3

Troubleshooting Storage Visibility

Certainly by now we have a solid understanding of how vSphere handles storage as well as a lot of the common tools we can use to troubleshoot and view storage. Now it’s time to deep dive into the primary topic of this book which is troubleshooting vSphere storage. This chapter will hit on on3 of the most common topics I’ve seen when it comes to vSphere storage; visibility. We will go over some common steps to take when the various types of storage are not seen by ESXi. We will also cover how to troubleshoot individual fiber channel paths to storage, storage Claimrules and LUN masking as well as LUN re-signatures. After than we will move into network storage and discuss the various troubleshooting steps to undertake when dealing with both iSCSI and NFS visibility issues.

This chapter will cover the following:

* Troubleshooting Fibre Channel Visibility

Fiber Channel Switch Zoning

ESXi Claimrules and LUN Masking

Troubleshooting Paths and Path Selection

LUN signatures and vCenter Storage Filters

Partition Tables

* Troubleshooting Network Storage Visibility

Verifying network connectivity to your iSCSI/NFS Array

Firewalls and port requirements

Jumbo Frames and MTU Size

Troubleshooting Paths and Path Selection (iSCSI)

CHAP Authentication (iSCSI)

NFS Share names (NFS)

# Troubleshooting Fiber Channel Storage Visibility

Fibre Channel has been around since 1994 which makes it tried and true and one of the most reliable storage transports around today. It’s for these reasons that it is also one of the most common storage transports found in virtualization environments and datacenters around the world. Fibre Channel is deployed in somewhat of a network fashion, utilizing switches in between the storage arrays and the hosts accessing the storage. This allows for multiple links to be established from both the storage array to the fibre channel switch and the switch to the hosts, which in turn provides multiple paths to the same storage, or redundancy.

If you are unable to see your fibre channel storage I’ve always found that most the time it is due to latency, which we will talk about in Chapter 4 however there are times when the connection issues are related to either zoning or masking issues which are explained below.

## Fibre Channel Switch Zoning

Fibre Channel zoning is the process of partitioning off a Fibre Channel fabric into smaller chunks to restrict connectivity, visibility, and add a level of security to your Fibre Channel network. I often try to think of the zoning functionality to that of a firewall, where we specify who can see who and who can’t see who. We normally zone to restrict access. Say you had a couple of storage arrays (one for your physical windows infrastructure and one for your vSphere infrastructure); you would not want your physical windows servers to gain access to a vSphere datastore. That would not make any sense at all – so we zone. We restrict our Zoning is performed at the switch level and can be applied in a couple of different scenarios. We can zone the ports, which basically authorizes one switch port to see another switch port. This means that if the device connected to the port moves, it will no longer have access. On the flip side if an new device is connected to the zoned port it will have access. Another type of zoning is called WWN zoning. WWN is essentially the same as port-based zoning just replacing the ports with WWNs. This allows to maintain access to the zone as devices are moved around on the switch as the WWN stays with the device. Also, if a new device is plugged into a port where a WWN zone once existed, it will not automatically gain access.

Zoning is different across the various vendors switches so it is hard to come up with a generic solution when troubleshooting storage visibility. Always follow your vendors recommendations for zoning configuration and always remember zoning when investigating Fibre Channel connectivity and visibility issues.

## ESXi Claimrules and LUN Masking

As we discussed in Chapter 1 each storage device managed by vsphere is done so by loading a plugin inside of the vSphere Pluggable Storage Architecture.  The process inside of the PSA that associates our storage array devices with the proper storage plugins, whether that be the default NMP provided by VMware of a third party plugin provided by a third-party is called claiming. Claiming is essentially a group of rules called clam rules which contain associations between the storage devices and either the MPP or the NMP.

So why are clam rules important when it comes to troubleshooting vSphere storage visibility?  Well just as we took a look at zoning and how it can block a vSphere host from seeing the storage, claim rules can utilize what is called a mask\_path plugin to do the same.  Most commonly these per-lun masking techniques are done directly on the storage processor of your SAN however ESXi does have this technology implemented within its own stack. Claimrules, along with the mask\_path plugin can be loaded into runtime of the ESXi host in order to remove visibility to a certain lun or a single path to a LUN.  To list out the claim rules currently loaded we can use the “esxcli storage core claimrule list” command as shown in figure 3.1  Claimrule access is currently not available inside of the vSphere web client.

2062\_03\_01



Figure 3.1 – Claimrule List

As we can see in figure 3.1 claimrules must be assigned a certain type.  The available types in vSphere are vendor, model, transport, and driver.  “Type” essentially describes a target or a match for where we would like the claimrule to be applied.  Meaning we can apply a claimrule to a specific vendor or model such as HP EVA or Dell EqualLogic, a specific transport, meaning iSCSI or Fibre Channel, or a specific driver or path to a LUN.  Another notable property of a claim rule is its ID. Every claimrule must be assigned an ID between 1 and 65534.  The highest claimrule number is actually 65535 however this is reserved as a catch-all rule to allow vSphere to claim any device that does not get claimed by any previous rules in order for it to be managed by the NMP.

Claim rules are executed in sequential order from smallest to largest.  Once a rule is matched no more rules are applied.  Meaning if you have a mask\_path definition at number 34, then a MPP claim at number 40 - the path will never be seen by vSphere as it will be masked away before the actual claim of the storage array.

As you can imagine claim rules can be a bit tedious to analyze when troubleshooting vSphere storage, and for the most part they are not there unless you have explicitly created them.  That being said, if you are having an issue with discovering storage in vSphere a lot of the time there are claimrules that are masking paths away causing the issue.  This is especially true when looking at environments that you aren`t familiar with.  Administrators may have masked away a complete storage array or certain individual paths during an upgrade or maintenance.  So what do we do if we have a claimrule that is causing the issue?  If we look at the claimrules listed in Figure 3.1 we can see that we have the MASK\_PATH plugin loaded for LUN 0 on vmhba32. If we were trying to access the datastore that sits on this LUN we would need to remove the claimrule before we would even be able to see it.  Taking into account the information displayed in Figure 3.1 we could remove the rule with the following command.

esxcli storage core claimrule remove –rule 400

Use caution when deleting claim rules as they aren’t always as they appear.  You may think you are solving an issue and end up creating another.  Definitely take caution inside of any production environment and always run the vm-support command to dump your configuration before removing any claimrules in the event that VMware support will need this information.

Simply running the remove command will only remove the ‘file’ class from the claimrule, not the runtime rule. In order for our removal to take affect we will need to first load our current claimrule set with.

esxcli storage core claimrule load

Secondly we will need to unclaim the device or target in question from the MASK\_PATH plugin. Again, using the information in Figure 3.1 we can unclaim our LUN with the following command.

esxcli storage core claiming unclaim -t location -A vmhba32 -T 1 -L 0

Only after reloading and unclaiming will the claimrule be completely removed from the ESXi host. A simple rescan of the hba in question now should result in us seeing the datastore that we were looking for.

If you are trying to add a new datastore and vSphere is having issues seeing it, aside from looking at your Fibre Channel zoning, Claimrules are a great spot to start.

Troubleshooting Paths and Path Selection

Whether it be the result of zoning, LUN masking, or other environmental issues a down or inactive path can definitely result in a datastore not showing up within vSphere if the proper redundancy design has not been implemented.  Most Fibre Channel SANs today have at least two storage processors, allowing for redundant links on the storage array to be present.  In the event that one fails the other can be activated depending on the failover method implemented.  In most cases there will be at least 2 paths to the storage, more commonly we see 8 or 16 depending on the number of storage processors, hba's and ports on the hba. vSphere manages these paths using the pluggable storage architecture and more-so the satp and psp associated with the storage array.  This is why we need to pay attention to the path selection policy assigned to each array.

Although vSphere provides a default Path Selection Policy for different SAN configurations (IE Active-Active SANS use fixed, Active-Passive SANs use Most Recently Used) there are times when these will need to be changed. Storage vendors know best how their hardware will integrate with vSphere so always check their recommendations and best practices. For instance, the HP MSA 1500cs has an Active-Active firmware, which in terms of VMware would be best suited with a fixed path selection policy. That being said, HP themselves recommend utilizing the Most Recently Used setting due to the nature of how the HP controllers handle the ownership and presentation of LUNS to ESXi. This is why it is important that we always check our storage vendors recommendation and best practices when implementing storage in vSphere in order to minimize the number of storage connectivity issues we run into.

Another factor that can play a key role in vSphere storage visibility is path states. As mentioned earlier most environments are designed with redundancy in mind, allowing individual paths to fail and recover without affecting our production workloads. That being said we should know how to view, configure, and monitor a paths state in order to pro-actively prevent outages and loss of data within our environments. Examples of a paths state could be active, standby, dead, or disabled and basically describes the current flow of data through the path. To view the current state of a LUNs paths inside of the vSphere Client follow the following steps.

1. Click on the host containing the desired LUN and select Storage from the Configuration tab
2. Select your desired LUN and then click ‘Properties’

Click ‘Manage Paths’

We should now see output similar to that of Figure 3.2. In the ‘Manage Paths’ screen we are able to change the Path Selection Policy associated with the LUN as well as disable/enable paths and mark paths as preffered if applicable. The equivalent to this inside the command line is the ‘esxcli ???? ‘ command which will list out all available paths on the system. Adding a –d and the device indentifier will limit our result to a specific device. This helps us see the status of paths and how all of our LUNS are configured in terms of their Path Selection Policy but is not very helpful in terms actively monitoring the paths for a state change. We don’t want to have to go into these screens every day to see if something has changed. In order to take a pro-active approach to path state change we can setup an alarm within vCenter to monitor for ????? The steps to setup an alarm are as follows.

STEPS TO SET UP ALARM ??????

## Storage Filters

Although not commonly disabled, another portion of vSphere that will cause headaches when dealing with our storage connectivity and visibility are vCenter Storage Filters.  vCenter provides us with four storage filters; VMFS Filter, RDM Filter, Same Host and Transports Filter and the Host Rescan Filter.   These filters are all enabled by default and the details of the functionality they provide is listed in table ?? below.

vmfs filter - filters out luns that are already contain a vmfs partition.

rdm filter - filters out luns that have already been mapped inside of vsphere as a RDM, whether it be physical or virtual mode.

same hosts and transports filter - filters out luns that cannot be used as a datastore extent - meaning those luns that are not presented to all hosts as well as luns that may use a different storage type as the primary datastore extent.

host rescan filter - automatically scans and updates datastores after any datastore operations have been performed.

Aside from the Host Rescan Filter the others do just as they describe; filter out LUNS. Although these filters are in place to protect us from inadvertantly destroying or corrupting data, sometimes they bring troubles when we actually have a legitimate reason to see those LUNS.  Take for instance an administrator tyring to setup a Microsoft cluster inside of vSphere. In order to do so he would need to map the same raw LUN or RDM to both all virtual machines in the cluster. With the RDM Filter running under its’ default configuration only the host which owns the first VM to see the RDM would see this storage. Certainly this is a valid scenario where we would need to disable the RDM Filter in order to gain visibility to the LUN from another VM on another host.

The storage filter settings are all contained inside of the advanced vCenter Server settings and can be accessed by performing the following steps.

????????

## Disk resignaturing

When we mount or present a VMFS datastore to an ESXi host we most commonly generate a new signature for that datastore.  A signature is composed of many things, but most importantly is the UUID that is generated inside of the header of the datastore.  This UUID provides the ESXi host with the ability to determine that each and every datastore is unique, most importantly, virtual disk configuration from the VMs point to this UUID in order to ensure compliance and security around the fact that the disks sit on a certain datastore.  The most important thing to remember around disk resignaturing and UUIDs is that we never want to have two datastores mounted with the exact same UUID at the same time.  If this were to happen, ESXi would be very confused about where to send its SCSI commands, reads, and writes and would simply start to send I/O down each path at random.

In most cases we will always want to to generate a new UUID for each and every datastore that we present to a host and this is the default action selected by vSphere when dealing with a duplicate UUID and datastore mount, however there are some niche cases, mostly revolving around SAN snapshotting and replication where we will not want to generate a new UUID or resignature the datastore but keep the original UUID.

For example, say we had an issue that caused one of our datastores to become corrupt, and we want to present and mount a SAN snapshot that had been taken moments before the corruption occured.  When doing so we would probably choose not to resgnature the LUN and mount the datastore with the same UUID as was before (SAN Snapshots contain the same UUID as the LUN that the snapshot was performed on).  By keeping the existing UUID we save ourselves a bit of time in terms of virtual machine managment.  As mentioned before, VMs store a config on where their virtual disks live and use the datastore UUID to do so.  If we had chosen to generate a new UUID we would either have to go into each VMs config file and update to the new UUID or remove and re-add all VMs to inventory - Either option would take quite a bit of time which does not come without a price in an enterprise environment.

\*\*\*NOTE

Only mount a datastore without resignaturing when you are absolutely sure that the oringal datastore will not come back online within vSpere.

\*\*\*\*

The options for resignaturing and adding datastores are explained below uusing both the vSphere Client as well as the ESXi command line.

??????????

# Network Storage (iSCSI)

Although most the troubleshooting techniques between Fibre Channel and Network storage will overlap (mostly between FC and iSCSI since they both use a block transport and VMFS partitition) there are quite a few differences between the two storage types.  Since we are now using network adapters inside our hosts and network switches (as opposed to HBAs and FC swithces) most of the same concepts around network security such as firewalls, routes, etc will be applied to our iSCSI and NFS storage.

## Verifying network connectivity to our iSCSI and NFS array.

The first step we should always take when experiencing connectivity and visibility issues to our network storage is to always verify connectivity between our two endpoints; the host and the storage.  Whether that be properly assigning network information to our hardware iSCSI initiators or properly creating a vmkernel port group for the software iSCSI initiator or NFS client, we need to ensure that connectivity is available.  To do this I almost always use a simple, traditional command; ping.  However when using ping inside of ESXi it always utilized your management network as its transport; best practices normally state we have a seperate network for storage, or at the very least a seperate VLAN so in order to ping out of our other vmkernel interfaces we need to use a slightly modified, VMware command called vmkping as shown in figure ??.??

INSERT FIGURE

If we arent receiving a response from vmkping from our storage array then we will have to dig a little deeper.  Most commonly this is a result of not having the corrrect network information applied to either our hardware initiators or vmkernel ports.  This is the first thing I would check.  Secondlly it could be an issue with static routes set up on the host itself.  In order to view our routing tables on our hosts we can use the following comand.

esxcfg-route -l

Always check to see if there is a route setup on your host that could be causing your iscsi or nfs traffic to be routed through the wrong nic or network altogether.  In the case of the above figure we can see that the 10.10.1.0/24 network is routed through a gateway of 10.10.18.1 - in my case this is wrong and I need it to be routed through 10.10.1.1.  To remove the incorrect route we can use the following command

esxcfg-route -d <ROUTE\_NUMBER>

and to re-add the correct route (if needed) we can use

esxcfg-route -n 10.10.1.0/24 -g 10.10.1.1 -a -f

Figure ?? now shows a succesfully vmkping to my iSCSI arrray located at 10.10.1.250

INSERT FIGURE

Once we have verified network connectivity to our storage array the next logical step to take if we are still experiening connectivity issues is the port requirements of both iSCSI and NFS.

## iSCSI and NFS Port Requirements.

Since we are traversing our stroage through our ethernet network, the same fundamentals for troubleshooting networking must be applied to our storage.  Meaning we must ensure that the required ports are open not only on the hosts and storage arrays themselves, but also on any firewalls, proxies, or network devices that sit inbetween the hosts and their storage.  Table ???? shows the required ports that are need for iSCSI and NFS.

Tablee with all required ports for iscsi and nfs.

Since there is a wide variety of firewalls on the market today its utterly impossible to explain how to create rules on them all - just esure that tcp and udp ???/ is open ???

When you configure iSCSI on an ESXi host the required local rules are enabled on ESXi.  If however for whatevver reason you need to modify these rules or ports ou can use the following example

example of how to create/delete/modify firewall rules on ESXi

## Jumbo Frames and MTU Size

although not required, administrators will often increase the size of the MTU from 1500 to 9000 in order to support jumbo frames and ultimately provide some higher throughput to their iSCSI array.  This is done inside of the vSphere Client by modifying the MTU size on the vmkernel port associted with the iSCSI network.

example of jumbo frames

although this is easy enough to change within vSphere itself, the trickier parts come in the fact that jumbo frames must be enabled on every single network device that the iscsi packet traverses as well as the iscsi array itself.  If it isnt enabled everywhere you will often see dropped packets or disconnects from the storage.  New in vsphere 5 is some sort of jumbo fram detection ???  a great way to monitor jumbo frames is by setting an alarm to monitor the network dropped packet aborts on the vswitch contianing the vmkernel port groupp

examople of setting upthe alarm.

## Troubleshooting iSCSI paths

Just as we can have multiple paths for fibre channel storage we can do the same for iSCSI by bonding physical NICs and vmkernel ports together to provide a load balancing solution.  iSCSI will also leverage the Pluggable storage archictecture to determine the best path possible to send the data down, as well as determine when a path has failed and a failover will need to occur.  So, as with fibre channel it is essential that we monitor the path states and failovers on an iSCSI connection as well.  First however lets quickly go over how we setup multi-nic iSCSI connections

example of setting up multi-nic connections - quick one