

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

04/16/2014

An Introduction to Virtual Storage Networks

I recently started a [VSN blog series](#) that describes (in great detail) a new concept that we're referring to as a Virtual Storage Network or VSN. As I was writing the third installment in the series, I realized the approach I was using to define the VSN concept was analogous to trying to describe a forest by starting with a description of soil composition. Although this approach might have eventually led to some kind of understanding of what I was talking about, I began to feel that I was probably going to lose the vast majority of readers if I didn't take a different tack. With this in mind, this post will provide a very high level overview of the VSN concept, describe what we're trying to accomplish and hopefully provide a framework to allow readers to understand the more detailed [VSN series](#).

Why VSN?

Since Fibre Channel was first introduced, the need to manually configure FC connectivity (e.g., [create zones](#)) has not only proven to be unpopular with customers, it has also proven to be a barrier to innovation.

From a customer's point of view, many view zoning as just an additional administrative burden and have been asking for a way to eliminate the [manual process](#) of zoning for years. We've responded to these requests by:

- introducing products such as Volume Logix, ECC SAN Manager and ViPR;
- participating in industry initiatives (e.g., [FC-SCM](#)) intended to eliminate zoning; and
- inventing a new mechanism ([TDZ](#)) that automates zone creation

As we were standardizing TDZ, we asked other EMC engineers for input and were surprised to discover that many of them were interested in the approach because they had run into problems with zoning in the past. In fact, several of them had considered adding functionality but avoided doing so specifically because of zoning. On the opposite end of the spectrum there was at least one case of an engineer scripting the zoning commands required for their particular

[Erik Smith](#)

[ARCHIVES](#)

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.

use case. While their script worked fine, ultimately I think they were just cleaning up someone else's mess and thought it was a shame that they had to spend time working around a limitation that should have been addressed long ago.

A more recent example of the barrier to innovation is the [FAST sideways](#) use case; it takes no less than four connectivity (e.g., zoning) changes in order to migrate a storage volume from one array to another. Until these connectivity changes can be automated (e.g., via ViPR); FAST sideways will probably not be all that useful as a automated remediation mechanism for chronic issues with IOPS or RT due to an overloaded array.

Although zoning is something that is specific to FC and FCoE, other types of storage protocols also suffer from the need to manually configure connectivity. I described the challenges with iSCSI in the [FC and FCoE versus iSCSI - "Network-centric" versus "End-Node-centric" provisioning blog post](#) and I described the challenges for FC, FCoE, iSCSI and File in the [Universal Fabric Controller blog post](#).

While these manual connectivity configuration steps have proven to be burdensome to our Enterprise customers, to our Service Provider customers, they're anathema. This is because many of these Service Providers are gearing up to compete with the likes of [Amazon](#) and in order for them to complete effectively, they need to be able to:

1. Completely automate the end-to-end provisioning process (including storage provisioning); and
2. Provide services at a price that are difficult to match when using traditional infrastructure components.

As a result, many of these customers are not interested in FC or FCoE because they perceive it to be a more expensive and less flexible solution, due to the perceived manual provisioning steps, specialized skill sets and specialized hardware associated with it.

Actually, if you press them for what they really want in regards to storage features, many just want storage arrays to go away and have the storage capacity provided via a software based approach. I'll generically refer to this approach as [Server SAN](#) and point out that EMC has a product in this space called [ScaleIO](#). I'll also point out that VMware has a solution called [VSAN](#) which has been on our radar for some time. Both of these products (along with [CEPH](#)) have proven to be very popular (at least conceptually) with Customers in both the Enterprise and Service Provider spaces. I think the main reason for the interest is [ease of use](#).

I'll admit that I'm intrigued by the Server SAN concept and I see use cases for both ScaleIO and VSAN. But I also have to admit, as I discuss in [part 1](#) of the VSN series, I have concerns about the ability of Server SAN solutions to provide Enterprise class features. That having been said, the message from our customers is crystal clear **"Make it easier to consume your array based storage or we'll find another solution"...**

I'll spare you some of the details and simply say that we took this message to heart. We did a bunch of work gathering connectivity requirements from customers (including at EMC World last year) to figure out what the best approach would be. As we were doing this an interesting pattern started to emerge:

Enterprise Customers (in general):

- Really like Fibre Channel
- Really do NOT like iSCSI
- Are willing to use FCoE at least from the Server to the ToR switch
- Are interested in investigating approaches to automate zoning
- Are increasingly looking at File
- Are looking at Server SAN to see what it can do for them but generally like the automation aspect of it
- Are typically very conservative when it comes to new technology
- See connectivity from Server to a Storage array as something that is static.
- Have a wide range of requirements with regards to multi-tenancy. Generally they see it as something that is handled through orchestration and not necessarily something that needs to extend to the data plane.

Service Providers (in general):

- Really like the Server San concept
- Like iSCSI, File, Object - will use whatever provides adequate performance and is easiest
- Really do not like FC - There are exceptions (e.g., Rackspace hybrid cloud and at least to some degree vCHS)
- Are not conservative when it comes to new technology
- See connectivity from Server to Storage as dynamic
- Have many multi-tenant storage requirements. Generally they see multi-tenancy as something that is handled through orchestration but also want data plane isolation as I describe in [Part 1](#).

In order to satisfy both the Enterprise and SP requirements, it became clear that we would need a solution that could support multiple protocols and automate connectivity tasks for all of them. Not only that, we would need something that makes our arrays as simple to use as a Server SAN solution.

Simple, right? </sarcasm>

With these requirements in mind, we set a goal for the VSN project to:

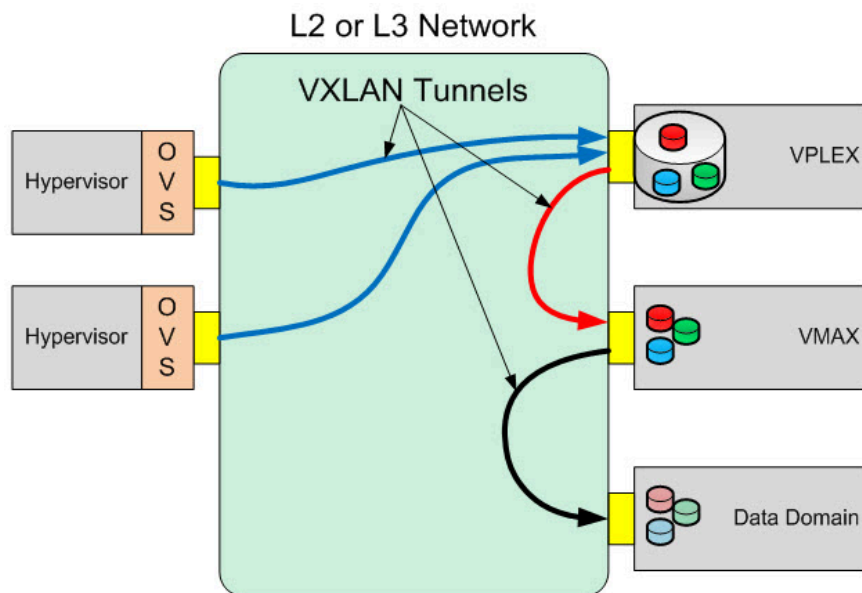
- enable the dynamic creation of storage service compositions;
- automate connectivity between compute and these storage service compositions; and
- do so in a way that is compatible with the concept of multi-tenant storage.

In case you're wondering, a storage service composition is a collection of storage services that are presented to an end user as a single storage volume. For the purposes of this discussion, think of a storage service composition as a storage volume that has specific attributes. These attributes could include:

- [Input/Output Operations Per Second \(IOPS\)](#).
- [Replication](#)
- [Data Deduplication](#)
- [Recovery Time Objective \(RTO\)](#)

- [Recovery Point Objective \(RPO\)](#).
- And many more...

One example of a storage service composition that has been automatically attached to a compute instance is shown below and is described in [Part 2 of the VSN series](#) in the "Storage Service Insertion" section.



The key take away from the above diagram is each hypervisor would only see a single "gray" Virtual Volume on the VPLEX. The storage capacity for the Virtual Volume actually resides on the VMAX. The VMAX could then be getting deduplication services from Data Domain. I refer to the Virtual Volume, the capacity service on the VMAX and the deduplication service on Data Domain as a "Storage Service Composition".

I have not (yet) attempted to calculate how many possible storage compositions could be created from just EMC products alone, but there are many others possible. It's important to note that from a connectivity automation point of view, the number and type of services doesn't really matter as long as the two services can use a common protocol (e.g., FC, iSCSI, File, Object, etc) to communicate with one another.

Conclusion

Well, that's about as close to the forest of Virtual Storage Networks that I can get without getting back into a study of soil composition. If you were to remember only a couple of things about this post, I would like them to be:

- The purpose of the Virtual Storage Network concept is to make storage and storage service compositions as easy to use as possible
- VSNs will be compatible with both Enterprise and Service Provider environments
- A Virtual Storage Network can be created using any of the supported protocols today
 - There are limits to this and they will be discussed in parts 3 and 4 of the series.
- Virtual Storage Networks do not have inherent orchestration layer dependencies

If you're interested in more information please see the more detailed [VSN series](#).

Thanks for reading!

Posted at 04:21 PM | [Permalink](#)



Comments

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

The comments to this entry are closed.

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