

Pseudowire Concepts and troubleshooting

Contents

[Introduction](#)

[Prerequisites](#)

[Components Used](#)

[Pseudowire Concept](#)

[Troubleshooting a Pseudowire](#)

Introduction

Pseudowires(PW) are used to provide end-to-end services across an MPLS network. They are the basic building blocks that can provide a point-to-point service as well as a multipoint service such as VPLS, which is practically a mesh of PWs used to create the bridge domain across which the packets flow.

Edited by: Kumar Sridhar

Prerequisites

Readers of this document should be knowledgeable of the following:

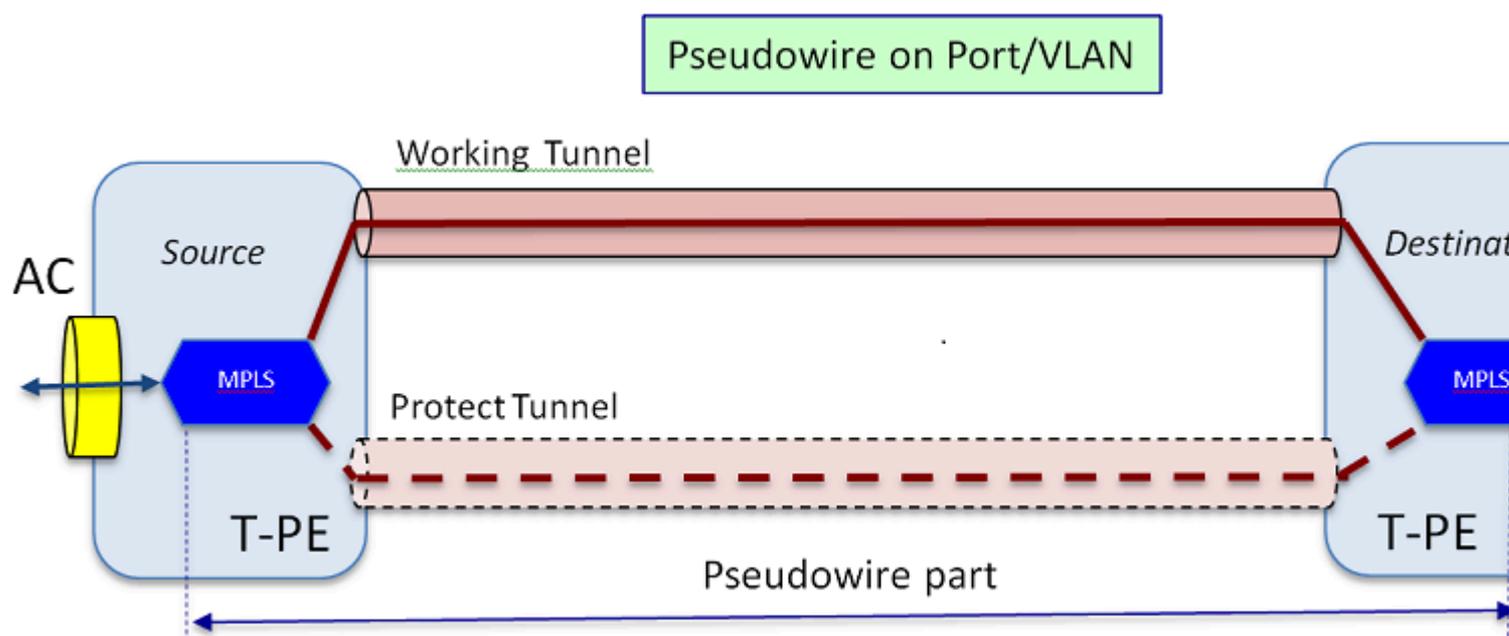
- MPLS Tunneling concepts

Components Used

The information in this document is based on The Cisco® Carrier Packet Transport (CPT) Product Family and in particular CPT50.

Pseudowire Concept

Pseudowires conceptually look as follows:



The end-to-end service is composed of 2 parts. The Attachment Circuit (AC) part and the Pseudowire part. The whole circuit end-to-end is still referred to as Pseudowire in Cisco Transport Controller(CTC), but bear in mind the two part distinction exhibited here for the troubleshooting that follows.

Also remember that a tunnel must have been created to house the Pseudowire service that is configured above. The tunnel may be protected (as depicted here) or unprotected.

Pseudowire part practically starts and stops at the tunnel end points (if you exclude the MPLS encapsulation block shown here).

The AC part starts from the tunnel end point all the way toward the client facing interface, where the Ethernet Flow Point (EFP) is defined, to identify the specific client traffic that is being transported through this Pseudowire. There are 2 ACs; one on each end.

The AC carries the customer traffic in its native form, i.e. Ethernet frames with or without VLAN tagging depending on whether we are creating a VLAN based Pseudowire or Ethernet Based Pseudowire (AC Type box in the PW creation wizard). The MPLS labels for the specific PW service as well as for the tunnel over which it is riding are then added. Packets are then sent across the Pseudowire part of the circuit into the MPLS cloud. This process is called Label Imposition in MPLS terminology. On the far end, the reverse process occurs, i.e. the labels are removed or the Label Disposition occurs, and the packets, which are now back to native Ethernet frames, are then delivered to the other end through the far end AC part of the Pseudowire circuit.

Troubleshooting a Pseudowire

For the Pseudowire service to work end to end, the Pseudowire part and the 2 AC parts have to work together. Troubleshooting the circuit involves each part, where each of the AC-PW-AC parts are debugged separately to identify where the problem is.

In the following troubleshooting discussion, it is assumed that the PW has been configured correctly, and all Layer 1 or physical layer issues have already been debugged and ruled out.

First, debugging the PW part is easy. Start by identifying the circuit through the command `show mpls l2 vc` run in IOS window on an end node. Note the Virtual Circuit Identifier(VCID) as well as the Destination node address of the connection.

```
10.88.130.201#show mpls l2 vc
```

Local intf Status	Local circuit	Dest address	VC ID
Gi36/2 UP	Eth VLAN 200	202.202.202.202	12
VFI vfi::1 UP	VFI	202.202.202.202	124
VFI vfi::1 UP	VFI	204.204.204.204	124

Here, the PW of interest is the first PW which was configured as VLAN 200 based on interface Gi36/2. Ensure the interface Status is UP.

show mpls l2 vc 12 detail command gives you a lot of information on the PW. Highlighted below are the important fields such as **tunnel id, remote node id, label stack, PWID number and statistics.**

```
10.88.130.201#show mpls l2 vc 12 detail
```

```
Local interface: Gi36/2 up, line protocol up, Eth VLAN 200 up
```

```
Destination address: 202.202.202.202, VC ID: 12, VC status: up
```

```
Output interface: Tp102, imposed label stack {16 19}
```

```
Preferred path: Tunnel-tp102, active
```

```
Default path: ready
```

```
Next hop: point2point
```

```
Create time: 00:32:52, last status change time: 00:05:42
```

```
Signaling protocol: Manual
```

```
Status TLV support (local/remote) : enabled/N/A
```

```
LDP route watch : enabled
```

```
Label/status state machine : established, LruRru
```

```
Last local dataplane status rcvd: No fault
```

```
Last BFD dataplane status rcvd: Not sent
```

```
Last local SSS circuit status rcvd: No fault
```

```
Last local SSS circuit status sent: No fault
```

```
Last local LDP TLV status sent: No fault
```

```
Last remote LDP TLV status rcvd: No fault
```

```
Last remote LDP ADJ status rcvd: No fault
```

```
MPLS VC labels: local 18, remote 19
```

```
PWID: 7
```

```
Group ID: local 0, remote 0
```

```
MTU: local 1500, remote 1500 <---- The local and  
remote values must match
```

```
Sequencing: receive disabled, send disabled
```

```
Control Word: On
```

```
SSO Descriptor: 202.202.202.202/12, local label: 18
```

```
SSM segment/switch IDs: 20513/12320 (used), PWID: 7
```

VC statistics:

transit packet totals: receive 10, send 0

transit byte totals: receive 1320, send 0

transit packet drops: receive 0, seq error 0, send 0

If the PW is down, then ensure the tunnel (here tunnel 102) is in good shape, and if not, then troubleshoot the tunnel issue. Troubleshooting the tunnel is beyond the scope of this article.

Ensure the labels in the stack are defined as shown above, i.e. they are not blank. Make sure the PW is programmed in the hardware by executing the command **show platform mpls pseudowire pwid** using the appropriate PWID number.

```
10.88.130.201#show platform mpls pseudowire pwid 7
```

PW Id: 7

PW VC Key: 7

PW AC Key: 786434

Is PW bind receive in HW: yes

Is PW setup in HW: yes

Is currently standby: no

--AC data --

Is AC Setup in HW:yes

AC interface : GigabitEthernet36/2

AC circuit id : 2

AC- Inner VLAN: 0

AC- Outer VLAN: 200

AC- MPLS Port Id: 0x1800000A

AC- Port Id: 31

AC- Mod Id: 36

AC- Is efp: yes

AC- Encap: Single Tag

AC- Ing RW Oper: none

AC- Egress RW Oper: none

AC- Ing RW TPID: 0
AC- Ing RW VLAN: 0
AC- Ing RW Flag: 0x0

--ATOM Data--

Interworking type: Vlan
Peer Requested Vlan id for type 4 PW 4091
MPLS Port Id: 0x1800000B
SD tag enabled : yes
Control Word enabled : yes

--Imposition data--

Remote VC label : 19

Outgoing int Num: 9
BCM port: 28
BCM ModId: 4
Tunnel egress object : 100008
Failover Id : 1
Failover Tunnel egress object : 100009
Failover BCM port: 0
Failover BCMModId: 0

--Disposition data--

Local label: 18

IF Num: 12
Is this MSPW : No

-- IMPOSITION SIDE --

Entry for VlanId 200 not found in VLAN_XLATE table

SOURCE_VP[10]

dvp: 11

ING_DVP_TABLE[11]

nh_index: 411

ING_L3_NEXT_HOP[411]

vlan_id: 4095

port_num: 28

module_id: 4

drop: 0

EGR_L3_NEXT_HOP[411]

mac_da_profile_index: 1

vc_and_swap_index: 4099

intf_num: 22

dvp: 11

EGR_MAC_DA_PROFILE[1]

DA Mac:

1

80.C20 .0 0

EGR_MPLS_VC_AND_SWAP_LABEL_TABLE[4099]

mpls_label(VC Label): 19

EGR_L3_INTF[22]

SA Mac:

4055.3958.E0E1

MPLS_TUNNEL_INDEX: 4

EGR_IP_TUNNEL_MPLS[4]

(lsp) MPLS_LABEL0

(lsp) MPLS_LABEL1

(lsp) MPLS_LABEL2

(lsp) MPLS_LABEL3

-- DISPOSITION SIDE --

MPLS_ENTRY[1592]

Label: 18

source_vp: 11

nh_index: 11

SOURCE_VP[11]

DVP: 10

ING_DVP_TABLE[10]

nh_index: 410

ING_L3_NEXT_HOP[410]

Port_num: 31

module_id: 36

drop: 0

EGR_L3_NEXT_HOP[410]

SD_TAG:VINTF_CTR_IDX: 134

SD_TAG:RESERVED_3: 0

SD_TAG:SD_TAG_DOT1P_MAPPING_PTR: 0

SD_TAG:NEW_PRI: 0

SD_TAG:NEW_CFI: 0

SD_TAG:SD_TAG_DOT1P_PRI_SELECT: 0

SD_TAG:RESERVED_2: 0

SD_TAG:SD_TAG_TPID_INDEX: 0

SD_TAG:SD_TAG_ACTION_IF_NOT_PRESENT: 0

SD_TAG:SD_TAG_ACTION_IF_PRESENT: 3

SD_TAG:HG_L3_OVERRIDE: 0

SD_TAG:HG_LEARN_OVERRIDE: 1

SD_TAG:HG_MC_DST_PORT_NUM: 0

SD_TAG:HG_MODIFY_ENABLE: 0

SD_TAG:DVP_IS_NETWORK_PORT: 0

```
SD_TAG:DVP: 10
```

```
SD_TAG:SD_TAG_VID: 0
```

```
ENTRY_TYPE: 2
```

```
Error: Entry not found in EGR_VLAN_XLATE table!
```

```
EGR_VLAN_XLATE[-1]
```

```
soc_mem_read: invalid index -1 for memory EGR_VLAN_XLATE
```

The logs indicate that the PW is bound and setup in the hardware, with the correct VLAN and labels, in agreement with what was seen before.

If any data point does not match or is missing, then the issue is in the driver, which did not setup and bind the PW in hardware. This points to a software or hardware defect.

If so far all is well, then you can try to ping the PW part internally by using the IOS command `ping mpls pseudowire 202.202.202.202 12 reply mode control-channel`. Note again that this pings the PW part only from one tunnel end point to the other and does not touch to the AC part of the circuit.

```
10.88.130.201#ping mpls pseudowire 202.202.202.202 12 reply mode control-channel
```

```
Sending 5, 100-byte MPLS Echos to 202.202.202.202,
```

```
timeout is 2 seconds, send interval is 0 msec:
```

```
Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
```

```
'L' - labeled output interface, 'B' - unlabeled output interface,
```

```
'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
```

```
'M' - malformed request, 'm' - unsupported tlvs, 'N' - no label entry,
```

```
'P' - no rx intf label prot, 'p' - premature termination of LSP,
```

```
'R' - transit router, 'I' - unknown upstream index,
```

```
'l' - Label switched with FEC change, 'd' - see DDMAP for return code,
```

```
'X' - unknown return code, 'x' - return code 0
```

```
Type escape sequence to abort.
```

```
!!!!!
```

```
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
```

```
Now check the statistics on the PW as we have done before:
```

```
10.88.130.201#show mpls 12 vc 12 det | beg statistics
```

VC statistics:

```
transit packet totals: receive 5, send 0
```

```
transit byte totals:   receive 650, send 0
```

```
transit packet drops: receive 0, seq error 0, send 0
```

Note that the ping was successful and that the 5 ping echo packets are recorded as received. Also, note that the ping request packets are not recorded as sent. It seems the echo request/reply packets are sent by the CPU into the stream post the counter, and thus are not recorded.

If the pings do not work, then we should step back and debug the tunnel to ensure it is operational.

If the PW part still looks good, then focus on the AC part on each end. This is the difficult part since there is not much debug support for it, and the AC path may include several cards and interfaces as in the case with Cisco CPT50.

But there are few things that can be checked.

You can send a pattern from a tester or do a ping from the client side equipment and watch for the packets being received by the client facing interface on the CPT box. This would be easy to do for a port based PW, but not for a VLAN based PW since the interface does not display packets per VLAN. In any case the command `show int` for the client facing interface should show packet count incrementing at least as a sign that packets are ingressing properly and if no other VLAN based circuits are active.

Bear in mind that these packets ingressing through the AC, are supposed to be MPLS labeled, and then sent across the PW to the other side. Thus, they should show in the statistics of the PW part as packets sent. So look for them in the command `show mpls l2 vc 12 detail | beg statistics`

```
10.88.130.201#show mpls l2 vc 12 detail | beg statistic
```

VC statistics:

```
transit packet totals: receive 0, send 232495
```

```
transit byte totals:   receive 0, send 356647330
```

```
transit packet drops: receive 0, seq error 0, send 0
```

And they should show as packets "receive" in the same command on the far end. So the send PW packets on this end and the receive PW packets on the far end should match the number of packets sent from the client equipment. Using the same command `show mpls l2 vc 12 detail | beg statistics` on the far end shows:

```
10.88.130.202#show mpls l2 vc 12 detail | beg statis
```

VC statistics:

```
transit packet totals: receive 232495, send 0
```

```
transit byte totals:   receive 356647330, send 0
```

```
transit packet drops: receive 0, seq error 0, send 0
```

You can see the match in the packets between the send on one end and receive on the other.

In case you need to clear the MPLS counters, use the command `clear mpls counters`.

Another way to check the statistics is to use the SPAN feature to replicate the incoming EFP traffic to a spare port on the CPT node and then look for the statistics on this port to monitor the packets received from the customer interface.

And finally you can run BCM shell commands on the different fabric and line cards to track the packets internally, but that is beyond the scope of this article.