2014

University of North Carolina Wilmington

Master of Science in

Computer Science and Information Systems

Proceedings

https://csbapp.uncw.edu/mscsis

AN IMPLEMENTATION OF A GREEN STORAGE AREA NETWORK

Tyler Loftis

A Capstone Project Submitted to the
University of North Carolina Wilmington in Partial Fulfillment
Of the Requirements for the Degree of
Master of Science in
Computer Science and Information Systems

Department of Computer Science
Department of Information Systems and Operations Management

University of North Carolina Wilmington

2014

Approved by

Advisory Committee

_____	_____
Dr. Ron Vetter	Dr. Ulku Clark

_____
Dr. Doug Kline, Chair

Abstract

A reliable and useful storage area network (SAN) was created using two legacy servers and low-cost commodity hardware. Care was taken to create a "reference" implementation for education purposes by using color coded cables, careful cable management, labels, and diagrams. The SAN is intended to support a scalable number of drive-less servers that will host virtual machines. The SAN servers boot from USB to Open Solaris, leaving the hard drives dedicated to SAN use. Comstar iSCSI was used to communicate between the servers, and the Zetta File System (ZFS) with RAIDZ2 was used to protect against drive failure. IOMeter performance measures indicate that the SAN performs as well as a standard direct-attached hard drive, and approaches some local SSDs. The end result is redundant, fast storage using decommissioned and commodity hardware.

Contents

Chapter 1: Introduction / Motivation

The goal of this project is to build a low-cost eco-friendly hardware platform to host a self-serve virtual environment utilizing legacy hardware that had previously been destined for the landfill. The project involved server hardware, virtualization software (Hyper V), SAN system architecture, network design, and the iSCSI configuration. The platform is designed to host a virtual machine environment that is designed to support faculty and student research projects.

The end goal of achieving a green IT platform for virtualization utilizing the hardware, protocols, software, file systems, and operating systems proposed was achieved. However, some of the methods and techniques discussed have been altered from the original design as knowledge and experience were gained. The following pre-implementation methods and techniques were theoretical and based on research, literature, and to some extent educated assumptions.

Legacy servers were repurposed to create a load-balanced, redundant SAN system providing reliability. The SAN system features a dedicated SAN-only subnet and the state-of-the-art Zettabyte File System (ZFS) to achieve performance, ease of management, scalability, reliability, and reduced power consumption. Isolated network subnets were used to reduce network congestion, increase reliability, and improve performance.

The legacy host servers that were used to host the virtual machine guests are all drive-less; all data and operating systems are stored on the SAN. This reduces power consumption and heat produced by the host servers. Three new servers were purchased as host servers because the legacy servers could not support current virtualization software. The new servers

are also all drive-less and individual components were purchased and hand-built in an effort to keep costs low.

The environment will also be used as an educational resource for future students and visitors. Color-coded network cables and the use of labels, signs, and diagrams were used to illustrate not only the components of the physical architecture, but also how the internal systems come together to produce the environment. The resulting platform, based on repurposed hardware is reliable and scalable. The entire project is low cost and capable of hosting a self-serve virtual machine farm.

Chapter 2 reviews the technologies and terms used in this project. Chapter 3 presents the results of several expert interviews. Chapter 4 describes the initial architecture of the project, as well as the implementation plan. Chapter 5 presents the actual implementation and how it differs from initial plan. Chapter 6 considers the project retrospectively, reiterates the accomplishments, and suggests future work.

Chapter 2: Review

In this chapter we will discuss and define virtualization along with the infrastructure that will be created to support the virtual environment and the associated terminology of each. We will review some relevant literature and briefly review the history and background of virtualization. Also, we will cover some common technologies that are used to support virtual environments. Specifically, the technologies to be discussed will be storage area network (SAN) architecture, the zetta file system (zfs), and internet small computer system interface (iSCSI). Also, we will analyze how a virtualized environment can be utilized to solve some common information technology problems. Before we get started with the analysis we should define some terminology.

Definition of Terms

This section is intended to give readers a clear and consistent understanding of the terminology and jargon that is commonly used to describe and reason about virtual environments. Some of the terminology may be used interchangeably outside of this document, however for our purposes, the following terms and their definitions will be used throughout this report.

Virtualization Configuration Terminology

Virtual. The term virtual comes from the Latin word virtualis meaning virtue. The definition of virtual is: In effect or essence, if not in fact or reality. Another translation for virtual is “nearly” or “almost”. Put these two definitions together and you get something like this; something is virtual if it is in effect, nearly or almost the same as reality, but is still not

reality. For example, we will create virtual machines, that is, machines that only exist as software programs that run in an actual physical machine. Many virtual machines can be created and ran in one physical machine.

Virtualization. The definition of virtualization is the act of virtualizing or creating a version of something, such as a hardware platform or storage device, that is not the actual, physical item. An example of virtualization may be a virtual server. Rand Morimoto and Jeff Guillet, authors of Windows Server 2008 HyperV Unleashed, define server virtualization as:

"…the ability for a single system to host multiple guest operating system sessions, effectively taking advantage of the processing capabilities of very powerful servers."

Virtual Machine (VM). A software implementation of a computer that behaves like and appears to the network like a separate physical machine, but that can be one of multiple virtual machines running on the same physical machine.

Host. In order to draw a clear distinction, let's define the word "host" in terms of both a client/server relationship and in terms of virtual computing. In a client/server relationship the machine that provides services would be the host, aka the server. The client, aka the guests, would be the machine(s) that received the service in a networked environment. In terms of virtualization, the host refers to the physical machine on which guests may reside.

Guest. A guest, in a client/server relationship, is the machine that receives the services from the host, aka the server, in a networked environment. In terms of virtual computing, the guest refers to the virtual machine(s) residing within the physical host machine.

Hypervisor. A virtualization platform that enables running multiple operating systems on the same physical computer, also known as a virtual machine monitor (VMM). There can be two types of Hypervisors, hosted and bare metal. For this report, a hypervisor will refer to the bare metal hypervisor unless otherwise specified. The two different types of hypervisors are defined below.

Hosted Hypervisor. A hosted hypervisor is virtualization software that runs on a host operating system on which virtual machines run. Some examples would be Microsoft's Virtual PC and Virtual Server or VMware's VMware.

Bare Metal Hypervisor. A bare metal hypervisor is a virtualization platform that runs directly on the hardware and does not require a separate host operating system. Examples are Hyper-V, ESX Server, and Citrix XenServer.

Host or Parent Operating System. This refers to the base operating system installed on a physical machine, on top of which the guest operating systems are installed in virtual machines, and which interacts with the hardware.

Guest or Child Operating System. This refers to the operating system that is installed in a VM that runs on top of the host or parent operating system.

Snapshot. An image of the state of a virtual machine at a specific point in time that includes all the data plus the configuration information for that VM. This allows one to return to a previous state in the future after changes have been made.

Partition. A partition is an abstraction of a container that is managed by the hypervisor. This abstract container is isolated from the rest of the operating system and it consists of

processor and memory resources and of policies and device access. A partition is a lighter weight concept than a virtual machine and could be used outside of the context of virtual machines to provide a highly isolated execution environment.

Root Partition. This is the first partition on the computer. Specifically this is the partition that is responsible for initially starting the hypervisor. It is also the only partition that has direct access to memory and devices.

Parent Partition. The parent partition is a partition that is capable of calling the hypervisor and requesting that new partitions be created. In the first release of Hyper-V the parent and root partitions are one and the same - and there can only be one parent partition.

Child Partition. Child partitions are partitions that have been made by the hypervisor in response to a request from the parent partition. There are a couple of key differences between a child partition and a parent/root partition. Child partitions are unable to create new partitions. Child partitions do not have direct access to devices (any attempt to interact with hardware directly is routed to the parent partition). Child partitions do not have direct access to memory (when a child partition tries to access memory the hypervisor / virtualization stack re-maps the request to different memory locations).

iSCSI Configuration Terminology

iSCSI. (Internet Small Computer System Interface) An IP based storage networking standard used to facilitate the linking of data storage devices.

iqn. (iSCSI Qualified Name) The name issued when an iSCSI target or initiator is created.

Initiator. Device responsible for issuing SCSI Input/Output commands to a SCSI target and logical unit.

Target. Device responsible for receiving SCSI Input/Output commands for a logical unit.

Logical Unit. Device responsible for executing SCSI Input/ Output commands within the target.

Logical Unit Number. Set of one or more initiators that are combined for the purpose of being applied to a view.

View. Defines the association of an initiator group, a target group, and a logical unit number (LUN) with a specified logical unit. Any view entry added to a logical unit must not be in conflict with existing view entries for that logical unit. A view entry is said to be in conflict when an attempt is made to duplicate the association of any given initiator, target and logical unit number.

Target Group. A set of one or more SCSI target ports that are treated the same when creating a view. The set of logical units that a particular SCSI initiator can see is determined by the combined set of views. Each logical unit has a set of view entries, and each view entry specifies a target group, host group, and a LUN. An initiator from that host group, when connecting through that target group, is able to identify and connect to that logical unit using the specified LUN. You can use views to restrict the set of logical units that a specific initiator can see, and assign the set of LUNs that will be used.

Target Portal Group. List of IP addresses to determine upon which interfaces a specific iSCSI target will listen.

Host Group. List of initiators (iqn's) that will be able to "view" the logical units (LU).

Brief History

The idea of virtualization has been around for quite some time and was conceived through necessity to solve a problem. The initial problem was underutilization of computing resources. In the late 1950's organizations and institutions that were employing computer technology noticed that the processing resources of their computers, the majority of the time, was not being fully used. Researchers began to try to solve this problem. Their first attempt was known as Time Sharing. This was an attempt to divide the computer's resources so that multiple programs could be run concurrently on one machine. The limitation of this was that all of the programs would have to run on the same kernel; the result is that any program that needs to communicate with the bare metal of a machine, would have to be run on a separate machine. In the 1960's, IBM and MIT researchers jumped on the time sharing band wagon and implemented a series of machines that ultimately culminated in the Compatible Time Sharing System (CTSS). CTSS monitored input/output (I/O), scheduled jobs, and handled memory and is the framework on which modern virtualization technologies and management consoles are based. CTSS implemented a supervisor program that isolated virtual machines from the physical host on which they were mounted. Now, programs that need to communicate with the bare metal of a machine *think* that they are physical machines; therefore multiple virtual machine programs can now be run on the same physical machine.

Although the initial motivation for exploring virtualization was underutilization of computer processing resources, as virtualization matured, the IT community began to realize additional benefits over time. We will go over a few of the more outstanding of these:

Consolidation of physical hardware, Isolation of a machine or environment for testing purposes, Decreased configuration management, and Disaster recovery.

Consolidation of physical hardware

Virtualization allows for consolidation of physical hardware by allowing one machine to run several different operating systems (OS) and/or programs that can communicate with the bare metal of the host on which they reside without interfering with one another. Thus taking several underutilized machines and combining their respective OS's and programs together on one physical machine, to get one fully-utilized machine. In the past a server may have only one application on it that may not be used by many employees and not very often. An organization may have several machines being underutilized in this same way. Virtualization technologies can create and manage virtual machines that support those applications. Now one physical host can have many different virtual machines or guests that facilitate the seldom used applications. Those applications are still isolated within their own virtual environment, but because there are now several of them on the same host, the physical machine's full potential can be realized. Other benefits realized by this type of configuration are - now there are fewer physical machines to power, to cool, and to manage. All of these techniques can add up to surprisingly powerful, cost-reducing results.

Isolation of a machine or environment for testing purposes

Isolation of a machine or environment can be very helpful for many reasons; such as testing new applications, new configurations, new upgrades/updates, and verifying compatibility to name a few. An organization may want to implement a new application but may not want to risk downtime in their production environment or may not have a good

understanding of what the overall costs and risks might be. The only way to test the new application, configuration, or upgrade fully is to replicate that specific environment with those specific configurations. This would be quite costly, and in most cases, is impossible to replicate exactly. However in a virtual environment system images and snapshots can be used to create identical copies of the real-world system and the new application, configuration, or upgrade can be tested without ever making any changes to the actual production environment. As a result, the organization can analyze the outcome of the testing phase to see if the new implementation is necessary, feasible, or within their means.

Decreased configuration management and disaster recovery

All organizations that employ information systems have at least one thing in common, configuration management. Configuration of the machines and the environment is necessary in order to ensure a system that provides the services needed and the performance desired. Generally the larger the organization, the more complex the system, which means the more difficult configuration management will be. In order to support a production environment, an organization may have many different machines with many different configurations. Some of these configurations may need to be repeated exactly across many different machines. These configurations must be intensely recorded and documented so that in the event of a failure, loss, or disaster the system can be recovered and restored to its identical state previous to the event. Utilizing virtualization technologies, the configurations of all of the machines in the system can be saved as images, templates, or snapshots and cloned as many times as necessary. If the organization has facilities in a different geographical location, the virtual

machines can simply be transferred on to the host machines at the alternate location experiencing little, if any down time.

With configuration management time being a major cost for organizations, virtualization alleviates the time involved by being able to create configuration templates. These templates can be created from scratch or from system images that already exist. Templates allow for particular configurations to be created only once; then when they are needed they can be cloned over and over again without the possibility of human error when attempting to replicate existing configurations. An unlimited number of templates can be saved so the process of cloning can be utilized for all different machine and operating system configurations. This significantly reduces the time involved in configuration management.

At one point or another all organizations are going to suffer through a loss or disaster event that can be very costly and even fatal to the organization. In the past, in order to protect against disaster, organizations have had to purchase additional servers or hire third-party services to back-up their data. This adds to costs and to an already over-crowded server room. Through the consolidation of servers, as mention earlier, an organization can now use those repurposed servers as disaster recovery back-ups without any addition hardware cost. Also, system images (which exist simply as files) can be used to quickly restore a machine to its identical state before the disaster occurred allowing the organization to get back to production as soon as possible. In the event of simple failure of a physical machine, for any number or reasons, the virtual guest sessions on that machine can be paused and the guest images can be transferred to another host and then be restarted to continue to run as normal with very little or no down time at all.

HyperV

HyperV is Microsoft's virtualization technology software, it comes in two forms; as a stand-alone product that can be implemented or as a role that can be enabled in MS Server 2008 R2. HyperV is a product that allows you to virtualize your physical machines so that they can operate on one physical machine in order to take advantage of the many benefits of virtualization. Some of which have been discussed in the previous section of this document. HyperV manages the virtual guest sessions and controls their access to the physical machine's bare metal resources on which they depend. HyperV began as Virtual PC. Microsoft acquired Virtual PC through a company named Connectix in 2003 and developed it into Virtual PC 2007. However the Virtual PC product line did not scale for organizations that wanted to run several virtual processes at once. Microsoft developed its first internal virtualization product with MS Server 2005. This product provided better scalability, but the hypervisor still ran on top of the installed operating system. This meant that any failure suffered by the host's operating system was also suffered by all of the virtual guests on that host as well. Changes made in MS Server 2008 R2 allowed for HyperV to run on a hypervisor layer. This hypervisor layer allows for the guests operating systems to talk directly with the bare metal hardware of the physical machine, bypassing the host's operating system all together. This allows for better performance, scalability, and reliability of the guest sessions.

Storage Area Network with ZFS

A storage area network (SAN), as defined by the Storage Networking Industry Association (SNIA), is as follows:

> *"A network whose primary purpose is the transfer of data between computer systems and storage elements and among storage elements. A SAN consists of a communication infrastructure, which provides physical connections, and a management layer, which organizes the connections, storage elements, and computer systems so that data transfer is secure and robust. The term SAN is usually (but not necessarily) identified with block I/O services rather than file access services."*

It accomplishes this through the use of high-speed, dedicated storage network devices that allow access to data storage. It is common to refer to the physical devices as SANs, but they are only the tools used to build the SAN. A SAN is actually the communication infrastructure that provides and organizes the physical connections and the data. A SAN can use a single dedicated server or can share many heterogeneous servers. SANs grew in popularity because they solve the problem of "islands of information". By allowing access to multiple types of vendors' products such as Windows, Linux, HP and IBM; SANs provided consolidation of data. This freed up valuable bandwidth because now information did not have to be transferred from one box to another, it could all be stored in one place. SANs also provided better fault tolerance and disaster recovery performance because the data to be backed up could be written to different types of storage arrays which could consist of disks or tapes or any other type of storage medium. This interconnectivity can be provided locally through a LAN or over long distances through a WAN and can be Ethernet or Fiber based. The devices that are connected to a SAN can be connected directly through the network and can be isolated within different subnets to

improve performance and ease administrative complexity. This allows for greater flexibility of the storage network with increased reliability, robustness, and security.

The file system that will be used is the Zettabyte File System (ZFS). ZFS was developed by Sun Microsystems and is a new kind of file system that uses a fundamentally different approach to data management. ZFS allows for increased simplicity, data integrity, and scalability. ZFS has a 128-bit capacity of 256 quadrillion zettabytes. One zettabyte is equal to one billion terabytes. This capacity was designed to surpass physical limits not theoretical, meaning that there is not enough usable matter on Earth to reach the maximum capacity of ZFS. Scalability is but one of the advantages of using ZFS. ZFS allows for the use of storage pools as opposed to volumes. This is beneficial because now there are no partitions to manage; the pools grow or shrink automatically based on the size of the data set. All the storage space in the pool is shared, and all the bandwidth stays available at all times. This decreases administrative overhead and increases I/O performance. Snapshots and clones are easy to create and restore. This is good for backing up and the cloning aspect will be particularly useful for this project. Further discussion about the cloning aspect will be discussed in the next section of this document.

Chapter 3: Expert Interviews

This project began with initial research into the ways that real-world organizations use virtualization, their motivation for adopting the technology, and the benefits gained. Interviews were conducted with several organizations that either employ virtualization on a daily basis or are otherwise dependent on virtualization in order to conduct business.

A standard on-line survey questionnaire was created and distributed to each company before face-to-face interviews took place to gather basic information pertaining to each company. The survey questions were not specific to any industry. Some questions asked in the survey were, for example, “what is the main industry of your organization?”, “how long have you used virtualization?”, or “what virtualization platform does you organization use?”. These were baseline questions to have some idea as to how much experience and investment an organization has with virtualization and how dependent they are on virtualization. See the appendix of this document for a copy of the survey questionnaire [5b], interview questions [1c], and responses.

Interviews were conducted with several organizations that utilize virtualization technologies in an effort to understand what motivated them to adopt the technology, how they are using the technology, and what benefits are gained. The companies interviewed were from different industries to gain an understanding of what aspects were common to all and what aspects were specific to the particular needs of each company.

Some common reasons for adopting virtualization technology were server sprawl, resource utilization, configuration management, and disaster recovery. All organizations with

in-house IT know the pains of server sprawl. Having to buy, store, power, and cool a growing collection of servers is expensive and difficult to manage; especially when the equipment has a usable lifespan of about three years at best. Virtualization helps to alleviate this problem by consolidating the many physical machines into one. The physical machines can then be virtualized and ran as guests on one physical host. This saves an organization a considerable amount of money in several ways. Now, there is no need to purchase a new physical machine when one is needed, just create a VM with the proper configurations. Since there were no physical machines added to the environment, there is no additional power or cooling costs. Additionally, upon adoption of virtualization, the amount of physical machines used will be reduced, lowering power consumption and reducing cooling needs from the very beginning. Also, the physical machines take up valuable real-estate within a business; now that space can be reallocated or reduced.

Resource utilization and configuration management, were also among the most common problems that virtualization can help to alleviate. Most servers do not run at full capacity at all times. This means that the majority of the time there are CPU and storage resources being wasted by sitting idle. With virtual technologies the VMs can be managed in order to take advantage of the full resource capacity of the physical machine on which they reside. For example, a VM that normally utilizes 25% of a CPU's capacity can be placed on a host with three other guests that have similar CPU usage requirements. Now the host will run at or near full capacity, wasting very little resources. Virtualization also helps with configuration management. Often hardware fails, when it does, new hardware has to purchased, built, and configured exactly as the previous. This is a time consuming processing taking hours at minimum and days to weeks at maximum. Utilizing virtualization, templates of common

machines can be built once and then saved. This means that now if hardware fails, an exact configuration is only a file transfer away eliminating hours of additional work and any possible human errors that may occur in trying to match the previous configuration. If the hardware is being used as a host and begins to fail, the guests on that host can be migrated to another host while running without having any downtime at all. This is known as hot-swapping. If it's the guest that begins to fail, snapshots or images can be taken of that guest so that another VM with matching configuration can be quickly spun up and used to restore the failure. This is also a good example of disaster recovery. In the event of a natural disaster like a flood or a hurricane, all of the guests can be hot-swapped to an alternate location safe from the disaster with little or no downtime. This is imperative for organizations that have high-availability and uptime requirements like hospitals, police, and fire departments.

Chapter 4: Initial Architecture

Deliverables

The deliverables for this project will be the fully built and configured "Green IT" hardware platform using all the technologies and techniques discussed in this document. The hardware platform will be color coded, labeled, and diagramed so that it may be used as an educational resource. Full documentation of the configuration setup and commands executed to achieve proper iSCSI configuration will be included. Diagrams of the physical architecture, network architecture, virtual architecture, and boot architecture will also be included.

Risks

There are many ways to build and configure the "Green IT" platform and with that comes many risks associated with each decision. The most prominent risk would be the decision to virtualize all the machines necessary to have a virtual environment and use physical machines as generic hosts only. This decision was based on the idea that if the entire environment is virtualized it will be much more robust, flexible, and reliable. A risk associated with this is, for example, the decision to virtualize the DHCP machine. It would be easier, in terms of initial configuration, to have the DHCP machine be a physical machine. However, if the piece of hardware on which the DHCP resides were to fail, then the entire virtual environment would not work at all. Therefore, if DHCP is virtualized it can be easily moved to another generic host with little effort and down-time increasing the reliability of the system as a whole.

Another risk is the decision to use all drive-less machines for the generic hosts. This is a risk because of the complex boot architecture involved. This decision was made in an effort to

lower costs through less power consumption and cooling needs. Again, this is more of an up-front initial investment, but may prove to be difficult to achieve within the given time constraints.

One final example of a risk involved is that this project involves cutting-edge technologies and techniques. There are few, if any, resources where one can look to for similar examples as a guide to proper setup and configuration. This project was conceived and designed through thought experiments and hypothetical scenarios. All of these technologies and techniques do currently exist, though not in this specific setup to achieve these specific goals. Only experimentation and full implementation of the project will tell if these ideas can be realized in order to complete this project using the ideas, techniques, and technologies discussed in this document.

There are many other risks involved in this project; far too many to list. However, if the project can be completed in the manner discussed it will result in an eco-friendly, highly reliable, highly scalable IT platform on which to host a virtual environment that will provide not only functionality to faculty and students, but also serve as an educational resource as a working example of green IT techniques and cutting-edge technologies.

A hardware platform will need to be created to host the virtual environment. Since the goal of this project is to be eco-friendly or “Green”, the majority of the hardware will be recovered from legacy equipment that would have otherwise ended up in a landfill. Some of the hardware will be recovered from UNCW’s campus while other parts rescued from local businesses that were simply throwing out old equipment.

A hardware configuration plan was designed to determine how this rescued legacy equipment will be utilized. A major factor in the configuration is determining how to get the highest level of performance while mitigating the risk of using legacy equipment. A networking scheme was developed in order to maximize performance and ease administrative complexity. A goal is also to reduce the dependence on physical hardware. More will be discussed about this dependence later in this chapter.

Questions were raised as to who would be the stakeholders, administrators, and users as well as how the system would be utilized by the end-users. All users and use cases were identified and the environment was built in a way that satisfies all users and use cases.

It is necessary to plan the configuration of the virtual environment. If legacy equipment is used, where would the VM templates be stored and how would the VMs boot? A boot sequence architecture would have to be developed in order to not only get the physical environment operational, but to get the VMs up and running as well. More information on the boot sequence of the system will continue in the Virtual Architecture section of this chapter.

Physical Architecture

The goal of this architecture is to create a hardware platform that is load-balanced and has a redundant configuration design that achieves eco-friendly reuse of equipment while still providing reliability, scalability, affordability, low power consumption, and decreased management complexity. Another important attribute of the physical architecture will be reduced dependence on hardware and the ability to extend the life of hardware.

The majority of the physical infrastructure will use legacy machines that will be rescued from the landfill. This will keep the initial investment costs low. Some of the machines are UNCW surplus machines and others are from local businesses that are simply throwing out old equipment. These legacy machines cannot run current virtualization software; therefore some new host servers were purchased. The new host servers are drive-less, purchased as parts, and built by hand in order to keep power consumption and costs low. A redundant SAN was created and features EON Open Solaris using the Zettabyte File System (ZFS). This will provide reliability and performance. ZFS provides excellent data integrity and allows for high performance at the storage level.

A network was designed with performance and configuration management in mind. The network has isolated subnets for traffic between the two nodes of the SAN, traffic between the hosts and the SAN, and traffic to the internet and other remote networks. This design prevents traffic bottlenecks, will provide a layer of security, and will ease network management.

The only machines that have hard drives are the two SAN nodes. This is where all c: drives for all virtual machines are located and where all data will be stored. In the initial design all other drive-less host machines were to boot-to-SAN through PXE and access all other data from the SAN through iSCSI. The exception is that the host machines now boot-to-USB. The drive-less machines will *think* they have a directly attached hard drive. Getting the drive-less machines to boot-to-SAN was to be accomplished by using a pre-installation execution environment (PXE) script that would be stored on USB. This would tell the machines the location of their hard drives on the SAN and instruct them to boot from there. The machines

would from then on access their hard drives and store data at these locations. The exceptions will be expanded on in chapter 5.

In order to accomplish the boot-to-SAN setup, a boot architecture was designed. The boot sequence would have to be adhered to in order for the machines to start correctly both physical and virtual. The SAN will boot up first; it will boot-to-USB. A physical machine, referred to as special host, will boot next. Special host would boot-to-SAN via a PXE script on a USB. The PXE script would indicate the hard drive location, which will be on the SAN, and configure static IP address. Once special host's hard drive is located, it will boot Windows Server 2008 R2. HyperV will be enabled as a role on Windows Server 2008 in order to begin hosting VMs. The first guest that will be needed will be a VM to provide DHCP services and will be hosted on special host. DHCP will boot-to-SAN in the same way as special host, but will use a virtual USB to locate its hard drive and get a static IP configuration. Once the hard drive is located, Windows Server 2008 boots and DHCP is enabled as a role. The DHCP guest can now provide IP addresses and PXE information to the rest of the generic drive-less host machines. Generic host machines will boot-to-SAN as well, but will get their IP address and PXE information from the DHCP guest already running. Generic host can now boot to whatever operating system desired and will be available to host multiple VMs. A boot sequence diagram is available in the appendix of this document [1a].

Internet Small Computer System Interface (iSCSI) will be used to provide communication between machines through a local area network (LAN) or a wide area network (WAN). At is most basic level; iSCSI involves an initiator and a target. The initiator sends, or initiates, commands to the target, where the commands are executed. This technology will provide the

host machine with the illusion that they have a hard drive. iSCSI will provide better performance, reliability, and management capabilities. Performance and security will be increased because each subnet is isolated. Reliability and management capabilities will be increased. If a piece of hardware fails, the administrator can replace it and quickly configure it as the same initiator or target that it replaces. A more thorough explanation of the iSCSI configuration will be discussed in chapter 5. iSCSI diagrams are available in the appendix of this document [5a] [6a] [7a] [8a].

Another goal is to design and build this hardware platform in such way that it can be used as an educational reference for future students and visitors alike. Labels and signs will be used to identify various parts of the system. Servers will be labeled and described. Network cables will be labeled and color coded. Diagrams of the various architectures and how they work together will be on display so that observers get an idea as to not only the physical setup, but also how the internal systems come together to create this environment. See appendices 2d and 3d for pictures of rack.

Virtual Architecture

Virtual machines are represented as software abstractions of “real” machines, and physical machines are only used to host these software abstractions. Having physical machines represented as software allows the system to be more reliable, scalable, energy efficient, and more easily managed.

If there is a failure of one of the physical host machines the VMs can quickly be migrated to other hosts and the damaged physical machine can be repaired or replaced. This also makes the system highly scalable. To increase the scale of the environment, simply add more generic

host machines; now many more VMs can be up and running with very little time and effort. Hard drives are a consumer of power in a computer and increase the heat produced by a server cluster. Because the only machines that will have hard drives are the two SAN nodes, this reduces power consumption and cooling needs. The reduction of physical machines by consolidating them as VM's on a single host server also greatly reduces power consumption. The effort involved in managing the virtual environment is mainly a one-time up-front investment where the configurations of the VMs are built and saved as templates. This means that the time involved in configuring future physical host machines should be minimal at best, as the templates can be quickly cloned and added as guest on the new host.

Use Cases

Use cases were created based on users and user requirements of the system. The use cases are: System Administrator, Library Manager, Guest Manager, and a User Manager.

The System Administrator will be responsible for managing the physical resources of the hardware platform and for migrating guests between different hosts when needed. If a physical machine fails, the System Administrator will migrate all VM guests to a working host machine. It will then be up to the System Administrator to decide to repair or replace the physical machine. Since the hosts are surplus machines it should be relatively inexpensive to completely replace the host with another surplus machine or to find another surplus machine within the organization. This is a great benefit and allows for much more flexibility when deciding with what to replace the failed machine. Any machine with the proper CPU and RAM requirements can serve as a new host. The System Administrator will also be responsible for creating and managing any and all reports needed as well.

The Library Manager will be responsible for creating new configuration templates, ISO images, and virtual hard disks (VHD) and storing them in the library for future use. Any editing or deleting of library items will be handled by the Library Manager as well.

Users of the system will have to be added to the system in order to interact with the self-serve environment. The User Manager will be responsible for adding and deleting users to and from the system. Once users are authorized by the User Manager they can login to the system and self-serve VMs.

Guest Management will be handled by authorized users of the system. Because this is a self-serve environment it will be the role of users to act as Guest Management. Users will be able to create VM guests from library items created by the Library Manager; start that guest and be responsible for it throughout the life of the VM. Users will have the ability to shutdown the guest, reset the guest, and delete the guest when no longer needed. Ultimate control will fall under User Management if a user fails to administer the guests properly. A copy of the Use Case Diagram is available in the appendix of this document [4a].

Chapter 5: Implementation / Results

Implementation

The finished product ended up being different than originally planned. When this project was conceived, I could only hypothesize results based on literature and theory as few, if any, real world case studies were available. I will discuss my path of discovery and attempt to explain what aspects are different from the original design and explain how and why those changes came to be. I will discuss the hardware used, coding implemented, and methodology practiced in order to create this platform so as to create a "cook book" of sorts.

The original motivation for this project was to create a green it platform for virtualization that utilized cutting edge technologies that would be scalable, reliable, fault tolerant, and easily managed that could also be used as an educational resource. The goal was to keep cost low and reduce the power consumption levels that a traditional architecture would incur. The storage area network (SAN) would be built using surplus equipment that was donated and destined for the landfill and as for the physical servers used as host for the virtual machines; I would use piece-milled parts and assemble those parts by hand.

First let's discuss the boot sequence. The original plan for the boot sequence was overly complex. This complexity was seen as an issue from conception, but other issues came to light later in the project as well. One of such issues was that this method involved the physical infrastructure being dependant on the virtual infrastructure. Meaning that first the SAN would have to boot, Special Host would then boot using custom gPXE script embedded on a USB, then a DHCP enabled VM that resides on Special Host would have to boot using gPXE scripting on a

vUSB. Finally the remaining physical hosts would then look to the DHCP VM for IP configuration. This was not a best practice and it was cumbersome to implement.

The solution was to have all the physical hosts boot to USB, SAN nodes included. This would simplify the boot sequence and eliminate the need for custom gPXE. Another great advantage is that the USB is seen as the c: drive on any drive-less physical host and changes that are made, for instance, enabling role and services, are saved on the USB. So in the event of a failure of one of these physical hosts, all you have to do is remove the USB, rack mount another host, insert the USB into a port (ideally a port embedded directly into the motherboard inside the form factor) and power on the host. That host will now see that USB as its c: drive and will have the same exact configuration as the previous. No reliance at all on the physical machine. Any virtual machines and iSCSI configurations you create on that OS are saved as well. There is an inherent limitation of any solid state drive and that is the number of overwrites it can take until it is no longer usable. This is addressed during the process of creating the USB by disabling and removing the paging files. This reduces the overwrites to the USB to only changes made to the OS like enable/disabling roles/services. These changes are minimal. The OS only facilitates the physical infrastructure so once you have it configured correctly; you do not make many changes. Booting to USB was a much more simple and elegant implementation while still adhering to the original goals and mission of the project. A more detailed description of installing Server 2008 to USB is covered in chapter 6.

Storage Area Network

The SAN is the heart of the infrastructure and where the majority of work and time were spent. Without a working SAN none of the remaining infrastructure could operate to achieve the goals of the project. I have a two-node SAN with two 1TB disk in each allowing for 4TB of raw storage. A “node” is the physical server. The communication protocol used to create the connection between the 2 nodes would be Internet Small Computer System Interface or iSCSI; more on iSCSI later. Before I could configure iSCSI I would need to setup the 2 nodes for the SAN.

I installed EON Open Solaris 10 with the Comstar package that includes the ZFS file system on both nodes. This was installed to USB. Installing to USB accomplished two things: 1) no dependency on the hardware and 2) now I could use the entire capacity of the internal hard drives for storage without wasting any space for the OS. Now I would need to configure the network interfaces on both nodes. As discussed earlier in the document, there would be three network domains: one for SAN traffic, one for hosts-to-SAN traffic and one for remote access to other networks. This would seem like a pretty straight forward network configuration, and it was, except for an issue of persistence had that caused some setbacks. More on how I overcame the persistence issues will be discussed in chapter 6.

SAN to SAN iSCSI

The original plan was to create a mirror on Node2 that would directly reflect the data on Node1. I found that the mirror would not achieve the goals of the project so a different method was utilized. I used ZFS to create a RAIDz2 zpool named VMStore using all four available disks. The ZFS commands were all issued from Node1. This is possible because of the iSCSI connection between the two nodes. Due to the RAIDz2 configuration and other OS overhead, I now had

1.8TB of useable storage capacity. This allows for the failure of at least two disks with the ability to recover all data. Not the way I had originally planned, but this provides the features and benefits intended. This also allows for easy scalability. All you have to do is add more storage devices and ZFS allows you to add that new storage with a simple command. Redundancy, failover, scalability, and ease of management; all of the goals are now met with this configuration. There were a few iterations of different iSCSI designs along with some obstacles that had to be overcome that are discussed in further detail in chapter 6.

SAN to Host iSCSI

I want make the entire 1.8TB zpool available to the host devices. A zpool is ZFS's answer to partitions and volumes. The zpool creates a block of raw storage that can grow and shrink as needed. You can then slice that raw storage in any way necessary by creating datasets of any size. For my purposes I create a ZFS raw dataset that is 1.8TB that is associated with the RAIDz2 zpool. I name the dataset VMStore. Once the dataset is created you can associate logical units (LU) of any size that point back to that dataset and ultimately to the zpool. In this case, I create a LU that utilizes the entire 1.8TB of storage in the dataset. I create the targets, target group, target portal group, and host group configuration. On the physical hosts, I open the iSCSI initiator administrator to get the necessary information needed. I will need the iqn numbers from each physical host. I add those iqn's to the host group on Node1. I return to the physical host iSCSI initiator administrator and setup the connections by entering the IP address of the target group, Node1, and click quick connect. The connection is successful. All three physical host now have 1.8TB available that can be managed via the disk management administrator. I enter into the disk management administrator, initialize a new simple volume, ensure it has a

master boot record (MBR) and format with the NTFS file system. There are 1800GB available so I create a partition with 600GB and click finish. This partition could have been any size up to the 1800GB available. I repeat this process for the second and third hosts. Now the physical hosts see a drive named VMStore with 600GB available and 1200GB unallocated.

Results: Performance

This project yielded respectable results. The goals were to keep cost low and reduce power consumption while not sacrificing on performance, reliability, and scalability. I was able to keep cost low by using donated repurposed machines that were destined for the landfill that cost nothing. Approximately $3000 USD was spent on three drive-less, hand-built, piece-milled servers that are being used for the physical host machines. The costs are nominal when you look at the performance that we are seeing. I created a RAIDz2 zpool using the ZFS file system on four Western Digital WD1001FALS 1TB hard drives spinning at 7200 RPM. Using the I/Ops software application iometer to read and write to and from the disk I was able to get some impressive numbers. See figure 1. In the best run, I saw over 2500 total IO/ps (per second) with over 1700ps reads and over 800ps writes. Not bad for old equipment pulled out of the trash. The average was approximately 1875 total IO/ps over four tests. In figure 2 I included some additional results provided by a university professor gathered during experimentation in a storage administration course. This serves as a good benchmark as all test, including my own, were performed with the same IO software with the same parameters.

Figure 1

'Target Type	Target Name	Access Specification Name	# Managers	# Workers	# Disks	IOps	Read IOps	Write IOps
ALL	All	Default	1	1	1	2578.35	1733.49	844.86
MANAGER	SPECIAL-HOST	Default		1	1	2578.35	1733.49	844.86
PROCESSOR	CPU 0							
PROCESSOR	CPU 1							
PROCESSOR	CPU 2							
PROCESSOR	CPU 3							
PROCESSOR	CPU 4							
PROCESSOR	CPU 5							
PROCESSOR	CPU 6							
PROCESSOR	CPU 7							
WORKER	Worker 1	Default			1	2578.346	1733.488	844.85862
DISK	F: "VMStore"					2578.346	1733.488	844.85862

Figure 2

Device	Capacity	Type	FS	Total Iops
SanDisk Cruzer Fit USB 2.0 Flash Drive	32GB	USB 2.0	FAT32	9979.93
Striped boot drive on office desktop machine	560 GB	SATA 2.0	NTFS	3341.32
IOMega External Hard Drive	160GB	USB 2.0	NTFS	2966.10
HP 8GB Flash drive	8GB	USB 2.0	FAT32	858.11
LEEF Supra 32GB USB 3.0 flash drive	32GB	USB 2.0	FAT 32	646.78
lenovo laptop	320GB	USB 3.0	NTFS	630.55
SD Card				453.34
HP 20				408.86
Sandisk Cruzer	32GB	SATA 3	FAT32	326.29
PNY Flash	128GB	SD 2	FAT32	324.64
Toshiba Q Series SSD	128GB	SATA 2	NTFS	215.97
Toshiba Canvio HDD external	500GB	USB 2	FAT32	204.00

Results: Power Consumption

To measure power consumption I used an appliance load tester that measures wattage used by the device plugged into it. For my test subject I used one of my host servers. I tested the server without the drive and saw an average of 60 watts being used when accessing the SAN. I then installed a Seagate Barracuda 7200 RPM drive on that same server. I was able to consistently see a steady 5 watt increase in power (65 watts total) while the hard drive was spinning. So this 5 watt increase is per drive. If we assume that the platform will be consistently accessed 24 hours a day, 7 days a week, then we can reasonably assume that one drive would add to the power consumption at a rate of 120 watts per day. In its current state my platform has three drive-less servers. So the power savings is as follows: 3 drives x 120 watts = 360 watts per day x 7 days = 2520 watts per week x 52 weeks = 131,040 watts per year. To convert this to KWh: 131,040 / 1000 = 131.04. Our current per KWh rate is $0.12. The results are as follows: 131.04KWh * $0.12 = $15.72. Keep in mind that this is only for three drives so while this may not seem that significant of savings, the benefits would be exponential in large organizations and data centers.

The most substantial savings is when you consider the amount of physical machines that can be eliminated by virtualizing these machines. In this platform each generic host has 16GB of RAM. RAM is a limiting factor when determining how many VMs can be run on a particular host. If we intend to allocate 2GB of RAM to each VM, then we can safely run 12 VMs on each host. This is because we can over-subscribe the resources that are available on the host. The reasoning behind this is that under most circumstances a machine does not fully utilize the amount of RAM that it has available. With VMs you can estimate your over-subscription rate to

roughly 75%. You can monitor this resource and if necessary either add more RAM to the host, reduce the amount of VMs on the host, or add another host and migrate the VMs. If we go with the 75% over-subscription rate, then we can reduce 12 physical machines into one. During testing I saw that a machine running, while one drive was being accessed pulled 65 watts of electricity consistently. Five of those watts are used to spin the hard drive as mentioned in the previous paragraph. The potential savings achieved by this reduction in the total amount of machines is calculated below assuming the machines run 24 hours per day, 365 days per year.

12 machines * 65 watts per machine = 780 watts * 24 hours per day = 18,720 watts * 365 days per year = 6,832,800 watts per year. Electricity is sold per Kilowatt hour (KWh). To determine total KWh, divide the total watts per year by 1000. 6,832,800 watts / 1000 = 6,833 KWh per year (numbers have been rounded up). The current rate per KWh in this part of the country is approximately $0.12. So the total savings achieved by reducing the total number of machines by 12 is as follows: 6,833KWh * $0.12 = $820 per year. If we take into account that there are three generic host available, the total potential savings achieved by utilizing all three generic host to full capacity is $820 *3 = $2,460 per year. Also note that this is not taking into account the reduced cooling needs of which I do not have any numbers, but would only add to the affect.

Results: Scalability

The platform is highly scalable. You can easily increase performance capacity by adding RAM to the physical hosts. RAM is the limiting factor of how many VMs a host can handle. Install more RAM and the host can take on more VMs. You can easily add storage capacity as well. This is one of the major benefits of using the ZFS file system. You can go about this in a

many different ways; you could add another physical server with more drives and add those drives to the RAIDz2 zpool. You could add a NAS type device and have ZFS see it as raw block storage that could be added to the zpool. Other zpools can be created as well to better allocate dedicated or isolated storage if needed. Basically, as long as it is raw block storage ZFS will treat it the same way as it does any other while managing load balancing, multi-pathing, and performance caching.

Results: Reliability/failover

Storage reliability comes into play with the RAIDz2 zpool. The SAN could lose up to two disks at any time and still maintain data integrity. The SAN also boots from USB, so if one of the physical machines were to experience an equipment failure, like a motherboard failure, all you would have to do is remove the hard drives and the USB drive, find a suitable replacement machine and install the drives. The SAN would pick up right where it left off. The physical host machines that provide the processing and RAM for the VMs boot to USB as well. So it is a similar scenario to the SAN. This platform was built to not be dependent on the hardware. As machines fail we can replace with another suitable machine and continue. There is a single point of failure that needs to be addressed; the USB drives. The USB drives should be copied upon creation and when changes are made. Another benefit of this design is that changes are rarely made to the USBs. Once the platform is up and running, the data on the USBs remain the same until storage is added or modified.

Chapter 6: Retrospective / Conclusion

In this chapter I will examine the project retrospectively. I will discuss some of my opinions and address some of the differences regarding the original architecture. I will also discuss some of the issues that arose throughout the project that were not highly technical, but were “show stoppers” so to speak. If these issues were not resolved, then the platform could not be completed. This is important because one could use the exact same equipment with the same operating systems following the same methodology and coding line for line and still not be able to achieve the build. Every build is different so I cannot hope to cover all that may occur, but will describe my experience. The following paragraphs describe the issues, the troubleshooting tactics, and the resolution of said issues.

Show Stoppers – Boot to USB

Upon attempting to install Server 2008 R2 to a USB, I found that Microsoft does not allow this. That is still the case today for a traditional install. However there is a loop hole. Microsoft will allow this for original equipment manufacturers (OEM), but will not support the operating system if this method is used. Microsoft even provides instruction on how to create a bootable USB instance of Server 2008 R2, but adamantly states no support for this method in there documentation. In essence you have to create a virtual disk (vdisk) on your local computer. Then attach or mount the drive. Using Windows AIK tools you copy the install.wim file to create a VHD; a very small foot print of about 12GB. You then have to “trick” the BIOS into thinking that the USB is not removable media by editing the registry settings while the USB is inserted into a port on the computer. You then attach the VHD to the USB. You now have a bootable instance of Server 2008 R2 on a USB device. This information was not available at the

time of conception of this project, nor was it considered, but it works well and solves multiple problems. A USB device can now be used to boot the physical host servers.

On the SAN nodes I was also booting to USB. I used an update image command (updimg.sh) to apply changed configuration settings to the USB. This is fairly common. I executed that command and for some reason the network settings would still not persist. I began to troubleshoot and research thinking this would be an easy fix. That assumption was highly inaccurate. I tried a couple of different methods to fix this issue. I tried manually configuring static IP addresses; I tried using the internal setup that should pre-configure on each reboot. Neither of these methods would persist after reboot. All of the information I found online only directed me back to the updimg.sh command that was currently not working. The next method I attempted was to write a script that would force the pre-configuration settings to be applied on each boot. The problem was I did not know in which file to place this script. After teaming through page after page on the internet and also consulting with a university Linux administrator I finally found It. The file resides in /etc/rc3.d/S99local. I wrote the configuration script in this file, ran the updimg.sh command, rebooted the system… and still the changes did not persist. The script I wrote was still saved in the S99local file, but the network interface would not persist. It was seemingly “small” things like this that caused major delays in the project. By this time I had spent close to 25 hours troubleshooting this problem and it still remained. I continued to research. I dug deeper into what the updimg.sh command really does. After tracing the code, I realized that updimg.sh uses a file called.backup to identify which files to save to the USB that would result in that file being persisted after reboot. I searched for the file for quite a while only realize it was a hidden file that required the –al option added to the list command (ls). I had found the file and the solution. I added the path to

S99local to the .backup file and now the updimg.sh command worked; the network configuration settings were now persisting after reboot. Although this issue caused much frustration and delay, it did teach me a few things about the Linux platform that I will not forget anytime soon. Linux is not very forgiving to someone with little experience working with it. Also, this lesson would serve me well later in the project.

Show Stoppers - iSCSI

iSCSI can be confusing in the way that all the pieces and layers have to fit together for it work properly. See the definitions section of this document and appendix 5a, 6a, 7a, and 8a for reference. Even the definitions themselves are a bit convoluted. Once you have experience working with iSCSI for a while it makes sense, but it has to be hands-on experience. In my opinion you can't get a real understanding of iSCSI from reading the definitions alone; only a conceptual overview. At its simplest level iSCSI is made up of targets and initiators. In essence, targets are the devices where you want the data to be transferred and initiators are the devices that send that data. In the example of the two node SAN, I was originally attempting to create a mirror of Node1 on Node2. Node1 (initiator) is sending its data and Node2 (target) will receive it. I attempted a few of different configurations to accomplish this design. In the first configuration I created a logical unit (LU) for each of the raw disk in Node2. The LU was directly associated with the disk devices themselves as opposed to a raw dataset. After configuring an initiator on Node1 using the itadm create-initiator command, I moved to Node2 to begin the iSCSI configuration set up. Once complete I was not able to get Node1 to see Node2's disks. Node1 could see the targets that I had created on Node2, but not the devices I needed to use as my mirror. I checked and re-checked the setting on both Nodes. After days of

troubleshooting and researching with no luck I began to destroy, build, destroy, and re-build the iSCSI configuration with no success. Once again I began to dig deeper into the iSCSI protocol. I found that there is already an initiator that is created by default when the system boots, the iSCSI initiator node. You can use the iscsiadm list initiator-node command to see this initiator. I went back to Node2 and removed the initiator that I had created and used the iSCSI initiator node in its place. That worked, now Node1 could see Node2's disks. Problem solved, right? Not quite…it's never that easy. Upon reboot the iSCSI initiator node changes its name so the settings in Node2 no longer match the initiator node on Node1. I would pull from past experience to resolve this issue. In the file S99local, I would write in the command to modify the iSCSI initiator-node to match the settings in Node2. This, in theory, would force the iSCSI initiator node in Node1 to be the same no matter what, after each reboot. It worked and Node1 could now see Node2's disks and the configuration would persist upon reboot. To come to this resolution would take almost two weeks.

The next obstacle in the way would be actually using Node2's disks. From Node1 I was not able to issue any commands on Node2's disk. I kept getting errors stating the device did not exist in /dev/dsk. After checking in that directory, I confirmed that the devices were indeed not in that directory. As long as this was the case I would not be able to create the mirror between Node1 and Node2. From Node1, when issuing the format command, to see the available disks, the disks names for the Node2 devices was a long string that included part of the GUID of the LU from the iSCSI configuration on Node2. What I expected to see was something like, c2d0 or c3d0. I thought this may have something to do with why the devices where not being populated in the /dev/dsk directory. My solution was to try a different methodology. I would destroy the current iSCSI configuration on Node2 and utilize ZFS zpools to accomplish the mirroring.

After destroying the iSCSI configuration on Node2 I created a zpool that included both of Node2's disks. I then created a zfs raw dataset that was associated with that zpool and utilized the entire capacity of storage. I then created a LU associated with that dataset and tied that LU back to the view so that Node1 could communicate with it. This part worked. From Node1 I could now see only one device. This was because I had in essence combined those two disk from Node2 with the zpool configuration. Now I would create a similar zpool combining both of Node1's disks and attempt to mirror the Node1 zpool to the Node2 zpool. This did not work. This method of mirroring zpools is not supported with ZFS; at least as far as I could tell. In this case I had to learn the hard way, by building and experimentation. Not a total lose though, I just learned one more way to *not* create a mirrored SAN; back to the drawing board.

Now I would return to my previous iSCSI configuration by destroying and rebuilding. Now Node2 has two LU's. One for each of its disks and both are visible via iSCSI to Node1. I am still running into the issue. Node2's disks have those long string names and are not being populated in the /dev/dsk directory. I still cannot issue any commands to manipulate those disk. After more research I find that these long string names are associated with the multi-pathing feature. Multi-pathing is used for performance and has great redundancy and failover capabilities. You can use multi-pathing between multiple NICs to load balance and as failover if one NIC experiences a failure. For the purposes of this project multi-pathing is not a necessary feature so in an effort to resolve my issue, I disable multi-pathing, but this will require a reboot. And once again, the changes made to multi-pathing do not persist after reboot so the change has had no effect. Once again I go back to that hard lesson learned at the beginning of this project and add the path directory and file to the .backup file that will now make the updimg.sh command save the changes made to that file persist after reboot. It worked. Now Node1 can

now see Node2's disks and that long string name has now been reduced to something much more familiar. I check the /dev/dsk directory and those devices are now in that directory. I issue a couple of test commands on the disks and I can now manipulate Node2's disks from Node1. A successful iSCSI connection has been made. Now I will attempt to create the mirror between the two SAN devices.

I now have a SAN that has four disks available with a storage capacity of a little less than 4TB. In creating a mirror I will lose half of my capacity as available active storage as the other half will be for the mirror. So in total, I should have roughly 2TB available for active storage, but that will be reduced by overhead so I expect to end up with approximately 1.8TB or so. The mirror will allow the SAN to lose two of the four disks and still be able to recover all data. I use the zpool mirror commands to attempt the mirroring. I try several different configurations of such and I find that I cannot use the zpool mirror commands to accomplish a full mirror from the two disks on Node1 to the two disks on Node2. The problem is that when you create the zpool mirror, it creates the mirror for only one of the disks. I could have a one-to-one mirror, where Node1 disk1 mirrors directly to Node2 disk1 and set up a similar mirror on the other 2 disks, but that is not what I am trying to accomplish. I want the two disks on Node1 to be mirrored as a single unit to the two disks on Node2. I have already tried mirroring them as zpools and know that does not work. I fall back to an alternative that accomplishes the goals, but uses different methodology.

I return the iSCSI configuration to its original configuration. I created the iSCSI connection so that Node1 can see and manipulate Node2's disk along with its own. I achieve this by creating a logical unit (LU) for each of Node2's disks. I setup the proper iSCSI target and

host configuration and apply them to a view. From Node1, I issue the send targets command and I can now manipulate Node2's disk from Node 1. See appendix 1d for code "cook book". I create a RAIDz2 zpool as mentioned earlier in the document. This does not create the mirror as originally intended but does offer the same failover, redundancy, and scalability with approximately the same amount of storage available.

Future Work

This project did achieve the goals set. There is always room for improvement however. I would like to see how the platform handles the stresses of an actual production environment. I believe that there is also some fine tuning that could be accomplished in the storage configuration. For example, fine tuning the block sector sizes based on use case. Smaller block sectors for smaller files and applications and larger sector sizes for streaming media files and graphic intensive applications. Security could also be improved upon. The only security aspect applied is the isolated subnets on the network. This is a good start however other aspects were neglected, such as CHAP authentication between targets and initiators and encryption for files traversing public networks. Network intrusion detection and prevention techniques could also be applied. This platform provides a blank canvas of sorts for faculty and students to build upon while also providing a fully functional example of SAN based technologies and techniques.

Conclusion

The result of this project is a working green IT platform that is capable of supporting a production virtual environment utilizing a 4TB RAIDz2 SAN created with EON Open Solaris, the ZFS file system, and the iSCSI protocol. All from hand built drive-less piece-milled equipment and old antiquated legacy equipment that was destined for the landfill. The goals for this

project – performance, scalability, and ease of management -- were met and in some cases the results exceeded expectation. This project has been a successful implementation of what was only an idea in the beginning.

References

1) Guillet, Jeff. Morimoto, Rand. "Window Server 2008 HyperV Unleashed" SAMS Publishing September 13, 2008.

2) http://en.wikipedia.org/wiki/Virtualization

3) Rouse, Margaret. "Virtualization" Tech Target. Published: 22 Dec 2010. http://searchservervirtualization.techtarget.com/definition/virtualization

4)Webster Dictionary. http://www.webster-dictionary.org/definition/virtual

5) http://en.wiktionary.org/wiki/Wiktionary:Main_Page

6) CBS Interactive http://www.techrepublic.com/blog/10things/mini-glossary-10-virtualization-terms-you-should-know/440

7) Armstrong, Ben. "Hyper-V Terminology: Microsoft's Virtualization Lexicon ". SYS-CON Media, Inc. JUNE 10, 2008. http://soa.sys-con.com/node/534928

8)Oracle Solaris. ZFS FAQ page http://hub.opensolaris.org/bin/view/Community+Group+zfs/faq

9) Wikipedia http://en.wikipedia.org/wiki/Storage_area_network

10) Rouse, Margaret. "Storage Area Network (SAN)." Tech Target. October 2007. http://searchstorage.techtarget.com/definition/storage-area-network-SAN

11) Singh, Amit. "An Introduction to Virtualization" January 2004. http://www.kernelthread.com/publications/virtualization/

12) Harrisdy. "History of Virtualization." Hinzie Media Inc. Oct 19, 2009 http://www.infobarrel.com/History_of_Virtualization

13) Moellenkamp, Joerg. "Less known Solaris Features: iSCSI with COMSTAR." NOV 22. 2009. http://www.c0t0d0s0.org/archives/6140-Less-known-Solaris-Features-iSCSI-with-COMSTAR.html

14) "EON ZFS Storage." http://sites.google.com/site/eonstorage

15) Rouse, Margaret. "iSCSI (Internet Small Computer System Interface)." Tech Target. May 2011. http://searchstorage.techtarget.com/definition/iSCSI

16) Microsoft. “Overview of Hyper-V.”
http://technet.microsoft.com/en-us/library/cc816638%28WS.10%29.aspx

17)Oracle Solaris. “Configuring iSCSI Devices with COMSTAR.”
http://docs.oracle.com/cd/E23824_01/html/821-1459/fnnop.html

18) Oracle Solaris. “Man Pages Section 1M: System Administration Commands.”
http://docs.oracle.com/cd/E23824_01/html/821-1462/index.html

19) “COMSTAR iSCSI howto.”
http://www.tek-blog.com/main/index.php?blog=2&title=comstar_howto&more=1&c=1&tb=1&pb=1

20) Wikipedia. “Hyper-V.”
Dec 7, 2009. http://en.wikipedia.org/wiki/Hyper-V

21) Tate, Jon. Beck, Pall. Ibarra, Hector Hugo. Kumaravel, Shanmuganathan. Miklas, Libor
IBM_Introduction To Storage Area Networks
Nov 2012. https://www.redbooks.ibm.com/redbooks/pdfs/sg245470.pdf

22) SNIA. “Definition of SAN.” SNIA
http://www.snia.org/education/dictionary/s#storage_area_network

23)Oracle Solaris. ZFS info page
http://hub.opensolaris.org/bin/view/Community+Group+zfs/whatis

24) Oracle Solaris. ZFS slides from Open Solaris
http://hub.opensolaris.org/bin/download/Community+Group+zfs/docs/zfslast.pdf

25) Microsoft. “Deploying Microsoft Hyper-V Server 2008 R2 on USB Flash Drive.”
http://technet.microsoft.com/en-us/library/ee731893(WS.10).aspx

26) FRANKOVIĆ, MARIN. “Install Window Server 2008 R2 or Windows 7 on USB Stick.”
June, 2010. http://blog.frankovic.net/2010/06/install-windows-server-2008-r2-or-windows-7-on-usb-stick/

27) Dunn, Eddie. University of North Carolina Wilmington, Systems Administrator

28) Oracle Solaris.
http://docs.oracle.com/cd/E23824_01/html/821-1462/stmfadm-1m.html

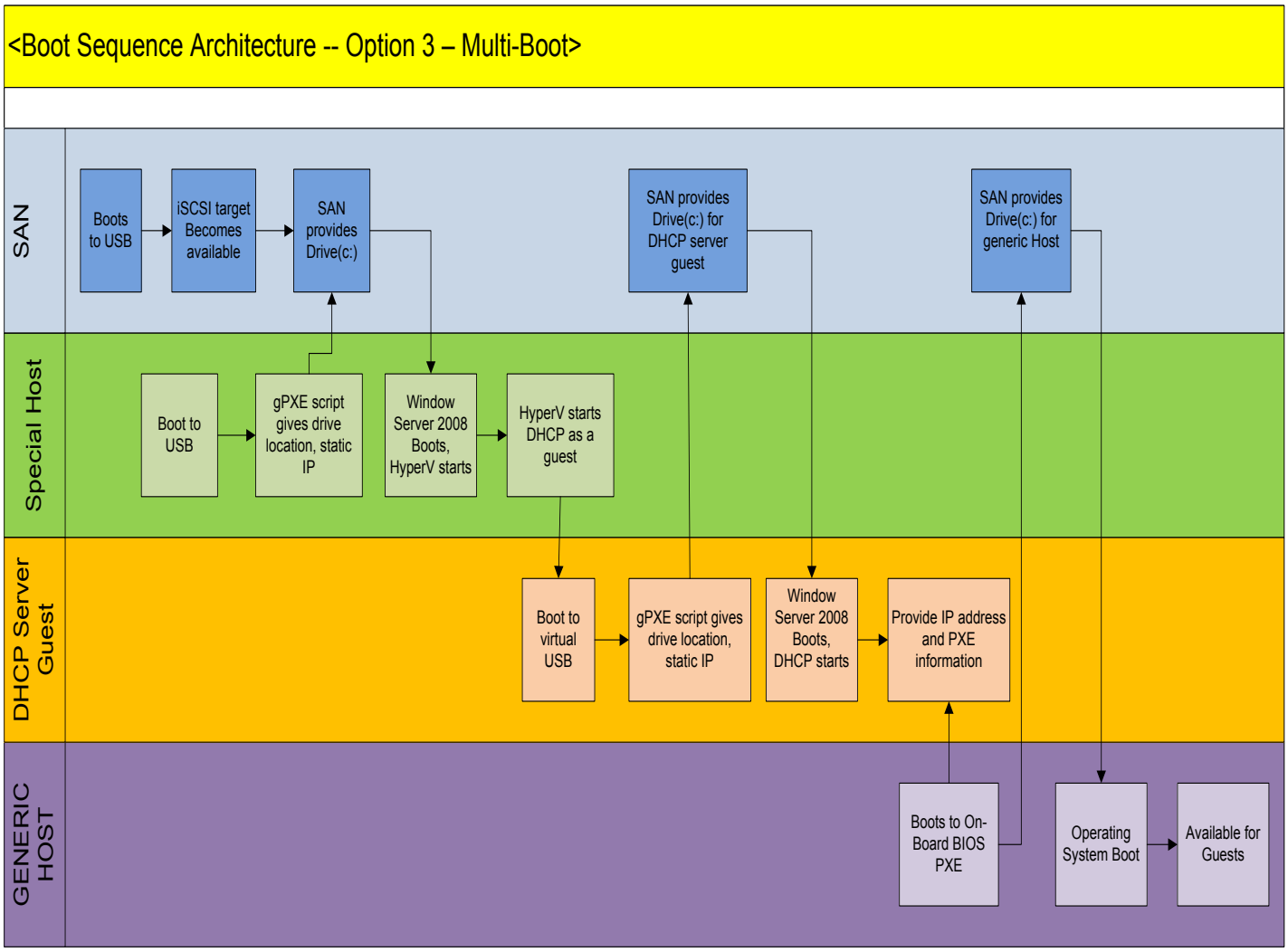
29) Duke Energy Carolinas.
http://www.considerthecarolinas.com/pdfs/NCScheduleSGS.pdf

30) Toponce, Aaron. “ZFS.”
https://pthree.org/category/zfs/

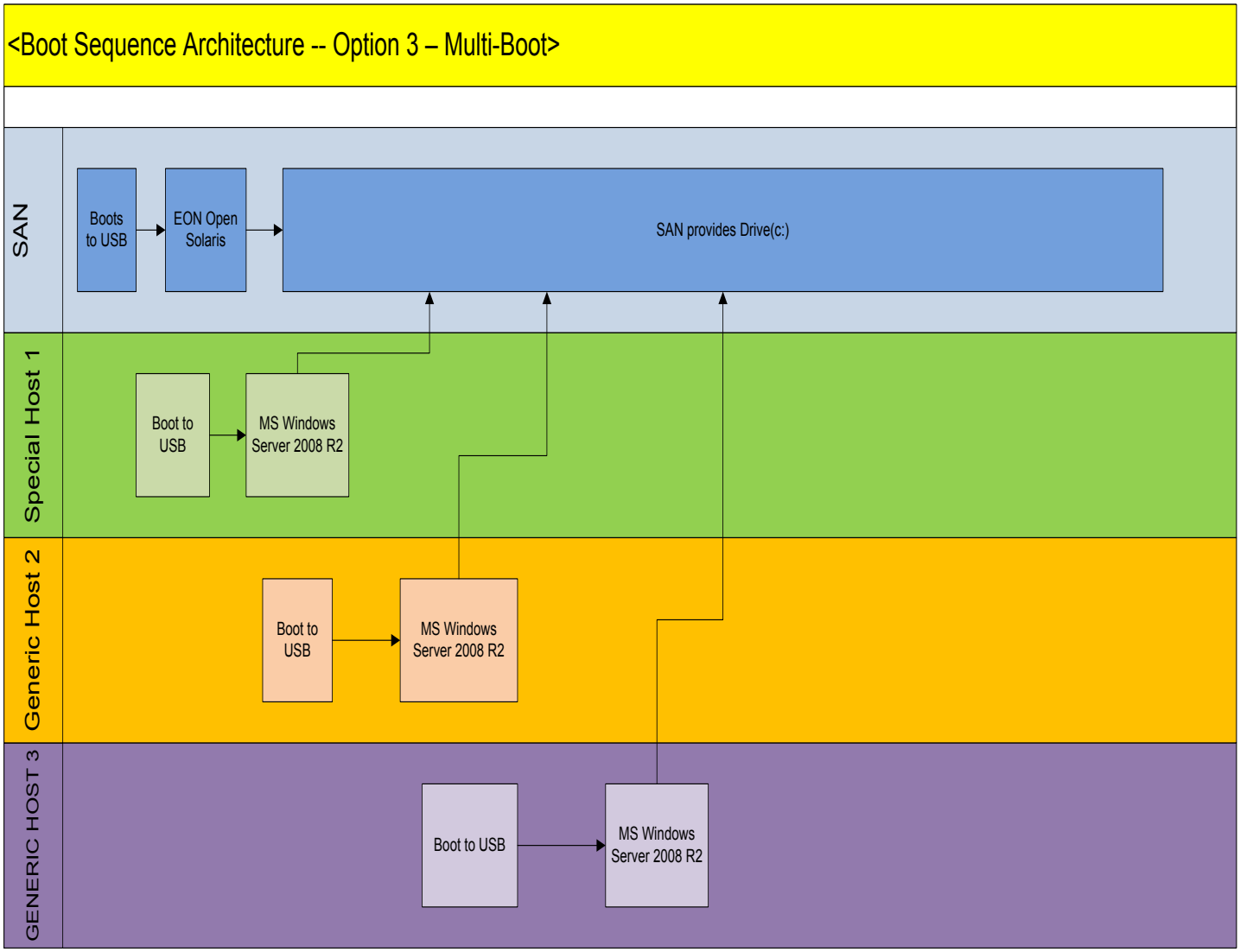
31) Toponce, Aaron. “ZFS Administration, Part XVII- Best Practices and Caveats.”
https://pthree.org/2013/01/03/zfs-administration-part-xvii-best-practices-and-caveats/

Appendix A

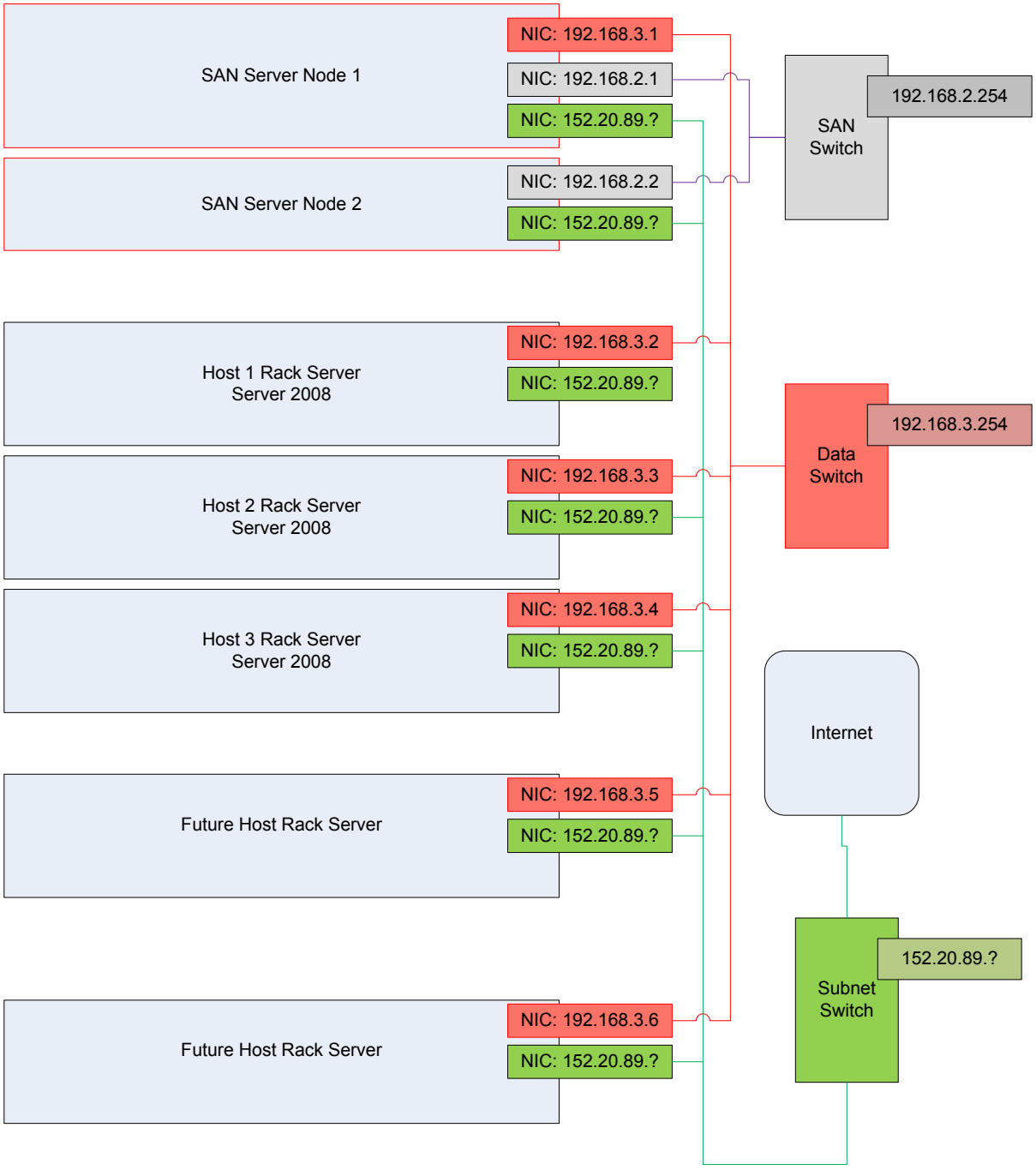
1a) Pre-implementation boot sequence architecture



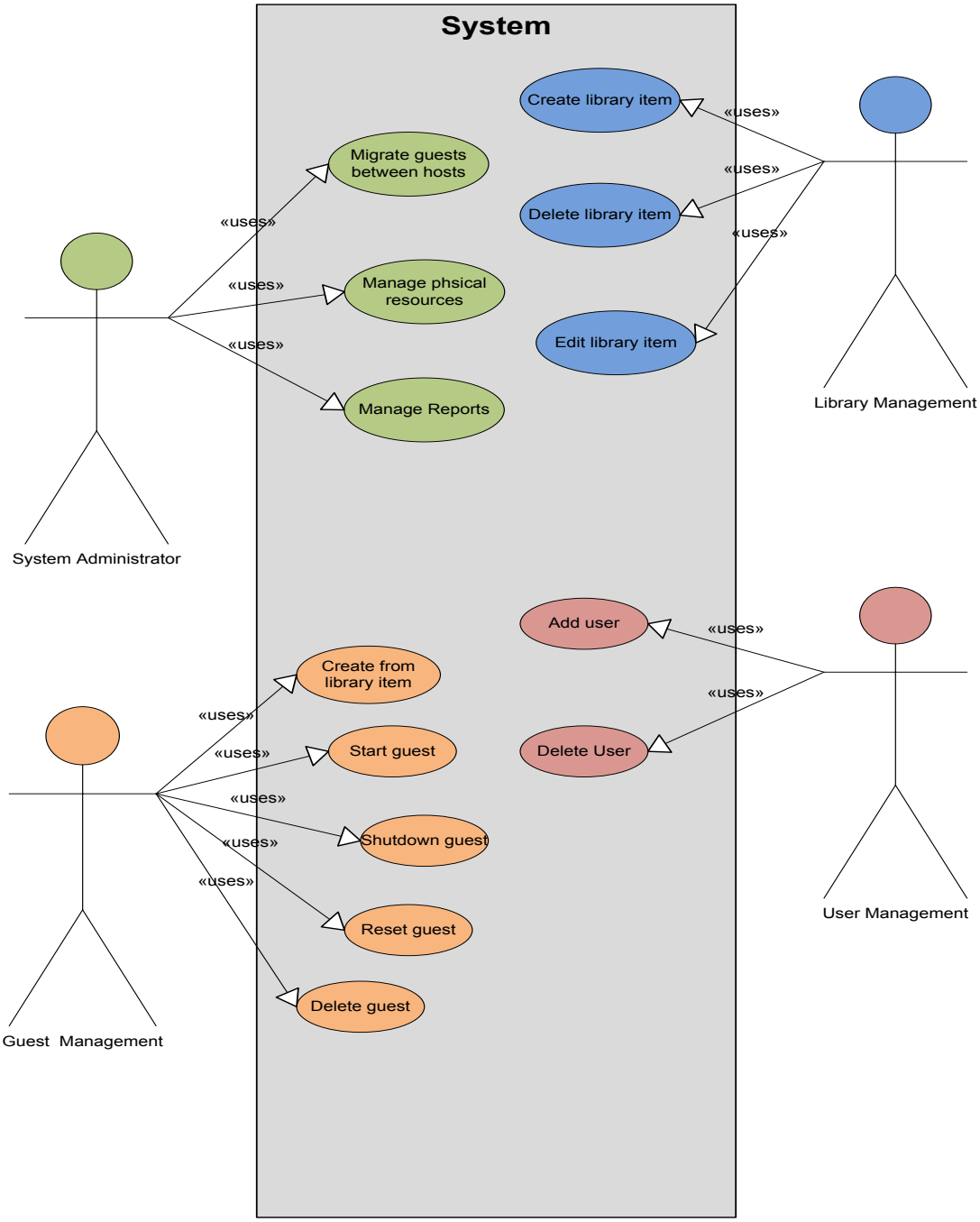
2a) Post-implementation boot architecture



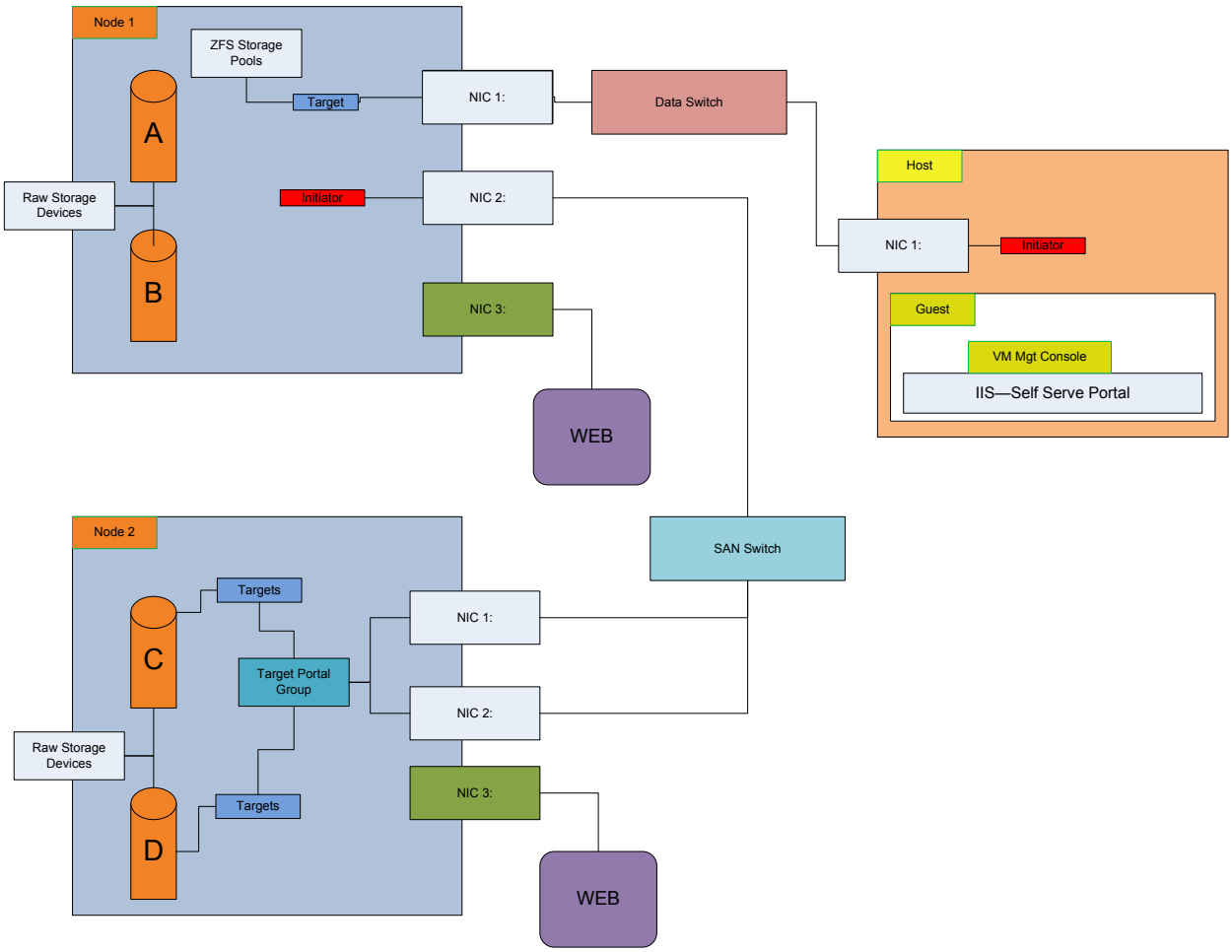
3a) Network diagram



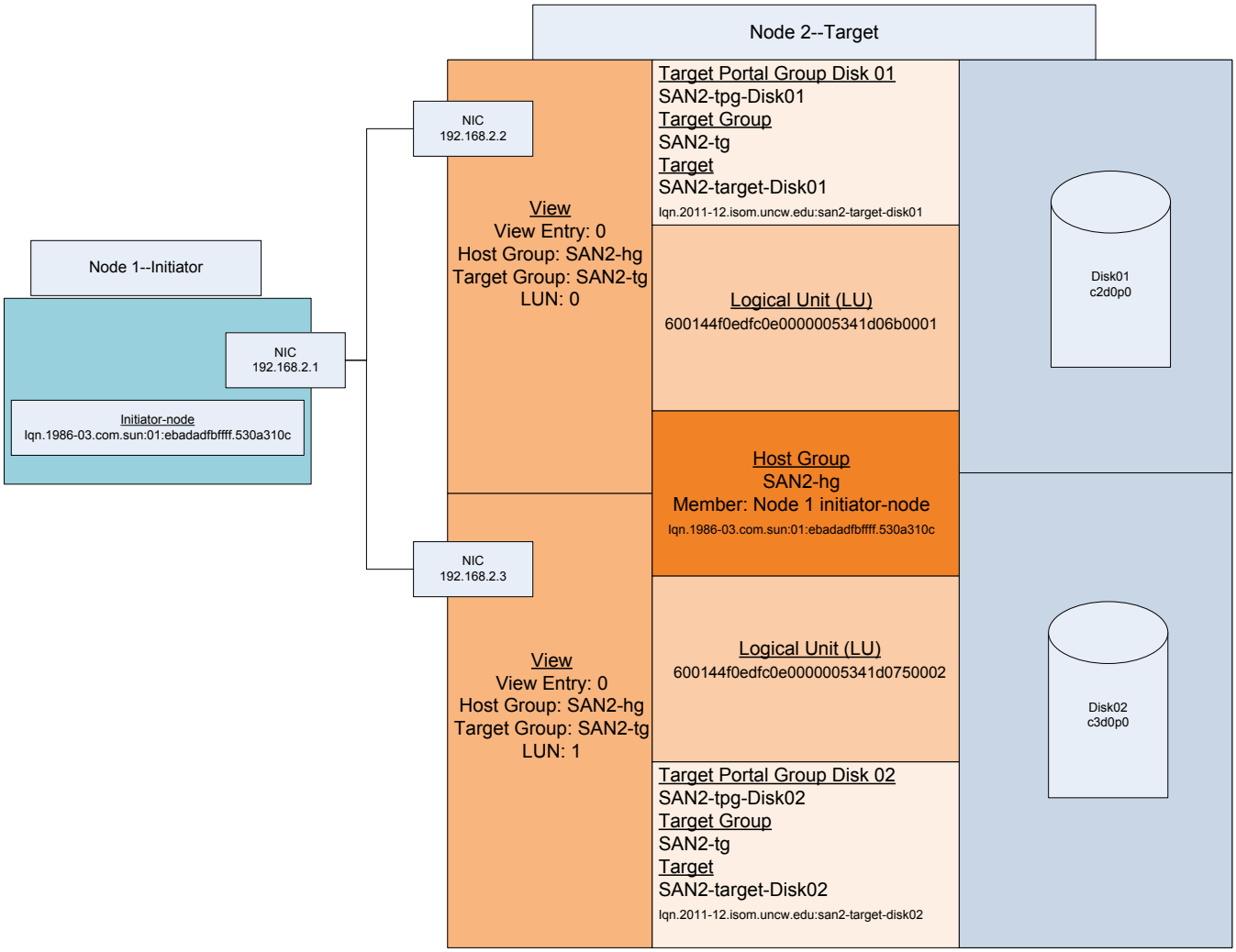
4a) Use case diagram



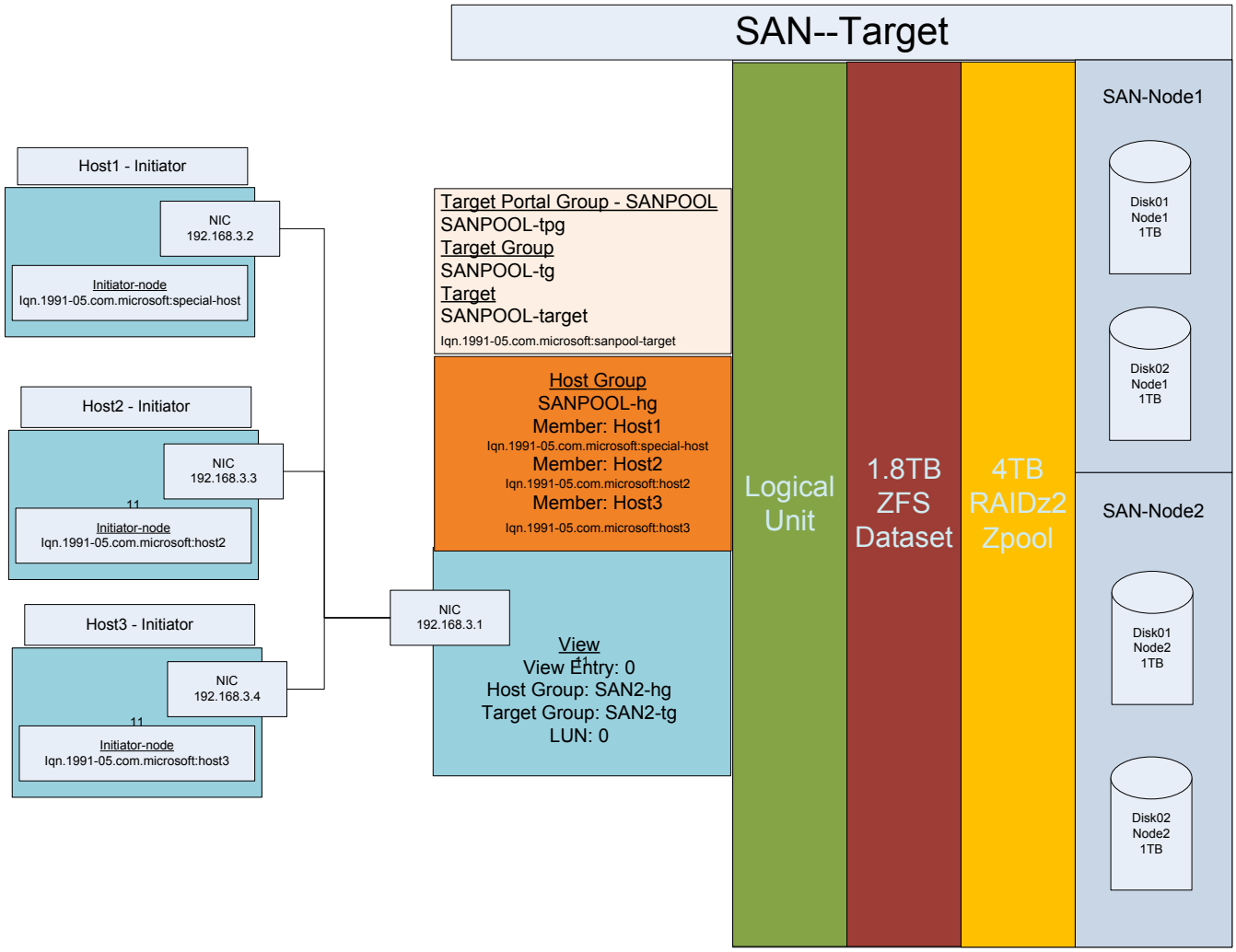
5a) Target/Initiator Diagram



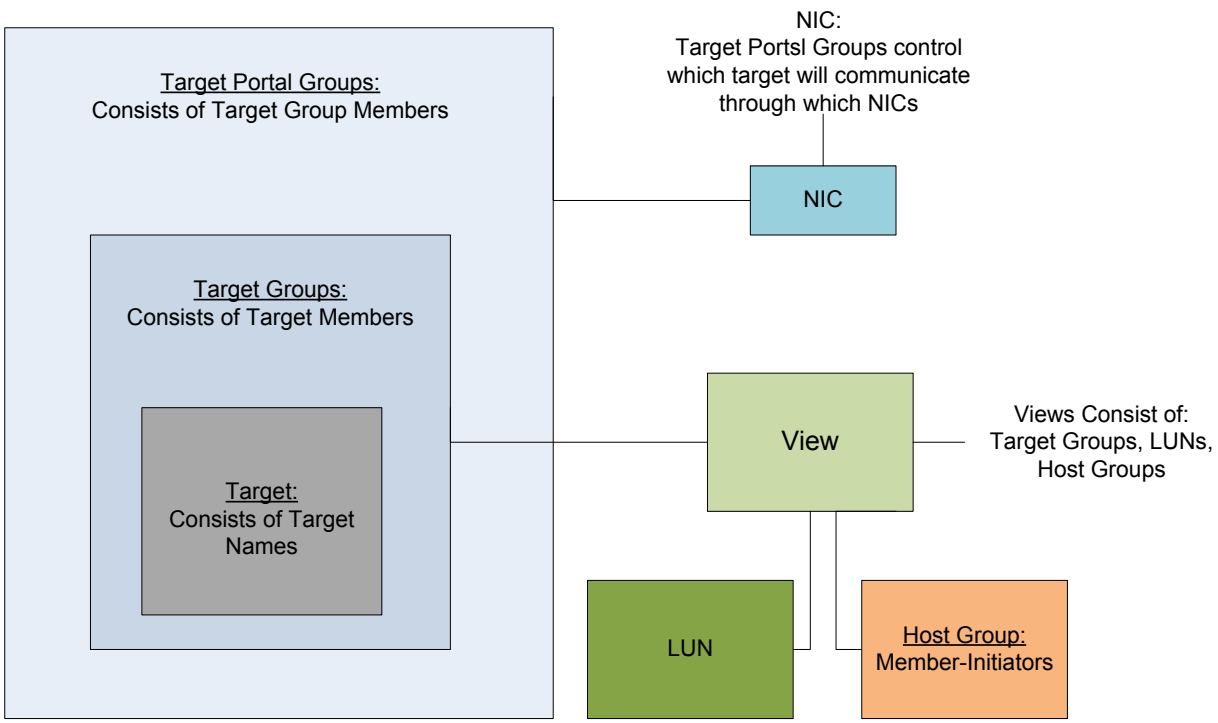
6a) Low Level View of SAN Node1-SAN Node2/Target-Initiator Relationship



7a) Low Level View of SAN Node1-Hosts / Target-Initiator Relationship

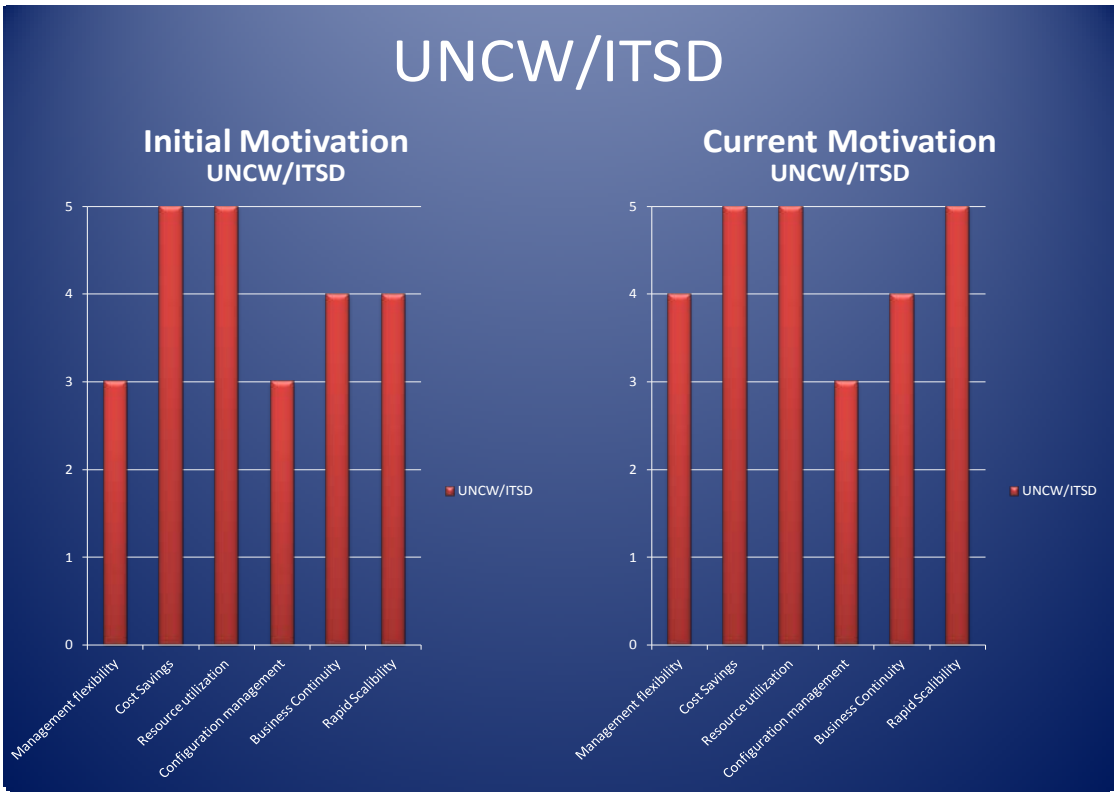


8a) Low level view target/initiator relationship

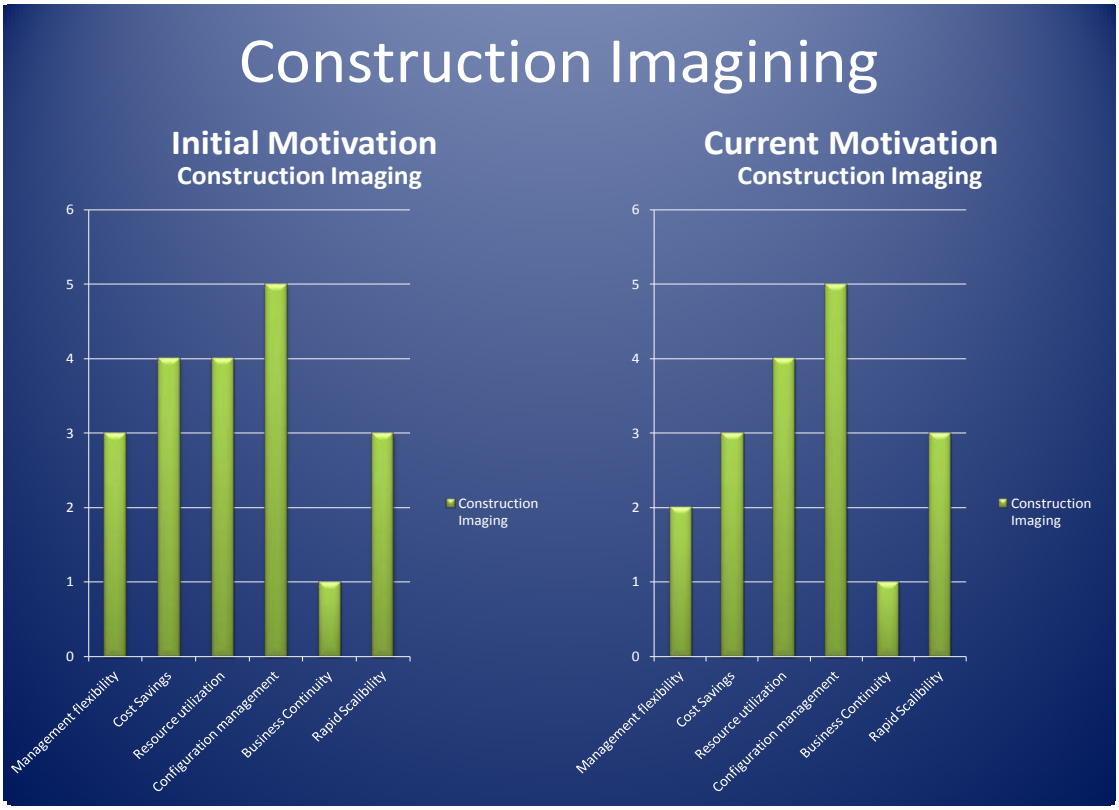


Appendix B

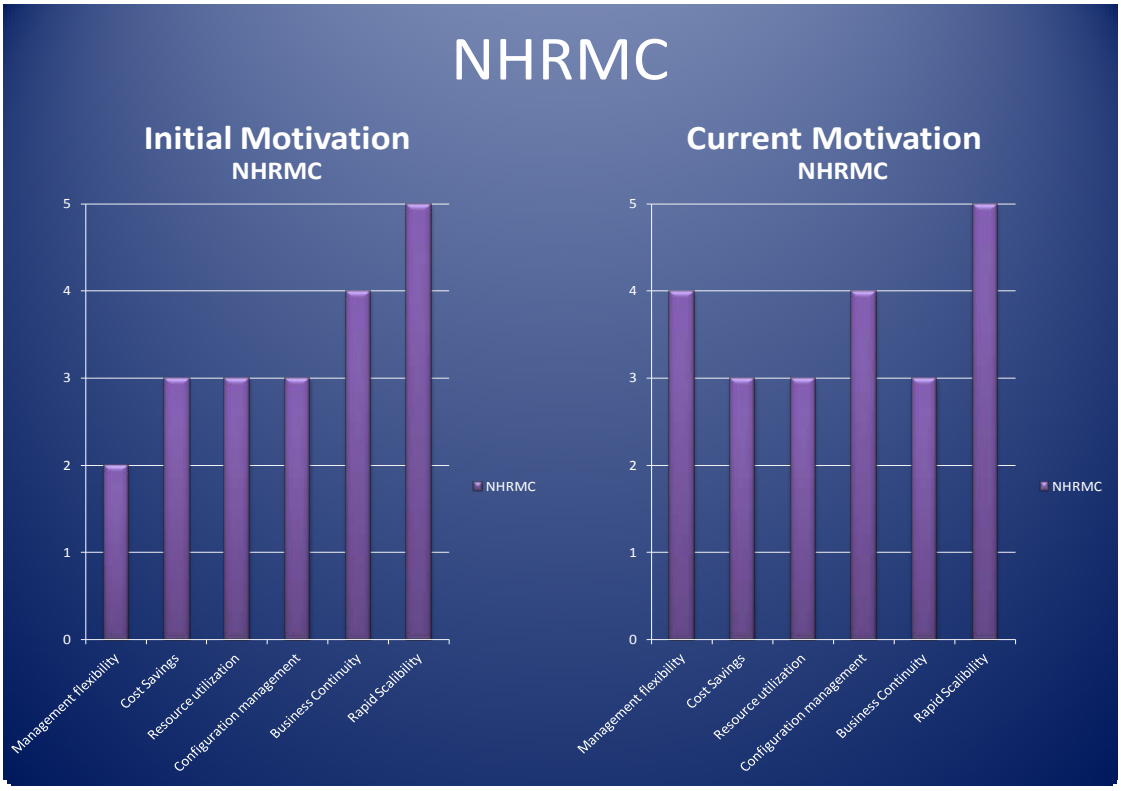
1b) Expert Interviews—UNCW / ITSD



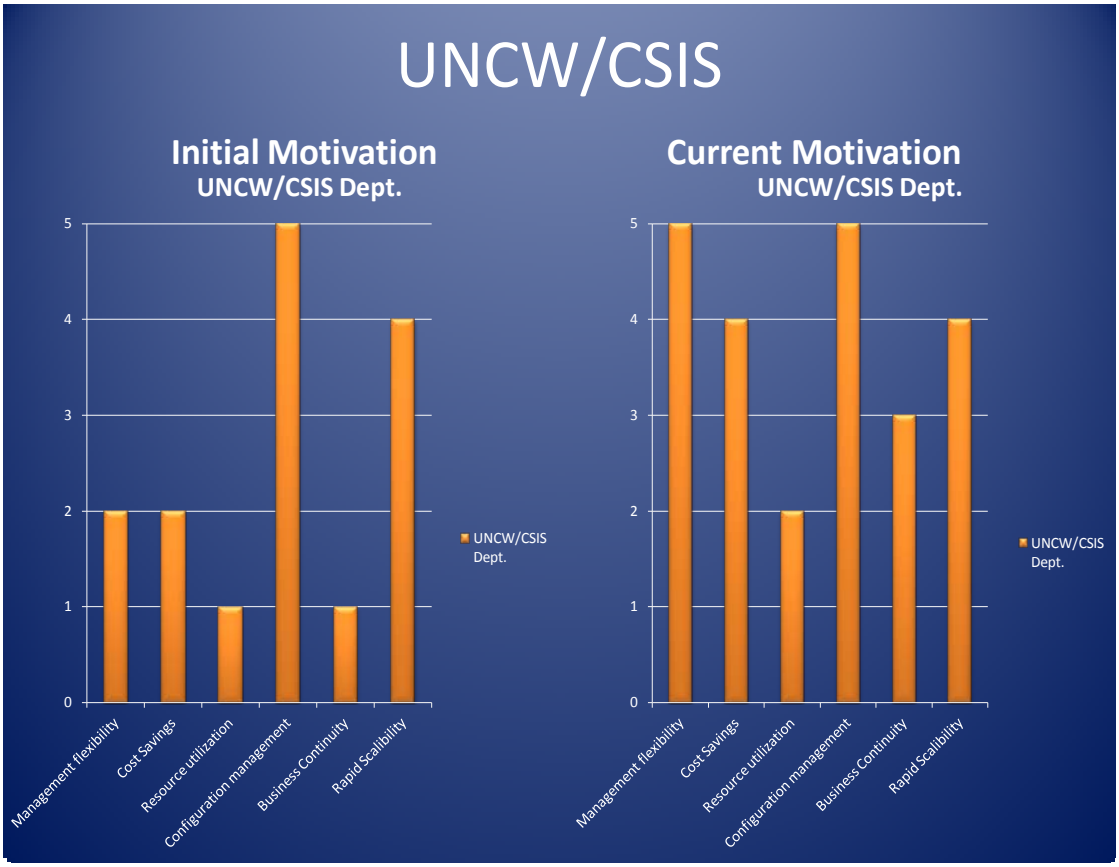
2b) *Expert Interviews – Construction Imagining*



3b) *Expert Interviews- NHRMC*

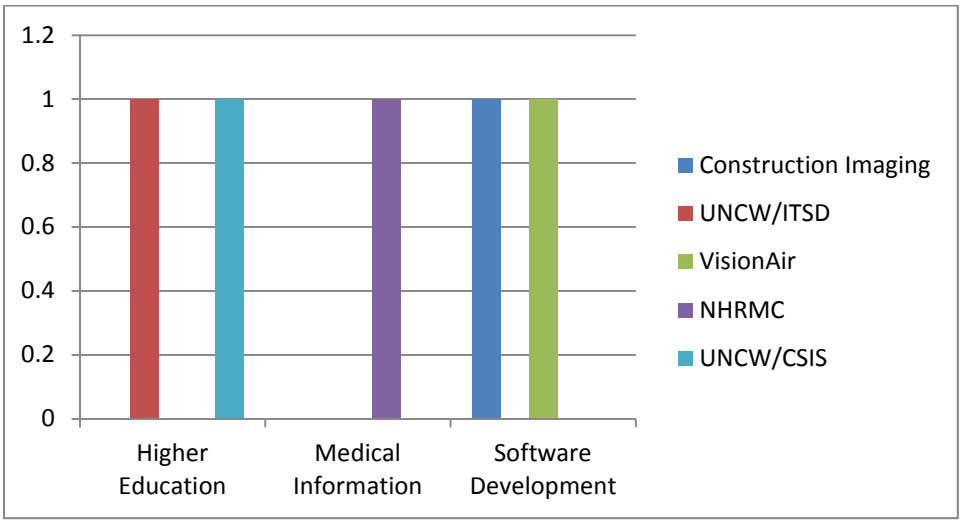


4b) *Expert Interviews – UNCW / CSIS*

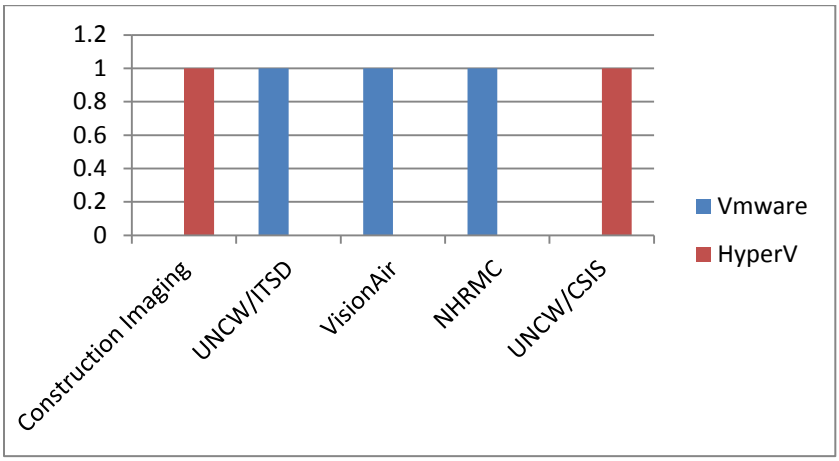


5b) Survey questions/responses – Responses depicted in charts

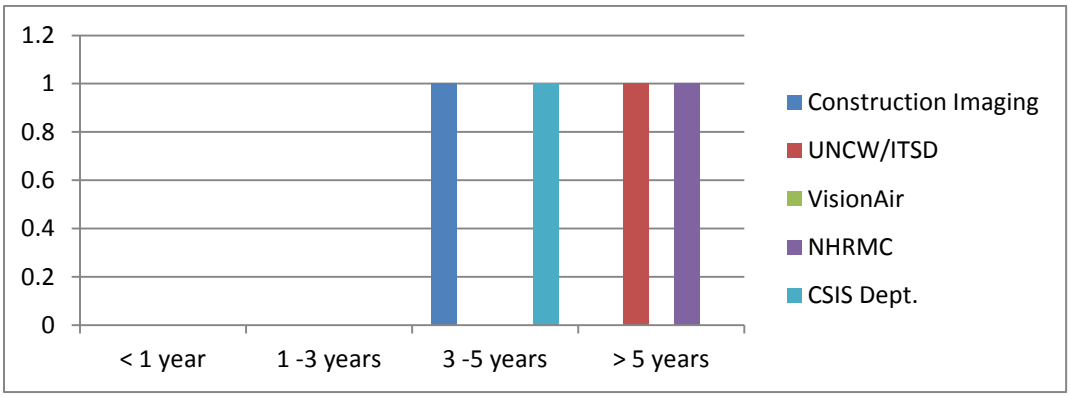
What is the main industry of your organization?	Higher Education	Medical Information	Software Development
Construction Imaging			1
UNCW/ITSD	1		
VisionAir			1
NHRMC		1	
UNCW/CSIS	1		



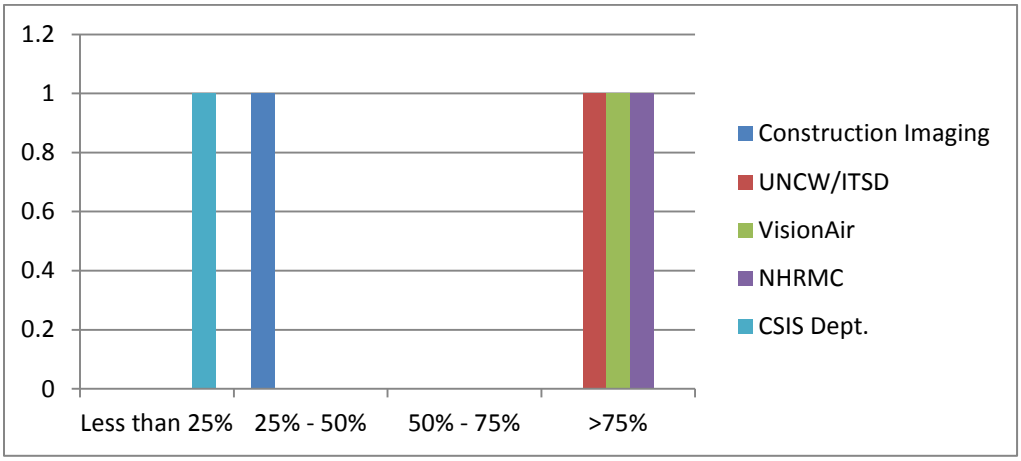
Which virtualization platform does you organization use?	Vmware	HyperV
Construction Imaging		1
UNCW/ITSD	1	
VisionAir	1	
NHRMC	1	
UNCW/CSIS		1



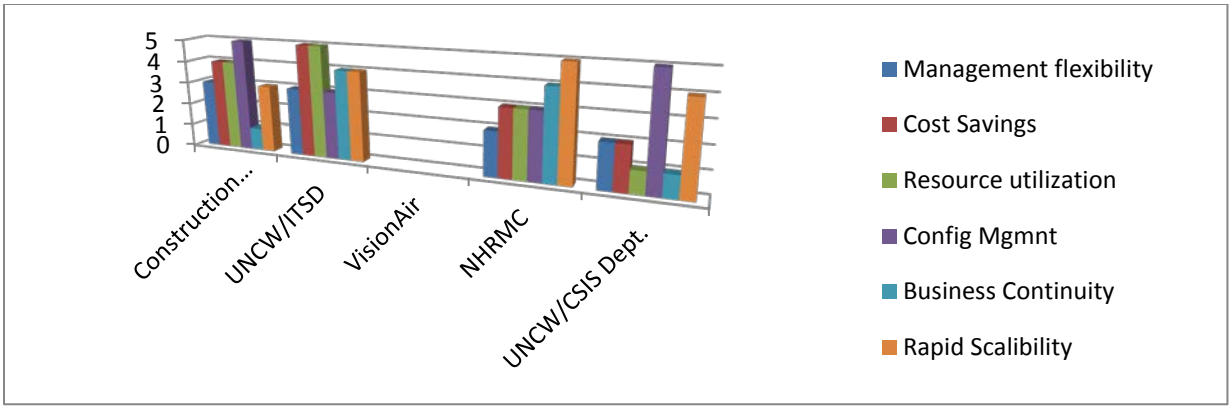
How long has your organization been using Virtualization?	< 1 year	1 -3 years	3 -5 years	> 5 years
Construction Imaging			1	
UNCW/ITSD				1
VisionAir				
NHRMC				1
CSIS Dept.			1	



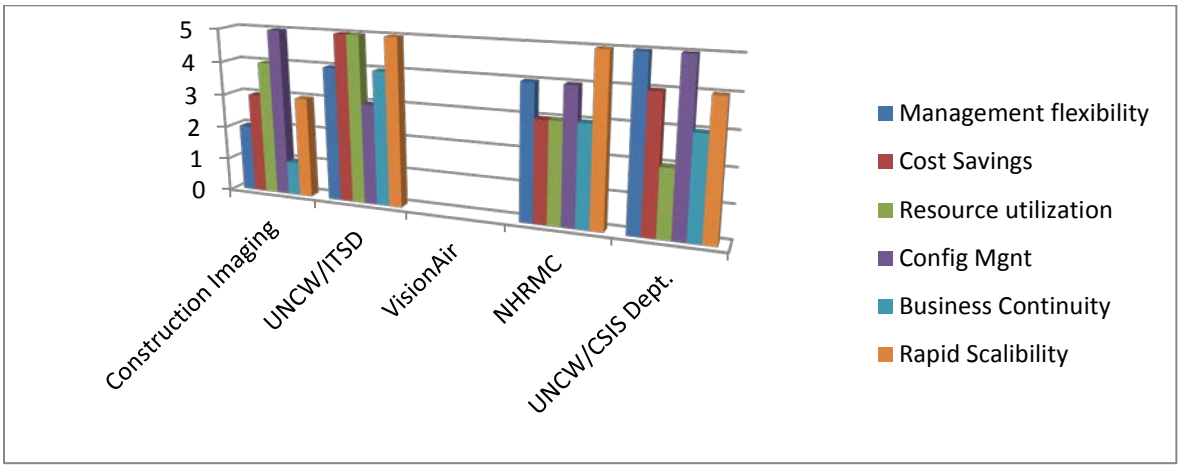
How much of your organization depends on Virtualization?	Less than 25%	25% - 50%	50% - 75%	>75%
Construction Imaging		1		
UNCW/ITSD				1
VisionAir				1
NHRMC				1
CSIS Dept.	1			



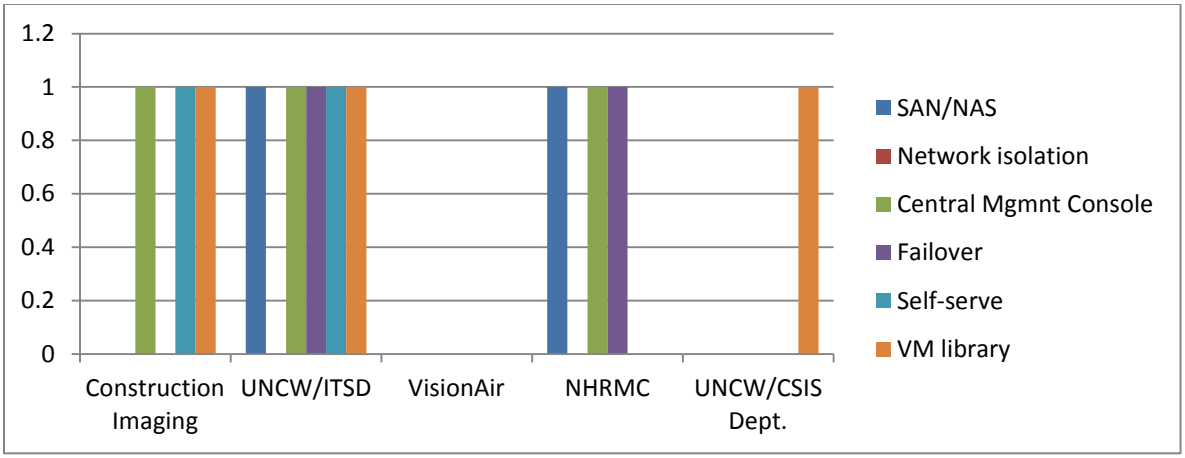
Rate in importance the Initial reason for adopting Virtualization. (1 = least important, 5 = most important)	Management flexibility	Cost Savings	Resource utilization	Config Mgmnt	Business Continuity	Rapid Scalibility
Construction Imaging	3	4	4	5	1	3
UNCW/ITSD	3	5	5	3	4	4
VisionAir						
NHRMC	2	3	3	3	4	5
UNCW/CSIS Dept.	2	2	1	5	1	4



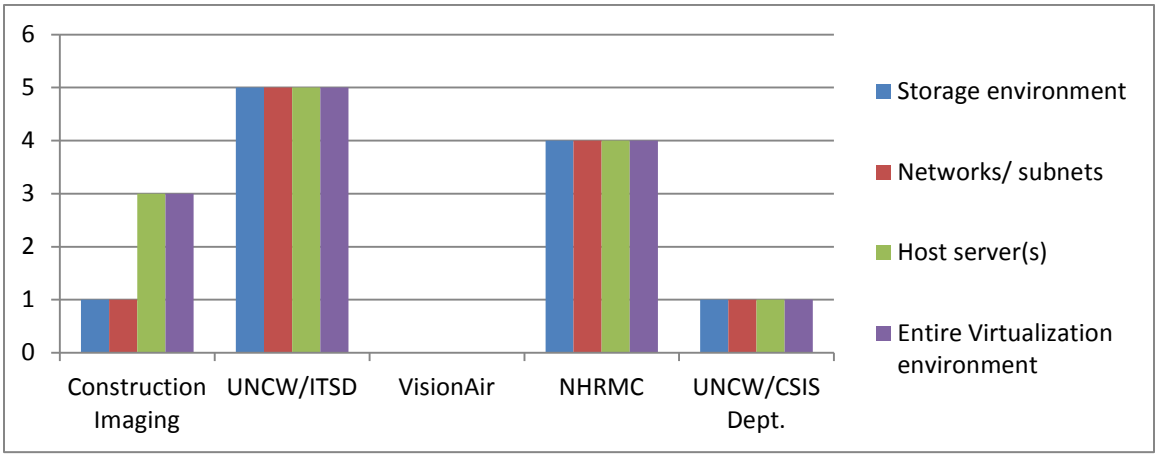
Rate in importance the Current reason for adopting Virtualization. (1 = least important, 5 = most important)	Management flexibility	Cost Savings	Resource utilization	Config Mgnt	Business Continuity	Rapid Scalibility
Construction Imaging	2	3	4	5	1	3
UNCW/ITSD	4	5	5	3	4	5
VisionAir						
NHRMC	4	3	3	4	3	5
UNCW/CSIS Dept.	5	4	2	5	3	4



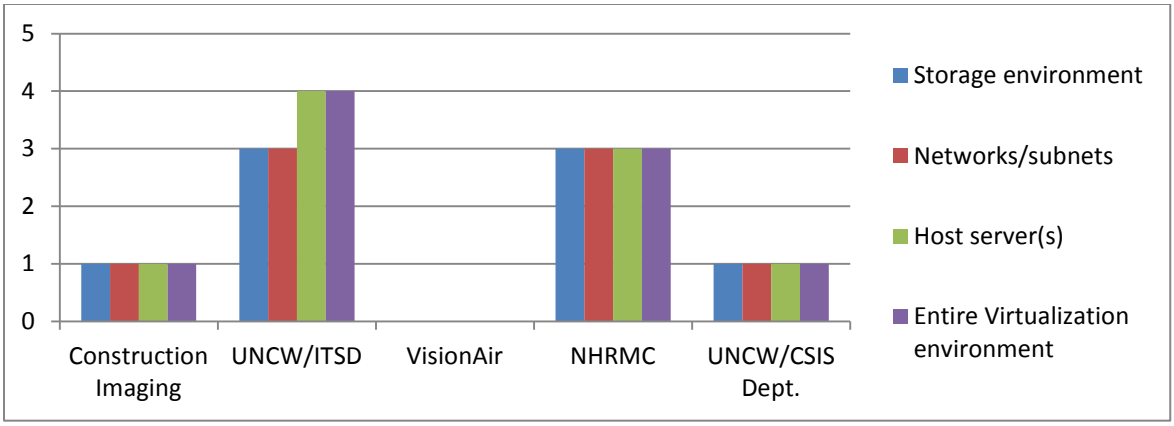
Which of these technologies is being used in your virutal enviroment?	SAN/NAS	Network isolation	Central Mgmnt Console	Failover	Self-serve	VM library
Construction Imaging			1		1	1
UNCW/ITSD	1		1	1	1	1
VisionAir						
NHRMC	1		1	1		
UNCW/CSIS Dept.						1



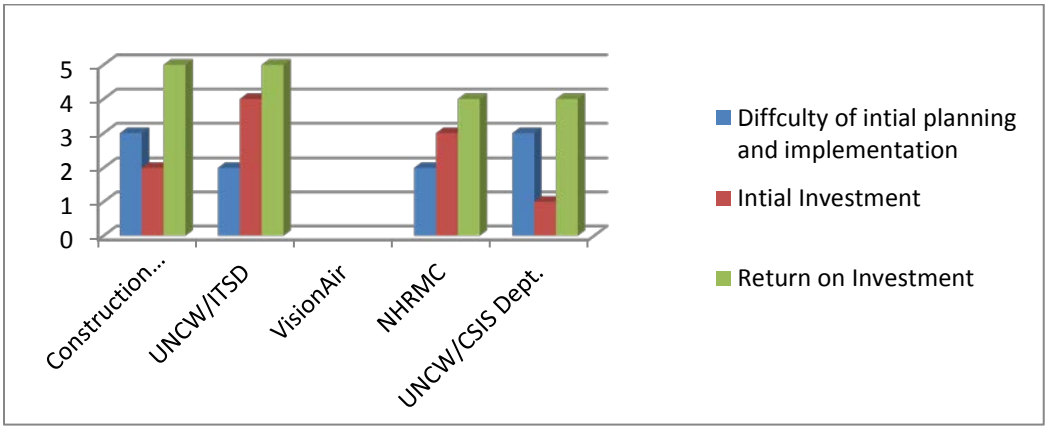
Rate the following on scale of home-grown to turn-key vendor solution: (1 = Completely home-grown, 5 = Completely turn-key vendor solution)	Storage environment	Networks/ subnets	Host server(s)	Entire Virtualization environment
Construction Imaging	1	1	3	3
UNCW/ITSD	5	5	5	5
VisionAir				
NHRMC	4	4	4	4
UNCW/CSIS Dept.	1	1	1	1



Rate the folowing pertaining to your organization's virtualization infrastructure equipment: (1 = General commidity, 5 = Specialized virtualization)	Storage environment	Networks/subnets	Host server(s)	Entire Virtualization environment
Construction Imaging	1	1	1	1
UNCW/ITSD	3	3	4	4
VisionAir				
NHRMC	3	3	3	3
UNCW/CSIS Dept.	1	1	1	1



Answer the following pertaining to your organization's virtual environment: (1 = Very low, 5 = Very High)	Diffculty of intial planning and implementation	Intial Investment	Return on Investment
Construction Imaging	3	2	5
UNCW/ITSD	2	4	5
VisionAir			
NHRMC	2	3	4
UNCW/CSIS Dept.	3	1	4



Appendix C

1c) Questions that were asked in Expert Interviews

Interview Questions

Date: Today
Interviewer: Tyler Loftis
Interviewee: Expert
Company: XYZ

1. What problem(s) does virtualization solve?

2. How is virtualization utilized to solve the problem?

3. What benefits are gained from virtualization?

4. Are there any drawbacks to using virtualization?
    - Were there any training issues?

5. How is the virtual environment managed?
    - Who configures VMs
    - Who uses VMs
    - Who distributes VMs
    - Who creates VMs

6. Where are the VMs stored or are they created ad-hoc?
    - What data storage is used

7. What management console is used?
    - Strengths
    - Weaknesses

8. What platform does your virtual environment run on?

9. What physical machines are used?

10. What types of VMs are being used?

11. How is the domain setup to manage traffic?

12. What are considered to be 'critical features' of the system?
    - What **must** the system perform or achieve in order to be deemed successful

13. What would you like to see improved upon?

Appendix D

1d) Code- Final Implementation

Node2

sbdadm create-lu <path to disk or dataset>
    - do this for each disk or dataset
itadm create-tg <name>
Itadm create-tpg <name IP:Port>
Itadm create-target –n <name>
Svcadm disable stmf
    - Must disable stmf to add tg members
Stmfadm add-tg-member –g <tg name> <target iqn>
Svcadm enable stmf
Stmfadm create-hg <name>
Stmfadm add-hg-member –g <hg name> <host initiator iqn>
    - Host initiator is the initiator-node from Node1
Stmfadm add-view –t <tg name> -h <hg name> <GUID of LU>
    - Do this for each LU

Node1

Cd /kernel/drv
    - Inside this directory edit iscsi.conf with the vi editor; it needs to read
        - Mpxio-disable=”yes”;
Iscsiadm modify discovery --send targets
    - The command must be as is written above
        - (2) dashes (--) and targets is plural
Iscsiadm add discovery-address <IP:Port>
    - Do this for all NICS being used
Iscsiadm list target
    - Check to see if discovery is working
Echo | format
    - This will show all devices/disks accessible from Node1
Zpool create <zpool name> raidz2 <disk1> <disk2> <disk3> <disk4>
Zfs create –V <size> <path to dataset>
    - Example: zfs create –V 1800GB /dev/zvol/rdsk/SANPOOL/VMStore
Sbdadm create-lu <path to dataset>

- Example: sbdadm create-lu /dev/zvol/rdsk/SANPOOL/VMStore

Now create iscsi configuration on Node1 to connect Node1 to host servers

itadm create-tg <name>

Itadm create-tpg <name IP:Port>

Itadm create-target –n <name>

Svcadm disable stmf

- Must disable stmf to add tg members

Stmfadm add-tg-member –g <tg name> <target iqn>

Svcadm enable stmf

Stmfadm create-hg <name>

Stmfadm add-hg-member –g <hg name> <host initiator iqn>

- Host initiator is the initiator-node from host servers

Physical host servers

- Start
- Administrative tools
- Iscsi initiator
- Add IP address of Node1 data network domain
- Click “quick connect”
- On volume and devices tab – click “auto-configure”
- Start
- Right click computer – click manage
- Right click with disk area with/in the display
- Click New simple volume
- Make sure it has a boot record (MBR)
- Chose size
- Chose drive letter
- Format with file system
- Drive is now accessible from windows explorer as if it is direct attached storage
- Repeat for other hosts
    - As other host are added, their iqn will need to be added as a member of the host group on Node1

2d) IT Platform Picture - Front

3d) IT Platform - Back