Efe Philip Osazee-Obazee

Data Management in an Elastic Storage Environment

Backup and Archive

Helsinki Metropolia University of Applied Sciences

Bachelor of Engineering

Information Technology

Thesis

30 May 2015

Helsinki Metropolia University of Applied Sciences

Author(s) Title Number of Pages Date	Efe Philip Osazee-Obazee Data Management in an Elastic Storage Environment: Backup and Archive 38 pages + 2 appendices 30 May 2015
Degree	Bachelor of Engineering
Degree Programme	Information Technology
Specialisation option	Communication and Data Network
Instructor(s)	Matti Puska, Senior Lecturer Stephan Schorn, (IBM GmbH) Manager Infrastructure and Lab Solutions

This project aimed to implement a storage solution with a backup and archiving functionality in a scalable environment. An educational institution was used, processing large amount of data of both staff and student as a use case scenario. The project showed how a scalable storage system can be setup and used to meet the needs of the institution.

In the project, IBM Spectrum Scale a storage file system was used along with IBM Spectrum Protect and IBM Spectrum Archive, for backup and archiving respectively. During implementation, a single node IBM Spectrum Scale cluster was set up with direct attached disks, and then connected to Tape Library via SAN. Also was IBM Spectrum Archive for archiving on the IBM Spectrum Scale node. The IBM Spectrum Protect Server and Client which was used for backup was also configured.

The result of this project was an elastic storage environment with a backup and archiving functionality. In the test, the backup functionality provided by IBM Spectrum Protect allowed a policy-based controlled backup and the IBM Spectrum Archive made it easy to migrate data from disk to tape with no additional software. This makes it easy to store data in a less expensive way. With IBM Spectrum Scale included in the setup, the system has possibility to add/remove client node without affecting the overall setup.

Some new skills were acquired during this project and on my old ones improved, for example; project planning, time management, IBM Spectrum Scale configuration (GPFS), IBM Spectrum Archive (LTFSEE) configuration and IBM Spectrum Protect client and server configuration.

The project was a success and all the goals were achieved.

Keywords	storage, archiving, backup, GPFS, LTFS

Contents

Abbreviations

CIFS	Common Internet File System
DAS	Direct Attached Storage
GPFS	General Parallel File System
GUI	Graphic User Interface
HSM	Hierarchical Storage Manager
IBM	International Business Machine
IP	Internet Protocol
iSCSC	Internet Small Computer System Interface
ITDT	IBM Tape Diagnostic Tool
LTFS	Linear Tape File System
LTFS EE	Linear Tape File System Enterprise Edition
LTO	Linear tape Open
NAS	Network Attached Storage
NBT	NetBios Over TCP
NFS	Network File System
NSD	Network Shared Disk
RAID	Redundant Array of Independent Disk
SAN	Storage Area Network
SDS	Software Define Storage
SSH	Secure Shell
SSL	Secure Socket Layer
SNC	Shared Nothing Cluster
TCP	Transport Control Protocol
TSM	Tivoli Storage Manager
UDP	User Datagram Protocol

List of Figures

1 Introduction

The world is witnessing a tremendous growth of the Internet, from social media platforms to household device connectivity. It is expected that Internet users will reach four billion by the year 2020, thereby increasing the amount of data on the Internet. This growth in data has also come with challenges, such as, how these data will be stored, and what these data will be used for. Companies are continuously looking for ways to manage and store data, and have identified the need to have a storage system that is able to scale. Some have termed it “Elastic Storage” while others “Hyper-scale Data Storage”. Storing data, either structured or unstructured is important for the future of analytics. It will transform the way companies do their business, how doctors and healthcare providers administer treatments, and how governments and organizations make decisions.

Companies like IBM have developed smart systems that will make it possible for large companies to store, manage, secure and access their data regardless of size. The General Parallel File System (GPFS) tagged “Elastic Storage” is a technology developed by IBM which provides high performance, allowing data to be accessed over multiple hosts at the same time as well as providing management and administrative tools for the shared files. Elastic Storage (IBM GPFS) provides a high availability platform supporting database application. It provides a global namespace for files to be accessed in parallel over different connection types, networks or the Storage Area Network (SAN) infrastructure. When integrated with IBM Tivoli Storage Manager (TSM) and IBM Linear Tape File System (LTFS), Elastic Storage (IBM GPFS) can manage the life cycle of data automatically in a tiered storage pattern.

The importance of storage cannot be emphasized without discussing about Backup and Archive. Backup and Archive is an important task for storage system, this provide availability and reliability of data. This project is about data backup and archiving in an Elastic Storage environment. I will compare how Backup and Archive is done with IBM Tivoli Storage Manage (TSM) and IBM Linear Tape File System (LTFS). I will also discuss how an Elastic storage (GPFS) environment is set up and the different kinds of infrastructure available.

1.1 Motivation

Innovation is changing the world and it is happening really fast. Today the number of devices connected to the Internet is estimated to be around 25 billion, which is three times the number of people on earth and this number is expected to reach around 50 billion in the year 2020. Over the years, I have always been curious about how these devices communicate with each other, what the technologies behind them are, how data are stored and managed. These questions led me into the field of Information Technology where I particularly became interest in Communication and Data Network which is my current major. Having understood how computers interact with each other I was eager to know the method data are stored.

My working at IBM Mainz, Germany (EMEA Storage Competency Center) afforded me the opportunity to understand the world of storage, giving me an in-depth knowledge of what a storage system is, understanding what Storage Area Network (SAN) is, what the different types of architecture for Storage are, helping me to know the type of technology needed for the future Big Data, also helping me to understand the importance of a scalable storage system (Elastic Storage) as it relates to future growth of the Internet.

So with so much already learnt and even much more to learn, my bachelor's thesis is going to focus on how billions of data are being managed (backup and archive) with the help of technology that gives room for scalability and flexibility. It will also provide an insight on how variety of data (structured and unstructured) are being managed in large volumes.

I chose this topic to prepare myself for the future on how these stored data will change the way we interact, the way companies do business, the way doctors administer treatments and how governments and organizations make their decisions. Understanding the mechanism of how these "Big Data" are stored will help to me understand the future of Big Data Analytics as it relates to the Internet of Things (IoT).

1.2 Aim of Project

The goal of this project is to implement backup and archive using Tivoli Storage Manager (TSM) and Linear Tape File System (LTFS) in an Elastic Storage environment.

Prior to this project several evaluations and lab sessions have been done by different teams in IBM Mainz related to this topic. So this project is to show from scratch how elastic an storage environment is set up, how GPFS is installed and configured, and how to install LTFS and TSM in an Elastic Storage environment and the test backup and archive using this setup.

The following questions should be answered by this project.

1. What is required for a storage system to be called Elastic?
2. What kind of environment can this system be deployed to?
3. How would the system scale?
4. Will it be possible to know if the system fails?
5. How can an elastic storage system be implemented?
6. How is backup and archiving done with regards to LTFS and TSM?

1.3 Scope and Delimitation

This project is going to cover installation, configuration of TSM, GPFS and LTFS, and backup and archive testing. It will give a foundation about the infrastructure used and the planning, the functionality of TSM and LTFS, and some use cases will also be discussed. This thesis will be limited to the topics highlighted in the content and will not cover the following subject areas.

- Database design
- Security
- Cost
- Data Analytics
- Cloud

1.4 Overview

In chapter 2, I will discuss the background and basis, definition of various storage architectures, storage file systems, and the concept of elastic storage. The chapter will give the foundation needed to understand the thesis. Chapter 3 will introduce the software that is used, discuss Backup and Archive, and explain this based on TSM and LTFS.

Chapter 4 will cover installation, planning and configuration of TSM, GPFS and LTFS. Chapter 5, will discuss the test comparison between both TSM and LTFS, looking at the functionalities, and some use case. Chapter 6 and chapter 7 will include explanation of the results of the test carried out in chapter 5 and the conclusion, respectively.

2 Background

Data storage describes the term used for storing data in an electromagnetic form for use by a computer or another device. Data are stored for the read and write operation; depending on the technology used this operation can be fast or slow. There are different kinds of storage technologies in today's computer environments. These technologies make it possible to store several terabytes of data, and access them even remotely for example via the cloud, thereby promoting the expansion of data storage capability.

This chapter will introduce the different kinds of storage architectures available, the type of file systems used and what backup and archive means. It will also define and explain the concept of Elastic Storage and Data Management.

2.1 Storage Architecture

There are three main kinds of architecture that has evolved in the world of storage:

- Direct Attached Storage (DAS)
- Network Attached Storage (NAS)
- Storage Area Network (SAN).

The difference between these storage architectures is the way the storage device connects to the host server, and sometimes the interfaces used in the connection. Each architecture design has its own advantages and disadvantages; this can be measured in their performance, reliability and scalability.

2.1.1 Direct Attached Storage

Direct attached storage is commonly referred to as DAS. Direct attached storage was the first networked storage architecture and it is still in wide use today. It is an architecture in which the storage device is directly attached to a server (or a client, in most cases with consumer DAS devices), rather than going through the switched network, thereby giving the hosts direct access to disks. The storage device may include one or more disks drive (Just a Bunch Of Disk (JBOD)) built into a server and, with an appropriate Host Bus Adapter (HBA), may be configured as a Redundant Array of Inexpen-

sive Disks (RAID). With DAS storage architecture, clients connect directly to the server that contains the storage in order to access the data. If the server should be down for maintenance, installation of new hardware or an application of the operating system patches, or if it becomes infected with viruses, clients would not be able to access shared data. [3]

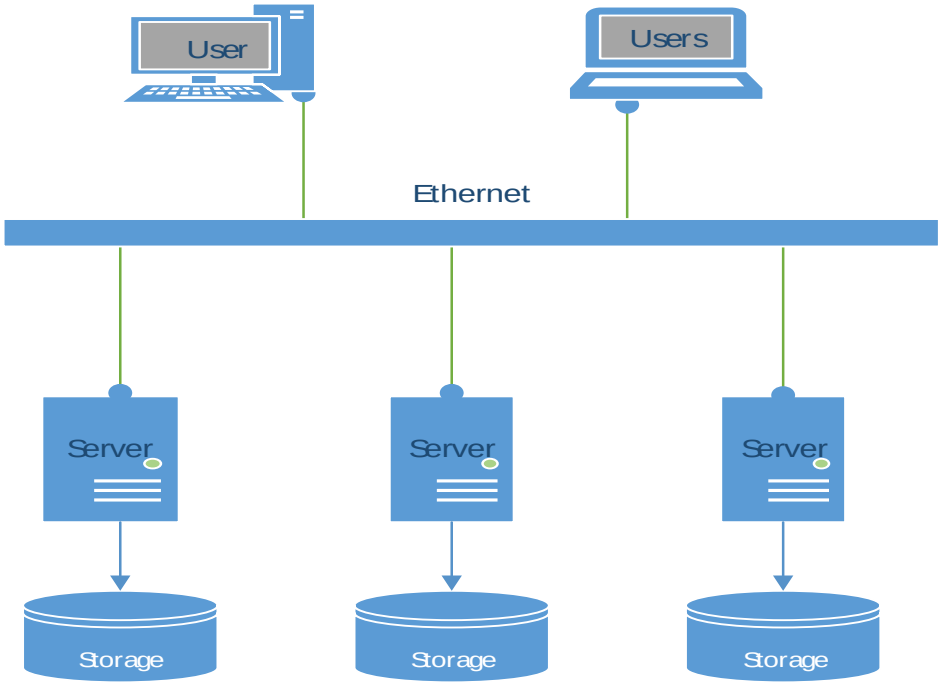

Figure 1 Direct Attached Storage Environment

Figure 1 describes DAS architecture. It is very simple and requires fewer technical skills to implement. It is mostly use by small businesses, but will become extremely difficult to expand if the data grows. If users need data, they can only connect to the server holding the data. This kind of architecture is ideal for local data provisioning.

2.1.2 Network Attached Storage

As the name implies, NAS are specialized devices that are dedicated to serving files on a network, rather than connecting host devices through a host bus adapter, it has its own operating system and storage. It can also be managed directly. This kind of architecture connects to the server using the Ethernet, and implements files sharing services. These files can be accessed using the Network File Service (NFS) protocol or Common Internet File System (CIFS). [1, 2]

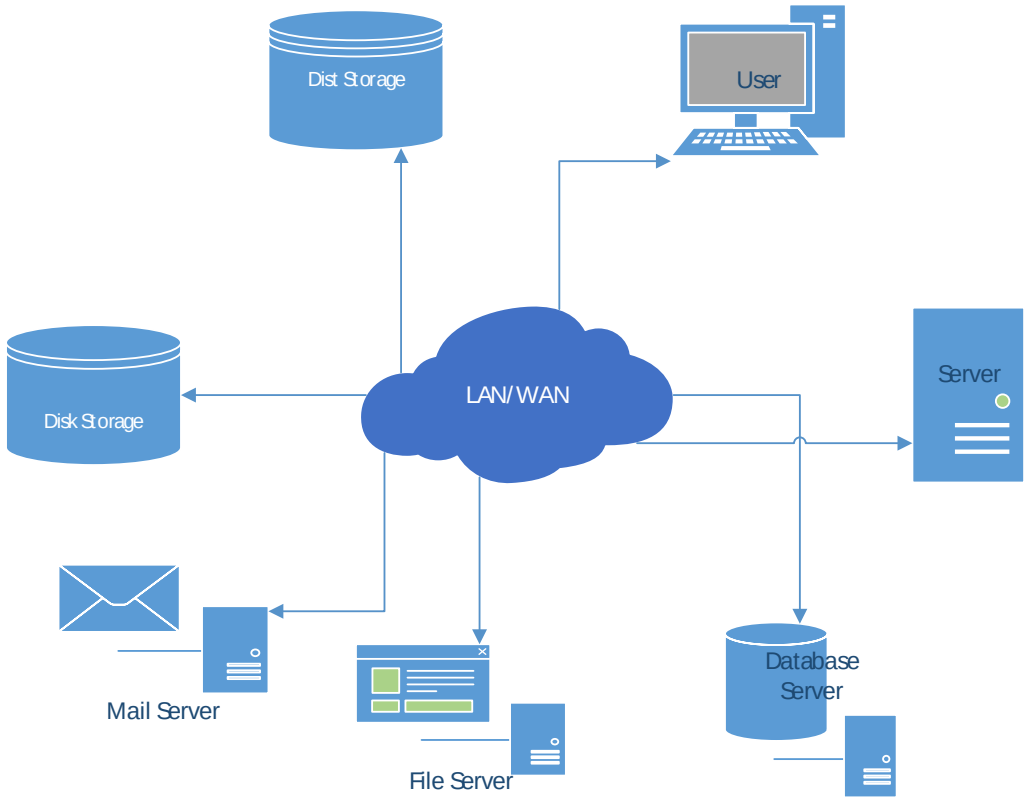

Figure 2 Network Attached Storage (NAS) Architecture

Figure 2 above shows how NAS device are connected, so it makes it easy to scale and add more storage and servers to the network, also giving it the possibilities for the different connection types to be used in the network. Some of the drawbacks of NAS are that it consumes much bandwidth since the storage device, servers and clients share the same Ethernet, and this might cause the network to congest and packets might get dropped. The more storage is attached to the network, the more bandwidth is required.

2.1.3 Storage Area Network

A SAN is a specialised high-speed network that connects servers and storage. It's primary purpose is to transfer data between a storage device and computer systems. SAN is mainly used for block-oriented data/file transfer. A SAN can be called the network behind the server, since it has communication infrastructures which provide physical connection means that help communication between the storage and computers. A SAN network might include one or several storage devices like tape libraries, disk drives or flash storage. One of the most common connection type uses by SAN is the Fibre Channel, which is very expensive to implement. The introduction of the Internet Small Computer System Interface (iSCSI) protocol made it also possible to connect

SAN less expensively. iSCSI performs encapsulation of SCSI in Transport Control Protocol/Internet Protocol (TCP/IP) while Fiber Channel encapsulate SCSI commands, so hosts and storage can use an Ethernet interface while the IP network serves as the SAN infrastructure.

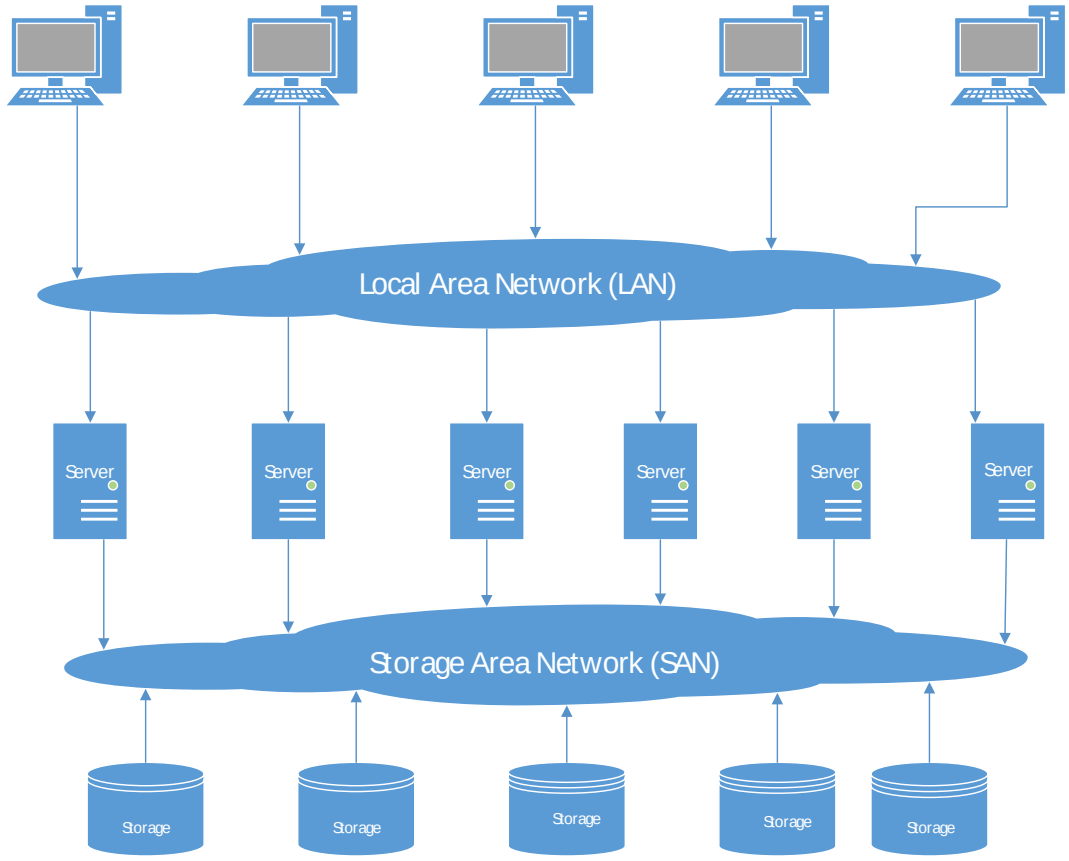

Figure 3 Storage Area Network (SAN)

Figure 3 is SAN architecture. SAN is commonly used in data centre design because it reduces the bottleneck of traditional networks. By increasing the data transfer between servers and storage via high-speed connections, it also makes it easier for applications which are responsible for data transfer (backup and archive) to move data with little interference with servers, thereby saving time and increasing performance. Another key benefit of using SAN is high usability; it is possible to access data through multiple paths making it reliable and flexible. Using a SAN can offer simplicity in storage management. Storage is centralised, and it is easy for data mirroring across different sites, which is important in the case of disaster recovery and data protection. Data replication is also possible with SAN infrastructure, as data can be offloaded from servers and move onto separate networks. [7]

One of the most important benefits of SAN is scalability. Servers and storage devices can be continuously added to or removed from the networks without problems, making it very flexible.

2.2 Storage File System

For storage devices to communicate with servers and share data in a specific format (file level), a storage file system is needed. It enables the sharing of the same copies of files stored on common storage media among multiple servers using different operating systems. A storage file system simplifies and streamlines storage management, minimizes storage and retrieval time. It optimizes the use of storage resources, allowing network components to be scaled individually, and eliminates the need for storage redundancy. [4]

In this project, four major storage file systems will be discussed: Network Files System (NFS); Common Internet File System (CIFS), General Parallel File System (GPFS); Linear Tape File System (LTFS)

2.2.1 Network File System (NFS)

Network File System (NFS) was developed in 1984 by SUN Microsystem. It is a distributed storage file system that allows the server to share directories and files with clients over a network. It was built on the Open Network Computing Remote Procedure Call (ONC RPC) system. NFS allows a remote host to mount the file system over the network and access files on this mount point like on a local storage.

There are three versions of NFS: NFSv2, NFSv3, NFSv4. All three versions use the Transport Control Protocol (TCP) over IP networks. NFSv2 and NFSv3 can use User Datagram Protocol (UDP) to provide a stateless network connection; this means that information about the client's access is not stored in the server, thereby increasing performance in a non-congested network. On the other hand, if the server is congested and it fails, the UDP client will continue to flood the network for a request for a server, and also because it is stateless; data consistency is not assured. [5]

NFSv4 improves performance, security and uses a stateful protocol. It listens from the TCP port, which eliminates the need for a portmapper interaction, because the mounting and locking protocols have been incorporated into this version.

In NFSv4, the client is granted access by the TCP wrapper. The NFS server refers to the configuration file /etc/exports to determine whether the client is allowed to access any of the exported file system. Once access is granted, all files and directory operation will be available to the user. [5]

2.2.2 Common Internet File System (CIFS)

CIFS is a distributed file sharing protocol that provides an open cross-platform mechanism for clients systems to request file services from the server system over a network. It was developed by Microsoft based on the Server Message Block (SMB) protocol. It is a high-level network protocol, which means that it is an application-level protocol which depends on other transport protocols. It mainly uses NetBIOS over TCP (NBT). [6]

CIFS defines a series of commands used to pass information between networked computers. The redirector packages requests meant for remote computers in a CIFS structure. CIFS can be sent over a network to remote devices. The redirector also uses CIFS to make requests to the protocol stack of the local computer [7]. It provides security and authentication between the client and the server. CIFS is mostly used by Windows OS.

2.2.3 General Parallel File System (GPFS)

GPFS was developed by IBM; it is a high-performance shared disk clustered file system that provides solutions for parallel computing problems. It was designed to achieve high bandwidth for concurrent access to a single file or collections of files, especially for sequential access from multiple nodes [8]. GPFS can contain either SAN or NAS, which enables high-performance access and a scale-out solution. A GPFS cluster can be created in an AIX, Linux, and Windows node, or a combination of different operating systems. GPFS can run also in a virtualized environment. With GPFS, data replication, policy-based storage management, and data access can be done over a multiple GPFS clusters, across LAN or WAN. [9]

GPFS allows sharing of files between separate clusters. This kind of access can be used for administrative purposes, which means a cluster does not have to own the storage, but can mount a file system from another cluster, thereby making it possible for separate management of the GPFS cluster. This kind of design is referred to as a multi clustered environment using GPFS remote mount features. Open Secure Socket

Layer (OpenSSL) is use to authenticate and check authorization for all network connections in a multi-clustered environment. [9]

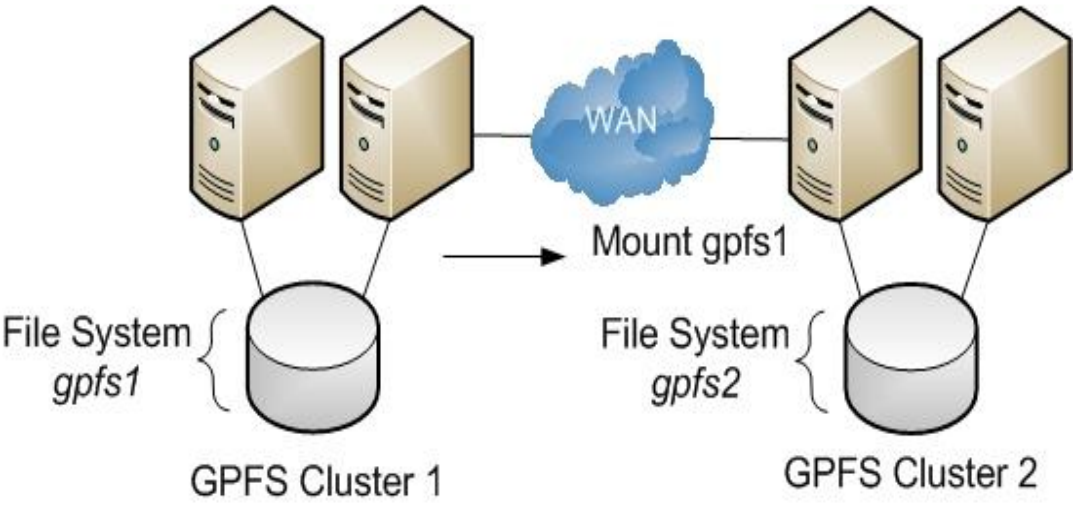

Figure 4 A multi-clustered architecture. Copied from IBM GPFS: Concept, Planning and Installation. [9]

Figure 4 above shows a multi-clustered GPFS architecture. GPFS is not a client-server file system like NFS or CIFS. The multi-clustered environment improves system performance by allowing multiple processes on every node in the cluster simultaneous access. Data consistency is guaranteed, with increase data availability. It also enhances system flexibility, which means it is possible to add and delete disks while the file system is mounted [9]. GPFS can be configured using different models, SAN model, NSD model and Share Nothing Cluster (SNC) model.

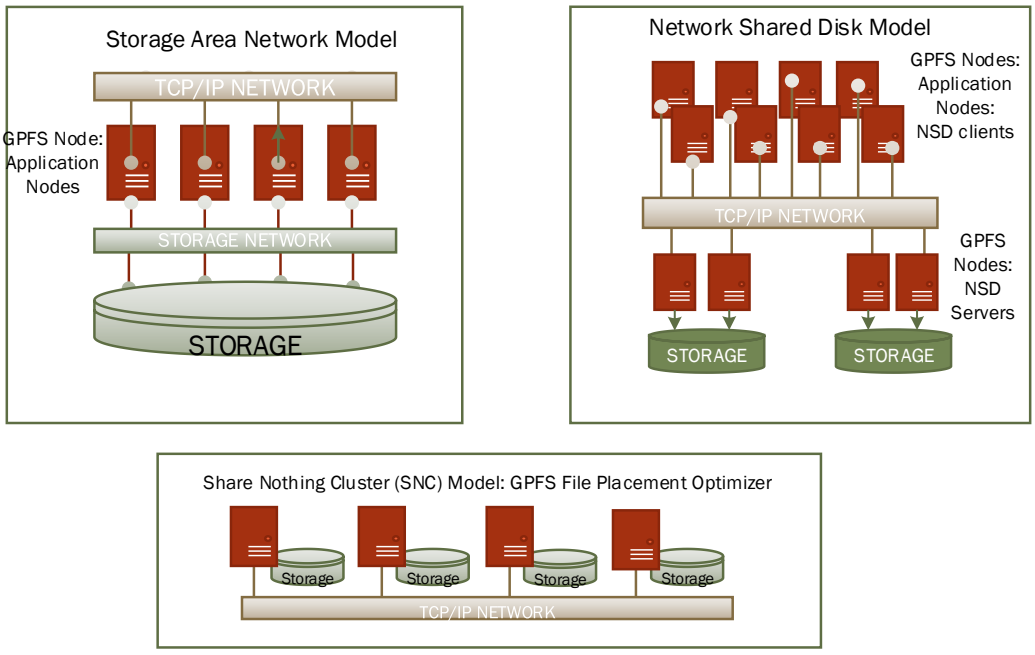

Figure 5 GPFS Models

Figure 5 shows the different environments where GPFS can be implemented. In the SAN model, the servers are connected to the TCP/IP network and the storage network, and this model has no GPFS client connected. The NSD model introduces the cluster. In Figure 5, the NSD has a total of four servers. Two servers are connected to storage and are called Cluster1, while the other two are also connected to storage and called Cluster2. In Figure 5, the NSD model shows that eight servers are used as GPFS NSD clients, four of the servers are connected to one cluster and the other four server are connected to the other clusters. The clients can access storage through the NSD server. There is really no difference in terms of configuration between a GPFS Server and Clients. The difference is in the licencing. In the Shared Nothing Cluster (SNC) design each GPFS server has its own storage and is directly connected to the network, which is the same principle as in DAS.

2.2.4 Linear Tape File System (LTFS)

As the name implies, LTFS is a tape-file system that allows tape drives to act like a disk base file system in order for files to be drag and drop to and from the tape. Developed by IBM to address tape archive requirements, LTFS works with LTO-5 tape technology and later versions. Currently IBM offers four different options of LTFS, Single Drive Edition, Library Edition, Storage Manager and Enterprise Edition. The Single Drive Edition adds support for IBM tape automation in addition to the single drive use, and all data are managed in a standalone environment like disk. Library Edition allows access to all the data in the tape library as if it were on disk in single or multiple access in a tape library. LTFS Storage Manager is integrated into the Library Edition to make it possible to manage, monitor and restore files both online and offline in tape libraries. It utilizes the hierarchical storage management functionality to balance resources for file write and recall. This provides a solution for storage life-cycle management. [10, 11, 12]

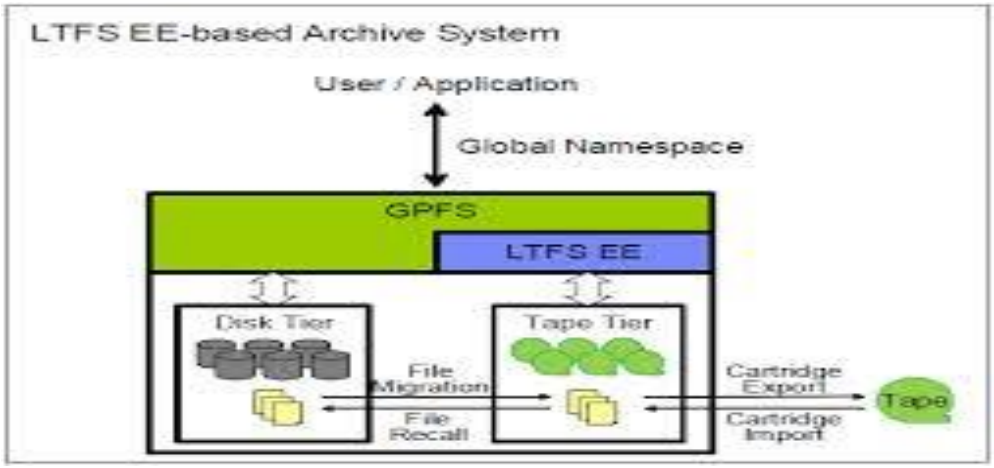

Figure 6 High-level overview of LTFS EE archive solution. Copied from IBM Linear Tape File System: Installation and Configuration [12]

LTFS Enterprise Edition enables applications to run on tapes even though they are designed for disk files. It can be used to replace disks with tapes in a tiered storage environment. With GPFS, LTFE EE provides the ability to add nodes and tapes devices as required, thereby making it highly scalable. It also supports migration of data to newer tapes managed by GPFS nodes. It simplifies management by providing a single infrastructure for administrators. [12]

2.3 Concept of Elastic Storage

Elastic Storage is part of the IBM Software Defined Storage (SDS) portfolio. This portfolio consists of mainly the IBM GPFS integrated into several software like IBM LTFS, TSM, and OpenStack to provide a capable software solution that automatically manage data locally or globally, with a high speed of access and a single point of administration. These software scale technology infrastructure quickly as the volumes of data grow. Elastic Storage virtualizes the storage allowing multiple systems and applications to share common pools of storage. This enables transparent global access to data without the need to modify applications, and without the need for additional storage management applications. Elastic Storage can automatically move infrequently-used data to less expensive, low-cost tape drives, while storing more frequently-accessed data on high-speed Flash systems for quicker access.[13]

The IBM Spectrum Storage includes different products that enhance the speed and efficiency of storage and simplify migration. Some of them include; IBM Spectrum Scale, IBM Spectrum Protect and IBM Spectrum Archive.

IBM Spectrum Scale is a scalable, high-performance data and file management solution. IBM Spectrum Scale provides storage management with extreme scalability, flash accelerated performance, and automatic policy-based storage tiering from flash through disk to tape. IBM Spectrum Protect is a data protection and data recovery solution which provides protection for virtual, physical, cloud and software defined environments, as well as core applications and remote facilities. IBM Spectrum Archive provides an easy to use GUI that enables one to automatically move infrequently accessed data from disk to tape without the need for proprietary tape applications. [14]

Subsequently, in some cases during this thesis IBM GPFS will be referred to as IBM Spectrum Scale; IBM TSM will be referred to as IBM Spectrum Protect and IBM LTFS will be called IBM Spectrum Archive.

3 Data Management

The importance of securing data cannot be overemphasized, as all data have value. There are several technologies available in storing data. The most common are two main operations, Archive (Data Discovery) and Backup (Data Recovery). For complete data management, backup and archiving solutions can be implemented. In this chapter, I will discuss the different terms and technologies used for backup and archive.

3.1 Backup

Backups are additional copies of data that are created in case of data loss, data corruption or other disasters. Backups are done periodically to create different copies of productions data to serve the purpose of disaster recovery or operational recovery. There are different types of backup methods: full backup, incremental backup, selective backup, logical backup and differential backup. Different applications help provide solution for this kind of policy based backup. In this project I will use the IBM Spectrum Protect.

3.1.1 Full Backup

Full backup is the process where all images, files and folders are selected for backup, it is usually the starting point of all backups and must be done first before other backup are done. This kind of backup makes restore process easy to manage. A full backup is done occasionally because it is possible that data will not change constantly, and if a full backup is done regularly it may cause several copies of the same data to be available, thereby wasting storage space. Also it takes a longer time compared to other types of backups and requires a large amount of storage space. [15]

3.1.2 Incremental Backup

In incremental backup, only files that have changed since the last backup whether full back up or incremental, will be backed up. It is done in order not to perform full back up on a regular basis which might affect the storage capacity. Incremental backup is done, because typically a very small percentage of files change between the periods of backup. Incremental backup is like an update of the last backup, only taking into account those files that have changed, so that during the restore process the recent copies of the files are found. It takes less time to perform and less storage space is required. It also helps to save bandwidth when done over the network. [15]

3.1.3 Logical Backup

Logical backup is a type of backup technique that allows one to back up a file system or a raw logical volume. It is similar to full backup, but in logical backup the entire backup is treated like in a single file. It is considered to be an export/import operation because it takes backed up logical volume and puts it into a file. It does not treat each file or folder individually during backup, so if a file is damaged it will not be able to identify it. Logical backup is done mainly to restore a system to its last state. [15]

3.1.4 Differential Backup

Differential backup is similar to Incremental backup, but different in the sense that, only files that have changed since the last full backup was done will be backed up. The advantage of this is that during file restore, only the Full backup and the differentials will be required to get the complete file restored. It also requires less storage capacity and it take shorter time to complete. [15]

3.1.5 Selective Backup

Selective backup is the manual form of incremental backup, which allow one to select the files they want back up. The advantage of this is that is allows one to select files from different directories that one wish to back up, saving storage space and time by avoiding files that are of less importance. Selective backup is useful in an environment where there is a large inflow of data but only few are needed. [15]

3.2 Archive

Archiving is a process of moving inactive or very rarely used data from the primary storage into a more cost effective storage tier. It is done to secure data and store them for a long time. Archiving services use a policy to make data easily accessible, by using a storage file system which makes managing the data in a file naming and directory structure easy. As more data are produced and stored, more files will be archived, which is why it is important for archiving to be done in an environment that can accommodate data growth and would be able to scale to adapt to the changing business requirements. Different vendors like EMC, Symantec, Oracle, NetApp and IBM have archiving solutions that provide policy-based archiving, but in this project I used IBM Spectrum Archive, which provides such a solution. This solution enables one to automatically move infrequently accessed data from disk to tape while retaining ease of use and without the need for proprietary tape applications. This reduces the load on the

local server, as old files are remotely stored and only recent files are kept on the local server. In policy based archiving, data can be retained for a certain period of time in a particular storage device, which is known as Data Retention. After that time has elapsed, the data is either automatically deleted or it will again be moved to another level of storage.

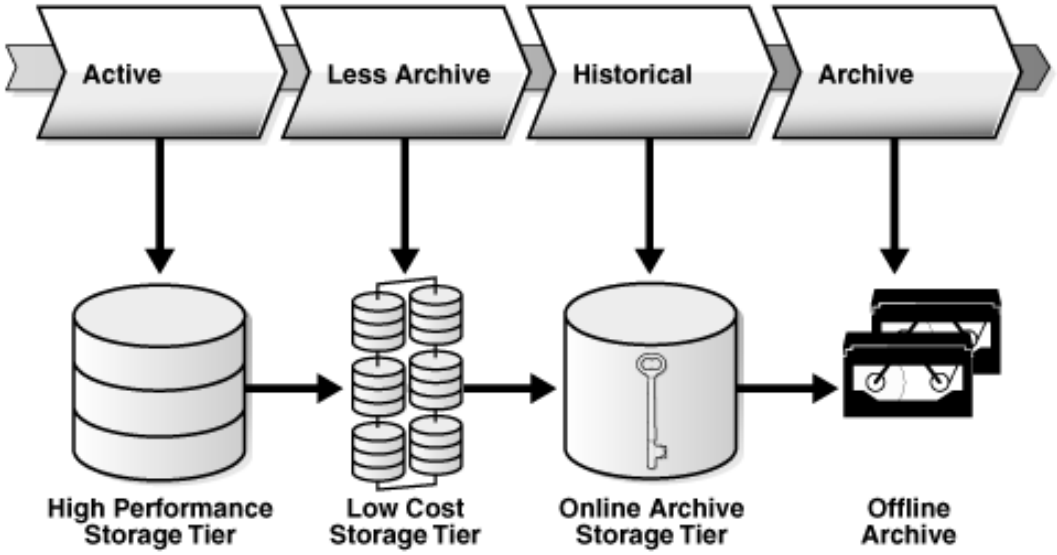

Figure 7 Tiered Storage. Copied from http://docs.oracle.com/cd/E11882_01/server.112

Figure 7 shows how data are stored in a tiered storage environment, putting inactive data in an offline media in order to create space for newly generated active media, because, as data become old, their usage will decline.

3.3 Hierarchical Storage Management

Hierarchical Storage Management (HSM) is a policy-based management that provides an automatism of managing and distributing data between different storage layers. This is done in such a way that it uses a tiered storage device economically, without users needing to be aware of when files are being backed up or retrieved from storage media. Typically, data files which are frequently used are stored on high performance storage media like disk drives. When these data become dormant, they will eventually be migrated to slower media like tape drives. If the files which are on tape need to be reused again, they will automatically be recalled and made available for use without the user noticing any slowdown or the storage device which the data are stored in. [16]

In HSM, data items are placed using a pattern that helps ease accessibility. HSM is mostly useful at sites that maintain large volumes of historical information but do not

require rapid access to that information. In a large-scale user environment, HSM makes it easy to manage storage media by using a single view storage pool to effectively manage devices. With HSM, multiple applications can have access to storage media. HSM can be implemented using different software solutions. [16, 17]

3.4 Terminology Associated with Backup and Archive

Migration is the process of transporting data between storage devices. During the process data are moved from fast high performance storage device to slower storage device. Migration helps to control the amount of free space within storage pool and can be used to move data to its permanent location.

Retention is the time period a file version is kept on a storage pool before being moved to another storage pool. Retention only applies to inactive files because an active file does not expire. When a file version is no more retained, then it will expired.

Data Recall is the process for bringing back a migrated file to its original location. In HSM, recall is used to bring back archive data or dormant data which are placed in lower storage media back to its original location or the location where it is needed.

4 Proof of Concept Implementation

This chapter is going to describe how Elastic Storage is implemented, from planning, setup, installation and configuration to testing.

4.1 Use Case Scenario

An educational institution, which processes the data of both staff and students, academic and non-academic need a data management solution that will meet it needs. More students enrol into the university the amount of data grows. And more data also become less active as older students completes their studies. It is required that the institution store some of the data of its student even after five years of graduation, and it is also expected to have copies of these data in case of disaster. Thus, the institution needs a data management system which will include backup and archive solutions that will meet its requirement.

This use case scenario will present an analytical view point for an enterprise solution as it relates to data lifecycle management.

4.2 Task

The goal of this project was to implement a scalable storage system for data management with archiving and backup solutions. Specifically, in this project, I implemented a scalable storage system using IBM Spectrum Scale (GPFS) and then integrate IBM Spectrum Protect (TSM) into that environment for Backup and IBM Spectrum Archive (LTFS) for Archiving. This project was done while I was working at IBM Mainz, for me to understand how an elastic storage system works. A successful completion of this project gave me the knowledge and skills required to implement an elastic storage system.

The projected was divided into three parts:

- Creating of an Elastic Storage System
- Integrating an archiving solution into the system
- Setup and configuration of IBM Spectrum Protect.

The first part of the project was to plan/create the infrastructure of an elastic storage system. Firstly, was to prepare the infrastructure for the elastic system by connecting the server and the storage device together and configuring the tape drive. Then I installed IBM Spectrum Scale, created the clusters on the IBM Spectrum Scale node, and then configured the NSD and files systems. The result of this stage should be a complete IBM Spectrum Scale Node, with the NSD mounted on the created file system.

The second stage was to integrate the archiving solution into the elastic storage environment. This will included installing and configuring IBM Spectrum Archive, then mounting the file system for space management, and after this part was completed, the system was ready to migrate data from one storage to another.

The last stage of the project was to configure the backup server and client using the IBM Spectrum Protect. This stage was important so that the system would be able to perform a backup to an IBM Spectrum Protect server before migrating the data to tape drive or the data can be a backup to the server before it will be migrated.

The second and last stage could be done in any order, but for this project these steps were followed to implement the solution.

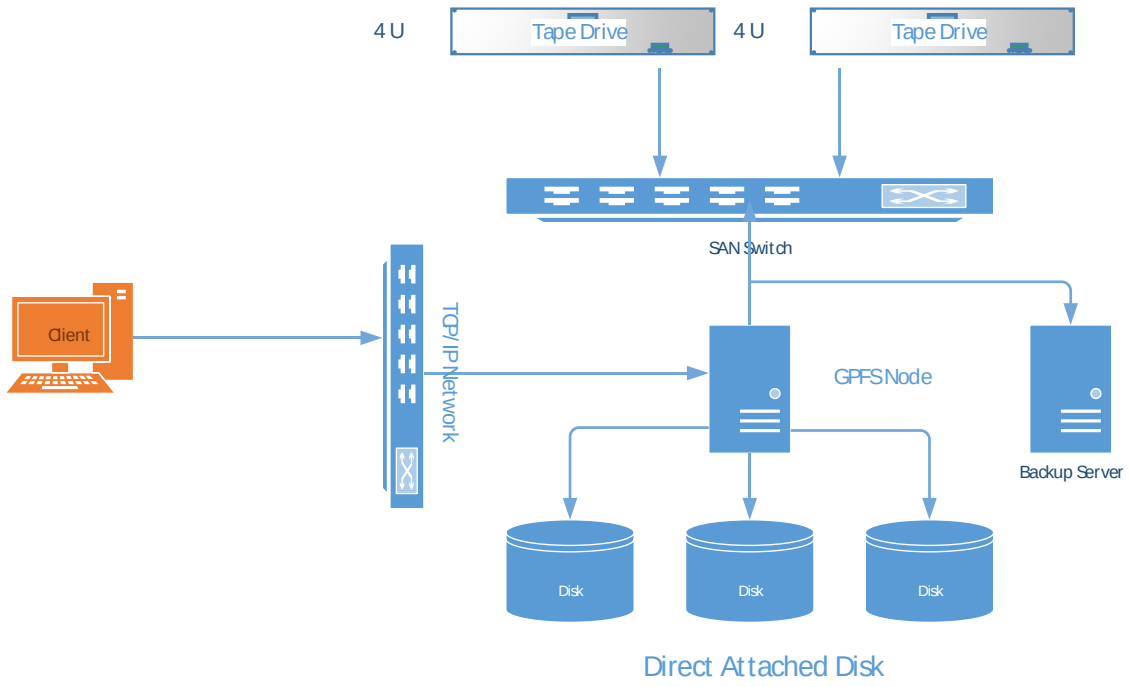

Figure 8 Setup of the Storage devices

Figure 8 shows the design used in implementing this project. In the design, I have one server which will be used as the IBM Spectrum Scale (GPFS) node. The server has its own DAS configured as RAID 1 in several pools. It was also connected to a Tape Library. The tape library was used for archiving; files were moved from disk to tape. The reason why tape was used was because tape is less expensive compared to disk and they are mostly used for archiving and a means to reduce the cost of storage.

4.2.1 Preparation

I started by installing Linux Red Hat version 6.5, which was my preferred operating system, though others could also be used. Then I connected the server to the tape drive using SAN switch. And checked with the command *cat /proc/scsi/scsi* to view if the tape drive was properly attached.

```
  Type:   Medium Changer                   ANSI  SCSI revision: 06
Host: scsi4 Channel: 00 Id: 00 Lun: 00
  Vendor: IBM      Model: ULT3580-TD6      Rev: E6R6
  Type:   Sequential-Access                ANSI  SCSI revision: 06
Host: scsi4 Channel: 00 Id: 00 Lun: 01
  Vendor: IBM      Model: 03584L32         Rev: E070
  Type:   Medium Changer                   ANSI  SCSI revision: 06
Host: scsi4 Channel: 00 Id: 01 Lun: 00
  Vendor: IBM      Model: ULT3580-TD6      Rev: E6R6
  Type:   Sequential-Access                ANSI  SCSI revision: 06
Host: scsi4 Channel: 00 Id: 01 Lun: 01
  Vendor: IBM      Model: 03584L32         Rev: E070
  Type:   Medium Changer                   ANSI  SCSI revision: 06
```

Figure 9 Attached Tape Drive.

Figure 9 shows that the media changer devices and the tape drives have been properly connected to the server using SCSI. They were connected through SAN because the Tape Library which held the Tape drives was connected to a storage network and other servers were attached to it as well. For the server to have control over the media changer and the tape drives, an IBM tape device driver (Lintape) and IBM Tape Diagnostic Tools (ITDT) were installed for diagnostic purposes and manual a library/tape drive control in case needed. These configurations were not necessary for the disk drive since it was directly attached to the server and only needed the appropriate RAID configuration. RAID 1 was selected because it consists of mirror functionality which means that in case any of the drive fails, the data will not need to be rebuilt. It will just be copied from the other disk. The next step was to install a scalable storage file system, and in this project I chose the IBM Spectrum Scale (GPFS). There are several versions of the IBM Spectrum Scale available. The IBM Spectrum Scale version 4.1 is

the newest version, and it provides enhanced security and better performance. The standard edition was used in the project.

4.2.2 IBM Spectrum Scale installation

IBM Spectrum Scale version 4.1.0 was installed on the server. It is not free software, so it requires a license. Initially, an internal network with IP aliases was created. This network will be the cluster network, which the node will use for communication. In this project, I used the same IP network for the cluster network and for the administration because it is a single node cluster. Though the project was only using a single IBM Spectrum Scale node, it was required to configure the internal network in case more nodes are to be added. Listing 1 below shows the commands used in installing IBM Spectrum Scale and the alias for the internal network, and the command to verify if the installation was correct.

```
#yum install gpfs*.rpm
#rpm -qa | grep gpfs
gpfs.docs-4.1.0-1.noarch
gpfs.ext-4.1.0-1.x86_64
gpfs.msg.en_US-4.1.0-1.noarch
gpfs.gpl-4.1.0-1.noarch
gpfs.gskit-8.0.50-16.x86_64
gpfs.base-4.1.0-1.x86_64

#vi /etc/hosts
#cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4
localhost4.localdomain4
::1         localhost localhost.localdomain localhost6
localhost6.localdomain6
9.155.113.90 RH_thesis
```

Listing 1. IBM Spectrum Scale installation

The next step was to configure password-less authentication on the node. To do this I generated a key using ssh-keygen command. After the key had been generated, I copied it to the node using `ssh-copy-id <node name>` command and specify the

node name, the node name used is RH_thesis. The authentication was necessary because other services that connect to the node need to be authenticated, and it also allows for a remote connection to the node via Secure Shell (SSH).

IBM Spectrum Scale Cluster Creation

IBM Spectrum Scale was designed as a clustered-based storage file system, which means it provides concurrent access to a single file system or a group of files system from multiple nodes [18]. After the base installation and configuration of IBM Spectrum Scale, the IBM Spectrum Scale cluster needs to be created. Table 1 below gives a brief explanation to some terms.

Table 1: Definitions of Terms

Terms	Brief Explanation
IBM Spectrum Scale Cluster	A group of independent running server that are grouped together to act like a single system
IBM Spectrum Scale Node	A single IBM Spectrum Scale server which acts as connection point to a file system or set a of file systems
Quorum	This is a cluster mechanism used by the IBM Spectrum Scale to maintain cluster operation for data consistency in event of node failure.
Cluster Configuration Repository (CCR)	A private, cluster-wide, distributed database for storing information that pertains to the configuration and state of a cluster.

The IBM Spectrum Scale cluster can be created using the mmcrcluster command. With this command all the necessary parameters that the cluster needs to function properly will be set. Creating the IBM Spectrum Scale cluster requires the cluster name, node name, node designation (role), primary server, secondary server, quorum node, remote shell command, remote file copy command and the IBM Spectrum Scale alias. There are several options that are available for the creation of IBM Spectrum Scale Cluster.

Table 2. IBM Spectrum Scale Cluster Creation Option copied from [10]

Options	Command to change the option	Default Value
Node in IBM Spectrum Scale Cluster	mmaddnode mmdelnode	None
Node designation: manager or client	mmchnode	client
Node designation: quorum or nonquorum	mmchnode	quorum
Primary Cluster Configuration	mmchcluster	none
Secondary Cluster Configuration	mmchcluster	none
Remote Shell Command	mmchcluster	/usr/bin/rsh
Remote File copy command	mmchcuster	/usr/bin/rcp
Cluster Name	mmchcluster	The node name of the primary IBM Spectrum Scale configuration server

Table 2 above shows the different options that are possible when creating an IBM Spectrum Scale cluster, also the commands to change the options are included in the table. I used the command below to create the cluster.

```
mmcrcluster -N <node_name>:manager-quorum --ccr-enable
-r /usr/bin/ssh -R /usr/bin/scp -C <group_name>
```

When using –ccr-enable the primary server does need not to be specified because the ***--ccr-enable*** command enables the configuration server repository, which stores copies of data files on all quorum nodes. For IBM Spectrum Scale to be active the licence has to be accepted on the node. It is required that the `mmchlicense server --accept -N RH_thesis` command be used to accept the license. Once the license has been accepted, I verify that the cluster has been created using the ***mmlscluster*** command

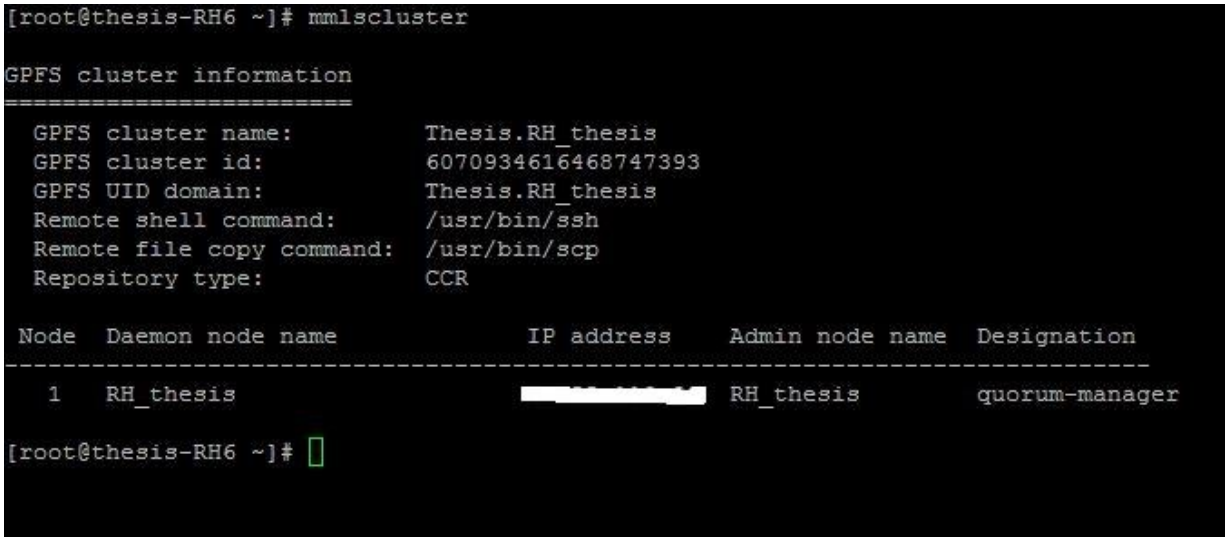

Figure 10 Cluster Creation

Figure 10 above shows the cluster name which was Thesis, with the node name RH_thesis. The NodeDesignations which is set as ***quorum-manager*** means that the node is included in a pool among which a file system manager can be selected from. Since it is a single node cluster and a file system will be created, it was necessary to designate it as manager, because by default all nodes are designated to be client. With a node designated as client it means it is not part of the pool to manage file system.

The next step is to create NSD and File System. The NSD is created because it allows the file system that will be created later have a shared storage pool to store data. It is very important that the NSD is planned before creation because it helps decide the kind of files to be stored on the pool. It is possible for the IBM Spectrum Scale to choose the kind of data needed to store on a storage pool. The USAGE Type parameter will allow selecting among the three choices ***DataOnly, DataAndMetadata, MetaDataOnly***. In this project, DataAndMetadata was selected to allow for all kind of data to be stored on the NSD. To create the NSD, I used the command *lsblk* to check the available disk and their content. It is important to know the content of the disk because the software will not create a file system on a disk that has a different file system format even if there is free space.

A text file with the necessary NSD parameters; DiskName, NSD Name, Usage Type and Pool, was created. This file will be referenced during the creation of the NSD. The command `mmcrnsd -F nsdfile.txt -v yes` was used to create the NSD file. After the NSD has been created, the file system that will use the NSD needs to be created and the NSD mounted to each file system. The IBM Spectrum Scale has the ca-

pacity to create several file systems. In the project two file systems were created to enable its functionalities to be tested. After the file system was created, the NSD was mounted on the file system, to enable the file system manage the NSD device.

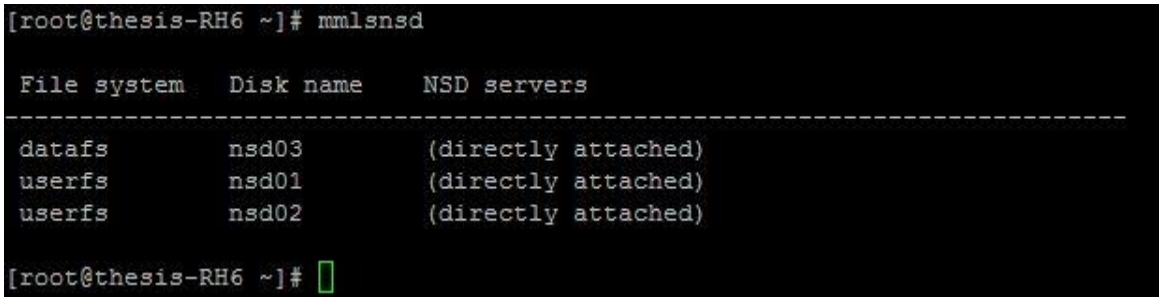

Figure 11 NSD and File System

Figure 11 shows the NSD and file systems that have been created. As a result of this stage, the IBM Spectrum Scale configuration is now complete. With this configuration, the next step is to install IBM Spectrum Archive (LTFS EE) which will be used as an archiving solution.

4.2.3 IBM Spectrum Archive Installation and Configuration

IBM Spectrum Archive is a tape file system; it is not free software, so installing it requires the right license. In this project, I used IBM Spectrum Archive Enterprise Edition version 2.1.3. The enterprise edition was a preferred choice because it has the ability to manage data in a tape environment without additional software. For IBM Spectrum Archive to be installed, it is required that all dependencies are installed and configured ahead. Some of these dependencies include lin_tape as a device driver for tape drives and the IBM Spectrum Scale configured with more than one file system created. The IBM Spectrum Archive needs the IBM Spectrum Scale file system, which allows the dcache to be selected. The file system also need to be Data Management API (DMAPI) enabled for it to be selected as dcache. If no file system in the IBM Spectrum Scale cluster is DMAPI enabled, the IBM Spectrum Archive cannot be configured. The dcache is for the IBM Spectrum Archive metadata to be stored. To verify if DMAPI is enabled in any of the file systems created, I used the command *mmlsfs <filesystem name> -z.* In this project, I enabled DMAPI in the file system name datafs, and once that was completed, then IBM Spectrum Archive will be configured.

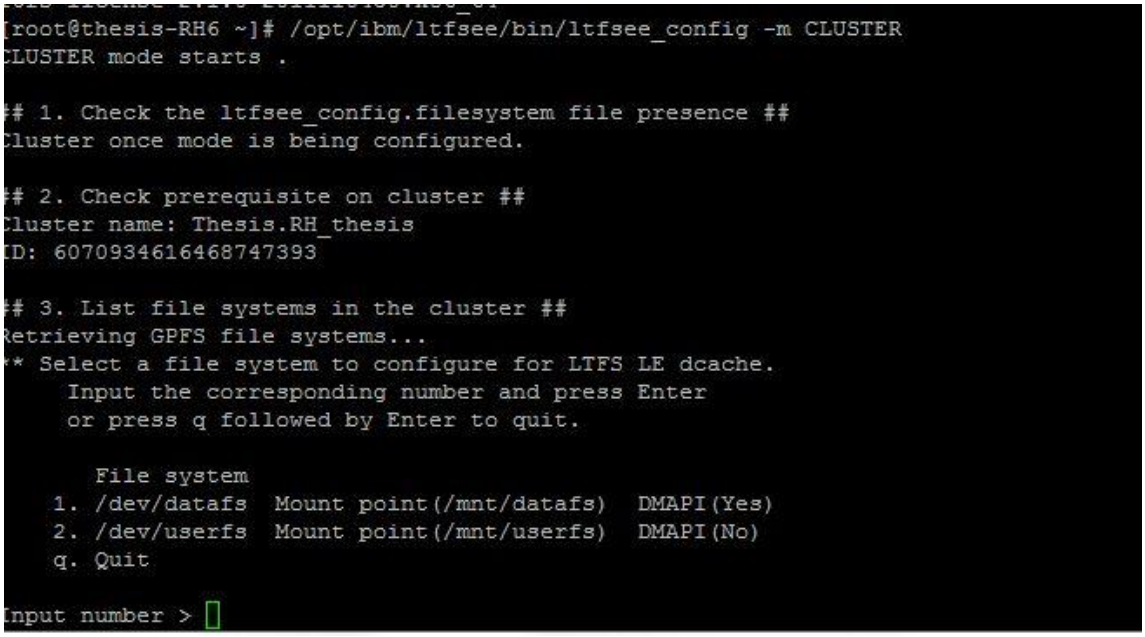

Figure 12 IBM Spectrum Archive configuration 1

Figure 12 shows the steps for configuring IBM Spectrum Archive, after the command /opt/ibm/ltfsee/bin/ltfsee_config – m CLUSTER had been entered. The /opt/ibm/ltfsee/bin/ PATH holds all the relevant commands for configuring IBM Spectrum Archive. For me to easily use the command without the full path name, I added the IBM Spectrum Archive bin directory to the system PATH environmental variable. After the command is entered, it checks the IBM Spectrum Scale cluster to see if it has all the prerequisites needed for configuration. This step also allows me to select the file system that is DMAPI enabled. Once the files system is selected, the IBM Spectrum Archive will add the file system mount point to space management. I also needed to select the tape drive that would be used. After this step, IBM Spectrum Archive configuration was completed and then be started. To start IBM Spectrum Archive, I used the command ***ltfsee start /mnt/datafs.*** The directory /mnt/datafs was the mount point of the file system that was selected during configuration. This mount point contains the ltfsee_config.filesystem file which was the configuration file of the IBM Spectrum Archive. When this step was completed and the IBM Spectrum Archive was running, tape pools were then created prior to data archiving.

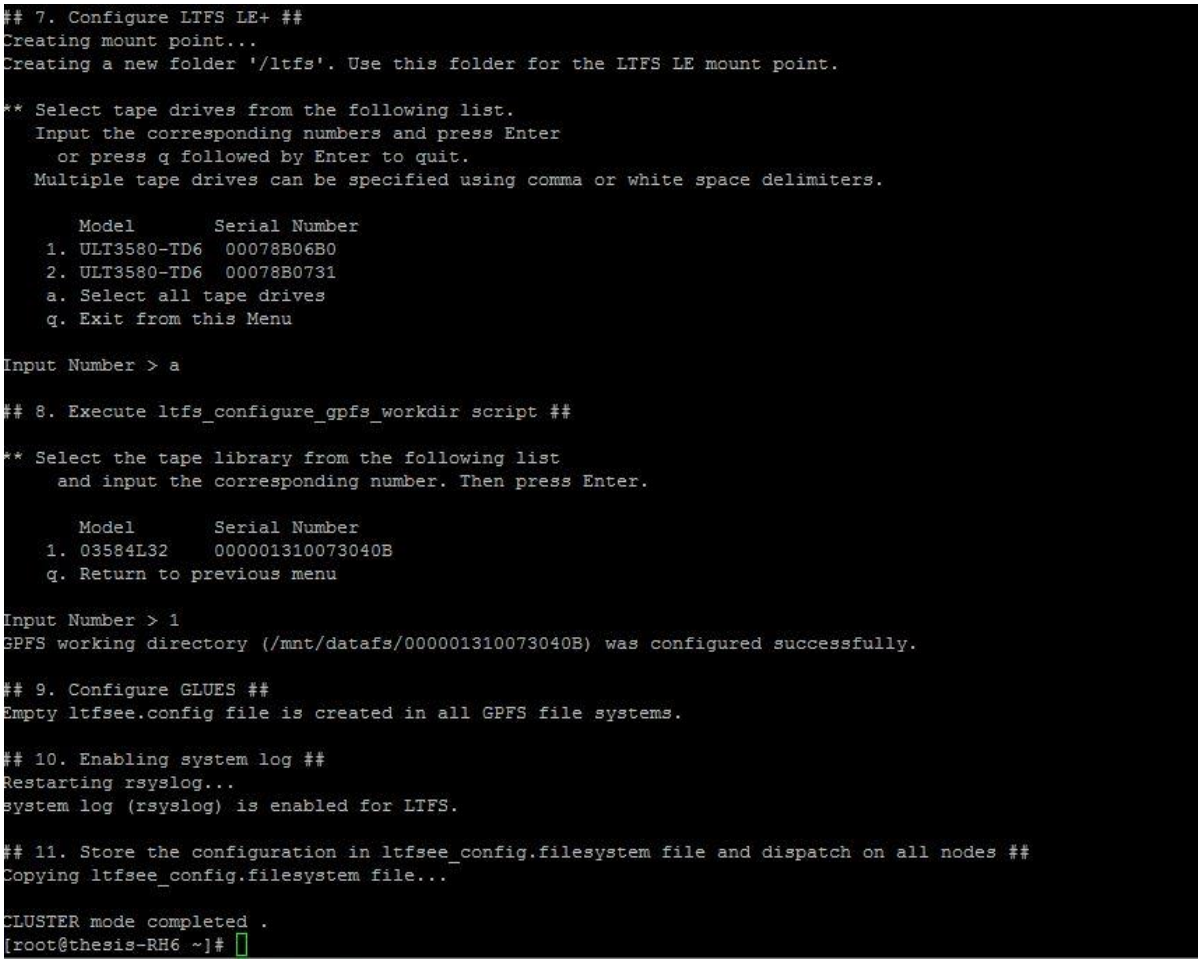

Figure 13 IBM Spectrum Archive configuration 2

Figure 13 shows the configuration as it was explained in the text above. The successful completion of this stage means that the archiving solution had been properly configured in the IBM Spectrum Scale environment. This stage makes it possible to move data from disk to tape without the need for additional software. Data are moved from disk to tape because tapes are less expensive, and therefore, using tapes in the project provide a cost effective data management plan for the use case scenario.

The next stage was to install and configure the other server; this server will be used as the backup server, so it will be configured with the IBM Spectrum Protect, backup solution software.

4.2.4 IBM Spectrum Protect Installation and configuration

IBM Spectrum Protect is software that can be used for both backup and archive, but for this project, I used it for backup functionality only. IBM Spectrum Protect function in a client-server relationship, which means that both an IBM Spectrum Protect client and a

server must be installed and configured. According to the architecture, a second server which will be used as the IBM Spectrum Protect server needs to be installed. In this project, I used IBM Spectrum Protect Server version 7.1.1 which is the latest version. After the basic installation of the software, the server needs to be configured. It is possible for several instances to be created in a single workstation. So a user ID needs to be created. This user ID will own the server instance and will be used to create it. After the user ID is created, the directories that the server instance will need will also be created. Then modify ***/usr/tivoli/tsm/server/bin/dsmserver.opt*** to the parameter needed for the IBM Spectrum Protect to start, because during installation the default configuration is not enough for the IBM Spectrum Protect to start. The following parameters were set for the project.

COMMmethod TCPIP
TCPport 9091
TCPADMINPORT 9091
VOLUMEHistory /tsm/sle1/instance/volhist.out
DEVCONFig /tsm/sle1/instance/devconf.out
ACTIVELOGDirectory /tsm/sle1/log/active
ARCHLOGDirectory /tsm/log/sle1/archive
ARCHFAILOVERLOGDirectory /tsm/log/sle1/failover

After the above parameters had been set, IBM Spectrum Protect would be started from the server instance, with the command **./dsmserver** in the directory **/opt/tivoli/tsm/server/bin/** . When the server is started, it is possible to set the high and low address and the Server name for Client:Server communication. The low address is the TCPport number that was configured in the dsmserver.opt file, and the high address is the IP address of the server. The storage device is then configured. These devices will be used to store the client's data, but before the client's data are stored in the server storage device, the client will be registered on the server, which makes it possible for the server to manage the client's operations and determine which storage device data will be stored on.

set serverlladdress 9091
set serverhladdress <IP Address>
set servername tsmsles1

The next step was to configure the client which will be used to initiate the backup operation to the IBM Spectrum Protect server. In this project the IBM Spectrum Scale node was the client. During the installation of the IBM Spectrum Archive, the IBM Spectrum Protect client version was installed as a prerequisite to install IBM Spectrum Archive, so what was needed was for the **dsm.opt** file and the **dsm.sys** files to be configure to in order to communicate with the server. The dsm.opt file will be a pointer to the dsm.sys file which was the system option file where the options for different stanza can be specified with the parameter of the targeted server.

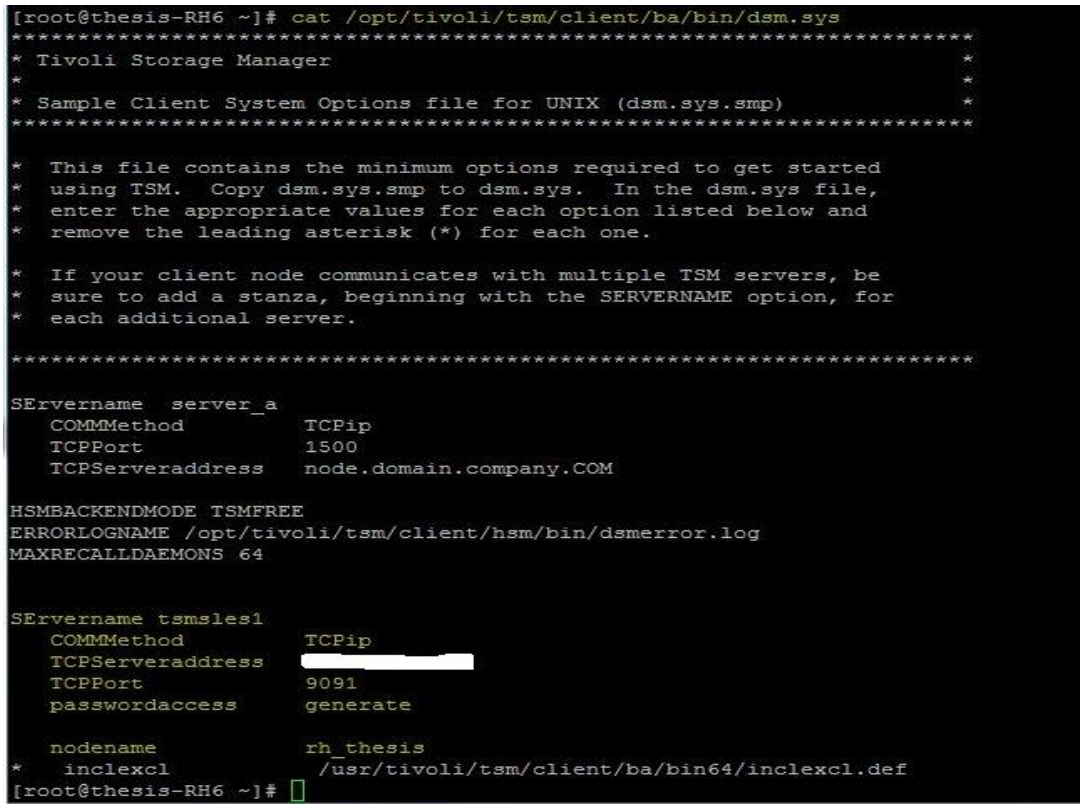

Figure 14 IBM Spectrum Protect Client config

Figure 14 shows the configuration of the dsm.sys file. The IBM Spectrum Protect servername was specified with the high and low level addresses. Also the node name of the IBM Spectrum Scale client was specified because the IBM Spectrum Scale server had already registered this client with this nodename.

```
[root@thesis-RH6 ~]# dsmc -optfile=/opt/tivoli/tsm/client/ba/bin/dsm_sle1.opt
IBM Tivoli Storage Manager
Command Line Backup-Archive Client Interface
  Client Version 7, Release 1, Level 1.3
  Client date/time: 05/14/2015 16:42:51
(c) Copyright by IBM Corporation and other(s) 1990, 2014. All Rights Reserved.

Node Name: RH_THESIS
Session established with server TSMSLES1: AIX
  Server Version 7, Release 1, Level 1.100
  Server date/time: 05/14/2015 17:55:54  Last access: 05/12/2015 16:39:34

tsm>
```

Figure 15 IBM Spectrum Protect client connectivity

In Figure 15 above, a connection between the IBM Spectrum Protect client and server was established; this means that the client can perform a backup operation to the target server. The server was attached to the disk and tape storage devices.

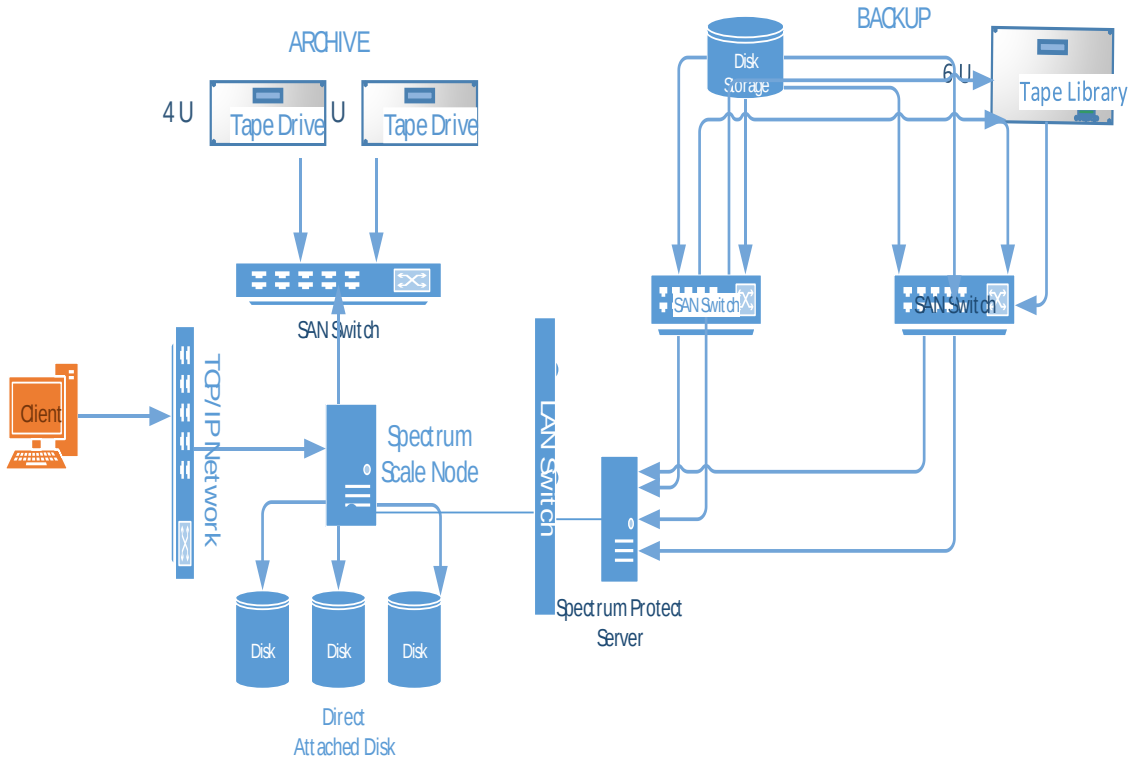

Figure 16 Complete Architecture

Figure 16 shows the complete architecture that was used in this project. In the diagram, backup and archive use two different solutions, although either can perform both tasks. The project specifically focused on using IBM Spectrum Archive for Archiving and IBM Spectrum Protect for Backup while using a scale storage file system, IBM Spectrum Scale. With this architecture, the data that need to be archived will be directly

migrated to tape, while the data that need to be backed up will be first backed up to the IBM Spectrum Protect server before being stored on the storage device. The sucessful completion of this part means that the solution is ready to be implemented. The next chapter will explain the functionalities and test result of the solution.

5 Test of IBM Spectrum Archive and IBM Spectrum Protect Functionality

This chapter will describe the functionality of the IBM Spectrum Archive and IBM Spectrum Protect. The setup shown in Figure 16 provides the possibility to use the archiving solution and the backup solution in the same environment.

During the backup test, the data were sent from the IBM Spectrum Scale node, which is also the IBM Spectrum Protect client. The data travels through a TCP/IP network to the IBM Spectrum Protect server. Inside the IBM Spectrum Protect Server there are storage pools, a random storage pool and a sequential storage pool. The random storage pool uses disk drives (primary storage pool), while the sequential storage pool uses tape drives/media. For disaster recovery a copy pool on tape was also configured. During the backup operation, data are stored in the random storage pool because they are fast, but the IBM Spectrum Protect has the functionality that allows a threshold of storage pool to be set. In the policy used in the storage pool to define the threshold, an alternative storage is defined. When the threshold on the primary storage pool is reached, all data within a given filling level on the pool to migrate will be automatically migrated to the next storage pool. In cases where the backup data coming in would not fit into the space of the primary storage pool, the data will directly be routed to the next specified storage pool.

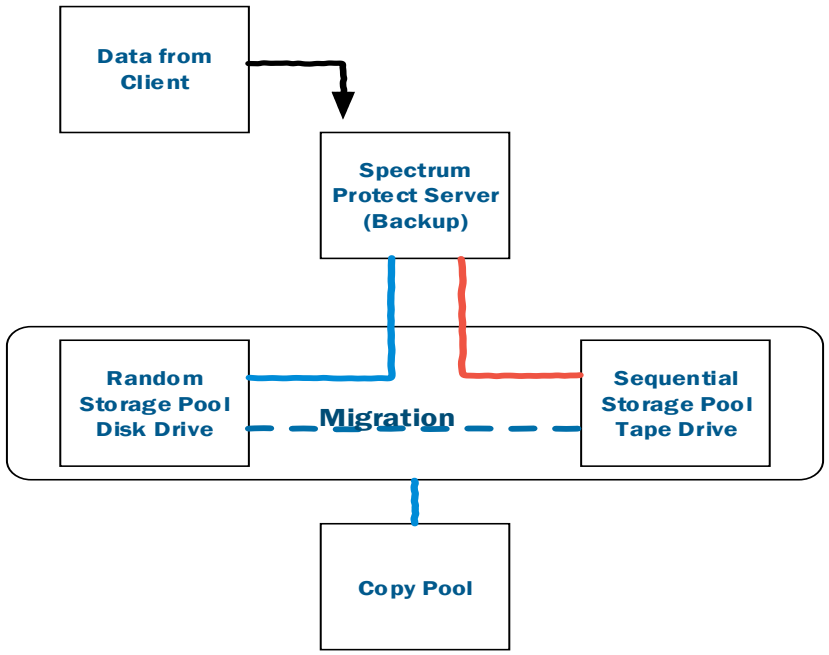

Figure 17 Backup Data Flow

The diagram in figure 17 shows how the data flows from client to server during the backup operation. This data flow is policy-controlled. When data comes into the IBM Spectrum Protect server, it will be stored directly in the disk drive depending on its threshold level. If the threshold level is reached, data will be migrated to tape media. The red line indicates that the threshold level was reached during the backup operation, so the data can be directly stored on the sequential storage pool (tape media).

In the backup test with the IBM Spectrum Protect server all these functionalities were achieved. It also provides the functionality of determining how long inactive data can be stored for before being deleted – the most recent backup version of a file is referred to as the Active version, provided the file still exists in the client. So when a new Active version was stored, IBM Spectrum Protect server would inactivate the previous active version, which will then be inactive. Files were created and deleted to also test the incremental backup functionality. It was also possible to backup a certain file or directory which means that selective backup was possible. Data could be recalled from the server even if it were deleted from the client machine. It is also possible to control the files that need to be backed up using policy. The mechanism for controlling this from the client side was by setting an include-exclude list – the include-exclude list is a set of references local to each client that controls which files are backed up and what management class is used.

In testing the archiving solution using the IBM Spectrum Archive, files were migrated from the local NSD, which is directly attached to the server to the external pool. IBM Spectrum Archive uses the space management functionality to move files from the file system which are on disk to tape media. Migration is only possible if the file system that holds the data in the IBM Spectrum Scale server is added to space management. It is also possible to set a threshold and create a policy that will automatically migrate files from disk to tapes. When data are migrated from disk to tape, no copy of the data is left on the disk, for the data to be used again; it has to be recalled from the tape storage pool back to disk. In Appendix 2, all result of the migration and the process followed are specified.

6 Discussion

The implementation of this project gave me an in-depth understanding of what a data storage system is about, and what it means for future data growth. During the research in the project, I learnt the importance of data backup and archiving, and also why it is important for a storage system to be scalable. I particularly learned more about IBM Spectrum Scale, which provides the functionality of a scalable storage file system, and also IBM Spectrum Archive and IBM Spectrum Protect, which are backup and archiving software.

At the beginning of this project some questions were raised and these question were properly addressed during the implementation.

1. What is required for a storage system to be called Elastic?
   From an IBM perspective, Elastic Storage was a code name for IBM Spectrum Scale. Technically, a system can be called Elastic if it provides the functionality that allows it to grow according to its needs without basically affecting its operation. A system which is scalable and flexible can be called Elastic, and the environment in this project presents such “Elasticity”.

2. What kind of environment will the system be deployed to?
   Organisational needs are always examined before such a system can be deployed. But for the data growth that we are currently experiencing, the system will be very useful for data life-cycle management.

3. How would the system scale?
   In the project, I used IBM Spectrum Scale, a storage file system, which works like a cluster file system, which means multiple nodes can be created and it will be managed by the file system as a single cluster. In the project, it will be possible to add and remove nodes from the cluster and manage storage devices easily.

4. Will it be possible to know if the system fails?
   For this project, a single node cluster was implemented, so even though if the IBM Spectrum Scale server fails, it will not affect the IBM Spectrum Protect

server. Just that data that have not been archived or backed up will be lost in case of failure, but in a real working environment a multi-clustered system will be used which will drastically reduce the possibility of data loss or failure.

5. How can an Elastic Storage System be implemented?
   Chapter 4 of this project explained how this storage system can be implemented.

6. How is backup and archive done with regard to LTFS and TSM?
   Chapter 5 also explains the functionality of IBM Spectrum Archive (LTFS) and IBM Spectrum Protect (TSM.)

This project gave me a good foundation for big data analytics and Big Data as a Service (BDaaS), which are innovative technologies and services for the future.

7 Conclusion

The goal of this project was to implement a scalable storage system for data management with an archiving and backup solution. The task was very interesting and challenging because I was not familiar with most of the technologies used prior to this project. So it required in-depth studying and expert consultations.

I started the project at the beginning of March 2015 and completed all the tasks by the end of May 2015. During this period, I studied about new storage technologies and how they worked. I used the IBM storage portfolio as a reference guide to implementing this project. It was somewhat difficult to understanding how the IBM Spectrum Scale functioned when integrated with the IBM Spectrum Protect and IBM Spectrum Archive because I had no idea about a storage file system before my internship at IBM.

The project was divided into three parts, and all of them were completed. The test results show that the backup and archive functionality of the storage system works correctly. It was possible to migrate data from disk drives to tape media for archiving purposes. It was also possible to send copies of data to the backup server. The environment was scalable because it was possible to add more nodes and storage devices to it.

In implementing this project, I faced some challenges which gave me a better understanding of best the practices to follow when implementing such a storage environment with the software used. I observed that when the IBM Spectrum Protect client is configured in the same node as IBM Spectrum Archive, there is a conflict in the space management functionality. The IBM Spectrum Archive will not be able to add file system that holds data which needs to be migrated to its local space management – this is due to the fact that the dsm.opt file will contain multiple server options that space management will try to connect to. In order to avoid such conflicts, it is better to use multiple nodes and configure them separately.

The project did not involve application development or policy creation for the IBM Spectrum Archive and IBM Spectrum Protect test. Also, the cost of implementing such a project, security steps for data protection, or data analytics was not covered in this project either.

The successful completion of the project gave me the technical skills and knowledge needed in implementing a scalable storage system in any environment. As future technologies move towards data analytics and Big Data as a Service (BDaaS), the skills I have acquired during this project will be useful for my career development.

References

1 Buffalo Technology Brief “Direct Attached Storage vs Network Attached Storage” [Online] 2010 URL:http://www.buffalotech.com/content/files/solutions_articles/DAS_vs_NAS.pdf Accessed: 2nd March 2015

2 Dinshan, He. Data Management in Intelligent Storage Systems. PhD Dissertation: UMI Microform. Michigan: 2006.

3 Tate, Beck, Ibarra, Kumaravel, Miklas. IBM Redbook: Introduction to Storage Area Network. IBM Corp; San Francisco: 2012

4 Tech Target “SAN File System” [Online] 2005. URL: http://searchstorage.techtarget.com/definition/SAN-file-system: Accessed: 2nd March 2015

5 Red Hat Inc. CentOS-5 Project “Chapter 5 Network File System” [Online] 1999. URL: https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-nfs.html Accessed: 6th March 2015

6 SNIA. Technical Proposal. CIFS Technical Reference. [Online] 2002. URL:https://www.thursby.com/sites/default/files/files/CIFS-TR-1p00_FINAL.pdf Accessed: 6th March 2015.

7 CodeFX: Common Internet File System (CIFS) Explained. [Online] 2001. URL:http://www.codefx.com/CIFS_Explained.htm. Accessed: 6th March 2015

8 Jones, Koniges, Yates. Performance of the IBM General Parallel File System. [Online] 2010. URL: https://computing.llnl.gov/code/sio/GPFS_performance.pdf Accessed: 9th March 2015.

9 IBM Redbook. GPFS: Concept, Planning and Installation Guide. IBM Corp. San Francisco: 2013

10 IBM Redbook. IBM Linear Tape File System Enterprise Edition: Installation and Configuration. IBM Corp. San Francisco: 2014.

11 Eurotech Computer Service: IBM Linear Tape File System. [Online]; URL: http://www.eurotech-computers.com/assets/files/Eurotech%20LTFS.pdf Accessed: 14th March 2015.

12 Coyne, Alavari, Browne, Hoffmann, Munoz, Schaefer. IBM Linear Tape File System: Installation and Configuration. IBM Corp. San Francisco. 2013.

13 The Register. Datacenter/Storage “IBM intro Elastic Storage”. [Online] 2014. URL:http://www.theregister.co.uk/2014/05/13/ibms_elastic_storage_lipstick_on_gpfs/m Accessed: 15th March 2015.

14 ESCC: IBM Storage Enablement Roadmap for Europe and MEA. IBM Corp. Mainz: 2015

15 Brien Posey. Data Backup Types Explained: Full, incremental, differential and incremental-forever backup. [Online] 2008. URL: http://searchdatabackup.techtarget.com/tip/Data-backup-types-explained-Full-incremental-differential-and-incremental-forever-backup. Accessed: 26th March 2015

16 Kujur, Chaurasia, S.Singh, P.Singh. Data Management using Hierarchical Storage Management (HSM). [Online] 2008. URL:http://www.spgindia.org/2008/388.pdf. Accessed: 1st April 2015.

17 IBM Redbook. IBM Tivoli Storage Manager Concept. IBM Corp. San Francisco: 2006.

Backup Test (Commands and Output)

```
tsm: TSMSLES1>q node *th*
Session established with server TSMSLES1: AIX
  Server Version 7, Release 1, Level 1.100
  Server date/time: 05/26/15   13:32:37  Last access: 05/26/15   11:13:25

Node Name                Platform     Policy Domain     Days Since     Days Since     Locked?
                            Name           Last Acce-      Password
                                                ss           Set
-------------------------  --------  --------------  -----------  -----------  -------
RH_THESIS                     Linux        DO_STD              <1           14       No
                           x86-64

tsm: TSMSLES1>q occu RH_THESIS

Node Name      Type     Filespace      FSID     Storage         Number of       Physical        Logical
                Name                            Pool Name          Files          Space          Space
                                                                 Occupied      Occupied
                                                                    (MB)          (MB)
-----------    ----    -----------     ----    -----------    -----------    -----------
RH_THESIS      Bkup    /mnt/userfs        3    CM_LTO6                  2       1,250.15       1,250.15

RH_THESIS      Bkup    /mnt/userfs        3    CM_PT                    5      40,004.89      40,004.89

RH_THESIS      Bkup    /mnt/userfs        3    TM_STD                   9      62,507.64      62,507.64

tsm: TSMSLES1>


tsm: TSMSLES1>q stgp dm_std f=d

Storage Pool Name: DM_STD
Storage Pool Type: Primary
Device Class Name: DISK
Estimated Capacity: 11 G
Space Trigger Util: 0.0
Pct Util: 0.0
Pct Migr: 0.0
Pct Logical: 100.0
High Mig Pct: 90
Low Mig Pct: 30
Migration Delay: 0
Migration Continue: Yes
Migration Processes: 2
Reclamation Processes:
Next Storage Pool: TM_STD
Reclaim Storage Pool:
Maximum Size Threshold: No Limit
Access: Read/Write
Description: Standard disk storage pool
Overflow Location:
Cache Migrated Files?: No
Collocate?:
Reclamation Threshold:
Offsite Reclamation Limit:
Maximum Scratch Volumes Allowed:
Number of Scratch Volumes Used:
Delay Period for Volume Reuse:
Migration in Progress?: No
```

Amount Migrated (MB): 9,979.34
Elapsed Migration Time (seconds): 267
Reclamation in Progress?:
Last Update by (administrator): xxxxxxxx
Last Update Date/Time: 12/18/14 11:25:02
Storage Pool Data Format: Native
Copy Storage Pool(s):
Active Data Pool(s):
Continue Copy on Error?: Yes
CRC Data: No
Reclamation Type:
Overwrite Data when Deleted:
Deduplicate Data?: No
Processes For Identifying Duplicates:
Duplicate Data Not Stored:
Auto-copy Mode:
Contains Data Deduplicated by Client?: No

tsm: TSMSLES1>q stgp tm_std f=d

Storage Pool Name: TM_STD
Storage Pool Type: Primary
Device Class Name: DC_LTO_LB1
Estimated Capacity: 18,917 G
Space Trigger Util:
Pct Util: 32.7
Pct Migr: 85.0
Pct Logical: 99.9
High Mig Pct: 90
Low Mig Pct: 70
Migration Delay: 0
Migration Continue: Yes
Migration Processes: 1
Reclamation Processes: 1
Next Storage Pool: TM_PT
Reclaim Storage Pool:
Maximum Size Threshold: No Limit
Access: Read/Write
Description: Standard tape storage pool
Overflow Location:
Cache Migrated Files?:
Collocate?: No
Reclamation Threshold: 100
Offsite Reclamation Limit:
Maximum Scratch Volumes Allowed: 20
Number of Scratch Volumes Used: 17
Delay Period for Volume Reuse: 0 Day(s)
Migration in Progress?: No
Amount Migrated (MB): 0.00
Elapsed Migration Time (seconds): 0
Reclamation in Progress?: No
Last Update by (administrator): xxxxxxxxxx
Last Update Date/Time: 05/20/15 10:18:16
Storage Pool Data Format: Native
Copy Storage Pool(s): CM_LTO6
Active Data Pool(s):
Continue Copy on Error?: Yes
CRC Data: No
Reclamation Type: Threshold
Overwrite Data when Deleted:
Deduplicate Data?: No

Processes For Identifying Duplicates:
Duplicate Data Not Stored:
Auto-copy Mode: Migration
Contains Data Deduplicated by Client?: No

[root@thesis-RH6 ~]# ls -l /mnt/userfs
total 0
drwxr-xr-x. 2 root root 4096 May 26 12:06 000001310073040B
drwx------. 2 root root 4096 May 26 10:49 staff1
drwx------. 2 root root 4096 May 26 10:51 staff2
drwx------. 2 root root 4096 May 26 10:53 staff3
drwx------. 2 root root 4096 May 26 10:46 Student1
drwx------. 2 root root 4096 May 26 10:46 Student2
[root@thesis-RH6 ~]#

[root@thesis-RH6 ~]# ls -l /mnt/userfs/staff1
total 23040000
-rw-r--r--. 1 root root 1310720000 May 26 10:48 data2.pdf
-rw-r--r--. 1 root root 1310720000 May 26 10:48 data3.pdf
-rw-r--r--. 1 root root 20971520000 May 26 10:51 data4.pdf
-rw-r--r--. 1 root root   12 May 26 00:16 data.pdf
[root@thesis-RH6 ~]#

---

select * from backups where node_name='RH_THESIS'

NODE_NAME: RH_THESIS
FILESPACE_NAME: /mnt/userfs
FILESPACE_ID: 3
STATE: ACTIVE_VERSION
TYPE: FILE
HL_NAME: /staff1/
LL_NAME: data4.pdf
OBJECT_ID: 754141987
BACKUP_DATE: 2015-05-26 12:32:16.000000
DEACTIVATE_DATE:
OWNER: root
CLASS_NAME: DEFAULT

---

select * from backups where node_name='RH_THESIS' and FILESPACE_NAME='/mnt/userfs' and HL_NAME='/staff1/' and LL_NAME='data4.pdf'
tsm: TSMSLES1>select * from backups where node_name='RH_THESIS' and FILESPACE_NAME='/mnt/userfs' and HL_NAME='/staff1/' and LL_NAME='data4.pdf'

NODE_NAME: RH_THESIS
FILESPACE_NAME: /mnt/userfs
FILESPACE_ID: 3
STATE: ACTIVE_VERSION
TYPE: FILE
HL_NAME: /staff1/
LL_NAME: data4.pdf
OBJECT_ID: 754141987
BACKUP_DATE: 2015-05-26 12:32:16.000000

```
DEACTIVATE_DATE:
OWNER: root
CLASS_NAME: DEFAULT

select * from backups where node_name='RH_THESIS' and FILESPACE_NAME='/mnt/userfs' and
HL_NAME='/staff1/' and LL_NAME='data4.pdf'

----------------------
----------------------
----------------------
new file on /mnt/userfs/staff1
testdata.txt

 dsmc i /mnt/userfs/staff1/* -optfile=/opt/tivoli/tsm/client/ba/bin/dsm_sle1.opt


 [root@thesis-RH6 staff1]#  dsmc i /mnt/userfs/staff1/* -optfile=/opt/tivoli/tsm/client/ba/bin/dsm_sle1.opt
IBM Tivoli Storage Manager
Command Line Backup-Archive Client Interface
  Client Version 7, Release 1, Level 1.3
  Client date/time: 05/26/2015 12:39:29
(c) Copyright by IBM Corporation and other(s) 1990, 2014. All Rights Reserved.

Node Name: RH_THESIS
Session established with server TSMSLES1: AIX
  Server Version 7, Release 1, Level 1.100
  Server date/time: 05/26/2015 13:52:51  Last access: 05/26/2015 12:30:12


Incremental backup of volume '/mnt/userfs/staff1/data2.pdf'

Incremental backup of volume '/mnt/userfs/staff1/data3.pdf'

Incremental backup of volume '/mnt/userfs/staff1/data4.pdf'

Incremental backup of volume '/mnt/userfs/staff1/data.pdf'

Incremental backup of volume '/mnt/userfs/staff1/dsmerror.log'

Incremental backup of volume '/mnt/userfs/staff1/testdata.txt'
Normal File-->                25 /mnt/userfs/staff1/testdata.txt [Sent]
Successful incremental backup of '/mnt/userfs/staff1/testdata.txt'

Successful incremental backup of '/mnt/userfs/staff1/data2.pdf'

Successful incremental backup of '/mnt/userfs/staff1/data3.pdf'

Successful incremental backup of '/mnt/userfs/staff1/data4.pdf'

Successful incremental backup of '/mnt/userfs/staff1/data.pdf'

Normal File-->               195 /mnt/userfs/staff1/dsmerror.log [Sent]
Successful incremental backup of '/mnt/userfs/staff1/dsmerror.log'


Total number of objects inspected:            6
Total number of objects backed up:            2
Total number of objects updated:              0
Total number of objects rebound:              0
Total number of objects deleted:              0
Total number of objects expired:              0
Total number of objects failed:               0
Total number of objects encrypted:            0
Total number of objects grew:                 0
```

```
Total number of retries:                0
Total number of bytes inspected:        21.97 GB
Total number of bytes transferred:      902  B
Data transfer time:                     0.00 sec
Network data transfer rate:             440,429.68 KB/sec
Aggregate data transfer rate:           0.28 KB/sec
Objects compressed by:                  0%
Total data reduction ratio:             100.00%
Elapsed processing time:                00:00:03
[root@thesis-RH6 staff1]#
```



```
select * from backups where node_name='RH_THESIS' and FILESPACE_NAME='/mnt/userfs' and
HL_NAME='/staff1/' and LL_NAME='testdata.txt'
```

```
tsm: TSMSLES1>select * from backups where node_name='RH_THESIS' and
FILESPACE_NAME='/mnt/userfs' and HL_NAME='/staff1/' and LL_NAME='testdata.txt'

NODE_NAME: RH_THESIS
FILESPACE_NAME: /mnt/userfs
FILESPACE_ID: 3
STATE: ACTIVE_VERSION
TYPE: FILE
HL_NAME: /staff1/
LL_NAME: testdata.txt
OBJECT_ID: 754142069
BACKUP_DATE: 2015-05-26 13:52:55.000000
DEACTIVATE_DATE:
OWNER: root
CLASS_NAME: DEFAULT
```

---

modify our test file

```
[root@thesis-RH6 staff1]# vi testdata.txt
[root@thesis-RH6 staff1]# cat testdata.txt
test
test
test
test
test
newly added
newly added
newly addrd
[root@thesis-RH6 staff1]#
```

```
[root@thesis-RH6 staff1]#  dsmc i /mnt/userfs/staff1/* -optfile=/opt/tivoli/tsm/client/ba/bin/dsm_sle1.opt
IBM Tivoli Storage Manager
Command Line Backup-Archive Client Interface
  Client Version 7, Release 1, Level 1.3
  Client date/time: 05/26/2015 12:42:19
(c) Copyright by IBM Corporation and other(s) 1990, 2014. All Rights Reserved.
```

```
Node Name: RH_THESIS
Session established with server TSMSLES1: AIX
  Server Version 7, Release 1, Level 1.100
  Server date/time: 05/26/2015 13:55:40  Last access: 05/26/2015 13:52:55


Incremental backup of volume '/mnt/userfs/staff1/data2.pdf'

Incremental backup of volume '/mnt/userfs/staff1/data3.pdf'

Incremental backup of volume '/mnt/userfs/staff1/data4.pdf'

Incremental backup of volume '/mnt/userfs/staff1/data.pdf'

Incremental backup of volume '/mnt/userfs/staff1/dsmerror.log'

Incremental backup of volume '/mnt/userfs/staff1/testdata.txt'
Normal File-->              61 /mnt/userfs/staff1/testdata.txt [Sent]
Successful incremental backup of '/mnt/userfs/staff1/testdata.txt'

Successful incremental backup of '/mnt/userfs/staff1/data2.pdf'

Successful incremental backup of '/mnt/userfs/staff1/data3.pdf'

Successful incremental backup of '/mnt/userfs/staff1/data4.pdf'

Successful incremental backup of '/mnt/userfs/staff1/data.pdf'

Successful incremental backup of '/mnt/userfs/staff1/dsmerror.log'


Total number of objects inspected:            6
Total number of objects backed up:            1
Total number of objects updated:              0
Total number of objects rebound:              0
Total number of objects deleted:              0
Total number of objects expired:              0
Total number of objects failed:               0
Total number of objects encrypted:            0
Total number of objects grew:                 0
Total number of retries:                      0
Total number of bytes inspected:         21.97 GB
Total number of bytes transferred:         402  B
Data transfer time:                     0.00 sec
Network data transfer rate:          392,578.12 KB/sec
Aggregate data transfer rate:             0.12 KB/sec
Objects compressed by:                       0%
Total data reduction ratio:                100.00%
Elapsed processing time:                   00:00:03
[root@thesis-RH6 staff1]#



tsm: TSMSLES1>select * from backups where node_name='RH_THESIS' and
FILESPACE_NAME='/mnt/userfs' and HL_NAME='/staff1/' and LL_NAME='testdata.txt'

NODE_NAME: RH_THESIS
FILESPACE_NAME: /mnt/userfs
FILESPACE_ID: 3
STATE: ACTIVE_VERSION
TYPE: FILE
HL_NAME: /staff1/
LL_NAME: testdata.txt
```

```
OBJECT_ID: 754142071
BACKUP_DATE: 2015-05-26 13:55:44.000000
DEACTIVATE_DATE:
OWNER: root
CLASS_NAME: DEFAULT

NODE_NAME: RH_THESIS
FILESPACE_NAME: /mnt/userfs
FILESPACE_ID: 3
STATE: INACTIVE_VERSION
TYPE: FILE
HL_NAME: /staff1/
LL_NAME: testdata.txt
OBJECT_ID: 754142069
BACKUP_DATE: 2015-05-26 13:52:55.000000
DEACTIVATE_DATE: 2015-05-26 13:55:44.000000
OWNER: root
CLASS_NAME: DEFAULT


tsm: TSMSLES1>
----
-----

[root@thesis-RH6 staff1]# vi testdata.txt
[root@thesis-RH6 staff1]# cat testdata.txt
test
test
test
test
test
newly added
newly added
newly addrd
once more added
[root@thesis-RH6 staff1]#


[root@thesis-RH6 staff1]#  dsmc i /mnt/userfs/staff1/* -optfile=/opt/tivoli/tsm/client/ba/bin/dsm_sle1.opt
IBM Tivoli Storage Manager
Command Line Backup-Archive Client Interface
  Client Version 7, Release 1, Level 1.3
  Client date/time: 05/26/2015 12:43:53
(c) Copyright by IBM Corporation and other(s) 1990, 2014. All Rights Reserved.

Node Name: RH_THESIS
Session established with server TSMSLES1: AIX
  Server Version 7, Release 1, Level 1.100
  Server date/time: 05/26/2015 13:57:15  Last access: 05/26/2015 13:55:44


Incremental backup of volume '/mnt/userfs/staff1/data2.pdf'

Incremental backup of volume '/mnt/userfs/staff1/data3.pdf'

Incremental backup of volume '/mnt/userfs/staff1/data4.pdf'

Incremental backup of volume '/mnt/userfs/staff1/data.pdf'

Incremental backup of volume '/mnt/userfs/staff1/dsmerror.log'

Incremental backup of volume '/mnt/userfs/staff1/testdata.txt'
Normal File-->              77 /mnt/userfs/staff1/testdata.txt [Sent]
Successful incremental backup of '/mnt/userfs/staff1/testdata.txt'
```

```
Successful incremental backup of '/mnt/userfs/staff1/data2.pdf'

Successful incremental backup of '/mnt/userfs/staff1/data3.pdf'

Successful incremental backup of '/mnt/userfs/staff1/data4.pdf'

Successful incremental backup of '/mnt/userfs/staff1/data.pdf'

Successful incremental backup of '/mnt/userfs/staff1/dsmerror.log'


Total number of objects inspected:           6
Total number of objects backed up:           1
Total number of objects updated:             0
Total number of objects rebound:             0
Total number of objects deleted:             0
Total number of objects expired:             0
Total number of objects failed:              0
Total number of objects encrypted:           0
Total number of objects grew:                0
Total number of retries:                     0
Total number of bytes inspected:         21.97 GB
Total number of bytes transferred:         418  B
Data transfer time:                    0.00 sec
Network data transfer rate:            0.00 KB/sec
Aggregate data transfer rate:          0.13 KB/sec
Objects compressed by:                    0%
Total data reduction ratio:            100.00%
Elapsed processing time:               00:00:03
[root@thesis-RH6 staff1]#



tsm: TSMSLES1>select * from backups where node_name='RH_THESIS' and
FILESPACE_NAME='/mnt/userfs' and HL_NAME='/staff1/' and LL_NAME='testdata.txt'

NODE_NAME: RH_THESIS
FILESPACE_NAME: /mnt/userfs
FILESPACE_ID: 3
STATE: ACTIVE_VERSION
TYPE: FILE
HL_NAME: /staff1/
LL_NAME: testdata.txt
OBJECT_ID: 754142072
BACKUP_DATE: 2015-05-26 13:57:18.000000
DEACTIVATE_DATE:
OWNER: root
CLASS_NAME: DEFAULT

NODE_NAME: RH_THESIS
FILESPACE_NAME: /mnt/userfs
FILESPACE_ID: 3
STATE: INACTIVE_VERSION
TYPE: FILE
HL_NAME: /staff1/
LL_NAME: testdata.txt
OBJECT_ID: 754142069
BACKUP_DATE: 2015-05-26 13:52:55.000000
DEACTIVATE_DATE: 2015-05-26 13:55:44.000000
OWNER: root
CLASS_NAME: DEFAULT
```

```
NODE_NAME: RH_THESIS
FILESPACE_NAME: /mnt/userfs
FILESPACE_ID: 3
STATE: INACTIVE_VERSION
TYPE: FILE
HL_NAME: /staff1/
LL_NAME: testdata.txt
OBJECT_ID: 754142071
BACKUP_DATE: 2015-05-26 13:55:44.000000
DEACTIVATE_DATE: 2015-05-26 13:57:18.000000
OWNER: root
CLASS_NAME: DEFAULT

----------------------
----------------------
-------------------
[root@thesis-RH6 staff1]# rm testdata.txt
rm: remove regular file `testdata.txt'? y
[root@thesis-RH6 staff1]#

[root@thesis-RH6 staff1]#  dsmc i /mnt/userfs/staff1/ -optfile=/opt/tivoli/tsm/client/ba/bin/dsm_sle1.opt
IBM Tivoli Storage Manager
Command Line Backup-Archive Client Interface
  Client Version 7, Release 1, Level 1.3
  Client date/time: 05/26/2015 12:52:40
(c) Copyright by IBM Corporation and other(s) 1990, 2014. All Rights Reserved.

Node Name: RH_THESIS
Session established with server TSMSLES1: AIX
  Server Version 7, Release 1, Level 1.100
  Server date/time: 05/26/2015 14:06:01  Last access: 05/26/2015 14:00:33


Incremental backup of volume '/mnt/userfs/staff1/'
Directory-->              4,096 /mnt/userfs/staff1 [Sent]
Expiring-->                  77 /mnt/userfs/staff1/testdata.txt [Sent]
Successful incremental backup of '/mnt/userfs/staff1/*'


Total number of objects inspected:            7
Total number of objects backed up:            1
Total number of objects updated:              0
Total number of objects rebound:              0
Total number of objects deleted:              0
Total number of objects expired:              1
Total number of objects failed:               0
Total number of objects encrypted:            0
Total number of objects grew:                 0
Total number of retries:                      0
Total number of bytes inspected:         21.97 GB
Total number of bytes transferred:            0  B
Data transfer time:                     0.00 sec
Network data transfer rate:             0.00 KB/sec
Aggregate data transfer rate:           0.00 KB/sec
Objects compressed by:                     0%
Total data reduction ratio:              100.00%
Elapsed processing time:                 00:00:01
[root@thesis-RH6 staff1]#


tsm: TSMSLES1>select * from backups where node_name='RH_THESIS' and
FILESPACE_NAME='/mnt/userfs' and HL_NAME='/staff1/' and LL_NAME='testdata.txt'

     NODE_NAME: RH_THESIS
```

FILESPACE_NAME: /mnt/userfs
FILESPACE_ID: 3
STATE: INACTIVE_VERSION
TYPE: FILE
HL_NAME: /staff1/
LL_NAME: testdata.txt
OBJECT_ID: 754142069
BACKUP_DATE: 2015-05-26 13:52:55.000000
DEACTIVATE_DATE: 2015-05-26 13:55:44.000000
OWNER: root
CLASS_NAME: DEFAULT

NODE_NAME: RH_THESIS
FILESPACE_NAME: /mnt/userfs
FILESPACE_ID: 3
STATE: INACTIVE_VERSION
TYPE: FILE
HL_NAME: /staff1/
LL_NAME: testdata.txt
OBJECT_ID: 754142071
BACKUP_DATE: 2015-05-26 13:55:44.000000
DEACTIVATE_DATE: 2015-05-26 13:57:18.000000
OWNER: root
CLASS_NAME: DEFAULT

NODE_NAME: RH_THESIS
FILESPACE_NAME: /mnt/userfs
FILESPACE_ID: 3
STATE: INACTIVE_VERSION
TYPE: FILE
HL_NAME: /staff1/
LL_NAME: testdata.txt
OBJECT_ID: 754142072
BACKUP_DATE: 2015-05-26 13:57:18.000000
DEACTIVATE_DATE: 2015-05-26 14:06:04.000000
OWNER: root
CLASS_NAME: DEFAULT

tsm: TSMSLES1>expire inventory
ANS8003I Process number 187 started.

05/26/15 14:03:15 ANR0984I Process 187 for EXPIRE INVENTORY started in the BACKGROUND at 14:03:15. (SESSION: 6520, PROCESS: 187)
05/26/15 14:03:21 ANR0985I Process 187 for EXPIRE INVENTORY running in the BACKGROUND completed with completion state SUCCESS at 14:03:21. (SESSION: 6520, PROCESS: 187)
05/26/15 14:06:04 ANE4970I (Session: 6544, Node: RH_THESIS) Total number of objects expired: 1 (SESSION: 6544)

tsm: TSMSLES1>q copy DO_STD PO_STD MC_STD

Policy Domain Name	Policy Set Name	Mgmt Class Name	Copy Group Name	Versions Data Exists	Versions Data Deleted	Retain Extra Versions	Retain Only Version
DO_STD	PO_STD	MC_STD	STANDARD	No Limit	No Limit	30	60

Archiving Test (command and outputs)

IBM Spectrum Scale Test

Before Migration

[root@thesis-RH6 ~]# mmdf datafs

disk name	disk size in KB	failure group	holds metadata	holds data	free KB in full blocks	free KB in fragments
Disks in storage pool: system (Maximum disk size allowed is 1.1 TB)						
nsd01	143373984	1	Yes	Yes	64342016 (45%)	2496 (0%)
nsd02	143373984	2	Yes	Yes	64344064 (45%)	2720 (0%)
(pool total)	286747968				128686080 (45%)	5216 (0%)
(total)	286747968				128686080 (45%)	5216 (0%)

Inode Information

Number of used inodes: 4060
Number of free inodes: 276004
Number of allocated inodes: 280064
Maximum number of inodes: 280064

For Migration

[root@thesis-RH6 ~]# cat migration_list.txt
/mnt/datafs/Student1/data3.pdf
Student

/mnt/datafs/Student2/data1.pdf
Student

/mnt/datafs/Staff1/records3.pdf
Staff

/mnt/datafs/Staff2/records2.pdf
Staff

/mnt/datafs/Staff3/records1.pdf

Staff

/mnt/datafs/Student3/data3.pdf
Student

[root@thesis-RH6 ~]# ltfsee migrate migration_list.txt
GLESL316E(01605): File /mnt/datafs/Staff1/records3.pdf does not exist.
GLESL316E(01605): File /mnt/datafs/Staff3/records1.pdf does not exist.
GLESL167I(01990): A list of files to be migrated has been sent to LTFS EE using scan id 3707961351.
GLESL159E(02037): Not all migration has been successful.
GLESL038I(02043): Migration result: 4 succeeded, 0 failed, 0 duplicate, 0 duplicate wrong pool, 2 not found, 0 too small to qualify for migration

---
During Migration

[root@thesis-RH6 ~]# mmlsfileset datafs
Filesets in file system 'datafs':
Name Status Path
root Linked /mnt/datafs
Student1 Linked /mnt/datafs/Student1
Student2 Linked /mnt/datafs/Student2
Student3 Linked /mnt/datafs/Student3
Staff1 Linked /mnt/datafs/Staff1
Staff2 Linked /mnt/datafs/Staff2
Staff3 Linked /mnt/datafs/Staff3
[root@thesis-RH6 ~]# ls -la /mnt/datafs/Staff3
total 8192256
drwx------ 2 root root 4096 May 28 10:56 .
drwxr-xr-x 9 root root 262144 May 28 10:27 ..
-rw-r--r-- 1 root root 4194304000 May 28 10:56 records2.pdf
-rw-r--r-- 1 root root 4194304000 May 28 10:56 records3.pdf
[root@thesis-RH6 ~]# ltfsee info jobs
job scan id pool node id tape id status not changed (sec) file name or i-node
Migration 3707961351 Student 1 SLE075L6 in progress 71 /mnt/datafs/Student1/data3.pdf
Migration 3707961351 Student 1 SLE076L6 in progress 70 /mnt/datafs/Student2/data1.pdf
Migration 3707961351 Staff - - unscheduled 118 /mnt/datafs/Staff2/records2.pdf
Migration 3707961351 Student - - unscheduled 117 /mnt/datafs/Student3/data3.pdf

```
[root@thesis-RH6 ~]#

--------------------------------
After Migration
--------------------------------
[root@thesis-RH6 ~]# ltfsee info pools
  stgp name  remaining capacity  unref space  list of tapes
    Student             6727GB          0GB  SLE075L6 SLE076L6 SLE077L6
      Staff             4485GB          0GB  SLE078L6 SLE079L6
[root@thesis-RH6 ~]# mmdf datafs
disk               disk size  failure holds    holds              free KB             free KB
name                   in KB    group metadata data        in full blocks        in fragments
---------------- ------------- -------- -------- ----- -------------------- -------------------
Disks in storage pool: system (Maximum disk size allowed is 1.1 TB)
nsd01              143373984        1 Yes      Yes       112470016 ( 78%)         2496 ( 0%)
nsd02              143373984        2 Yes      Yes       112472064 ( 78%)         2720 ( 0%)
              -------------                    -------------------- -------------------
(pool total)       286747968                      224942080 ( 78%)         5216 ( 0%)

              =============                    ==================== ====================
(total)            286747968                      224942080 ( 78%)         5216 ( 0%)

Inode Information
-----------------
Number of used inodes:            4062
Number of free inodes:          276002
Number of allocated inodes:     280064
Maximum number of inodes:       280064.
```