July 2023

CFM Fault Notification Enhancements

Anonymous

CFM Fault Notification Enhancements

ABSTRACT

This disclosure describes Connectivity Fault Management (CFM) which is the protocol used in Ethernet Networks for detecting, verifying, and isolating connectivity failures. The CFM protocol uses Continuity Check Messages (CCMs) to detect and report faults in a service. A CFM service may be created between 2 endpoints (called MEPs) or it can also be created between multiple endpoints. In case of a multipoint-to-multipoint service, the CCMs are exchanged between multiple MEPs (present on different devices). The fault notification reporting mechanism defined in the IEEE 802.1Q-2018 standard, defines faults based on the local endpoint. This is not an optimized method of managing CFM faults, because a local MEP (endpoint) can be communicating with multiple other MEPs (as in case of multipoint to multipoint CFM service). Thus, there is a possibility of reporting faults on services that are actually fault free. The present disclosure proposes enhancements to solve these problems present with the current Fault notification mechanism supported in the IEEE 802.1Q CFM standard.

DETAILED DESCRIPTION

A new mechanism is proposed to initialize and update the Fault Notification Generator state machine's variables on a per (local + remote) MEP basis. This will help to identify the faults per remote-end point in case of point to multi-point service. The new mechanism also proposes to handle misconfiguration faults where the CFM application would read the received packet and update the remote MEP database. Enhanced fault notification functionality is contemplated to add content which will inform about the faulty remote MEP, in case of misconfiguration faults. Proposed mechanisms provide methods to free up the remote MEP database by deleting the entries once the misconfiguration fault is cleared and no valid CCM packets are received for that remote end point. Updating fault notification messages to carry addition information to identify the 'type' of misconfiguration, to provide more information on the cause of the defect, leads to faster resolution of the fault.

Background of CFM:

- Used to Provide OAM functionality in Layer-2 Networks.
- Uses Multicast Destination MAC address based 'Heartbeat' messages (Continuity Check Messages) between Endpoint Entities called Maintenance Association End Points (MEPs).
- CCMs are used to detect faults in Ethernet services.

Problem statements including some deficiencies in fault notification handling (reporting or clearing of alarm notifications) for CFM services are described below.

Problem 1:
As per the CFM standard, fault notifications are tracked and reported on MEP's Maintenance Association. This behavior is fine untill the time a point-to-point service i.e.,

CFM service that has only two end points is present. This behavior fails to provide correct information in case of CFM Multi-Point service. It does not inform the network operator about the remote end point identifier, which has failed. In case of fault creation on multiple end points of a multipoint CFM service, fault notification is generated only for one of the end point failures at a time.

Problem 2:
Fault notifications can not be reported unless the local MEP has fully discovered at least one valid remote-MEP i.e., local MEP has been receiving valid CCMs from at least one remote MEP for a time period greater than the CCM time-out period in either a point-to-point service, or a multipoint service. Thus, the operator does not receive any fault notifications in case the service was mis-provisioned from the very first time of its creation.

Problem 3:
Some faults can have multiple causes of occurrence. Faults related to the configuration mismatch like the DefXconCCM and DefErrorCCM are the two such faults that do not provide the exact configuration mismatch parameter to the network operator due to which the fault notification was generated (true for both point to point as well as multipoint services). For example, fault notification for DefXconCCM can be generated either due to (i) MD level mismatch or (ii) MAID mismatch. Similarly, DefErrorCCM can be generated either due to (i) CCM interval mismatch or (ii) due to invalid MEP ID in the received CCM.

Detailed Explanation of the Problem 1:

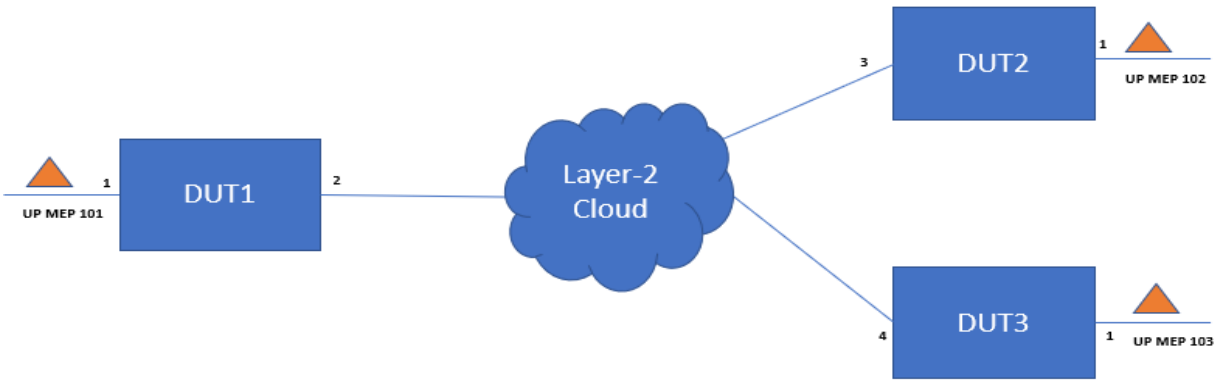

Figure 1

In figure 1, DUT1, DUT2 and DUT3 are connected in a multipoint topology with CFM services created on each of these devices. UP MEP is created on Port-1 of each DUT under same Maintenance-Association (MA). So local MEP on each DUT will discover two remote MEPs. If there is a connectivity loss on the link towards DUT2, then all DUTs i.e., DUT1, DUT2 and DUT3 will send a fault notification for DefRemoteCCM to indicate that there is a problem in MEP's MA as DUT1 & DUT3's MEP will detect a CCM timeout from DUT2's MEP and DUT2's MEP will detect CCM timeout from DUT1 & DUT3's MEP. Thus, if there is connectivity loss on the link towards DUT3 as well, then DUT1 and DUT3 will also detect a CCM timeout from each other's MEPs but fault notification will not be

generated in this case to notify the operator that another end point in a multipoint service has failed.

As per the current specification, even if the connectivity on the link towards DUT2 is restored, still the Network Operator will not be notified that the CFM service between DUT1 and DUT2 has been restored and another remote end point i.e. DUT3's MEP is in failed state. Since Fault Notification Variables is getting tracked only at MEP level and the value of Fault Notification Variables will remain same in this scenario, so no fault clear notification for remote MEP 102 (DUT2's MEP) and fault raise notification for remote MEP 103 (DUT3's MEP) will be generated. This leads to high Mean Time To Restore (MTTR) when multiple endpoints are in a faulty state in a multipoint service as the operator will never get to know about CCM timeout from DUT3. So, the service restoration for DUT3 end point will be delayed.

When there is a failure in the path to DUT-2, the loss of connectivity fault notification (remoteCCM) is reported in all the devices. However, there is no way of knowing just from the fault notification as to which path is impacted by the loss of connectivity as this fault notification would be reported on all the devices. This is where the need lies to precisely point out the faulted service in case of CFM Multipoint services.

Detailed Explanation of the Problem 2:

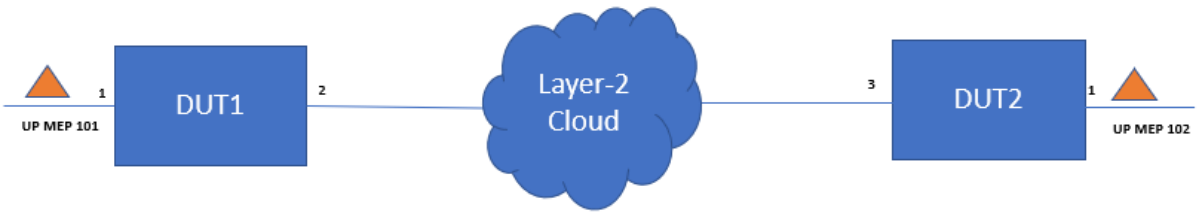

Figure 2

This problem applies to both point-to-point as well as multipoint services. In figure 2, DUT1 and DUT2 are connected in a point to point topology. Operator has configured a new UP MEP on DUT1 and DUT2, so none of them have discovered any remote end point yet. During UP MEP configuration on DUT1 and DUT2, operator did some misconfiguration mistake which may be related to either a) MD level mismatch, or b) MA ID mismatch or c)CCM interval mismatch or d)configured same MEP ID at both ends. In this scenario, fault notification for DefXconCCM (due to MD level mismatch, or MA ID mismatch) or DefErrorCCM (due to CCM interval mismatch or same MEP ID configuration at both end) will be generated but there is no information about the remote endpoint that is causing the problem as the device has not discovered any valid end point.

Detailed Explanation of the Problem 3:

Figure 3

This problem also applies to both point-to-point as well as multipoint services. In figure 3, DUT1 and DUT2 are connected in a point-to-point topology. Operator has configured a new UP MEP 101 on DUT1 and MEP 102 DUT2. So, both DUT1's MEP and DUT2's MEP has discovered each other as valid remote MEPs. Now operator modified some configuration on say DUT2, which caused misconfiguration related to either MD level mismatch, or MA ID mismatch or CCM interval mismatch or configured same MEP ID as DUT1's MEP ID. In this scenario, fault notification for DefXconCCM or DefErrorCCM will be generated depending upon the configuration mismatch but user won't be notified about the configuration parameter due to which fault is reported. For example, fault notification for DefXconCCM won't inform the operator if fault has been detected due to MD level mismatch or MA ID mismatch. Similarly fault notification for DefErrorCCM won't inform the operator if fault has been detected due to CCM interval mismatch or same MEP ID configuration at both end. This would require manual intervention to figure out the cause of the fault. The mismatch information, if provided along with the fault notification, can help in faster resolution of the problem (mismatch).

Current solutions:

Fault Notification Variables have a single instance defined for each Local MEP, irrespective of the number of remote MEPs present on a device. This creates problems during fault tracking for a multipoint service as a single instance of the variables can't be used to track faults and hence to generate fault notifications for multiple end points of the same service. The current implementation also does not specify any way to report fault notifications where remote end points have never been discovered. The current implementations also does not provide any information to identify the 'type' of misconfiguration present in the CFM configuration.

Proposed solutions:

Solution 1:
The Fault Notification variables would be instantiated on a per 'MEP-REMOTE MEP' basis for all types of CFM services. Since the variables would be present for each pair of MEP and Remote MEP entry, it will be possible to track and report CFM faults for this pair separately.

Solution 2:

Every incoming CCM packet will be examined to determine the MEP ID present in it. So, in case of mis-configuration scenarios where remote MEP has never been discovered, the read value from the CCM packet would be used to create a temporary entry in the REMOTE MEP database. Then, the Fault notification state machine's variables would be initialized based on the local MEP ID and the learnt (read from the packet) Remote MEP ID. The fault notification for the corresponding misconfiguration i.e., DefXconCCM or DefErrorCCM will be generated. Once the DefXconCCM or DefErrorCCM fault for this LMEP and RMEP pair is cleared and no CCMs are received for this Local MEP and Remote MEP pair, then the fault notification variables for this pair will be deleted to free up the utilized memory and resources.

Solution 3:
Fault notifications will be updated to carry additional text or attributes, to notify the 'type' of mis-configuration that has caused the fault. For example, in case of DefXconCCM fault, the fault notification message will provide additional information as to whether the fault has been detected due to MD level mismatch OR if it is caused due to MAID mismatch and also the corresponding parameter value configured on the remote MEP. Similarly in case of DefErrorCCM fault, the fault notification message will provide additional information about the fault has been detected due to CCM interval mismatch OR due to same MEP ID configuration at both ends and also the corresponding parameter value configured on the remote MEP.

Validation of the solution for Problem 1:

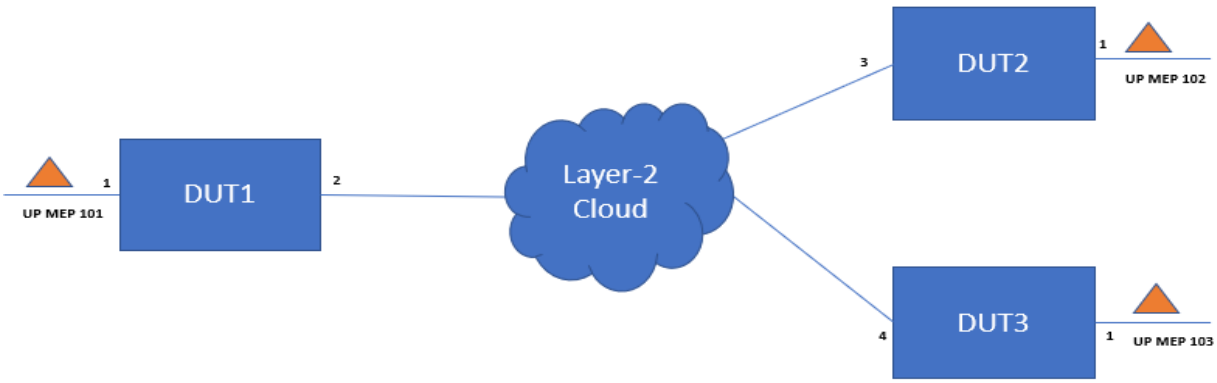

Figure 4

Here DUT1 has discovered two remote MEPs, MEP ID 102 from DUT2 and MEP ID 103 from DUT3. Each DUT will have two instances of the FNG variables running for each of the LMEP and RMEP pairs. For instance, on DUT1, it would have:

FNG variables for_LMEP-101_RMEP-102 -> To track the service faults between two CFM end points between DUT1 and DUT2

FNG variables for_LMEP-101_RMEP-103 -> To track the service faults between two CFM end points between DUT1 and DUT3 of the same CFM service.

Connectivity loss happens on the link towards DUT2. DUT1 will detect the CCM timeout from DUT2's MEP, update the FNG variables for_LMEP-101_RMEP-102 and send fault notification for CCM timeout from RMEP 102. Now connectivity loss happens on the link towards DUT3. DUT1 will detect the CCM timeout from DUT3's MEP, update the FNG variables for_LMEP-101_RMEP-103 and send fault notification for CCM timeout from RMEP 103. The operator can parallelly start working on restoration of both service end points at the same time as the operator has been notified about failure of both RMEPs. Once the service on DUT2 is restored, DUT1 will start receiving the CCMs from DUT2's MEP i.e. MEP ID 102. So DUT1 will update the FNG variables for_LMEP-101_RMEP-102 and send clear fault notification for CCM timeout from RMEP 102. Operator will still have outstanding fault notification for DUT3's MEP to work. Once the service on DUT3 is restored, DUT1 will start receiving the CCMs from DUT3's MEP i.e. MEP ID 103. DUT1 will update the FNG variables for_LMEP-101_RMEP-103 and send clear notification for CCM timeout from RMEP 103.

Use case example of Problem 1:

As per figure 4, A customer had a multipoint CFM service configured between the 3 devices. The network side had a work and a protection Pseudowire. They wanted to use CFM services for Performance monitoring. But, it was observed that after a PW switchover from work to protect, there was a constant CFM fault present on the CFM services on all the 3 devices. After much deliberation by the design, it was communicated to the customer, that this use-case is not possible as per the CFM standard's existing design. This is because fault between one endpoint was being propagated on all other endpoints.

With the present solution to initialize fault notification variables on a per Local + Remote endpoint basis, this customer use case becomes achievable. The Dut1-Dut2 CFM service instance can be used for PM on the 'work' pseudowire present between DUT1 and DUT-2. The Dut1-Dut3 CFM service instance can be used for PM on the 'Protect' pseudowire between DUT1 and DUT3.

Validation of the solution for Problem 2:

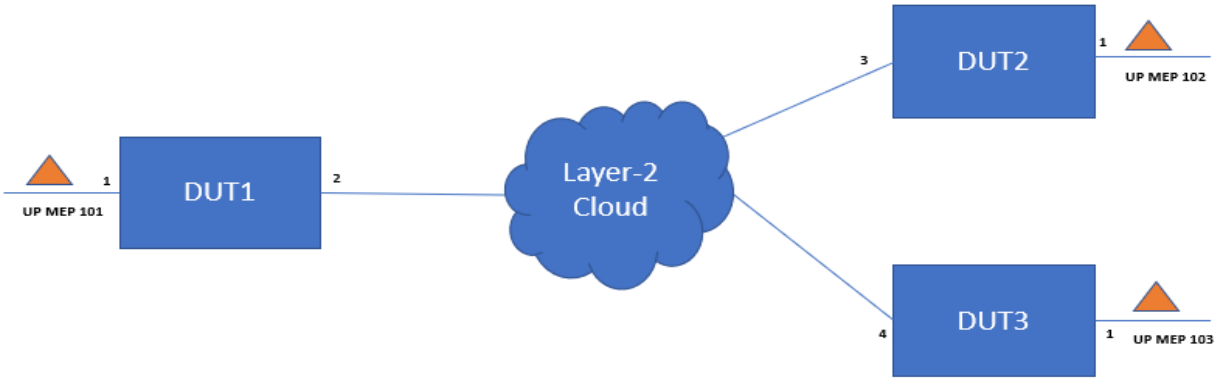

Figure 5

Here, DUT1 has not discovered any remote MEPs yet. Now DUT2 is provisioned with MEP ID 102 with same maintenance association ID but lower MD level from DUT1's MEP (i.e. there is a MD level mismatch between the MEPs). DUT1 will get the CCM from DUT2 with lower MD level. DUT1 will read the remote-MEP ID i.e. MEP ID 102 in DUT2's CCM and check that FNG variables for instance for LMEP (DUT1's MEP) and RMEP (DUT2's MEP) does not exist. Thus, DUT1 will instantiate a new instance of variables i.e. FNG variables for_LMEP-101_RMEP-102, and update them accordingly and then send fault notification for cross-connect defect i.e. DefXconCCM from RMEP 102.

Now DUT3 is provisioned with MEP ID 103 with the same maintenance association ID but lower MD level from DUT1 and DUT2's MEP. DUT1 will get the CCM from DUT3 with lower MD level. DUT1 will read the remote-MEP ID i.e. MEP ID 103 in DUT3's CCM and check that FNG variables for the instance for LMEP (DUT1's MEP) and RMEP (DUT3's MEP) does not exist. So DUT1 will instantiate a new instance of variables i.e. FNG variables for_LMEP-101_RMEP-103, and update them to send fault notification for cross-connect defect i.e. DefXconCCM. Thus, the operator will be notified about DefXconCCM from both the remote maintenance association end points at the same time even though these remote MEPs had never been fully discovered and the operator can parallelly start working on service restoration of both the remote end points at the same time.

Validation of the solution for Problem 3:

Figure 6

Consider that DUT2 is provisioned with MEP ID 102 with the same maintenance association ID as DUT1's MEP but lower MD level from DUT1's MEP. DUT1 will update FNG variables for_LMEP-101_RMEP-102 and send fault notification DefXconCCM from RMEP 102 with the additional information: about MD level mismatch and MD level received in the CCM packet from DUT2's MEP. Thus, by looking at the fault notification message itself the operator will get to know about the misconfigured parameter.

Conclusion:

The solutions provide important enhancements to the CFM fault notification process to help in achieving an important use case in the customer's network. No scalability concerns mean:

- Solutions have analyzed that the Fault Notification variables' initialization on (local + remote) MEP basis will consume only a few KB of memory in the database for a scale size of approximately 16000 Remote MEPs.
- The Fault Notification variables' initialization will happen at run time (whenever a CCM from a new RMEP is received) and hence, all the memory resources are not utilized at the initial state.
- Significant functionality enhancement achieved in CFM at the cost of minor increase in the memory resources needed for variable initialization.

Terms:

CFM: Connectivity Fault Management
IEEE: Institute of Electrical and Electronics Engineers
OAM: Operations, Administration and Maintenance
MEP: Maintenance Association End Point
RMEP: Remote MEP
DB: Database
CCM: Continuity Check Message

It will be appreciated that some embodiments described herein may include one or more generic or specialized processors ("one or more processors") such as microprocessors, digital signal processors, customized processors, and Field-Programmable Gate Arrays (FPGAs) and unique stored program instructions (including both software and firmware) that control the one or more processors to implement, in conjunction with certain non-processor circuits, some, most, or all of the functions of the methods and/or systems described herein. Alternatively, some or all functions may be implemented by a state machine that has no stored program instructions, or in one or more Application-Specific Integrated Circuits (ASICs), in which each function or some combinations of certain of the functions are implemented as custom logic. Of course, a combination of the aforementioned approaches may be used. Moreover, some embodiments may be implemented as a non-transitory computer-readable storage medium having computer-readable code stored thereon for programming a computer, server, appliance, device, etc. each of which may include a processor to perform methods as described and claimed herein. Examples of such computer-readable storage mediums include, but are not

limited to, a hard disk, an optical storage device, a magnetic storage device, a ROM (Read Only Memory), a PROM (Programmable Read-Only Memory), an EPROM (Erasable Programmable Read-Only Memory), an EEPROM (Electrically Erasable Programmable Read-Only Memory), Flash memory, and the like. When stored in the non-transitory computer-readable medium, the software can include instructions executable by a processor that, in response to such execution, cause a processor or any other circuitry to perform a set of operations, steps, methods, processes, algorithms, etc.

Although the present disclosure has been illustrated and described herein with reference to preferred embodiments and specific examples thereof, it will be readily apparent to those of ordinary skill in the art that other embodiments and examples may perform similar functions and/or achieve like results. All such equivalent embodiments and examples are within the spirit and scope of the present disclosure.