



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

[« Getting the most from your SAN with Dell EMC PowerMax | Main | Let's play in an AWS based SANdbos - Part 2 Setting up your AWS VPC and networks »](#)

09/23/2022

Let's play in an AWS based NVMe/TCP SANdbos!

One of the great things about NVMe/TCP is its flexibility.

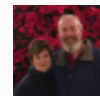
Not only can you use the NVMe/TCP protocol to connect hosts to storage systems over an arbitrary IP network topology; as originally described [here](#), you can also run it in a surprising number of operating environments, including:

- traditional enterprise environments that utilize array-based storage solutions (e.g., PowerStore and PowerMax),
- next generation operating environments that utilize Software Defined Storage (e.g., PowerFlex),
- completely new use cases, such allowing applications running on the Edge to access centralized block storage resources, and most recently,
- deploying your NVMe/TCP solution in the Cloud (e.g., AWS) and using the resulting "Virtual Lab" to simply get familiar with an NVMe IP-Based SAN, or more importantly test and validate your infrastructure automation software before you use it in production.

The last item, especially the part about using AWS as a Virtual Lab, needs a bit more explanation. Along these lines, this blog post series will:

1. Provide an overview of the automation tools being made available in the SANdbos Github repository,
2. Provide the step-by-step instructions required to configure a AWS based NVMe/TCP test environment, and then
3. Demonstrate how you can get started with automating your storage provisioning tasks by using the PowerShell, Python or Ansible scripts that we will provide in SANdbos.

Introducing SANdbos!



[Erik Smith](#)

[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.

[SANdbos](#) is a Github repo that provides access to the resources you will need when automating your NVMe IP-Based SAN. This includes:

Documentation

[Start here](#) if you're just coming up to speed on the concept of an NVMe IP-Based SAN. You'll find links to an IP-Based SAN introductory level blog post as well as links to two detailed SNIA presentations describing how Discovery Automation works at a protocol level.

There's also a link to Dell's SmartFabric Storage Software (SFSS) deployment guide. It covers not only how to install SFSS, but also the basic network topologies that are currently supported as well as step by step configuration examples.

Centralized Discovery Controller (CDC) [Downloads](#)

If you've been following the NVMe IP-Based SAN space for a little while, you probably already know that a group of companies identified a scalability problem related to the management of IP-Based SANs and decided to work together to solve it. The result of this work can be found in a couple of NVM Express Technical Proposals (i.e., TP8009 and TP8010).

- TP8009 describes how hosts can automatically discover subsystems using mDNS or DNS.
- TP8010 defines the concept of a Centralized Discovery Controller (CDC) that limits the number of storage subsystem interfaces visible to each host. As most people who have dealt with FC SANs can attest, this functionality is very important as environments scale up.

To help potential IP-Based SAN users get familiar with the usage of a CDC, Dell has made our implementation of a CDC, "SmartFabric Storage Software (SFSS)" available for download for trial purposes. Please note, this version is for trial and test purposes only and no support is available. If you want the latest, fully featured and fully supported version of SFSS, you need to purchase a license.

If you don't want to download and install SFSS, either on your own infrastructure or on AWS, you can always give our [Interactive Demo](#) a try.

Toolkit

Once you've installed SFSS and understand the basics of interacting with it, you'll probably be interested in learning how to automate the configuration steps and interact with your infrastructure programmatically. To this end, Dell has provided [sample scripts](#) that will help you learn the basics. Currently we're providing PowerShell and Python examples and will be adding Ansible soon.

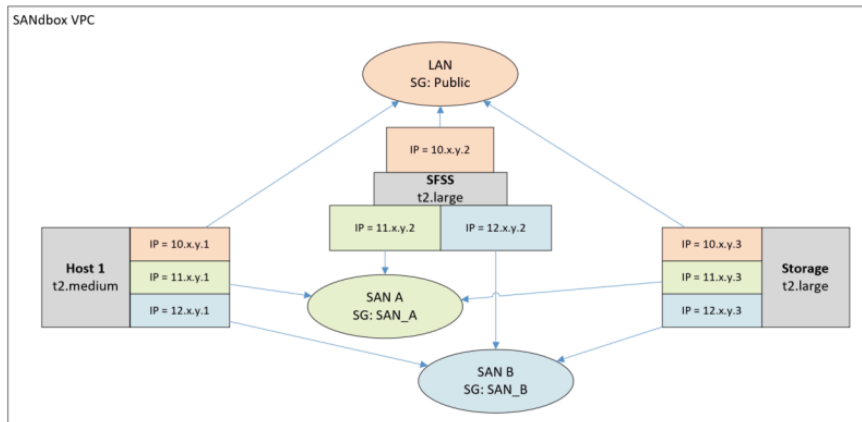
Both the PowerShell and Python scripts will allow you to:

1. connect to the SFSS via the REST API,
2. retrieve a copy of the name server database,
3. add all of the NS entries to a single zone, and then
4. activate the zoning configuration.

The experienced SAN admins amongst you will immediately recognize that zoning all of the ports together is NOT something that would typically be done! However, the example will give you enough information to do something a bit more realistic, such as allowing hosts to access a subset of the available subsystem interfaces.

Virtual Lab

If you're interested in experimenting with an NVMe IP-Based SAN but you don't have the HW infrastructure to do so, you can run all of it in AWS and experiment with it there on your own! The configuration I'm experimenting with is shown below.



In my AWS configuration I've created:

1. Three EC2 instances (Host 1, SFSS, Storage) all running Ubuntu 20.04
 - a. The **Host** instance utilizes the Dell maintained open source nvme discovery client "nvme-stas"
 - b. The **SFSS** instance is running our SFSS application and it handles all aspects of discovery automation
 - c. The **Storage** instance is running either nvmet or our "Dell End Point Simulator" that allows you to simulate pull and push registration from the command line.
2. A public facing network that I am using to access the instances from my desktop. This allows me to run my PowerShell or python scripts locally and configure my SFSS instance running in AWS.
3. Two private networks (i.e., SAN A and SAN B) that will allow you test the impact that things like zoning changes, or loss of connectivity to the CDC will have on your hosts ability to access block storage.

The [Virtual Lab directory](#) will eventually contain all of the information and configuration steps you'll need to setup an AWS based virtual lab of your own that is capable of using NVMe/TCP to connect from your host to your subsystem. Much more will be coming in this area.

Take a look at [SANdbosx](#) today and take what you need or feel free to contribute!

The next post in this series will provide the step-by-step instructions to setup your AWS based NVMe/TCP lab... Stay tuned.

#iworkfordell

Posted at 08:25 AM | [Permalink](#)

Reblog (0)

Comments

 [Feed](#) 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](#)