Review the Resilient Ethernet Protocol

Contents

Introduction

This document describes an overview of Resilient Ethernet Protocol (REP).

Supported Platforms

- Desktop Switching Business Unit (DSBU) Metro Switches (3750ME and ME3400) Release 12.2(40)SE and later
- Cisco Catalyst 4500 Series Switch Release 12.2(44)SG and later
- Cisco Catalyst 6500 Series Switch Starting in Whitney2 (12.2SXI)
- Cisco Catalyst 7600 Series Router Starting in Cobra (12.2SRC)

Note: Use Cisco Feature Navigator to find information about platform support and Cisco software image support.

Background Information

Why REP

REP is a protocol used in order to replace the Spanning Tree Protocol (STP) in some specific Layer 2 network designs. The most current STP specification is Multiple Spanning Trees (MST), defined in 802.1Q-2005. Users who want an alternative to MST have these legitimate concerns:

- STP considers a bridged domain as a whole. As a result, a local failure is recovered if you change the state of an arbitrarily remote link. The apparent unpredictability of STP is only mitigated if you segment the bridged domain in small, independent pieces. Unfortunately, this is complex, if not impossible, to achieve without the removal of some key features from the Spanning Tree (like loop prevention in all scenarios).

- STP convergence can seem slow for service providers who expect recovery times of 50 milliseconds (ms), which is common in circuit-switching technologies. This slowness is not caused by the protocol itself; the platforms require optimization in order to run STP in a more efficient way. In the meantime, there need to be new solutions that work around platform limitations.

- The MST load-balancing configuration is not flexible. In order for MST to achieve instance load-balancing, all the bridges must be part of the same region. Regions are defined by user configuration, and there is no way to modify the MST configuration on a switch without the introduction of some reconvergence in the network. This could be worked around by careful pre-configuration, and to a limited extent, by the use of other protocols such as VLAN Trunk Protocol (VTP) v3.

Benefits

Here are some of the benefits of REP:

- REP offers these convergence times:
  - 3750ME converges between 20ms and 79ms
  - ME3400 converges between 40ms and 70ms
- Works on current hardware
- Predictable, blocked ports
- Easy configuration

Limitations

Here are some of the limitations of REP:

- No plug-and-play
- No protection against misconfiguration (easy to create loops)
- Limited amount of redundancy (only able to withstand one link failure)
- Cannot discover global topology (only segment topology)
- Cisco proprietary

Protocol Operation

Segments

REP uses a segment as a minimal network building block. A segment is simply a collection of ports chained together. Only two ports can belong to a given segment on a bridge, and each segment port can have a maximum of one external neighbor. The definition of the segment is entirely achieved by user configuration. The segment is terminated by two edge ports that are also determined by the user. The REP protocol that runs on segments is as minimal as possible and only guarantees these properties:

- If all the ports in the segment are online and operational, a single one of them logically blocks traffic for each VLAN.
- If at least one port in the segment is not operational for any reason, all the other operational ports forward for all the VLANs.
- In case of link failure, unblocking all the operational ports that remain is achieved as fast as possible. Similarly, when the last failed port becomes operational again, when one logically-blocked port per VLAN is elected it must cause disruption in the network as little as possible.

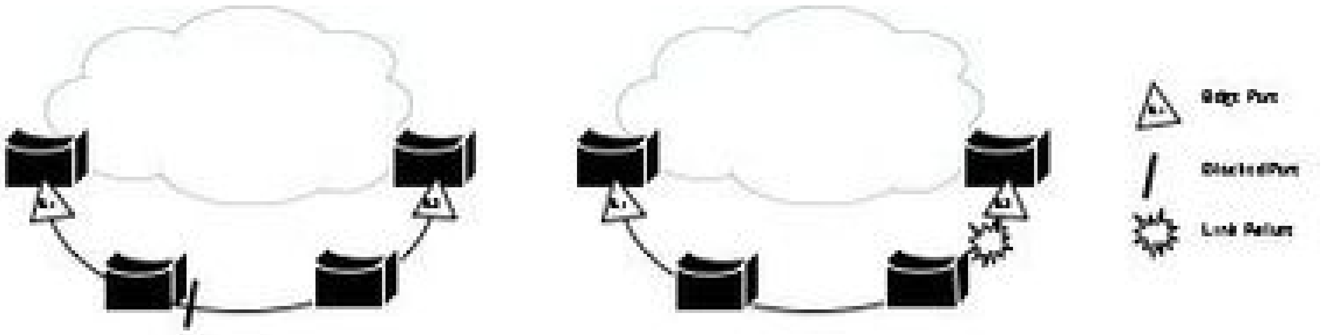

Image 1. A Segment as a Simple Building Block

Image 1 shows an example of a segment that includes six ports spread across four bridges. The configured edge ports E1 and E2 are represented with a triangle in the diagram, and the logically-blocked port is represented by a bar. When all the ports are operational, as pictured in the left, a single port is blocked. When there is a failure in the network, as shown in the diagram on the right, the logically-blocked port goes back to a forwarding state.

When the segment is open, as represented in Image 1, it never provides connectivity between its two edge ports. Connectivity between REP edge switches is assumed to be present outside of the segment (through STP). With optional configuration, a STP Topology Change Notification (TCN) is generated if a failure occurs in the REP segment in order to speed up convergence.

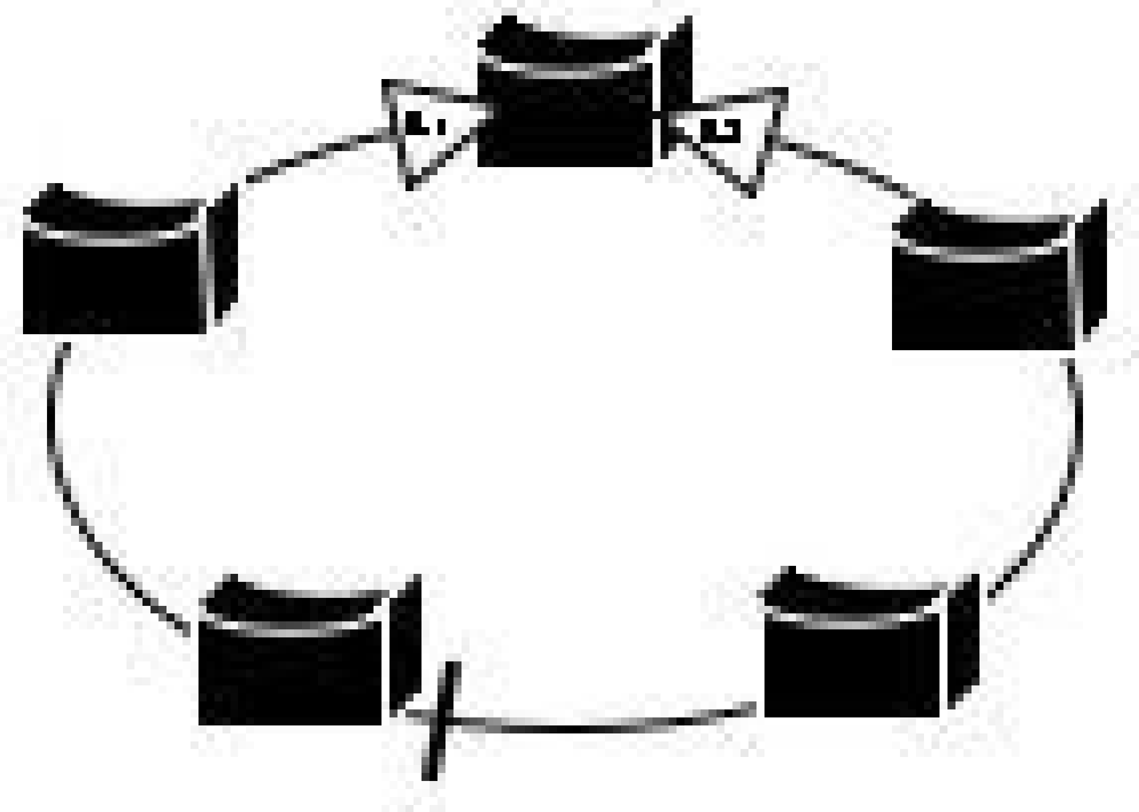

Image 2. A Segment can be Wrapped into a Ring

When the two edge ports are located on the same switch, as shown in Image 2, the segment is wrapped into a ring. In this case, there is connectivity between the edge ports through the segment. In fact, this configuration allows you to create a redundant connection between any two switches in the segment.

If you use combinations of open and closed segments, as represented in Image 1 and Image 2, you can achieve a variety of different network designs.

Link Status Layer

Responsibilities

- Establish connectivity with a unique neighbor.
- Periodically check the integrity of the connection with the neighbor.
- Send and receive messages for higher-layer state machines.
- Acknowledge data received from the neighbor.
- Limit rates of Protocol Data Units (PDUs).

Port States

When a port is configured for REP, it undergoes these states:

- Failed state (blocking)
- Neighbor relationship formed:
- Alternate port (blocking yet operational)
- Lost Access Point (AP) election:
- Open port (if a different port elected the 'AP')

A port does not become operational under these conditions:

- No neighbor detected on port
- More than one neighbor detected on port
- Neighbor does not acknowledge (ACK) the messages

Packet Detail

By default, REP sends hello packets to a Bridge Protocol Data Unit (BPDU) class MAC address on the native VLAN (untagged) so that they are dropped by devices that do not run the feature. Each Link Status Layer (LSL) PDU includes both a sequence number of the PDU that is sent and the remote sequence number of the last PDU received. This ensures reliable transmissions between ports. Each neighbor keeps a copy of each PDU sent until an ACK is received. If no ACK is received, it resends after a timer expires.

The actual LSL PDU contains:

- ProtocolVersion (currently 0)
- SegmentID
- RemotePortID
- LocalPortID
- LocalSeqNumber
- RemoteSeqNumber
- Higher Layer TLVs

LSL packets are sent at each hello interval, or when a higher-layer protocol requests it. When the LSL PDU is built, it first populates its own fields, such as SegmentID and LocalPortID. Next, it looks in the higher-layer protocol queues, such as Block Port Advertisement (BPA) or End Port Advertisement (EPA), in order to see if any additional data needs to be enqueued.

Hardware Flood Layer (HFL)

The HFL is the REP module that facilitates rapid convergence after link failures. It does not send PDUs to the BPDU MAC address like LSL, but rather, it sends multicast PDUs to a special MAC address

(0100.0ccc.ccce) on the REP administrative VLAN. This way, it is flooded in hardware to all switches in the segment.

The HFL packet format is simple:

- Protocol Version (still 0)
- SegmentID
- Higher Layer Type Length Values (TLVs)

At this time, the only TLVs sent via HFL are BPAs.

BPA

BPAs are sent by APs in order to advertise the VLANs they block along with their port priority. This helps notify the segment of link failures and ensures there is only a single AP per segment per VLAN. This is not easy to accomplish.

Considerations

In a stable topology, AP elections are simple. A port that comes online starts as an AP for all VLANs (blocking). When it receives a BPA from another port with a higher priority, it knows it can safely unblock. When a port on the segment fails, this same process is used in order to unblock the other ports. All failed ports generate a higher port priority (with a failed bit in the priority) than the current APs, which causes the current AP to unblock.

Problems happen, however, when this link comes back up. When this happens, the failed bit on the priority clears, and the priority returns to normal. Even though this port knows its new priority, other parts of the segment can have stale BPA information from this port. This diagram illustrates this scenario:

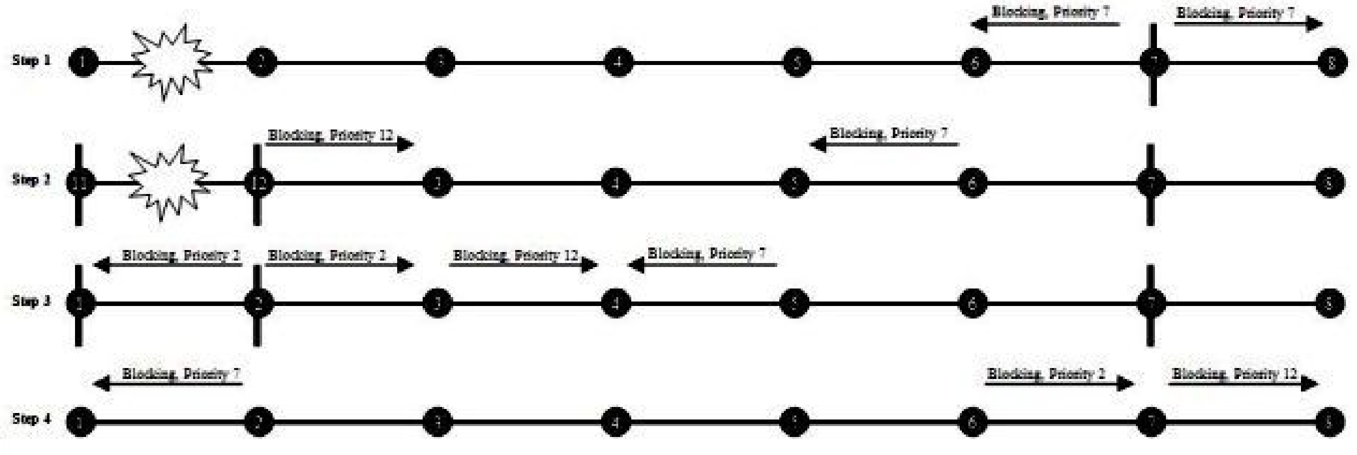

Image 3. Stale Information that Opens the Segment

At the beginning of this scenario, port 7 is blocking and advertises its priority as 7. Next, the link between 11 and 12 breaks, which causes 12 to send a BPA that indicates it is blocking with a priority of 12. Before these blocking ports receive the BPA of the other port, port 12 comes back up and is operational. Soon after, port 12 receives port 7's BPA with priority 7, so it unblocks. Port 7 then gets the stale BPA from port 12 with priority 12, so it unblocks. This causes a loop. This race condition is the reason BPA uses keys.

BPA Behavior

Each port calculates a port priority with this information:

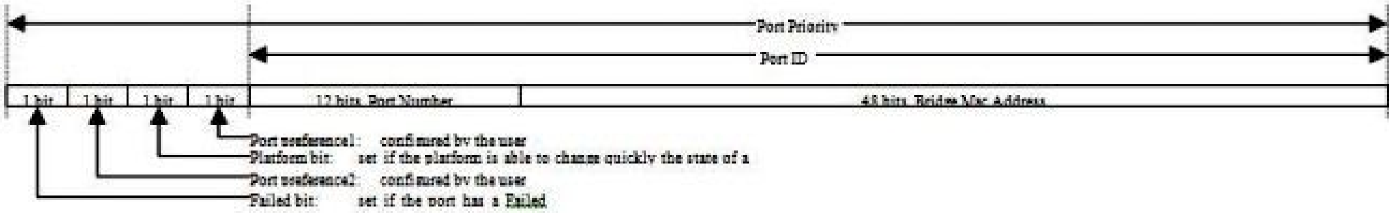
Port Priority
Port ID
1 bit
1 bit
1 bit
1 bit
12 bits Port Number
48 bits Bridge Mac Address
Port preference1: configured by the user
Platform bit: set if the platform is able to change quickly the state of a
Port preference2: configured by the user
Failed bit: set if the port has a Failed

Image 4. Port Priority

It is now apparent why failed ports are always elected APs on the segment. When a port moves from Failed to Alternate, it generates a unique key based on its port ID and a random number, and advertises it along with its port ID. An AP only unblocks if it receives a message from a blocked port that includes its local key. This mechanism helps prevent the race condition scenario described in the previous section. Here are diagrams that show what happens when ports come up and go down:

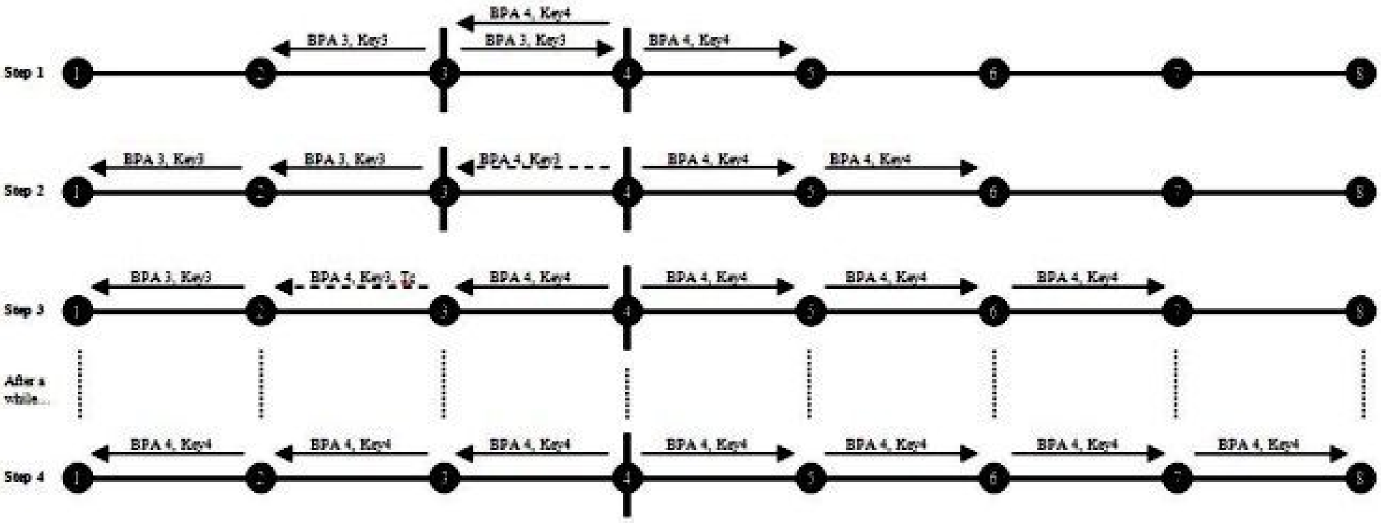

Image 5. BPA Operation at Link-Up

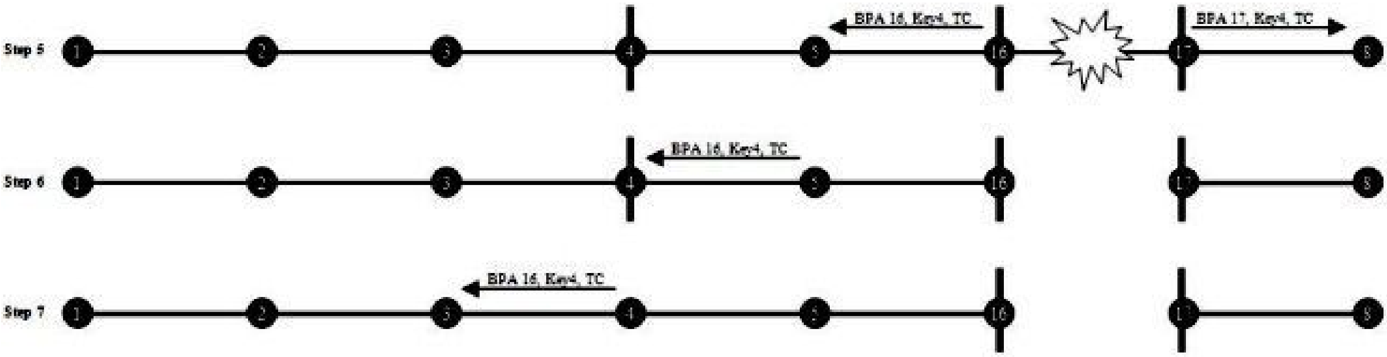

Image 6. BPA Operation after Link Failure

Hardware Assist

When a link failure occurs on a segment, a BPA is flooded to the rest of the segment through HFL. In order for this to be fully effective, the administrative VLAN must be carried on all segment ports, and it must be carried between edge ports outside of the segment. BPA also sends this information through LSL, because HFL cannot guarantee reliable transport. If there are any problems with HFL delivery, LSL makes sure reconvergence occurs.

EPA

An end port is either an edge port or a failed port. When a segment is terminated on both sides by an edge port, it is considered complete and VLAN load balancing is possible. When a segment is terminated by a failed port, no load-balancing is possible because all ports are open.

End ports periodically send EPAs which are relayed via LSL. These messages:

- Propagate statistics about the segment
- Detect the segment-complete condition
- Initiate VLAN load-balancing

Segment Statistics

Each end port sends a periodic EPA that contains information about itself through LSL. Each intermediate port adds its own information and relays the EPA. Since these messages move in both directions, each REP-participating switch has knowledge of the entire REP segment. The information contained in the EPA includes:

- Bridge ID
- Port ID and status for both REP-participating ports

Detect Segment Complete Condition

Each edge port sends a special election EPA message with its own edge priority and a special key (not related to the BPA key). The first port to receive this puts its own port priority in this message and relays it to the next switch. Each switch along the path compares its own port priority with the one in the EPA and replaces it with its own if the priority is higher. When the edge port receives an EPA, it compares the edge priority with its own. If the received EPA has a higher priority, the edge port sends its next EPA message with the key to the primary edge. This mechanism helps achieve two things:

- Ensures that the segment is complete
- Provides both edge ports with knowledge of the intermediate port with the highest priority

Initiate VLAN Load-Balancing

VLAN load-balancing is achieved with two different APs that block different VLANs. The primary edge is and is responsible for the AP on at least a subset of the VLANs, and it sends an EPA message that tells the highest priority port to block the rest. The information about the intermediate port with the highest priority was already fetched with the EPA election message. The type of message that is generated for this is an EPA command TLV that contains a bitmap of the VLANs that the highest-priority port needs to block.

PDU Format

EPA header:

- Type=EPA
- Instance #
- Optional TLVs

Election TLV:

- edgePriority

- edgeKey
- BestPortPriority

Command TLV:

- SelectedPortPriority
- SelectedVLANs

Information TLV:

- Bridge ID
- Two port IDs
- Port Roles

Troubleshoot

Broken Link Investigation

Here is an example of a good topology:

<#root>

```
SwitchA#
```

`show rep topology`

```
REP Segment 1
BridgeName       PortName   edge Role
---------------- ---------- ---- ----
SwitchA          Fa0/2      Pri  Alt
SwitchC          Fa1/0/23        Open
SwitchC          Fa1/0/2         Open
SwitchD          Fa0/23          Open
SwitchD          Fa0/2           Open
SwitchB          Fa1/0/23   Sec  Open
```

Here is an example where something is broken:

<#root>

```
SwitchA#
```

`show rep topology`

```
REP Segment 1
Warning: REP detects a segment failure, topology may be incomplete
BridgeName       PortName   edge Role
---------------- ---------- ---- ----
SwitchA          Fa0/2      Sec  Open
SwitchC          Fa1/0/23        Open
SwitchC          Fa1/0/2         Fail
```

Here is what it used to look like:

<#root>

```
SwitchA#
```

```
show rep topology archive
```

```
REP Segment 1
BridgeName       PortName   edge Role
---------------- ---------- ---- ----
SwitchA          Fa0/2      Pri  Open
SwitchC          Fa1/0/23        Open
SwitchC          Fa1/0/2         Open
SwitchD          Fa0/23          Open
SwitchD          Fa0/2           Open
SwitchB          Fa1/0/23   Sec  Alt
```

Enter this command in order to get more details on the link between SwitchC and SwitchD that failed:

<#root>

```
SwitchA#
```

```
show rep topology archive detail
```

```
REP Segment 1
<snip>
SwitchC, Fa1/0/2 (Intermediate)
  Open Port, all vlans forwarding
  Bridge MAC: 0017.5959.c680
  Port Number: 004
  Port Priority: 010
  Neighbor Number: 3 / [-4]
SwitchD, Fa0/23 (Intermediate)
  Open Port, all vlans forwarding
  Bridge MAC: 0019.e73c.6f00
  Port Number: 019
  Port Priority: 000
  Neighbor Number: 4 / [-3]
<snip>
```

Here is what it looks like after you bring the link back up:

<#root>

```
SwitchA#
```

```
show rep topology
```

```
REP Segment 1
BridgeName       PortName   edge Role
---------------- ---------- ---- ----
```

```
SwitchA          Fa0/2      Pri  Open
SwitchC          Fa1/0/23        Open
SwitchC          Fa1/0/2         Alt
SwitchD          Fa0/23          Open
SwitchD          Fa0/2           Open
SwitchB          Fa1/0/23   Sec  Open
```

Notice that the previously-failed port remains as the AP and continues to block. This is because AP elections only happen between blocked ports. When this link failed, all other ports in the topology opened. When the link came up, both SwitchC and SwitchD sent out BPAs with their priorities. SwitchC F1/0/2 had a higher priority, so it became the AP. This stays until another port in the topology fails, or until a preempt is performed.

Alternate (ALT) Ports

An ALT port blocks some or all VLANs. If there is a failure in the REP segment, there is no ALT port; all ports are open. This is how REP is able to provide an active path for the data traffic when there is failure.

In a complete REP segment (when there is no failure), there is either one ALT port, or there are two ALT ports. If VLAN load-balancing is enabled, then there are two ALT ports in the segment - one of the ALT ports blocks a specified set of VLANs, and the other ALT port, which is always at the primary edge, blocks the complementary set of VLANs. If VLAN load-balancing is not enabled, then there is single ALT port in the segment, which blocks all VLANs.

The order in which the ports come online, and the built-in port priorities determine which port in the segment becomes an ALT port. If you want a particular port to be the ALT port, configure it with the preferred keyword. Here is an example:

```
interface gig3/10
 rep segment 3 edge preferred
```

Suppose **gig3/1** is the primary edge, and you want to configure VLAN load-balancing:

```
interface gig3/1
 rep segment 3 edge primary
 rep block port preferred vlan 1-150
```

With this configuration, after preemption, port gig3/10 is an ALT port that blocks VLANs 1 through 150, and port gig3/1 is an ALT port that blocks VLANs 151 through 4094.

Preemption is done either manually with the **rep preempt segment 3** command, or automatically if you configure rep preempt delay <seconds> under the primary edge port.

When a segment heals after a link failure, one of the two ports adjacent to the failure comes up as the ALT port. Then, after preemption, the location of the ALT ports become as specified by the configuration.

Troubleshoot REP Adjacencies

Enter this command in order to see if there is an adjacency:

<#root>

```
SwitchC#

show interface fa1/0/23 rep


Interface              Seg-id Type           LinkOp      Role
---------------------- ------ -------------- ----------- ----

FastEthernet1/0/23     1                     TWO_WAY     Open
```

Enter this command in order to obtain more information:

<#root>

```
SwitchC#

show interface fa1/0/23 rep detail


FastEthernet1/0/23   REP enabled
Segment-id: 1 (Segment)
PortID: 001900175959C680
Preferred flag: No
Operational Link Status: TWO_WAY
Current Key: 000400175959C6808335
Port Role: Open
Blocked VLAN: <empty>
Admin-vlan: 1
Preempt Delay Timer: disabled
Configured Load-balancing Block Port: none
Configured Load-balancing Block VLAN: none
STCN Propagate to: none
LSL PDU rx: 255547, tx: 184557
HFL PDU rx: 3, tx: 2
BPA TLV rx: 176096, tx: 2649
BPA (STCN, LSL) TLV rx: 0, tx: 0
BPA (STCN, HFL) TLV rx: 0, tx: 0
EPA-ELECTION TLV rx: 870, tx: 109
EPA-COMMAND TLV rx: 2, tx: 2
EPA-INFO TLV rx: 45732, tx: 45733
```

Debugs

Most of the debugs print too much output to be useful. Here is the full list (some only available with service internal):

<#root>

```
SwitchB#

debug rep ?
```

```
  all               all debug options
  bpa-event         bpa events
  bpasm             BPA state machine
  database          protocol database
  epasm             EPA state machine
  error             protocol errors
  failure-recovery  switchover events
  lslsm             LSL state machine
  misc              miscellaneous
  packet            protocol PDU
  prsm              Port Role state machine
  showcli           show debug info
```

Useful Debugs

Here are some useful debugs:

- **debug rep showcli** (needs service internal) - This debug prints lots of extra information when you enter the regular **show rep** commands.
- **debug rep error** - This debug has the potential to be very useful.
- **debug rep failure-recovery** - This debug prints messages that go by when a link breaks.

```
*Mar  5 05:01:11.530: REP LSL-OP Rx EXT Local (Fa0/23 seg:1, tc:1, frs:0) prio:
*Mar  5 05:01:11.530: 0x80 0x00 0x19 0x00 0x17 0x59 0x59 0xC6
*Mar  5 05:01:11.530: 0x80
*Mar  5 05:01:11.530: REP Flush from Fa0/23 to REP, sending msg
*Mar  5 05:01:11.530: REP LSL-OP Rx INT Local (Fa0/2 seg:1, tc:1, frs:0) prio:
*Mar  5 05:01:11.530: 0x80 0x00 0x19 0x00 0x17 0x59 0x59 0xC6
*Mar  5 05:01:11.530: 0x80
*Mar  5 05:01:11.530: REP Flush from Fa0/2 to REP, sending msg
```

- **debug rep prsm** - This debug is good to troubleshoot adjacencies that do not form. It provides you with a play-by-play of what happens at link up/down.

```
4d05h: %LINK-3-UPDOWN: Interface FastEthernet0/2, changed state to up
4d05h: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/2,
changed state to up
*Mar  5 05:06:19.098:     rep_pr Fa0/2 - pr: during state FAILED_PORT,
got event 5(no_ext_neighbor)
*Mar  5 05:06:19.098: @@@ rep_pr Fa0/2 - pr: FAILED_PORT ->
FAILED_PORT_NO_EXT_NEIGHBOR[Fa0/2]rep_pr_act_no_ext_neighbor@272:
PRSM->fp_no_ext_neighbor state
[Fa0/2]rep_pr_lsl_event_handler@448: REP_MSG_EXT_PEER_GONE rcvd



4d05h: %REP-4-LINKSTATUS: FastEthernet0/2 (segment 1) is operational
*Mar  5 05:06:22.236:     rep_pr Fa0/2 - pr: during state FAILED_PORT_NO_EXT_
NEIGHBOR, got event 0(link_op)
```

```
*Mar  5 05:06:22.236: @@@ rep_pr Fa0/2 - pr:
FAILED_PORT_NO_EXT_NEIGHBOR ->
ALTERNATE_PORT[Fa0/2]rep_pr_act_ap@162: PRSM->alternate state
[Fa0/2]rep_pr_lsl_event_handler@431: REP_MSG_LINKOP_TRUE rcvd
```

```
*Mar  5 05:06:23.125:     rep_pr Fa0/2 - pr: during state ALTERNATE_PORT,
got event 2(pre_empt_ind)
*Mar  5 05:06:23.133: @@@ rep_pr Fa0/2 - pr: ALTERNATE_PORT -> UNBLOCK_VLANS_ACT
*Mar  5 05:06:23.133:     rep_pr Fa0/2 - pr: during state UNBLOCK_VLANS_ACT,
got event 3(no_local_block_vlan)
*Mar  5 05:06:23.133: @@@ rep_pr Fa0/2 - pr: UNBLOCK_VLANS_ACT ->
OPEN_PORT[Fa0/2]rep_pr_act_op@252: PRSM->active state
[Fa0/2]rep_pr_act_uva@222: PRSM unblock vlans
[Fa0/2]rep_pr_sm_prempt_ind@374: Posting pre empt indication
```

- **debug rep epasm -** This debug provides useful information during topology changes. Nothing is printed if the segment is stable.

Here is the output if a port goes offline:

```
*Mar  5 04:48:31.463:     rep_epa_non_edge Fa0/2 - epa-non-edge: during state
INTERMEDIATE_PORT, got event 1(lr_eq_fp)*Mar  5 04:48:31.463: @@@ rep_epa_non_
edge Fa0/2 - epa-non-edge: INTERMEDIATE_PORT -> FAILED_PORT[Fa0/2]rep_epa_non_
edge_act_failed_port@164:  Trigger archiving
[Fa0/23]rep_epa_set_peer_archive_flag@1084: set arch flag
[Fa0/2]rep_epa_non_edge_act_failed_port@171: no edge, failed port
*Mar  5 04:48:35.473:     rep_epa_non_edge Fa0/2 - epa-non-edge: during state
FAILED_PORT, got event 0(epa_hello_tmo)
*Mar  5 04:48:35.473: @@@ rep_epa_non_edge Fa0/2 - epa-non-edge: FAILED_PORT ->
FAILED_PORT[Fa0/2]rep_epa_non_edge_act_periodic_tx@90: archiving on port down
[Fa0/2]rep_epa_copy_topology@913: deip=0x3396F18,pe=0,se=1,fp=0,ap=0,op=2
[Fa0/23]rep_epa_non_edge_handle_info_tlv@1560: archiving on internal flag
[Fa0/23]rep_epa_copy_topology@913: deip=0x33961F0,pe=1,se=0,fp=0,ap=1,op=3
[Fa0/2]rep_epa_non_edge_act_periodic_tx@102: epa non edge, send info tlv
[Fa0/23]rep_epa_set_peer_archive_flag@1084: set arch flag
[Fa0/2]rep_epa_non_edge_handle_election_tlv@325: archiving on seg cfg change
[Fa0/2]rep_epa_copy_topology@913: deip=0x3396F18,pe=0,se=1,fp=0,ap=0,op=2
[Fa0/2]rep_epa_set_peer_archive_flag@1084: set arch flag
[Fa0/23]rep_epa_non_edge_handle_election_tlv@325: archiving on seg cfg change
[Fa0/23]rep_epa_copy_topology@913: deip=0x33961F0,pe=1,se=0,fp=0,ap=1,op=3
[Fa0/2]rep_epa_non_edge_handle_info_tlv@1560: archiving on internal flag
[Fa0/2]rep_epa_copy_topology@913: deip=0x3396F18,pe=0,se=1,fp=0,ap=0,op=2
```

Here is the output when a port is comes online:

```
*Mar  5 04:49:39.982:     rep_epa_non_edge Fa0/2 - epa-non-edge: during state FAILED_PORT,
got event 2(lr_neq_fp)
*Mar  5 04:49:39.982: @@@ rep_epa_non_edge Fa0/2 - epa-non-edge: FAILED_PORT ->
INTERMEDIATE_PORT[Fa0/2]rep_epa_non_edge_stop_timer@123: epa non edge, stop the timer
[Fa0/2]rep_epa_copy_topology@913: deip=0x32E2FA4,pe=0,se=1,fp=0,ap=1,op=1
[Fa0/2]rep_epa_copy_to_stable_topology@1040: copy to stbl
```

```
[Fa0/23]rep_epa_copy_topology@913: deip=0x3ACFFB8,pe=1,se=0,fp=0,ap=0,op=4
[Fa0/23]rep_epa_copy_to_stable_topology@1040: copy to stbl
```

Less Useful Debugs

- **debug rep bpa-event** - This debug tells you when you receive a BPA, and what you do with it. It has four lines per second.

```
[Fa0/23]: BPA: Sending ext pak to bparx
[Fa0/2]: BPA: Enqueued internal pak
[Fa0/2]: BPA: Sending int msg to bparx
[Fa0/2]: BPA: Relay pak
[Fa0/2]: BPA: Enqueue ext pak
```

- **debug rep bpasm** - This debug tells you what the BPA state machine does whenever a BPA is received. It has three lines per second.

```
*Mar  5 04:44:23.857:     rep_bpa_rx BPA RX sm: during state BPA_RX_IDLE,
got event 0(bpa_rx_bpa_msg)
*Mar  5 04:44:23.857: @@@ rep_bpa_rx BPA RX sm: BPA_RX_IDLE -> BPA_RX_IDLE
[Fa0/23]: BPA Rx sm: Received bpa: <internal> 0, <vlan_detail> 0
[Fa0/23]: BPA Rx sm: Role 2: TC 0; Internal 0; Frm Remote Segment 0



*Mar  5 04:44:23.857:     rep_bpa_rx BPA RX sm: during state BPA_RX_IDLE,
got event 0 (bpa_rx_bpa_msg)
*Mar  5 04:44:23.857: @@@ rep_bpa_rx BPA RX sm: BPA_RX_IDLE -> BPA_RX_IDLE
[Fa0/2]: BPA Rx sm: Received bpa: <internal> 1, <vlan_detail> 0
[Fa0/2]: BPA Rx sm: Role 2: TC 0; Internal 1; Frm Remote Segment 0
```

- **debug rep lslsm -** This debug dumps low-level LSL message processing.

```
*Mar  5 05:03:10.564: REP Fa0/23 seq:4411 ACK'ed (ref: 1)
*Mar  5 05:03:10.564: REP Fa0/23 seq:4412 ACK'ed (ref: 1)
*Mar  5 05:03:10.564: REP LSL: Fa0/23 rx expected seq# (4744),
process it (TLV: 0).

*Mar  5 05:03:10.782: REP Fa0/2 seq:440 ACK'ed (ref: 1)
```

Related Information

- Cisco Technical Support & Downloads