaf vsiId {
                        description
                            "Specifies the negotiation ip address of the
                             local PE.";
                        config "false";
                        type inet:ip-address;
                    }

                    container vpnTargets {
                        description "BGP-AD vpn-targets.";
                        list vpnTarget {
                            key "vpnRTValue";
                            description "BGP AD vpn targets.";

                            leaf vpnRTValue {

                              description
                                "Vpn-target:
                                 adds VPN target extended community attribute
                                 to the export or import VPN target extended
                                 community list. The vpn-target can be
                                 expressed in either of the following formats:
                                 (1)16-bit AS number:32-bit user-defined number
                                    For example, 1:3. The AS number ranges from
                                    0 to 65535. The user-defined number ranges
                                    from 0 to 4294967295. The AS number and the
                                    user-defined number cannot be 0s at the same
                                    time. That is, a VPN target cannot be 0:0.
                                 (2)32-bit IP address:16-bit user-defined number
                                    For example, 192.168.122.15:1. The IP addres
s

                                    ranges from 0.0.0.0 to 255.255.255.255. The
                                    user-defined number ranges from 0 to 65535."
;

                              mandatory "true";
                              type string {
                                length "3..21";
                              }
                            }

                            leaf vrfRTType {
                              description
                                "Specifies the vpn target type,
                                 export-extcommunity: specifies the extended
                                 community attributes carried in routing
                                 information to be sent. import-extcommunity:
                                 receives routing information carrying
                                 specified extended community attributes.";
                               mandatory "true";
                               type enumeration {
                               enum export_extcommunity {
                                 value "0";
                                 description "export-extcommunity:";
                               }
                               enum import_extcommunity {
                                 value "1";
                                 description "import-extcommunity:";
                               }
                               enum both {
                                 value "2";
                                 description
                                   "export-extcommunity &
                                    import-extcommunity:";
                               }
                             }
                           }
                        }
                    }

                    container bgpAdPeerInfos {
                        config "false";

                        list bgpAdPeerInfo {

                            key "peerRouterID";
                            leaf peerRouterID {
                                description
                                    "The Router ID of the remote router.";
                                type inet:ip-address;
                            }

                            leaf vplsId {
                                description
                                    "The vpls id. The value is a case-sensitive
                                     string of 3 to 21 characters without blank
                                     space.";
                                type string {
                                    length "1..21";
                                }
                            }

                            leaf sourceAII {
                                description
                                    "The source AII of the remote PE.";
                                type inet:ip-address;
                            }

                            leaf targetAII {
                                description
                                    "The target AII of the remote PE.";
                                type inet:ip-address;
                            }
                            leaf peerType {
                              description
                                "Specifies the peer type.Static,Dynamic.";

                              type enumeration {
                                    enum static {
                                      value "0";
                                      description "Static pw";
                                    }
                                    enum dynamic {
                                      value "1";
                                      description "Dynamic pw";
                                    }
                              }
                            }

                            container bgpAdPwInfo {
                                config "false";

                                uses vplsPwInfo;
                            }

                        }
                    }

                }

                container vplsBgpInst {

                    leaf bgpRd {
                        description
                            "Specifies the Route Distinguisher. The value is a
                            case-sensitive string of 3 to 21 characters without
                            blank space.";
                        type string {
                            length "1..21";
                        }
                    }

                    leaf ignoreMtu {
                        description
                            "Ignore the mtu when negotiate pw.";
                        type boolean;
                    }

                    container vpnTargets {
                        description "BGP vpn-targets";
                        list vpnTarget {
                            key "vpnRTValue";
                            description
                                "BGP vpn targets";

                            leaf vpnRTValue {

                              description
                                "Vpn-target: adds VPN target extended community
                                 attribute to the export or import VPN target
                                 extended community list. The vpn-target can be
                                 expressed in either of the following formats:
                                (1)16-bit AS number:32-bit user-defined number
                                   For example, 1:3. The AS number ranges from
                                   0 to 65535. The user-defined number ranges
                                   from 0 to 4294967295. The AS number and the
                                   user-defined number cannot be 0s at the same
                                   time. That is, a VPN target cannot be 0:0.
                                (2)32-bit IP address:16-bit user-defined number
                                   For example, 192.168.122.15:1. The IP address
                                   ranges from 0.0.0.0 to 255.255.255.255. The
                                   user-defined number ranges from 0 to 65535.";

                              mandatory "true";
                              type string {
                                length "3..21";
                              }
                            }

                            leaf vrfRTType {
                              description
                                "Specifies the vpn target type,
                                 export-extcommunity: specifies the extended
                                 community attributes carried in routing
                                 information to be sent.
                                 import-extcommunity: receives routing
                                 information carrying specified extended
                                 community attributes.";

                              mandatory "true";
                              type enumeration {
                                    enum export_extcommunity {
                                      value "0";
                                      description "export-extcommunity:";
                                    }
                                    enum import_extcommunity {
                                      value "1";
                                      description "import-extcommunity:";
                                    }
```

```
                                    enum both {
                                      value "2";
                                      description "export-extcommunity &
                                                   import-extcommunity:";
                                    }
                              }
                            }
                        }
                    }

                    container bgpSite {

                        leaf siteId {
                            description
                                "Specifies the ID of the site.";
                            mandatory "true";
                            type uint16 {
                                range "1..65535";
                            }
                        }
                        leaf siteRange {
                            description
                                "Specifies the ID of the site range.";
                            type uint16 {
                                range "1..65535";
                            }
                        }
                        leaf defaultOffset {
                            description
                                "Specifies the default offset of the site ID.";
                            type uint8 {
                                range "0..1";
                            }
                        }
                    }

                    container bgpPeerInfos {
                        config "false";

                        list bgpPeerInfo {

                            key "siteId";

                            leaf siteId {
                                description
                                    "The site id of the peer.";
                                type uint16 {
                                    range "1..65535";
                                }
                            }

                            container bgpPwInfo {
                                config "false";

                                uses vplsPwInfo;
                            }

                        }
                    }

                }

                container vplsAcs {

                    list vplsAc {

                        key "interfaceName";

                        leaf interfaceName {
                            description "Specifies the AC interface name. ";
                            config "true";
                            type leafref {
                                path "/if:interfaces/if:interface/if:name";
                            }
                        }
                        leaf hubModeEnable {
                            description
                                "Change the VSI attribute of the local
                                 interface from spoke to hub.By default,
                                 the AC side of a VSI has the attribute
                                 of spoke, and the PW side of a VSI has
                                 the attribute of hub.";
                            config "true";
                            default "false";
                            type boolean;
                        }
                        leaf state {
                            description
                                "Indicates the status of the AC.";
                            config "false";
                            type ifState;
                        }
                        leaf lastUpTime {
                            description
                                "Indicates how long the AC keeps the Up state.
                                 If the AC is currently in the Down state, the
```

```
                          value is 0.";
                      config "false";
                      type yang:date-and-time;
                  }
                  leaf totalUpTime {
                      description
                          "Indicates the total duration the AC is Up.";
                      config "false";
                      type string {
                          length "1..60";
                      }
                  }
                  container ifLinkProtocolTran {

                      description "lacp status";

                      leaf protocolLacp {
                          description "lacp status";
                          config "true";
                          type enumeration {
                              enum enable {
                                  value "0";
                                  description "enable:";
                              }
                              enum disable {
                                  value "1";
                                  description "disable:";
                              }
                          }
                      }
                      leaf protocolLldp {
                          description "lldp status";
                          config "true";
                          type enumeration {
                              enum enable {
                                  value "0";
                                  description "enable:";
                              }
                              enum disable {
                                  value "1";
                                  description "disable:";
                              }
                          }
                      }
                      leaf protocolBpdu {
                          description "bpdu status";
                          config "true";
                          type enumeration {
                              enum enable {
                                  value "0";
                                  description "enable:";
                              }
                              enum disable {
                                  value "1";
                                  description "disable:";
                              }
                          }
                      }
                      leaf protocolCdp {
                          description "cdp status";
                          config "true";
                          type enumeration {
                              enum enable {
                                  value "0";
                                  description "enable:";
                              }
                              enum disable {
                                  value "1";
                                  description "disable:";
                              }
                          }
                      }
                      leaf protocolUdld {
                          description "udld status";
                          config "true";
                          type enumeration {
                              enum enable {
                                  value "0";
                                  description "enable:";
                              }
                              enum disable {
                                  value "1";
                                  description "disable:";
                              }
                          }
                      }
                  }
              }

          }

          container vplsLoopDetectInfo {

              description "L2vpn admin vsi binding class.";

              leaf lastLoopType {
                      description
                          "Indicates the last mac withdraw loop type.";
                      config "false";
                      type enumeration {
                          enum detect-loop {
                              value "0";
                              description "detect loop:";
                          }
                          enum exceed-max-hop {
                              value "1";
                              description "exceed max hop:";
                          }
                      }
                  }
                  leaf sendPeer {
                      description
                          "Indicates the mac withdraw send peer.";
                      config "false";
                      type inet:ip-address;
                  }
                  leaf receivePeer {
                      description
                          "Indicates the mac withdrawreceive peer.";
                      config "false";
                      type inet:ip-address;
                  }
                  leaf lastLoopTime {
                      description
                          "Indicates the last mac withdraw loop time.";
                      config "false";
                      type string {
                          length "1..80";
                      }
                  }
              }

          }

      }

  }
}
</CODE ENDS>
```

5. IANA Considerations

This document makes no request of IANA.

6. Security Considerations

This document does not introduce any new security risk.

7. Acknowledgements

The authors would like to thank Guangying Zheng, Gang Yan for their contributions to this work.

8. References

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

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

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

Authors’ Addresses

Shunwan Zhuang
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: zhuangshunwan@huawei.com

Haibo Wang
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: rainsword.wang@huawei.com

Zhenbin Li
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: lizhenbin@huawei.com