



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

[« Slow Drain due to Bandwidth mismatch and Large Block READs](#) | [Main](#) | [Storage Protocol Stacks for NVMe »](#)

07/24/2017

NVMe over Fabrics: Not ready (yet)



Image credit:

<https://www.someecards.com/usercards/viewcard/MjAxMy03ZWxZjlxZmRkMjRmZGNI/>

If you're into IT infrastructure and you aren't yet familiar with NVMe and its potential benefits, now would probably be a good time to start reading. In fact, since there's already so much [high quality material](#) that describes what NVMe is, why it's useful and how it works, I'm not going to go into those details here.

This post is also not going to focus directly on NVMe over Fabrics. The reason is, based on our limited testing, from a system level point of view I think it's too early to say much about it. I will say many of the features that our customers would expect from an enterprise class storage protocol simply haven't been created for NVMe over Fabrics yet. As I'll describe below, we've found that the missing functionality varies based on the transport being used. For example, in the case of RoCE and iWARP they need to do a bit of work on



[Erik Smith](#)

[Follow](#)

[ARCHIVES](#)

[September 2022](#)

[August 2020](#)

[April 2019](#)

[December 2017](#)

[November 2017](#)

[July 2017](#)

[April 2017](#)

[September 2016](#)

[July 2016](#)

[February 2016](#)

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.

“Discovery”. With regards to Fibre Channel, it comes down to the lack of a GA quality NVMe over Fabrics driver for any of the Tier 1 Operating Systems.

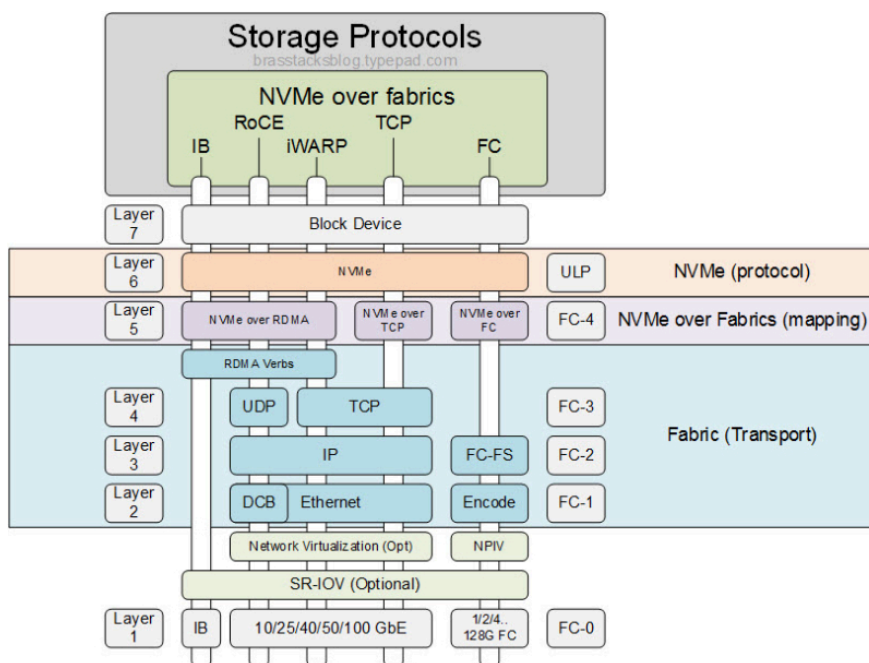
In an attempt to be fair to the people currently working to address these limitations, I'll say I fully expect the majority of these issues to be addressed by sometime in 2018.

This post will focus on fabrics.

More specifically, the fabrics that are capable of transporting NVMe. I'll also spend some time describing:

1. Why these fabrics aren't ready for the Enterprise; and
2. why we're watching them soooo closely.

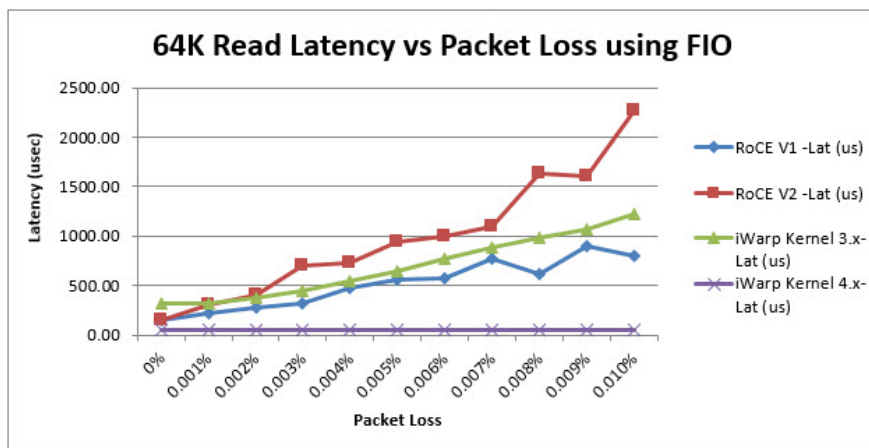
As shown in the following diagram, these fabrics can be based on RDMA, Fibre Channel or TCP (TCP is pre-standard at this point, although apparently there's a [demo at the Flash Summit](#)) and use IB, RoCE, iWARP, TCP or Fibre Channel (FC) as transports.



For the sake of this blog post, I'm going to focus on the three transports that my team and I feel are the most likely to be used in by our customers and have consequently been working on over the past several years; RoCE, iWARP and FC.

iWARP

When we first started working with RDMA, I thought iWARP was going to win the protocol war that was going on between it and RoCE. My reasoning was very simple, iWARP doesn't require anywhere near the amount of network configuration that RoCE does (much more on this in the next section) and it isn't susceptible to congestion spreading (A.K.A., Slow Drain) like all lossless protocols (e.g., RoCE, FC) are. In addition, some of the early work we did to compare the performance attributes of RoCE to iWARP showed that they were comparable. For example, the chart below was generated from data collected by two Engineers on my team (Alan Rajapa and Massarrah Tannous) during some testing that they performed to compare different implementations of the two transports.



When reviewing this chart, please keep in mind the following:

1. The chart was created about 18 months ago and represents a point in time when iWARP latency (with a 4.x Linux Kernel) was slightly less (under ideal conditions) than RoCE, and MUCH less than RoCE when any kind of packet loss was introduced.
2. Some in the RoCE community will complain that this isn't a fair comparison since RoCE requires a lossless network. My response to this concern has always been "We're interested in how the protocols react under packet loss conditions because configuring lossless behavior across a L3 Clos (leaf/spine) network is fairly complex and is not something that we (at the time), nor our Customers, have much experience with. As a result, we believe lossless (DCB) will be frequently misconfigured and many of our customers may inadvertently end up running RoCE on a lossy network." With this in mind, the purpose of our test was to find out just how screwed our customers would be in this kind of a mis-configuration. If RoCE flat out wouldn't work, these types of situations would be fairly easy to identify and then resolve. However, if this kind of mis-configuration resulted in a situation where things mostly worked and just sort of limped along, our support organizations would have a MUCH harder time troubleshooting and this was causing us great anxiety.
3. Over the past 18 months the RoCE community (Mellanox in particular) has done a [ton of great work](#) to address many of the issues we've raised with regards to packet loss. More on this in the next section.

With regards to iWARP going forward, the primary proponent of the protocol appears appears to have found [some use cases](#) for it. That said, they are really the only vendor pushing it and based on what I've heard from others in the industry as well as what our Customers are asking for, my personal opinion is that iWARP will not be the dominant Ethernet based transport for NVMe. As a result, I won't spend any additional time on it here.

RoCE

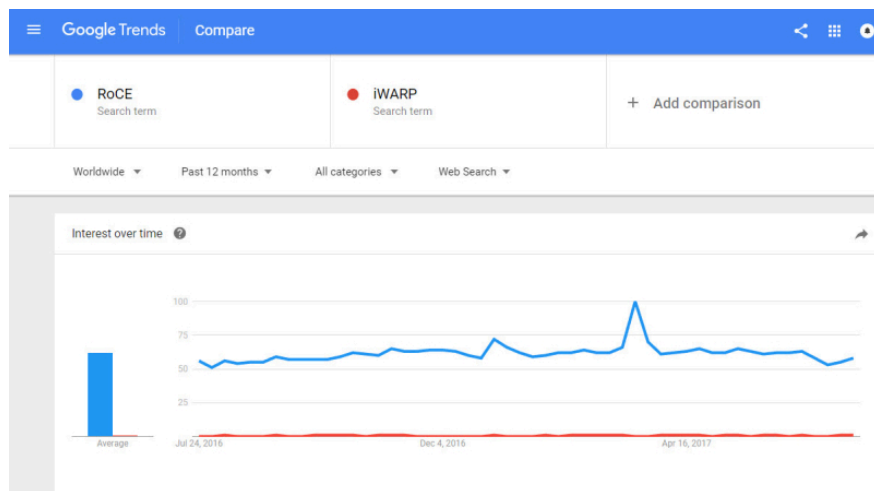
RoCE (RDMA over Converged Ethernet) is an interesting transport for a number of reasons:

1. Due to the need to configure DCB across an L3 network (when running at scale), it is much more complex to configure than FCoE (more on this below).
2. Since it requires DCB (which includes Priority Flow Control) it's susceptible to [Congestion Spreading \(a.k.a., Slow Drain\)](#). A partial

solution to this problem is to use ECN/RED (as described in the Azure/Sigcomm white paper that I linked to below). However, this does not address all potential causes of congestion spreading (especially the acute cases).

3. The NICs and the switches that are typically used to transport RoCE lack even the most basic counters to help troubleshoot congestion spreading events (i.e., something analogous to FC's "[time spent at zero transmit credit](#)" counter is needed).

Yet despite these limitations, there is more interest in RoCE than iWARP (see Google trends below) and I'm also hearing that it's much more popular in production environments as well. For example, RoCE is being used in production by Azure as [described in this white paper](#). BTW, I LOVE this white paper because it not only confirms that it's possible to use RoCE at scale, it also confirms the level of complexity required to properly configure it to avoid issues with congestion spreading.

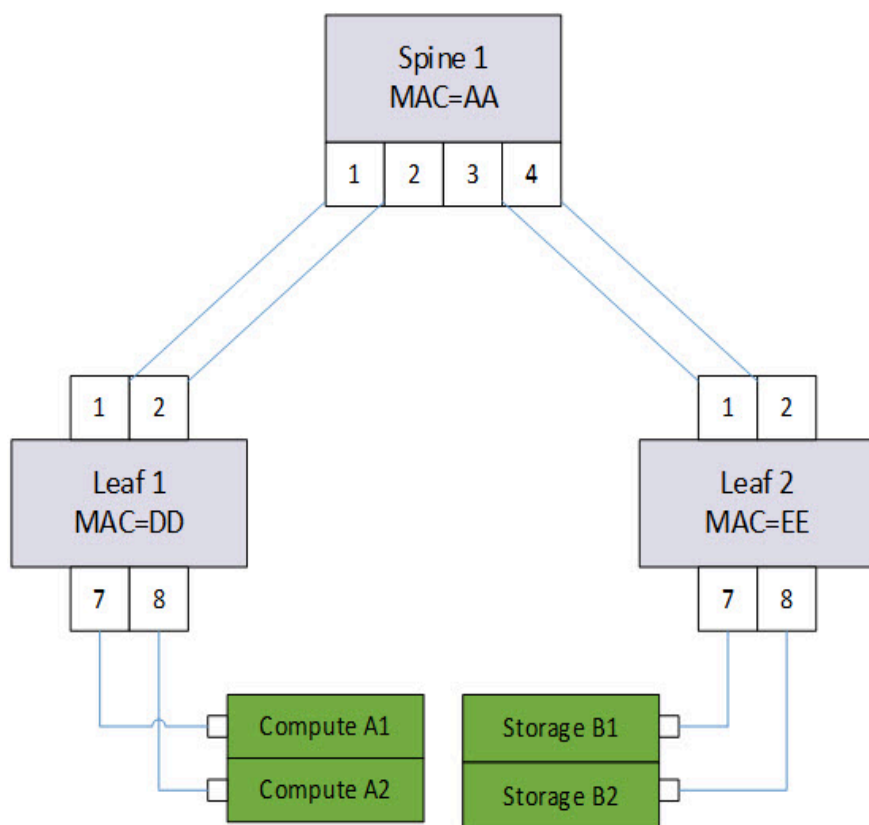


A few more points before I go on:

1. For the sake of this blog post, I'm going to focus only on the configuration complexity portion of RoCE.
2. This complexity is not specific to any one vendor but the following steps are specific to the switch vendor we were testing with at the time.
3. ECN/RED is being used for congestion avoidance.
4. In the example below we configure 3 Classes of service (0, 3 and 6) and only really 3 and 6 are required for RoCE and ECN.
5. A number of vendors have done a ton of work to automate these configuration steps and seem to be heading down the right track towards providing a complete end-to-end configuration tool. A notable example is [Neo](#) which Mellanox provided to us a no cost for evaluation purposes.

The actual topology that we used for testing RoCE is shown below.

It consists of a spine, two leaf switches, two "Compute" nodes and two "Storage" nodes.



A summary of the configuration steps that we performed to configure this topology is shown below.

Spine 1: MAC = AA

1. Configure buffer pools:
2. Configure Trust level per port:
3. Map DSCP levels to switch-priority
4. Map switch-priority to priority-groups
5. Map ingress priority group traffic to pools.
6. Map egress traffic class to pools
7. Map switch-priority to traffic class
8. Configure the scheduler
9. Set ECN/RED on the switch.
10. Enable PFC on the switch
 - a. Make sure Flow Control is disabled
 - b. Make sure Priority Flow Control is enabled

Leaf: MAC = DD and EE

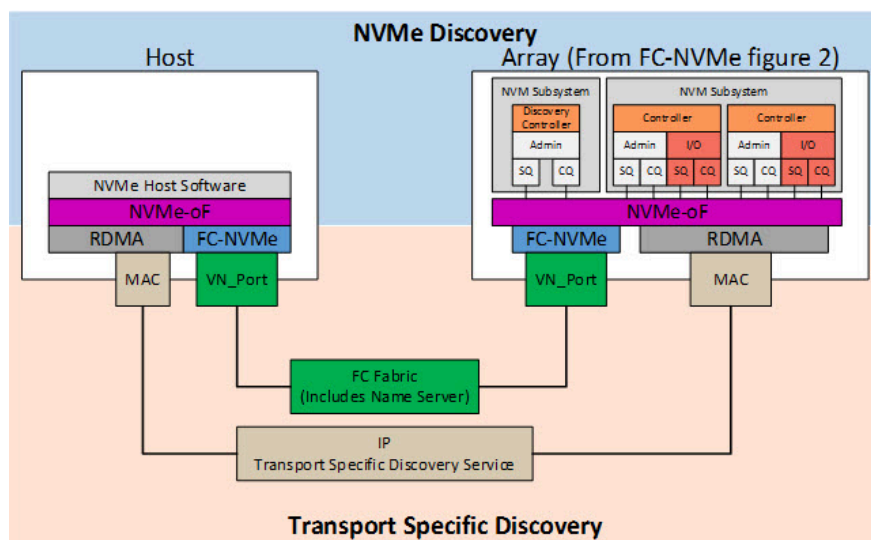
1. Configure buffer pools:
2. Configure Trust level per port:
3. Map DSCP levels to switch-priority
4. Map switch-priority to priority-groups
5. Map ingress priority group traffic to pools.
6. Map egress traffic class to pools

7. Map switch-priority to traffic class
8. Configure the scheduler
9. Set ECN/RED on the switch.
10. Enable PFC on the switch
 - a. Create a VLAN, set the end-station facing ports as switchports in trunk mode.
 - b. Make sure Flow Control is disabled
 - c. Make sure Priority Flow Control is enabled

Fibre Channel

Disclaimer: I've spent the past 20 years working as a member of Dell EMC's Connectrix Business Unit and my livelihood is pretty much tied directly to FC's success. That said, I suspect you may be surprised to learn that until recently I've struggled with the idea of using FC as a transport for NVMe.

When I first started reading about NVMe over Fabrics, one of the things that caught my eye was the need for a Discovery Service. After a little bit of digging, I realized that most of what NVMe over Fabrics needed for discovery could be provided by the FC Name Server and that this fact actually gave the FC transport a slight competitive advantage over the RDMA based transports. BTW, this is really just another way of saying FC uses a [network centric discovery](#) approach and RoCE/iWARP/TCP use an [end node centric](#) approach.



The diagram above illustrates the gap in functionality. Just as with SCSI and FCP, there are actually two different kinds of discovery that need to be performed, each operating at different layers in the stack.

The first kind of discovery is Transport specific. This discovery process allows two or more ports to discover each other over a network. With FC, the endpoints (i.e., VN_Ports) discover each other automatically via the use of the FC Name Server. With IP, there is no centralized in-band way to do this short of relying on iSNS (which they shouldn't). To get around this limitation, end users will basically need to do what they do today with iSCSI; individually configure each host to point at the correct targets.

The second kind of discovery is done at the NVMe layer and is fairly well-defined by the [NVMe standard](#). This isn't to say the NVMe discovery mechanism is perfect, but it is common across FC, RDMA and TCP transports, and as a result it doesn't make sense to use it to compare the benefits of different transports.

In terms of configuration complexity, a FC fabric can basically self-form and it supports lossless behavior out of the box. The only additional configuration that is typically done is to zone the host and storage ports together, but this doesn't necessarily need to be done manually (see [Introducing Target Driven Zoning](#)).

So you may be wondering, outside of easier discovery and less configuration why does using FC as a transport for NVMe make sense?

To be completely honest, I didn't think FC was a good fit! (initially) This is because in the early days all of the practical use cases for it seemed to be focused on platform 3 rather than more traditional enterprise applications. This causes a problem for FC because [platform 3 is generally hostile to FC and storage arrays in general](#), consequently I didn't see a real use case for FC, until recently.

This year at Dell EMC World, we included an NVMe section in our [SAN presentation](#). The key points about NVMe from that presentation can be found on slides 135-137. To summarize, NVMe SSDs are first being used in compute and they already provide efficiencies over what can be done with SATA based SSDs today. These same efficiencies can be realized on the back end of a storage array and [Jeff Boudreau has already indicated](#) he has plans to introduce a standards based NVMe solution soon. The next logical evolutionary step (shown on slide 137) would be to allow hosts to access the NVMe drives in an array via NVMe over Fabrics.

The reason this fabric should be Fibre Channel for our Enterprise customers is because the vast majority of them (85-90%) currently use FC to access their All Flash Arrays and they do so [for good reasons](#). So it only makes sense for them to continue to use the same Transport to access the next generation of ultra-low latency NVMe based SSDs.

However, as I mentioned previously, in order for customers to use NVMe over FC, we need GA quality NVMe over FC drivers. And then we have to put these drivers through an extensive qualification process (because data integrity). I'm thinking that the soonest this might actually happen would be mid 2018.

****BTW**, in case you were wondering, ALL FC switches that you can purchase today support NVMe over Fabrics.

Wrapping up

So what transport should you use? As always, I'll leave that decision up to you. But I will say the problem that keeps me up at night is how to address Congestion Spreading; and whether you choose RoCE or FC, you'll eventually need to deal with this problem. Since the FC community has been dealing with Congestion Spreading for years, have hardware support to help detect it and as shown below are continuing to innovate in this area, I would personally choose FC when it's ready.

Congestion Spreading Avoidance

	Prevention	Detection	Remediation
Short Term (today)	- BW Ratio - Modernize SAN - Fan-In/Out Ratio - BW Limits - Isolation	- Ratio: <code>bb_credit Zero(or TxWait) /tx frames</code> - Brocade FPI/MAFS - Cisco Port Monitor	- Congestion Spreading KB - Brocade SDDQ - Cisco Slow Drain - Virtual Link Isolation - Cisco no-credit-drop - Port fencing
Medium term	- BW Limit Enhancements - ViPR-C	ViPR-SRM	ViPR-SRM & ViPR-C
Long term	Future Fibre Channel Enhancements (TBD)		

BTW, “Future Fibre Channel Enhancements (TBD)” is being actively discussed and I hope to be able to share more details about it here at some point in the future.

Thanks for reading!

Posted at 10:47 AM | [Permalink](#)

 Reblog (0)

Comments

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



Fantastic blog entry as always, Erik. Can you provide any update to the status of TDZ? Is Dell/EMC, Cisco, and Brocade going to (or have they already) embrace it for their FC platforms? When you first blogged about it I got super excited but haven't seen much about it since. I've been leveraging MDS's Smart Zoning feature which certainly helps the config process but it's no TDZ. :-)

Posted by: Brett | [08/01/2017 at 06:48 PM](#)



Hi Brett, thanks!

With regards to TDZ. August 6th will mark the 10th anniversary of the day I came up with the idea. Since then, despite beating every bush and using every approach I could think of, the sad fact is none of the Dell EMC storage platforms support the feature (today).

That said Brocade has supported it since FOS 8.0.

Also, I'm watching (with a perverse sense of amusement) the stuff the HPE is doing with SmartSAN (which they've referred to as Target Driven Peer Zoning). It's just nice to know that the idea went somewhere... :-)

I've also heard of a couple of other Flash array vendors that support it but I haven't dug into it much.

I agree Smart Zoning isn't TDZ but it seems to be a step in the right direction and it is available today!

I will post the mother of all blog posts the day that Dell EMC supports the feature, which I still hope will happen at some point.

Regards, Erik

Posted by: [Erik Smith](#) | [08/03/2017 at 10:51 AM](#)

The comments to this entry are closed.

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