


[Home](#) [Archives](#) [Profile](#) [Subscribe](#)
[Main](#)

12/09/2010

FIP, FIP Snooping Bridges, and FCFs (Part 2 - FIP Snooping Bridges and Fibre Channel Forwarders)

In my last post I described what FIP is, how FIP bridges the gap between the expectations of the Fibre Channel protocol and the reality of an Ethernet network, and described how FIP works. In this post, I'll expand on this subject and describe why FCoE requires ENodes to be either directly connected to a Fibre Channel Forwarder (FCF) or connected to a FIP Snooping Bridge (FSB) and then to an FCF in order to function properly. To avoid any misinterpretation, let me explicitly state that **FCoE cannot be guaranteed to function properly when a non-FIP-aware Ethernet Bridge is used anywhere in the data path.**

Note: The terms “switch” and “bridge” are used interchangeably, not only in this post, but in the Ethernet space in general. My friends who bear the scars from working with SCSI to FC and FC to iSCSI bridges will almost certainly find this as perplexing as I did.

Let's begin with a definition:

FIP Snooping Bridge - An Ethernet Bridge that supports Priority Flow Control (PFC - 802.1Qbb), Enhanced Transmission Selection (ETS - 802.1Qaz), and Data Center Bridging Capabilities Exchange Protocol (DCBX - 802.1Qaz). In addition, it supports the functionality described in FC-BB-5 Annex C, which I refer to as *Dynamic ACLs*.

The issues I am going to focus on are basically new manifestations of two well-known networking exploits: “denial-of-service” and “learning” attacks. While these problems are well known and understood in the Ethernet community, their very presence is anathema to the bedrock principles of Fibre Channel and had to be addressed before FCoE could be taken seriously. In order to solve these problems, a new protocol (FIP) and a new Layer 2 feature (Dynamic ACLs) were created. The purpose of this post is three-fold: to describe the impact that each of these exploits has on the FC protocol; to show how FIP snooping and Dynamic ACLs resolve these problems; and to prove why a non-FIP-aware Ethernet bridge should *not* be used when using FCoE. The first case presented involves a user inadvertently causing a denial-of-service attack by joining two


[Erik Smith](#)
[Follow](#)
[ARCHIVES](#)

SEARCH

Search

[CATEGORIES](#)
[Subscribe to this blog's feed](#)

DISCLAIMER

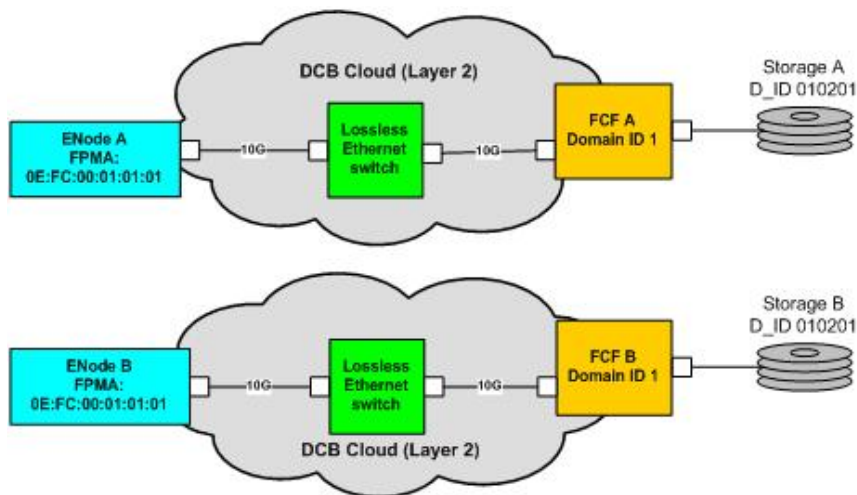
This is not an official EMC blog.

The opinions expressed here are my personal opinions. Content published here is not read or approved in advance by EMC and does not necessarily reflect the views and opinions of EMC nor does it constitute any official communication of EMC.

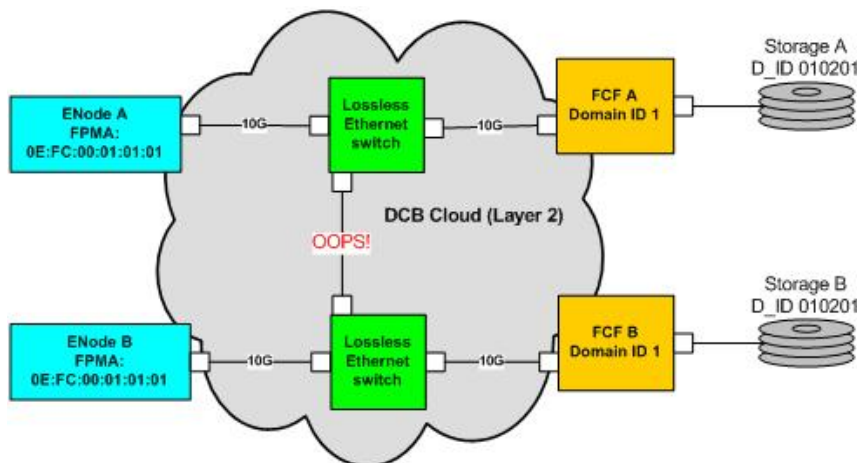
separate Layer 2 networks. The second case involves a malicious user attaching a “Rogue Host” to the network and performing a learning attack.

The problem with network joins

Consider the following topology. It consists of two physically separate, yet identical, network topologies.



Note that the FCFs have the same Domain ID of 1 and because of this it is POSSIBLE that each ENode will be assigned the same Fabric Provided MAC Address (FPMA). Ordinarily, this would not be a problem since the ENodes are located on two different L2 networks. However, what happens if someone accidentally connects the two lossless Ethernet switches together, as shown in the following diagram?



Were this to occur, the two ENodes would have the same MAC Address, which could cause a denial-of-service condition. In fact, whether or not a denial-of-service could occur in the above configuration is not the question, but rather, how long it would take before it happened! The reason has to do with how MAC Addresses are learned and is explained in a bit more detail in the following section.

Ethernet Frame delivery

If you want a good description of how frames are processed by a Layer 2 Ethernet switch, check out figure 7 (and the text that follows) in the [Nexus 5000 Architecture white paper](https://web.archive.org/web/20250912220019/https://brasstacksblog.typepad.com/brass-tacks/2010/12/fip-fip-snooping-bridges-and-fcfs-part-2-fip-sno...). For the purpose of looking at the denial-of-

service condition, we only need to consider the destination lookup and Ethernet learning portions of this diagram so as a result, I will only talk about the aspects of frame-forwarding that are relevant.

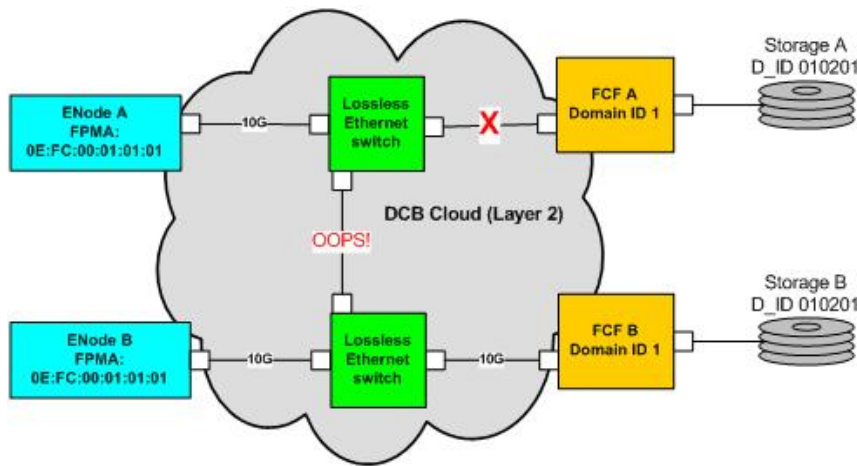
When an Ethernet Frame is received by an Ethernet switch, one of the many things it will do is to take note of the Ethernet Frame's Source Address (SA) and Destination Address (DA). If the SA is unknown to the switch, it will take note of the interface the frame was received on and "learn" the MAC Address by adding it to the station table. By doing so, any frames that are received with that MAC Address in the future will be forwarded back down the interface that it was received on. If the DA is unknown to the switch, it will perform a unicast flood and forward the frame on all interfaces that are in the same network segment (VLAN). This process of learning and unicast flooding will be repeated throughout the broadcast domain until either the frame is received by a switch that recognizes the DA of the flooded frame or until there are no interfaces left to forward the frame on. In this way, the path back to the frame originator will be built as the frame is flooded across the broadcast domain. If an end station with that DA exists and it transmits a response back to the originator of the flooded frame, the learning process will be repeated as the frame transits the network. However, since the path back to the originator of the unicast frame was built as the original frame was flooded, the response will not be flooded. Eventually, the response will be received by the end station that transmitted the original frame. At this point, the path between the end stations has been created and no more unicast flooding will need to be performed as long as the MAC Addresses are in the station table.

Let's refer back to the second topology diagram where the cable was accidentally connected between the two lossless Ethernet switches. In this case, frames would continue to be delivered to the proper end stations until at least one of the end station MAC Addresses are flushed from the station table. This could happen for a number of reasons, including:

- a) A MAC Address is removed from the station table due to a loss of physical connectivity to the end station;
- b) A MAC Addresses is aged out of the station table;
- c) An administrator manually clears the station table; or
- d) A Topology Change Notification (TCN) is received by the switch.

The final case, a TCN being received by the switch, would occur after the Spanning-Tree Protocol ran and one of the switches recognized a change in the network configuration. When the TCN is received, the forwarding entries in the station table would be rapidly aged out and the process of unicast flooding would need to be performed in order to rebuild the forwarding entries.

Relating this to the case that I am interested in proving, let's look at case "a" and assume that for some reason FCF "A" was physically disconnected from the lossless Ethernet switch.

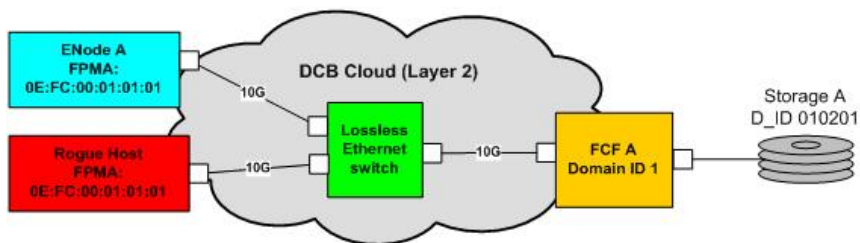


When this happens, the FPMA for FCF A will be cleared from the station table and subsequent frames from ENode “A” to array “A” will be flooded. As a part of the flooding process, this frame would be forwarded across the link between the two switches. When the frame is received by the ingress port on the switch at the other end of the link, the ingress port will not recognize the DA since it belongs to FCF “A” and the frame will need to be flooded. It will, however, take note of the SA and update its station table to indicate that frames destined back to this FPMA should be forwarded across the link back to the other switch. This means any frames from Array “B” going back to ENode “B” will be incorrectly forwarded back across the link and on to ENode “A”. Of course, as soon as the next frame is received from ENode “B”, the station table would be correctly updated and frames would be forwarded to ENode “B” again. The problem is, if frames were received from ENode “A” frequently enough, it could cause many frames to be incorrectly forwarded to ENode “A” so ENode “B” would spend all of its time in error recovery, get no meaningful work done and therefore a denial of service condition would be created.

A FIP Snooping Bridge prevents this problem by only allowing frames either from or to an FCF to be forwarded out of a **FIP Snooping uplink**. As indicated in [FC-BB-5 annex C](#) and [annex D](#), allowing the forwarding or reception of a frame to or from the FCF is something that should be explicitly enabled on a per-interface basis. As a result, an FIP snooping bridge would have prevented the scenario described above by not allowing these frames to be flooded out non-FIP Snooping uplinks. In addition, even if one of the FIP Snooping uplinks were accidentally connected to an interface on another FIP Snooping Bridge, the receiving interface would not forward frames with an FCoE Ethertype. For more information, refer to [FC-BB-5 annex C](#) and [annex D](#).

The Rogue Host

As shown in the following topology diagram, this problem is actually very simple to understand and is fairly easy to write code for and perform in the lab. The topology consists of a host labeled ENode “A”, a non-FIP-aware Lossless Ethernet switch, an FCF labeled FCF “A”, a storage port labeled Storage “A” and a “Rogue Host”. Assume that ENode “A” is logged into Storage “A”.



If a person were interested in using a Rogue Host to gain access to specific data on Storage “A”, all they would need to know is the FPMA of an ENode that has access to that data. The FPMA of any ENode can easily be found by using “show fcoe database” on a Nexus 5k (for example). Once in possession of the FPMA, the person in control of the Rogue Host could transmit a single Ethernet Frame (e.g., FIP Solicitation), setting the Source Address (SA) to the value of the FPMA that they are interested in. This would cause a station table update as described above and then allow them to capture whatever frames come back. These frames would contain the FCF-MAC Address and any storage port FCIDs (labeled D_ID in the diagram) that happen to be transmitting frames back to ENode “A” at the time. Once the FCF-MAC and the D_ID of the storage port is known, the Rogue Host could transmit a SCSI-FCP READ command to the storage port and read data or a SCSI-FCP WRITE command to alter (corrupt) the data.

To prevent this problem from happening requires the use of an ACL as defined in FC-BB-5 annex C and D. Rather than repeat the relevant text here, I’m just going to provide an overview of the functionality and encourage those of you who are interested in more information to [download a copy of FC-BB-5](#).

Dynamic ACLs

By default, an interface on a FIP Snooping bridge will not forward FIP or FCoE frames and will not learn any MAC Addresses contained within these frames. Because of this, the Rogue Host problem I described above is prevented from happening by default. However, this default ACL does not prevent a whole bunch of other problems, such as someone making use of a port where the ACL was removed to allow an FCoE host to be attached, or someone hijacking an existing host that is already logged into the fabric. To prevent these types of problems, the ACL would need to specify the exact MAC Addresses that can be used on the interface and this ACL would need to be modified whenever a port was added or removed.

The downside to this solution is that it would create an enormous administrative burden on the network administrator since they would need to manually modify each ACL entry to allow FIP and FCoE frames. Because of this administrative burden and the critical nature of the problem should the ACLs not be used, a solution that would allow for the ACLs to be updated automatically at certain points in the protocol was created. I call the solution that was eventually agreed upon “Dynamic ACLs” and it works as follows:

1. By default, FIP and FCoE frames will not be forwarded from the ENode onto the network.
2. Once a Discovery Advertisement is transmitted to the ENode, the switch will take note of the Discovery Advertisement SA and modify

the ACL to allow the attached ENode to transmit a FIP frames to that address.

3. For each successful FIP FLOGI, the ACL will be updated to allow FCoE frames between the ENode's MAC Address and the FCF that it completed login with.

Implementing this behavior as described would prevent the previously described learning attack from being successful.

An important point to note is that the Dynamic ACLs are not actually visible in any of the implementations I am familiar with. For the most part, this has been implemented by forwarding all FIP Frames to the Control Plane of the switch for processing or by discarding FCoE frames until a login has been completed.

Conclusion

Hopefully, I've managed to convince you that using a non-FIP-aware Ethernet Bridge in an FCoE environment would be problematic, or at least less than optimal! The only additional piece of advice I can give you is this: In a production environment don't even consider using a non-FIP aware bridge with FCoE.

Thanks for reading!

Posted at 11:34 AM | [Permalink](#)

 Reblog (0)

Comments

 You can follow this conversation by subscribing to the [comment feed](#) for this post.



Hi there, You've done a great job. I will definitely digg it and personally recommend to my friends. I'm confident they'll be benefited from this web site.

Posted by: [bmx](#) | [10/04/2012 at 08:25 PM](#)

The comments to this entry are closed.

[Brass Tacks](#) [Powered by TypePad](#)