Microwave ACM Signaling Configuration and EEM Integration

This feature module describes the Microwave Adaptive Code Modulation (ACM) Signaling and Embedded Event Manager (EEM) integration, which enables the microwave radio transceivers to report link bandwidth information to an upstream Ethernet switch and take action on the signal degradation to provide optimal bandwidth.

Prerequisites

- The microwave transceiver in the network topology must support adaptive bandwidth modulation and bandwidth vendor specific message (BW-VSM)/Ethernet Bandwidth Notification Message (ETH-BNM), and the microwave transceiver must support the Ethernet Connectivity Fault Management (CFM) extension for microwave devices as defined by Cisco. The BW-VSM/ETH-BNM is defined to report the available bandwidth information from the microwave radio to the Ethernet switch.
- In a heterogeneous ring topology, all devices connected directly to the microwave transceiver must support signal degradation (SD) functions. Devices not connected directly to the microwave transceiver can be standard-compliant nodes or enhanced SD-capable nodes.
- In a homogeneous ring topology, all links must be microwave links and all devices must support microwave SD-based ring protection.
- A ring topology with multiple microwave links can experience a signal degradation condition on one or more of the microwave links. Only one signal degradation condition per ring instance is supported. This support is provided on a first-come, first-serve basis, per ring instance.
- The source MAC address must be an unique MAC address. It can be the MAC address of the Ethernet port or the Bridge.
- The destination MAC address must be set to the CCM multicast address for the associated maintenance level (a multicast address is used to avoid discovery of MAC addresses).
- The microwave transceiver in the network topology must support bandwidth vendor specific message (BW-VSM) (The BW-VSM is defined to report the available bandwidth information from the microwave radio to the Ethernet switch.).
- The BW-VSM/ETH-BNM may be sent untagged, or it may be transmitted with a configurable valid IEEE 802.1Q VLAN tag.

- The BW-VSM/ETH-BNM must be associated with maintenance level 0. The microwave equipment should allow the network operator to associate the message with a valid maintenance level in the range 0 to 7 per ITU-T Y.1731 / IEEE 802.1ag-2007.

Benefits

- The IP/MPLS access network adapts intelligently to the microwave capacity change by:
  - optimizing routing
  - controlling congestion
  - enabling loss protection.
- Microwave ACM changes are signaled through a Y.1731 VSM/G.8031/Y.1731 (ETH-BNM) to the IP/MPLS access node.
- The IP/MPLS access node adapts the IGP metric of the link to the new capacity.
- The IP/MPLS access node can change the H-QOS policy on the interface with the microwave system allowing EF traffic to survive.
- The IP/MPLS access node can remove a degraded link from SPF triggering a loss protection.

Microwave ACM Signaling Configuration and EEM Integration

This section describes how to configure Microwave ACM Signaling and EEM Integration:

Configuring Connectivity Fault Management

To configure CFM between the microwave outdoor unit (ODU) and the router, complete the following steps:

Note For a ring topology, you should configure CFM between the microwave ODU and the router. You must configure two VLANs to the two microwave ODUs, to process the vendor specific message (VSM)/Ethernet Bandwidth Notification Message (ETH-BNM) and trigger the Embedded Event Manager (EEM).

Procedure

Step 1 **enable**

Example:

```
Router> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Router# configure terminal
```

Enters global configuration mode.

Step 3 **ethernet cfm domain** *domain-name* **level** *level-id*

Example:

```
Router(config)# ethernet cfm domain outer level 3
```

Defines a CFM maintenance domain at a particular maintenance level and enter Ethernet CFM configuration mode.

- domain-name—String of a maximum of 154 characters that identifies the domain.
- level-id—Integer from 0 to 7 that identifies the maintenance level.

Step 4 **service** *csi-id* **evc** *evc-name* **vlan** *vlan-id* **direction down**

Example:

```
Router(config-ether-cfm)# service microwave1 evc V60 vlan 60 direction down
```

Sets a universally unique ID for a customer service instance (CSI) within a maintenance domain.

- csi-id—String of a maximum of 100 characters that identifies the CSI.
- evc—Specifies the EVC.
- evc-name—String that identifies the EVC.
- vlan—Specifies the VLAN.
- vlan-id—String that identifies the VLAN ID. Range is from 1 to 4094.
- direction—Specifies the service direction.
- down—Specifies the direction towards the LAN.

Step 5 **continuity-check**

Example:

```
Router(config-ecfm-srv)# continuity-check
```

Enables the transmission of continuity check messages (CCMs).

Step 6 **exit**

Example:

```
Router(config-ecfm-srv)# exit
```

Exits Ethernet CFM service configuration mode and enters global configuration mode.

Step 7 **ethernet evc** *evc-id*

Example:

```
Router(config)# ethernet evc V60
```

Defines an EVC and enters EVC configuration mode.

- evc-id—String from 1 to 100 characters that identifies the EVC.

Step 8 **exit**

Example:

```
Router(config-evc)# exit
```

Exits Ethernet EVC configuration mode and enters global configuration mode.

Step 9 **interface** *type number*

Example:

```
Router(config)# interface GigabitEthernet0/0/1
```

Specifies an interface type and number, and enters interface configuration mode.

Step 10 **service instance** *id* **ethernet**

Example:

```
Router(config-if)# service instance 60 ethernet 60
```

Configures an Ethernet service instance on an interface.

- id—Integer that uniquely identifies a service instance on an interface.

Step 11 **encapsulation dot1q** *vlan-id*

Example:

```
Router(config-if)# encapsulation dot1q 60
```

Enables IEEE 802.1Q encapsulation of traffic on a specified interface in a VLAN.

- vlan-id—Virtual LAN identifier.

Step 12 **rewrite ingress tag pop 1 symmetric**

Example:

```
Router(config-if)# rewrite ingress tag pop 1 symmetric
```

Specifies the encapsulation adjustment to be performed on a frame ingressing a service instance.

- pop—Removes a tag from a packet.
- 1—Specifies the outermost tag for removal from a packet.

- symmetric—Indicates a reciprocal adjustment to be done in the egress direction. For example, if the ingress pops a tag, the egress pushes a tag and if the ingress pushes a tag, the egress pops a tag.

Step 13 **bridge-domain** *bridge-domain-id*

Example:

```
Router(config-if)# bridge-domain 60
```

Enables RFC 1483 ATM bridging or RFC 1490 Frame Relay bridging to map a bridged VLAN to an ATM permanent virtual circuit (PVC) or Frame Relay data-link connection identifier (DLCI).

- bridge-domain-id—Bridge domain identifier.

Step 14 **exit**

Example:

```
Router(config-if)# exit
```

Exits interface configuration mode.

Configuring an Embedded Event Manager Applet

Before you begin

- One switch virtual interface (SVI) or bridge domain is required per physical link.
- One EEM script is required per physical link.
- A dedicated line VTY without AAA is required for the EEM script to perform without any interruption.

Note The EEM script configures the metric on the microwave link and adjusts the QoS policy based on the Ethernet event parameters.

EEM built-in environment variables are a subset of the Cisco-defined environment variables and the built-in variables are available to EEM applets only. The built-in variables can be read-only or can be read and write and these variables may apply to one specific event detector or to all event detectors. For more information about built-in environment variables, see Embedded Event Manager Configuration Guide, Cisco IOS XE Release 3S.

Procedure

Step 1 **enable**

Example:

```
Router> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Router# configure terminal
```

Enter global configuration mode.

Step 3 **event manager applet** *applet-name*

Example:

```
Router(config)# event manager applet ACM61
```

Registers an applet with the Embedded Event Manager (EEM) and enters applet configuration mode.

- *applet-name*—Name of the applet file.

Step 4 **event tag** *event-tag* **ethernet microwave clear-sd** {*interface type number*}

Example:

```
Router(config-applet)# event tag event_cd ethernet microwave clear-sd interface
GigabitEthernet0/0/1
```

Specifies the event criteria for an EEM applet that is run by matching a Cisco IOS command-line interface (CLI).

- *event-tag* —Specifies a tag using the event-tag argument that can be used with the trigger command to support multiple event statements within an applet.

Step 5 **event tag** *event-tag* **ethernet microwave sd** {*interface type number*} **threshold** *mbps*

Example:

```
Router(config-applet)# event tag event_sd ethernet microwave sd interface GigabitEthernet0/0/1
 threshold 1000
```

Specifies the event criteria for an EEM applet that is run by matching a Cisco IOS CLI.

Step 6 **action** *action-id* **set** *variable-name variable-value*

Example:

```
Router(config-applet)# action 110 set ifname "vlan $_svi61"
```

Sets the value of a variable when an EEM applet is triggered.

- *action-id*—Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
- *variable-name*—Name assigned to the variable to be set.
- *variable-value*—Value of the variable.

Step 7 **action** *action-id* **cli command** *cli-string*

Example:

```
Router(config-applet)# action 458 cli command "event manager applet ACM61"
```

Specifies the action of executing a Cisco IOS CLI when an EEM applet is triggered.

- *action-id*—Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
- **cli command**—Specifies the message to be sent to the Cisco IOS CLI.
- *cli-string* —CLI string to be executed. If the string contains embedded blanks, enclose it in double quotation marks.

Step 8 **exit**

Example:

```
Router(config-applet)# exit
```

Exits applet configuration mode.

Configuring Event Handler

To configure the microwave event handler, which runs hold-off timer, loss threshold, and fading wait-to-restore (WTR) timers that are configurable per interface, complete the following steps:

Procedure

Step 1 **enable**

Example:

```
Router> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Router# configure terminal
```

Enters global configuration mode.

Step 3 **interface** *type number*

Example:

```
Router(config)# interface vlan 40
```

Specifies an interface type and number, and enters interface configuration mode.

Step 4 **ethernet event microwave hold-off** *seconds*

Example:

```
Router(config-if)# ethernet event microwave hold-off 30
```

Configures the settings of the Ethernet microwave event.

- hold-off—Specifies the microwave bandwidth degradation hold-off time, in seconds. This time is used to prevent changes in the state of the network node as a result of signal degradation (SD) occurrences.
- seconds—Hold off time, in seconds. The valid values range from 0 to 600, with a default value of 0.

Step 5 **ethernet event microwave loss-threshold** *number-of-messages*

Example:

```
Router(config-if)# ethernet event microwave loss-threshold 100
```

Configures the settings of the Ethernet microwave event.

- loss-threshold—Specifies the number of bandwidth Vendor-Specific Messages (VSM)/Ethernet Bandwidth Notification Message (ETH-BNM) sent from the microwave transceiver to the Cisco device.
- number-of-messages—Number of bandwith VSMs/ETH-BNMs. The valid values range from 2 to 255, with a default value of 3.

Step 6 **ethernet event microwave wtr** *seconds*

Example:

```
Router(config-if)# ethernet event microwave wtr 45
```

Configures the settings of the Ethernet microwave event.

- wtr—Specifies the wtr time. This time is used to prevent changes in the state of the network node as a result of recovery events after an SD occurrence.
- seconds—WTR time, in seconds. The valid values range from 0 to 600, with a default value of 10.

Verifying the Microwave ACM Signaling and EEM Integration Configuration

To verify the microwave ACM and EEM integration configuration, use the show commands described in the following examples.

To display microwave bandwidth status information of an interface, use the following show command.

```
Router# show ethernet event microwave status [interface]
Microwave Bandwidth Status for GigabitEthernet0/0/1
  State:                    SIGNAL_DEGRADED
  Hold Time:                0 seconds
  Restore Time:             10 seconds
  Loss Threshold:           3
```

```
  Total VSM Receive Count:  0
  Total VSM Drop Count:     0
  Total BNM Receive Count:  4
  Total BNM Drop Count:     0
  Sender Address 64f6.9d67.ac02
    State:                          SIGNAL_DEGRADED
    Elapsed time in this state:     00:00:25
    Nominal Bandwidth:              500 Mbps
    Current Bandwidth:              100 Mbps
    Lowest Bandwidth:               100 Mbps
    Last VSM Received:              Never
    VSM Receive Count:              0
    VSM Drop Count:                 0
    VSM Period:                     60 second
    Last BNM Received:              Tue Jul 25 14:36:09.895
    BNM Receive Count:              4
    BNM Drop Count:                 0
    BNM Period:                     60 seconds
    Hold Timer:                     Not running
    Wait-to-Restore Timer:          Not running
    Periodic Timer:                 184 seconds remaining
    Transitions into degraded state: 4
```

To display microwave bandwidth statistics of an interface, use the following show command.

```
Router# show ethernet event microwave statistic [interface]

Microwave Bandwidth Statistics for GigabitEthernet0/0/2
 Total VSM Receive Count : 145
 Total VSM Drop Count : 0
 Number of transitions into Degraded state : 2
```

Configuration for Microwave ACM Signaling and EEM Integration Examples

Sample configurations of Microwave ACM Signaling and EEM Integration feature.

Example: Configuring CFM

The following is a sample configuration of CFM.

```
!
ethernet cfm domain outer level 3
service microwave1 evc V60 vlan 60 direction down
  continuity-check
!
ethernet evc V60
!
interface GigabitEthernet0/0/1
!
service instance 60 ethernet V60
  encapsulation dot1q 60
  rewrite ingress tag pop 1 symmetric
  bridge-domain 60
!
```

Example: Configuring EEM Applet

The following is a sample EEM script to configure metric on a microwave link and adjust a QoS policy according to the ethernet event parameters sent through OAM.

Note You should have one SVI/BD per physical link. Also, one EEM script is required per physical link. In all, there should be two EEM scripts and two SVI/BDs.

Note The threshold in the EEM script should be set to the nominal bandwidth value. If this value is unknown, we recommend setting the threshold to 1000. The EEM script adjusts the nominal bandwidth using the following vendor-specific message (VSM)/Ethernet Bandwidth Notification Message (ETH-BNM): **action 460 cli command "event tag event_sd ethernet microwave sd interface GigabitEthernet0/3/0 threshold $nb"**

Note The EEM script supports the **bandwidth percent** command, but does not support the **bandwidth remaining percent** command.

```
no event manager applet ACM62
Router#show run | sec event manager
event manager environment _eem_mode 1
event manager environment _bdi60 60
event manager environment _ring_nodes 5
event manager applet ACM62
 event tag event_cd ethernet microwave clear-sd interface GigabitEthernet0/0/1
 event tag event_sd ethernet microwave sd interface GigabitEthernet0/0/1 threshold 400
 trigger
  correlate event event_cd or event event_sd
 action 100 set olc "100"
 action 102 set dlc "1"
 action 104 set n "$_ring_nodes"
 action 106 set cb "$_ethernet_current_bw"
 action 108 set nb "$_ethernet_nominal_bw"
 action 110 set ifname "bdi $_bdi60"
 action 112 set cpmap_bw "0"
 action 114 set pri_bw "0"
 action 116 set ppmap "0"
 action 118 set s1 "EEM-"
 action 120 set zeros "000000"
 action 122 set cb_bps "$cb$zeros"
 action 124 set nb_bps "$nb$zeros"
 action 126 set ifcfg "1"
 action 127 set class-type "0"
 action 130 cli command "enable"
 action 132 cli command "conf t"
 action 160 if $cb eq "$nb"
 action 162  cli command "interface $_ethernet_intf_name"
 action 163  cli command "no service-policy output $s1$ppmap"
 action 164  cli command "service-policy output $ppmap"
 action 180 elseif $_eem_mode le 1
 action 181  if $ppmap eq "0"
```

```
action 182   cli command "do show run int $_ethernet_intf_name | i service-policy output"
action 186   regexp "service-policy output (.*)\n" "$_cli_result" line pmap
action 192   string trimright "$pmap"
action 196   set pmap "$_string_result"
action 197  else
action 198   set pmap "$ppmap"
action 199  end
action 200  syslog msg "s1pmap 200: $s1$pmap"
action 214  cli command "do show run policy-map $pmap | i service-policy"
action 216  regexp "service-policy (.*)\n" "$_cli_result" line cpmap
action 217  string trimright "$cpmap"
action 218  set cpmap "$_string_result"
action 220  cli command "do show run policy-map $cpmap"
action 221  regexp "class .*!" "$_cli_result" string
action 223  cli command "policy-map $s1$cpmap"
action 226  foreach var "$string" "\n"
action 228   regexp "class (.*)" "$var" match cname
action 230   if $_regexp_result eq "1"
action 233    syslog msg "233: cname: $cname"
action 234   end
action 236   regexp "(police) (.*)" "$var" line ef_bw_perc
action 238   if $_regexp_result eq "1"
action 256    string trimright "$ef_bw_perc"
action 263    set bw_demand "$_string_result"
action 264    add $cpmap_bw $_string_result
action 266    syslog msg "266: cpmap_bw: $_result, bw_demand: $bw_demand"
action 268    set cpmap_bw "$_result"
action 274    add $pri_bw $bw_demand
action 282    set match1 "police $bw_demand"
action 283    set match2 "police $bw_demand"
action 284    set class-type "1"
action 286   end
action 288   regexp "(bandwidth) percent (.*)" "$var" line cmd ef_bw_perc
action 290   if $_regexp_result eq "1"
action 291    string trimright "$ef_bw_perc"
action 294    divide $nb_bps 100
action 296    multiply $_result $_string_result
action 298    set bw_demand "$_result"
action 300    add $cpmap_bw $_result
action 302    syslog msg "266: cpmap_bw: $_result, bw_demand: $bw_demand"
action 304    set cpmap_bw "$_result"
action 306    syslog msg "269: cpmap_bw sub-sum: $cpmap_bw"
action 308    set match1 "$match"
action 310    set match2 "bandwidth percent 1"
action 312    set class-type "2"
action 314   end
action 316   if $class-type eq "1"
action 318    append cfg_out1 "priority"
action 320    append cfg_out1 "$match1 \n"
action 322    append cfg_out2 "priority"
action 324    append cfg_out2 "$match2 \n"
action 325    set class-type "0"
action 326   elseif $class-type eq 2
action 328    append cfg_out1 "$match1 \n"
action 330    append cfg_out2 "$match2 \n"
action 331    set class-type "0"
action 332   else
action 334    append cfg_out1 "$var \n"
action 336    append cfg_out2 "$var \n"
action 338   end
action 340   syslog msg "310: cpmap_bw sum: $cpmap_bw"
action 342   if $cpmap_bw lt "$cb_bps"
action 344    set cfg_out "$cfg_out1"
action 346   elseif $pri_bw lt $cb_bps
```

```
 action 348    set cfg_out "$cfg_out2"
 action 350   else
 action 352    set metric "1000000"
 action 354    set ifcfg "0"
 action 356   end
 action 358   if $ifcfg eq "1"
 action 360    foreach var "$cfg_out" "\n"
 action 362     cli command "$var"
 action 364    end
 action 366   end
 action 367  end
 action 368  cli command "policy-map $s1$pmap"
 action 370  syslog msg "config 334: policy-map $s1$pmap"
 action 372  cli command "class class-default"
 action 374  cli command "shape average $cb_bps"
 action 376  cli command "service-policy $s1$cpmap"
 action 378  cli command "int $_ethernet_intf_name"
 action 380  cli command "no service-policy output $pmap"
 action 382  cli command "service-policy output $s1$pmap"
 action 384 end
 action 400 if $_eem_mode ge "1"
 action 402  multiply $n $cb
 action 404  divide $_result $nb
 action 406  syslog msg "406: cb: $cb nb: $nb result: $_result"
 action 408  set m "$_result"
 action 410  syslog msg "m: $m"
 action 412  increment n
 action 414  subtract $n $m
 action 416  multiply $_result $olc
 action 418  if $ifcfg eq "0"
 action 420   set dlc "$metric"
 action 422  else
 action 424   set dlc "$_result"
 action 426  end
 action 428  syslog msg "428: n:$n m:$m olc:$olc dlc:$dlc result:$_result intf: $ifname"
 action 434  cli command "int $ifname"
 action 436  cli command "do show run int $ifname"
 action 438  string first "ip router isis" "$_cli_result"
 action 440  if $_string_result ne "-1"
 action 442   cli command "isis metric $dlc"
 action 444   cli command "do show ip ospf int | i $ifname"
 action 446   string first "$ifname" "$_cli_result"
 action 448  elseif $_string_result ne -1
 action 450   cli command "ip ospf cost $dlc"
 action 452  end
 action 454 end
 action 456 syslog msg "The EEM script executed"
 action 458 cli command "event manager applet ACM62"
 action 460 cli command "event tag event_sd ethernet microwave sd interface
GigabitEthernet0/0/1 threshold $nb"
 action 462 if $ppmap eq "0"
 action 464  if $_eem_mode le "1"
 action 466   cli command "action 116 set ppmap $pmap"
 action 468  end
 action 470 end
Router#
```

Example: Configuring Event Handler

The following is a sample configuration of Event Handler.

```
event manager applet mw_ring_sd1
```

```
 event ethernet microwave sd interface gigabitethernet 0/0/1 threshold 400
 action 1 switch ring g8032 ringA instance 1
interface gigabitethernet 0/0/1
 ethernet event microwave hold-off 30
 ethernet event microwave loss-threshold 100
 ethernet event microwave wtr 45
```