UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ
TECNÓLOGO EM REDES DE COMPUTADORES

REJANE VASCONCELOS LOPES

AVALIAÇÃO DE SISTEMAS DE ARQUIVOS DISTRIBUÍDOS EM UM AMBIENTE DE NUVEM

QUIXADÁ
2016

REJANE VASCONCELOS LOPES

AVALIAÇÃO DE SISTEMAS DE ARQUIVOS DISTRIBUÍDOS EM UM AMBIENTE DE NUVEM

Trabalho de Conclusão de Curso submetido à Coordenação do Curso Tecnólogo em Redes de Computadores da Universidade Federal do Ceará como requisito parcial para obtenção do grau de Tecnólogo.

Área de concentração: Computação

Orientador Profº. Me. Alisson Barbosa de Souza
Coorientador Profº. Me. Paulo Antônio Leal Rêgo

QUIXADÁ
2016

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca do Campus de Quixadá

L855a Lopes, Rejane Vasconcelos
Avaliação de sistemas de arquivos distribuídos em um ambiente de nuvem/ Rejane Vasconcelos Lopes. – 2016.
53 f.: il. color., enc.; 30 cm.

Monografia (graduação) – Universidade Federal do Ceará, Campus Quixadá, Curso de Tecnologia em Redes de Computadores, Quixadá, 2016.
Orientação: Prof. Me. Alisson Barbosa Sousa
Coorientação: Prof. Me. Paulo Antônio Leal Rego
Área de concentração: Computação

1. Virtualização. 2. Computação em nuvem . 3. Armazenamento de dados. I. Universidade Federal do Ceará (Campus Quixadá). II. Título.

CDD 004.36

REJANE VASCONCELOS LOPES

AVALIAÇÃO DE SISTEMAS DE ARQUIVOS DISTRIBUÍDOS EM UM AMBIENTE DE NUVEM

Trabalho de Conclusão de Curso submetido à Coordenação do Curso Tecnólogo em Redes de Computadores da Universidade Federal do Ceará como requisito parcial para obtenção do grau de Tecnólogo.

Área de concentração: Computação

Aprovado em: _____ / fevereiro / 2016.

BANCA EXAMINADORA

_____
Profª. Me. Alisson Barbosa de Souza (Orientador)
Universidade Federal do Ceará-UFC

_____
Prof. Me. Antonio Rafael Braga
Universidade Federal do Ceará-UFC

_____
Prof. Me. Michel Sales Bonfim
Universidade Federal do Ceará-UFC

_____
Prof. Me. Paulo Antonio Leal Rego (Coorientador)
Universidade Federal do Ceará-UFC

À minha mãe, Regina Lúcia Rodrigues Vasconcelos, e ao meu pai, Francisco Esmerino Lopes Neto, que sempre priorizaram oferecer educação apesar das dificuldades, e às minhas irmãs por ajudarem sempre que foi necessário.

AGRADECIMENTOS

Agradeço à minha família, principalmente meus pais, minhas irmãs e minha avó, por acreditarem em mim e me ajudarem nesta importante conquista na minha vida.

Ao Professor Alisson Barbosa, pela orientação neste estudo, e ao Professor Paulo Antônio, por me auxiliar desde o início dessa pesquisa.

A todas as professoras e a todos os professores, que foram indispensáveis para a minha formação.

A todas as minhas amigas e a todos os meus amigos, que tive oportunidade de conviver e compartilhar ótimos momentos durante o período em que morei em Quixadá.

Em especial a Liliane Costa, que me acolheu como alguém da sua família durante quase três anos.

"Feliz daquele que transfere
o que sabe e aprende
o que ensina"
(Cora Coralina)

RESUMO

Os sistemas de arquivos distribuídos são comumente utilizados em organizações que trabalham com infraestruturas virtualizadas, através da virtualização de armazenamento. Cada sistema de arquivo possui características específicas que são exploradas por diferentes tipos de interesses. Este trabalho apresenta uma avaliação de desempenho dos sistemas de arquivo distribuídos Ceph, GlusterFS, e NFS, utilizando uma infraestrutura de nuvem computacional com a plataforma OpenNebula, abordando operações comuns para este tipo infraestrutura. Os resultados foram obtidos com a ferramenta IOZone e *scripts* desenvolvidos na execução deste trabalho. Os resultados mostram o GlusterFS e Ceph possuem alto nível de replicação de dados e o NFS se destaca em oferecer consistência de dados.

Palavras chave: Computação em Nuvem. Sistema de Arquivo Distribuído. Avaliação de Desempenho.

ABSTRACT

Distributed file systems are commonly used in organizations working with virtualized infrastructure through storage virtualization. Each file system that has specific characteristics which are exploited by different interests. This paper presents a performance evaluation of distributed file systems Ceph, GlusterFS, NFS and using a cloud computing infrastructure with OpenNebula platform, addressing common operations for such infrastructure. The results were obtained with the IOzone tool and scripts developed in the execution of this work. The results show the GlusterFS and Ceph have high-level data replication and NFS stands out in offering data consistency.

Keywords: Cloud Computing. Distributed File System. Performance Evaluation.

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1 INTRODUÇÃO

1.1 Introdução

O conceito de Computação em Nuvem vem sendo adotado pelas organizações como uma solução nos setores de Tecnologia da Informação (TI). Pois acredita-se que através dessa abordagem, os investimentos em recursos de TI são melhor aproveitados. (CARISSIMI, 2008).

Umas das maiores motivações para utilizar Computação em Nuvem é a redução dos custos com infraestruturas de TI e com manutenção de equipamentos, reduzindo assim o espaço físico. De acordo com Sousa, Moreira e Machado (2009), a Computação em Nuvem é uma tendência de tecnologia que pretende prover serviços em massa tanto para usuários finais, que desejam armazenar seus dados pessoais em um servidor na Internet, quanto para empresas que terceirizam serviços de infraestrutura de TI. Ela é um paradigma que possibilita o pagamento por demanda dos serviços utilizados, gerando economia para os usuários.

Um estudo que analisou o uso da Computação em Nuvem em empresas brasileiras mostrou que os serviços mais predominantes são e-mail corporativo, hospedagem, armazenamento de dados e planilhas e processamento de texto (RAMALHO, 2012).

Diante dessas novas perspectivas de inovações tecnológicas, a indústria de TI está cada vez mais avançando em soluções de *hardware* de alto desempenho. Com isso, surge a necessidade de haver um melhor aproveitamento desses recursos, tendo a virtualização como solução da Computação em Nuvem que provê uma melhor distribuição dos serviços que estão sendo executados (CARISSIMI, 2008).

A virtualização é uma tecnologia que provê a abstração das características físicas de uma plataforma computacional através de um *hardware* virtual, que dá suporte ao usuário acessar um determinado recurso, arquivo, sistema operacional, etc., sem a necessidade de estar no mesmo computador que este serviço se encontra (SOUSA, MOREIRA E MACHADO, 2009). Existem várias formas de prover virtualização, por exemplo, virtualização de *desktops*, virtualização de redes, virtualização de servidores e a virtualização de armazenamento (REGO, 2012). E é através da virtualização de armazenamento que é possível fazer o uso de sistemas de arquivos distribuídos para prover uma abstração de armazenamento lógico e físico.

Atualmente, existem vários tipos de sistemas de arquivos distribuídos que proporcionam diferentes soluções para o armazenamento distribuído em massa, tendo em vista que ao utilizar um servidor de arquivos distribuído na Internet ou na própria empresa, o espaço físico, utilizado pela infraestrutura computacional, sofre uma diminuição e, consequentemente, os gastos com manutenção da infraestrutura também diminuem. Estes sistemas geralmente contam com um sistema de replicação de dados, que dá garantia que os mesmos vão estar disponíveis, independente de qualquer falha que vier a ocorrer no sistema, pois permitir que o sistema suporte várias cópias de um arquivo em diferentes discos de armazenamento é uma das características de um sistema de arquivos distribuído (COULOURIS, 2007).

Sistemas de arquivos distribuídos possuem características que devem ser avaliadas para proporcionarem as melhores formas de acesso e de armazenamento em determinados cenários. Os sistemas de arquivos distribuídos abordados nesse trabalho são o NFS, o GlusterFS e o Ceph, pois são comumente utilizados para implantação de nuvens computacionais do tipo infraestrutura como serviço (IaaS) (OPENNEBULA, 2016).

O objetivo desse trabalho é verificar o comportamento desses sistemas de arquivos distribuídos em diferentes cenários comuns em ambientes virtualizados e avaliar o desempenho desses sistemas na nuvem computacional criada com o OpenNebula[1], uma solução de código aberto, simples e flexível para o gerenciamento de infraestruturas virtualizadas de *datacenters* ou *clusters*, que dá suporte a implementação de diferentes sistemas de arquivos distribuídos (OPENNEBULA, 2016). A avaliação se dá através da técnica de experimentação, utilizando o *benchmark* IOZone[2] e *scripts* desenvolvidos durante a execução do projeto.

1.2 Objetivo Geral

O objetivo deste trabalho é realizar uma avaliação entre os sistemas de arquivos distribuídos Ceph, GlusterFS e NFS ao executar tarefas típicas de uma infraestrutura de nuvem.

1.3 Objetivos Específicos

- Planejar e realizar experimentos em cada sistema de arquivos distribuído.
- Analisar os resultados para cada sistema de arquivo e métrica.

[1] http://opennebula.org/
[2] http://www.iozone.org/

- Identificar o sistema de arquivo que tem melhor desempenho para atividades específicas de um ambiente de nuvem do tipo infraestrutura como serviço (IaaS).

1.4 Organização do Trabalho

Este trabalho está organizado da seguinte forma: a Seção 2 apresenta os conceitos utilizados neste trabalho. A Seção 3 apresenta os trabalhos relacionados importantes para o direcionamento desta avaliação. A Seção 4 está relacionada aos procedimentos seguidos para a realização da avaliação. A Seção 5 apresenta o desenvolvimento da avaliação e a análise dos resultados coletado. E a Seção 6 apresenta as conclusões obtidas da avaliação realizada com este trabalho.

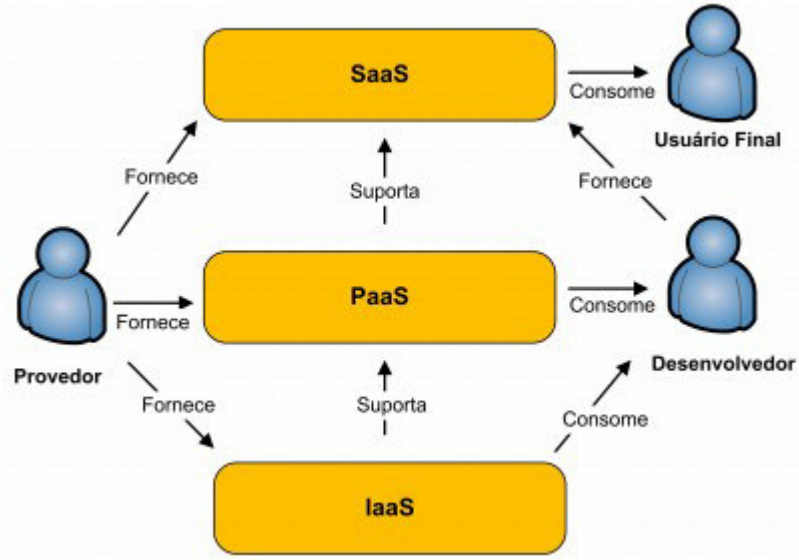
2 FUNDAMENTAÇÃO TEÓRICA

2.1 Computação em Nuvem e Virtualização

O grande avanço tecnológico e o grande número de serviços utilizados pelas organizações impulsionam cada vez mais o desenvolvimento de recursos de alto desempenho. A Computação em Nuvem permite que novas tecnologias sejam utilizadas de forma inteligente, baseada em uma arquitetura que oculta a complexidade da infraestrutura (TAURION, 2009).

Segundo Taurion (2009), Computação em Nuvem possui algumas características próprias, que definem o seu conjunto de tecnologias, como a abstração da infraestrutura que necessita de recursos administrativos e gerenciais para prover autonomia de gestão dos sistemas distribuídos geograficamente; a heterogeneidade, pois uma nuvem não pode exigir que os sistemas sejam de uma mesma tecnologia; a escalabilidade, que permite o crescimento da nuvem dependendo da demanda dos sistemas; e a adaptabilidade, onde entra a virtualização dos recursos computacionais, onde um sistema ou uma nuvem computacional permanence disponível mesmo se um recurso falhar.

Figura 1– Camadas do modelo de Computação em Nuvem



Fonte: SOUSA, MOREIRA E MACHADO (2009)

Como mostra a Figura 1, o modelo de Computação em Nuvem pode ser dividido em três camadas de arquiteturas:

- SaaS (*Sofware as a Service*): responsável por prover o serviço de *software*, onde são executadas as aplicações acessadas de qualquer dispositivo. É a camada mais visível para o consumidor, que não tem conhecimento das camadas abaixo, logo não gerencia nem controla serviços como rede, sistemas operacionais e armazenamento (MARTINS, 2010);

- PaaS (*Plataform as a Service*): camada que fornece infraestrutura para implantação e teste de aplicações. Auxilia a implementação de *softwares* com o fornecimento de um sistema operacional, linguagens de programação, um ambiente e ferramentas de desenvolvimento. Na PaaS, o usuário não é autorizado a controlar ou administrar os serviços da camada abaixo, como rede, servidores, sistemas operacionais e armazenamento (SOUSA; MOREIRA; MACHADO, 2009);

- E a camada IaaS (*Infrastructure as a Service*) que provê processamento, armazenamento, rede e outros serviços que possibilitam ao cliente implantar e executar qualquer tipo de *software* (MARTINS, 2010). Nesta camada, geralmente, o usuário não possui permissão de controle e administração da infraestrutura da nuvem, porém possui controle sobre sistemas operacionais, armazenamento,

aplicativos implantados e podendo eventualmente selecionar componentes de rede (REGO, 2012).

Além dos Modelos de Serviços, Computação em Nuvem é composta por Modelos de Implantação, que se trata dos diferentes modelos que definem o acesso e a disponibilização dos recursos de um ambiente computacional, tendo em vista a necessidade de restringir o acesso de determinados recursos apenas para usuários devidamente autorizados (SOUSA; MOREIRA; MACHADO, 2009). Segundo NIST (2016), podem ser divididos em quatro modelos:

- Privado: uma organização tem acesso exclusivo à infraestrutura, podendo ser uma nuvem local ou remota e os serviços possuem uma política de acesso (SOUSA; MOREIRA; MACHADO, 2009).
- Público: a infraestrutura pode ser acessada pelo público em geral através do serviço de pagamento pelo uso. Empresas que possuem grande capacidade de armazenamento e processamento oferecem esse tipo de serviço (VERDI, *et al.*, 2010).
- Comunidade: a nuvem é compartilhada por um grupo de consumidores que tem interesses em comum, podendo existir local e remotamente (SOUSA; MOREIRA; MACHADO, 2009).
- Híbrido: composto por duas ou mais infraestruturas de nuvens, que continuam sendo entidades únicas, mas conectadas por uma tecnologia proprietária ou padronizada (SOUSA; MOREIRA; MACHADO, 2009).

A virtualização permite que um servidor mantenha diversos tipos de sistemas executando em uma mesma máquina, como sistemas operacionais, sistemas de armazenamento e sistemas de banco de dados. Um ambiente virtualizado provê benefícios para uma empresa, pois reduz custos e complexidades, faz um melhor aproveitamento dos recursos de infraestrutura, melhora a segurança e a confiabilidade através da camada de abstração entre as aplicações e o *hardware* (CARISSIMI, 2008).

A IBM foi uma das empresas pioneiras a abordar o conceito de virtualização, ao possibilitar a criação de máquinas virtuais que operassem em seus *mainframes* e dessem suporte às aplicações que só executavam nos sistemas operacionais em que foram implementadas. Naquela época era comum que os *mainframes* tivessem seus próprios sistemas operacionais, por isso a importância do surgimento das máquinas virtuais. Máquina virtual é uma camada de *software* que possibilita a execução de sistemas operacionais e

aplicações em um ambiente bastante similar ao de uma máquina física (CARISSIMI, 2008). Esta solução possibilita que em um mesmo *hardware* sejam executadas várias instâncias de máquinas virtuais, diminuindo os custos com a infraestrutura física, como espaço, energia elétrica, cabeamento, refrigeração, suporte e manutenção a vários sistemas, pois há uma diminuição de máquinas físicas, segundo Carissimi (2008).

Existem várias formas de prover virtualização, por exemplo (REGO, 2012):

- Virtualização de *desktops*: um tipo de virtualização onde o usuário final possui uma configuração de *desktop* em uma infraestrutura virtual, onde as aplicações e os sistemas operacionais são executados em um *datacenter*, na forma de uma máquina virtual, podendo acessar seu ambiente de trabalho de diferentes locais através de qualquer estação de trabalho conectada a rede;
- Virtualização de redes: nesse tipo de virtualização ocorre um compartilhamento da rede física através da criação de ambientes de rede separados, onde a rede é vista para o usuário final como sendo uma rede própria, com recursos dedicados e políticas de segurança independentes;
- Virtualização de servidores: esse tipo de virtualização permite que vários ambientes virtuais sejam executados em uma mesma máquina física através do particionamento desta máquina em várias máquinas virtuais;
- E a virtualização de armazenamento: esse tipo de virtualização se refere à abstração de armazenamento lógico e físico, onde diversos discos físicos são vistos como um conjunto homogêneo de recursos de armazenamento.

Este trabalho abordará o Modelo de Serviço IaaS, por permitir um maior controle sobre a infraestrutura, especificamente sobre o armazenamento, possibilitando assim a instalação de uma nuvem privada, para executar os experimentos de análise dos sistemas de arquivos distribuídos.

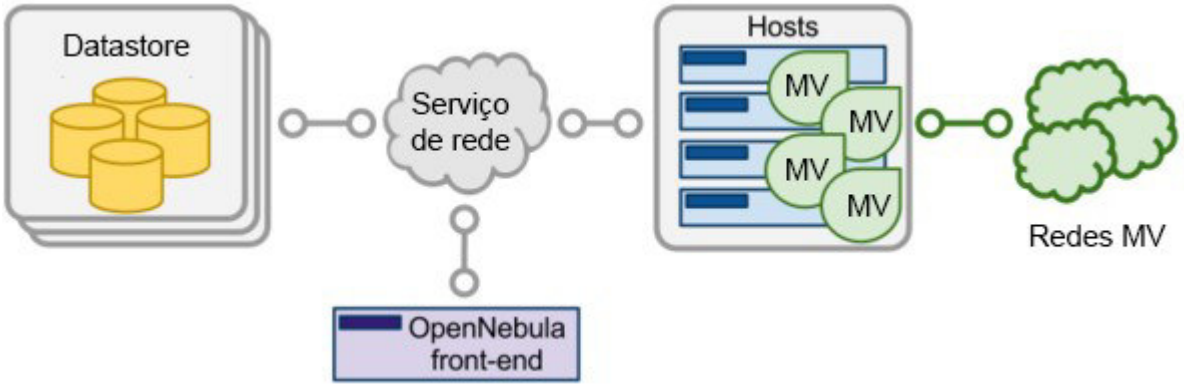
2.2 OpenNebula

Para fazer o gerenciamento do ambiente virtual proposto neste projeto será utilizado o OpenNebula, que é uma solução de código aberto, simples e flexível, que dá suporte a utilização de diferentes sistemas de arquivos distribuídos (OPENNEBULA, 2016).

O OpenNebula visa fornecer uma solução de alto nível de escalabilidade, adaptável para o gerenciamento de dados virtualizados, oferece componentes para personalizar o serviço de nuvem e dá suporte para a construção de nuvens privadas e híbridas.

É uma solução totalmente de código aberto, possui ferramentas que tornam a administração da nuvem simples e fácil de manter, é flexível, podendo adaptar a nuvem aos requisitos de um projeto, e permite o crescimento da nuvem com facilidade (OPENNEBULA, 2016).

Figura 2 - Visão geral da arquitetura OpenNebula



Fonte: OPENNEBULA (2016)

Como pode ser observado da Figura 2, a sua arquitetura possui quatro componentes básicos: o *Front-end*, os *hosts*, o *Datastore* e as redes. O *Front-end* é responsável pelos serviços e administração da infraestrutura de nuvem e possui conexão com cada *host* e com o *Datastore*. Os *hosts* fornecem os recursos para a execução das máquinas virtuais. O *Datastore* é um meio de armazenamento podendo ser usado para armazenar imagens de discos de máquinas virtuais e para o armazenamento de dados do sistema, onde são armazenadas as máquinas virtuais em execução (OPENNEBULA, 2016). E as redes são componentes facilmente adaptáveis e personalizáveis, necessárias para conectar o *Front-end* com os *hosts* e o *Datastore*, e para conectar as máquinas virtuais aos *hosts*.

A escolha deste *software* se deu também pelo fato de já existir uma nuvem computacional na Universidade Federal do Ceará, Campus Quixadá, com o OpenNebula fazendo o gerenciamento do compartilhamento dos recursos sobre infraestruturas distribuídas através de máquinas virtuais. E a avaliação deste trabalho pode auxiliar no melhoramento da nuvem computacional do campus.

2.3 Sistemas de Arquivos Distribuídos

Sistemas de arquivos distribuídos permitem, a partir de qualquer computador que esteja conectado à Internet ou a uma intranet, o acesso a arquivos e armazenamentos de um ou mais discos numa rede de computadores, exatamente como se estivesse em um disco da própria máquina que está fazendo as requisições. Com um servidor de arquivos distribuído a

necessidade de armazenamento em disco local é reduzida e economias poderão ser observadas no gerenciamento e no arquivamento de dados de uma organização (COULOURIS, 2007).

Os sistemas de arquivos assumem a responsabilidade de organizar, armazenar, recuperar, atribuir nomes, compartilhar e proteger os arquivos, portanto devem atender aos seguintes requisitos (COULOURIS, 2007):

- Transparência: o sistema deve ocultar a divisão dos componentes do sistema permitindo que o usuário final o perceba como um todo. Pode ser dividida em transparências de acesso, de localização, de mobilidade, de desempenho e de mudança de escala;
- Atualizações concorrentes de arquivos: deve haver o controle de concorrência enquanto um arquivo está sendo alterado e as atualizações não devem interferir no acesso a esse arquivo por outros clientes;
- Replicação de arquivos: permite que o sistema suporte várias cópias de um arquivo em diferentes discos de armazenamento;
- Heterogeneidade do *hardware* e do sistema operacional: os sistemas devem suportar diferentes implementações de *software* tanto do cliente como do servidor;
- Tolerância a falhas: o sistema deve continuar disponível enquanto falhas são detectadas e tratadas, por exemplo, reenviando uma mensagem perdida, replicando os arquivos em outros discos, etc.;
- Consistência: o conteúdo de um arquivo é o mesmo para todos que estão o acessando e após alguma alteração desse arquivo, as atualizações devem ser realizadas em todas as suas cópias;
- Segurança: os sistemas de arquivos distribuídos podem prover a segurança através de mecanismos de controle de acesso, como autenticação das requisições do cliente para proteger o conteúdo das mensagens; e
- Eficiência: o serviço de arquivo distribuído deve ter um nível de desempenho e confiabilidade comparado ao serviço de arquivo local.

Ao atender esses requisitos para um ambiente distribuído, os sistemas de arquivos podem ter um desempenho melhor em algumas dessas características, o que pode refletir negativamente em outras. Atualmente, existem vários sistemas de arquivos distribuídos, e cada um deles possui características específicas. A seguir serão apresentados os sistemas de arquivos NFS, GlusterFS e Ceph, que serão utilizados nesse trabalho.

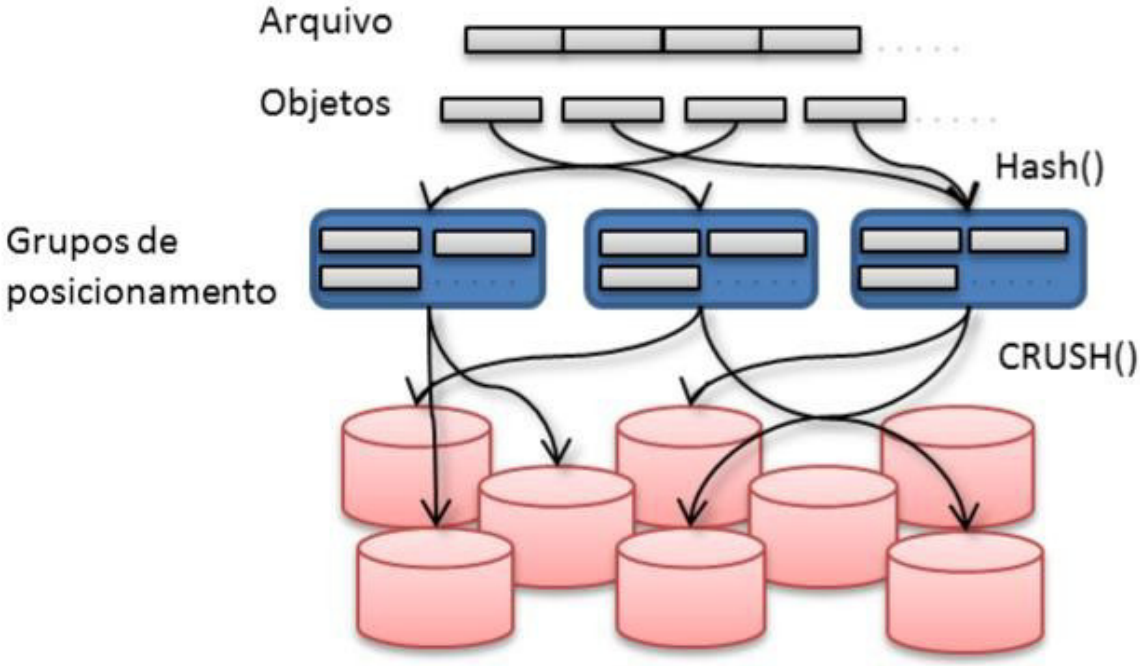
2.3.1 Ceph

Ceph é um sistema de arquivo distribuído orientado a objetos, desenvolvido pela Universidade da Califórnia no ano de 2006. Sua arquitetura é destina para ambientes que envolvem milhares de máquinas compartilhando um mesmo arquivo. Um dos principais objetivos do Ceph é prover alta escalabilidade, confiabilidade e eficiência, através do acesso aos servidores de metadados (*Metadata Servers* - MSD), que são os primeiros nós a serem consultados pelo cliente, responsáveis por manter informações do arquivo, como nomes, permissões, datas de acesso e as localizações dos blocos dos dados no sistema, e repassar o caminho do arquivo. Os dados e metadados são armazenados em dispositivos de armazenamento de objetos (*Object Storage Deamon* - OSD), sendo um aglomerado de OSD's responsável por replicar, detectar falhas, restaurar os dados armazenados e prover as principais funções de E/S (SOARES, 2012).

O Ceph utiliza uma função de distribuição de dados chamada CRUSH (*Controlled Replication Under Scalable Hashing*) para atribuir os arquivos dividido em objetos a dispositivos de armazenamento. Foi desenvolvido para ambientes de larga escala e permite que mais de cem servidores de metadados sejam gerenciados utilizando o conceito de árvore hierárquica, segundo Soares (2012).

O acesso ao sistema pelo cliente se dá a partir de uma interface POSIX, ou seja, a leitura deve refletir os dados gravados anteriormente e a escrita deve ser atômica. A localização dos arquivos é fornecida pelo MDS, que recebe as informações sobre o mapeamento dos objetos que compõem os arquivos repassadas pelos OSDs. A distribuição dos dados ocorre com o mapeamento dos objetos em grupos de posicionamento (*Placement Groups* – PG) e são atribuídos aos OSDs através da função CRUSH, que mapeia cada PG em uma lista ordenada de OSDs, afim de armazenar réplicas dos objetos. Qualquer componente do sistema pode localizar qualquer objeto de forma independente, pois a função CRUSH requer apenas o PG e um mapa de *cluster* de OSD para localizar objetos (SOARES, 2012). A Figura 3 ilustra o funcionamento da distribuição dos dados com a função CRUSH.

Figura 3 – Distribuição dos dados no sistema Ceph

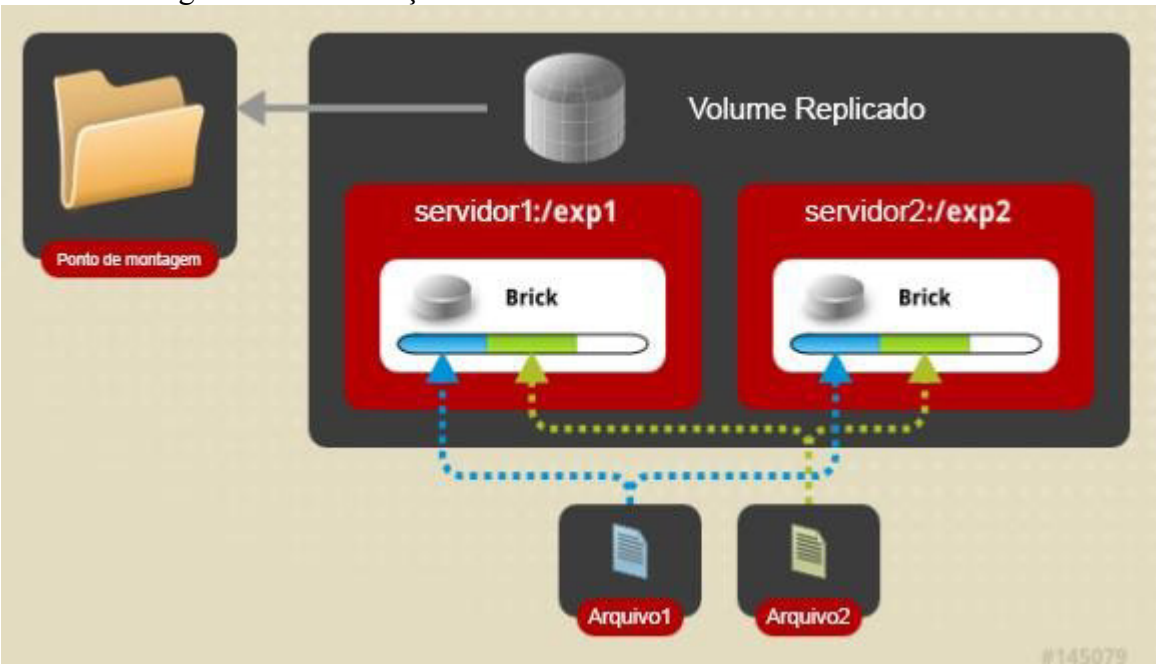


Fonte: SOARES (2012)

2.3.2 GlusterFS

Este sistema de arquivo distribuído de código aberto, desenvolvido pela empresa Z *Research*, permite aos seus usuários um alto nível de escalabilidade, performance e alto desempenho. O armazenamento é configurado em dois ou mais nós, onde apenas os clientes acessam os nós de armazenamento, permitindo que o lado cliente modifique a forma de acesso ao sistema de arquivo, aumentando a flexibilidade do sistema (BOUFLEUR *et al.*, 2007).

Figura 4 – Distribuição do armazenamento GlusterFS



Fonte: GLUSTERFS (2016)

Alguns dos termos utilizados na documentação do GlusterFS são: *trusted pool*, o termo utilizado para definir um conjunto de nós; *brick*, refere-se ao espaço utilizado para o armazenamento GlusterFS, como mostra a Figura 4; Gluster volume, uma coleção de um ou mais blocos, onde os dados são distribuídos uniformemente entre as *bricks* disponíveis (GLUSTERFS, 2016).

De acordo com Boufleur *et al.* (2007), os clientes têm acesso ao sistema ao montarem os diretórios compartilhados. A implementação do sistema é realizada no espaço do usuário do sistema operacional, utilizando o módulo FUSE (*Filesystem in Userspace*), proporcionando maior flexibilidade ao administrador, por não precisar de privilégios especiais para carregar o sistema.

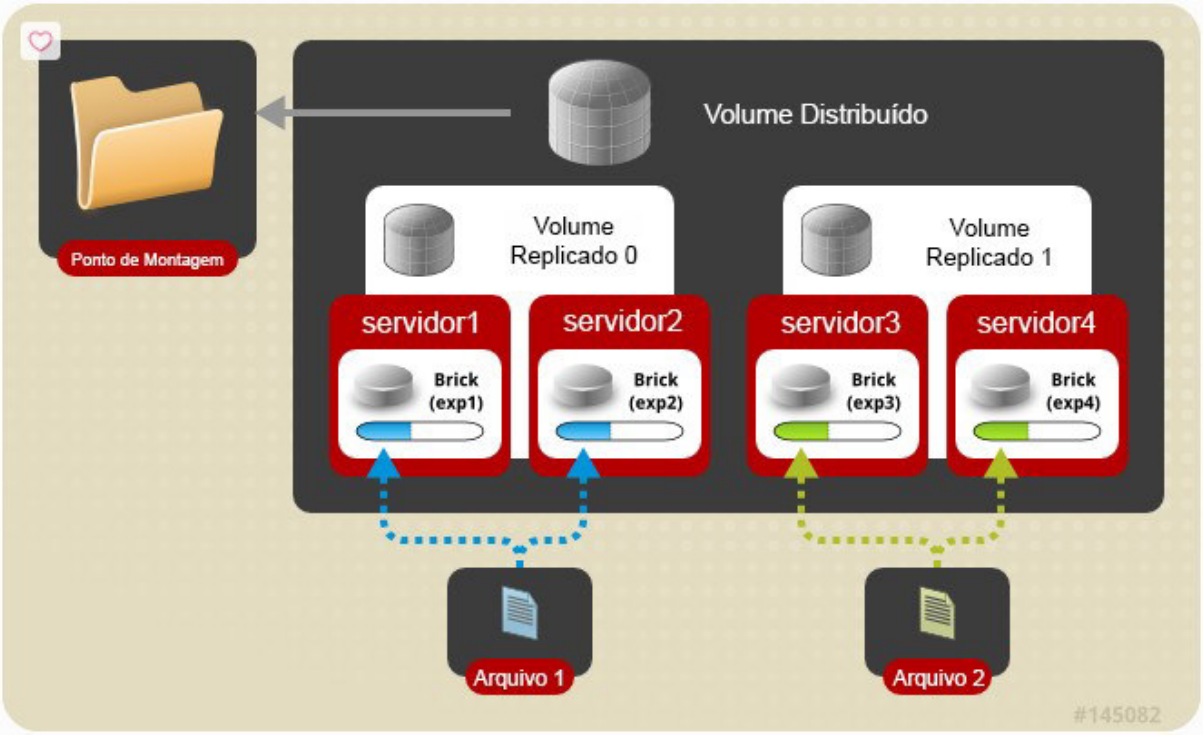
O GlusterFS possui um componente chamado de tradutores (*Translators*), que são responsáveis por converter as solicitações dos usuários para pedidos de armazenamentos. Os tradutores podem ser divididos em várias categorias, mas os principais são os tradutores de *cluster* e tradutores de desempenho. Os tradutores de *cluster* são o DHT (*Distributed Hash Table*), que é responsável por atribuir um arquivo em um dos subvolumes (*bricks*) e o AFR (*Automatic File Replication*), que é responsável por replicar os dados em *bricks*, manter os dados em ambos os *bricks*, mesmo em casos que estejam ocorrendo operações nos dados, e fornecer uma forma de recuperação de dados em caso de falhas. E os tradutores de desempenho são: *io-cache*, *io-threads*, *md-cache*, O-B (*open-behind*), QR (*quick read*), r-a

(*read-ahead*) e o w-b (*write-behind*) (GLUSTERFS, 2016), que são responsáveis por ajustar a carga de trabalho de operações de Entrada/Saída (E/S).

O GlusterFS disponibiliza diferentes soluções de armazenamento, cada uma com uma utilidade específica, alguns são destinados para um armazenamento escalável, alguns para melhorar o desempenho e outros para ambos (GLUSTERFS, 2016). Os diferentes tipos de volumes serão apresentados a seguir.

- Volume distribuído: é o volume padrão que GlusterFS cria se não for especificado qual o tipo de volume. Se volume for composto por dois *bricks*, um arquivo será armazenado em um deles, não em ambos. Não há redundância de dados. O objetivo deste volume é dimensionar de forma fácil e barata o seu tamanho.
- Volume replicado: este volume mantém cópias exatas dos dados em todos os *bricks*. Durante a criação do volume, é possível decidir o número de réplicas que o volume terá. É utilizado para melhor confiabilidade e redundância de dados, pois são necessários no mínimo dois *bricks* para haver redundância. Este trabalho utilizará essa abordagem de volume do GlusterFS.
- Volume distribuído replicado: como na Figura 4, os arquivos são distribuídos em conjuntos replicados de *bricks*, por exemplo, um arquivo é replicado em um conjunto de *bricks* e outro arquivo é replicado em outro conjunto de *bricks*. É recomendado quando há uma exigência de alta disponibilidade, redundância e escalonamento.

Figura 5 – Volume distribuído replicado



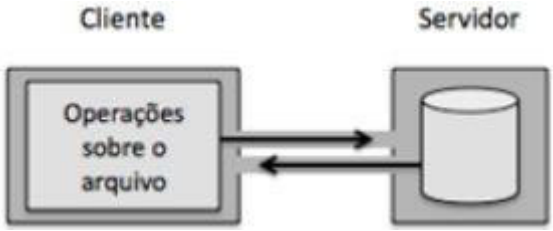
Fonte: GLUSTERFS (2016)

- Volume *striped*: neste volume os dados são divididos em tiras e armazenados nos *bricks*. Usado para armazenar arquivos grandes e acessados frequentemente por vários clientes ao mesmo tempo. Mas não oferece redundância.
- Volume *striped* distribuído: é semelhante ao volume *striped*. Neste volume as tiras são distribuídas nos *bricks*. O número de *bricks* deve ser um múltiplo no número de distribuições.

2.3.3 NFS

O NFS é um sistema de arquivo distribuído centralizado desenvolvido pela Sun Microsystems, de código aberto e amplamente utilizado na indústria e nos ambientes acadêmicos (COULOURIS, 2007).

Figura 6 – Modelo de acesso remoto



Fonte: MARTINS (2011)

Ele oferece transparência no acesso remoto a arquivos, como está ilustrado pela Figura 5, onde cada computador que esteja conectado a uma rede NFS pode se comportar como cliente e servidor, ou seja, qualquer máquina pode exportar alguns de seus arquivos e pode acessar arquivos de outras máquinas. Entretanto, é comum que em uma rede algumas máquinas são configuradas para atuarem como servidores dedicados e outras como estações de trabalho, de acordo com Coulouris (2007). Hoje, ele está na quarta versão (NFSv4), é implementado na maioria dos sistemas Unix e é considerado um dos principais sistemas de arquivos distribuídos pelo fato de permite que vários clientes possam acessar um dispositivo de armazenamento NFS.

O NFS, em sua quarta versão, dá suporte para funcionar sem informações de estado, onde o servidor não tem conhecimento dos arquivos abertos pelos clientes, e dá suporte para que cliente e servidor saibam quais operações estão sendo realizadas em determinados arquivos. Isso auxilia o trabalho do *cache*, que pode ser agregado a outras técnicas, como a *sync*, onde ocorre a cada 30 segundos a cópia dos blocos de arquivos para o disco (COULOURIS, 2007). No lado servidor, são armazenados blocos de arquivos e diretórios, e os atributos dos arquivos lidos no disco, em *cache* (MARTINS, 2011).

A transparência de acesso do NFS permite que os programas de usuários executem operações de arquivo, em arquivos locais ou remotos. Este sistema de arquivo funciona como o sistema de arquivo convencional do UNIX, transferindo blocos de arquivos para o servidor e colocando os blocos em *cache* na memória local. Essa característica, segundo Coulouris, resulta no melhoramento do seu desempenho.

No servidor, o arquivo /etc/exports contém o nome do sistema de arquivos e uma lista de acesso associada ao nome do sistema de arquivos, indicando quais máquinas podem ter acesso à ele (COULOURIS, 2007).

2.4 Avaliação/IOZone

Atualmente, se faz cada vez mais necessária a realização de avaliação de desempenho em sistemas computacionais, pois é através dessa avaliação, por exemplo, que se pode verificar a eficiência de uma determinada funcionalidade de um sistema, identificar quais as restrições que um determinado produto possui e possíveis aprimoramentos, avaliar se determinada aplicação possui um desempenho aceitável no sistema e se o sistema é adequado às necessidades de desempenho com menor custo (MACHADO, 2008). De acordo com

Johnson e Margalho (2011), existem basicamente três técnicas de avaliação de desempenho: a modelagem, a simulação, e experimentação ou aferição, esta técnica é comumente utilizada para gerar resultados mais precisos na avaliação de desempenho de um sistema, usando a carga de trabalho em uso atual. Uma das formas de se utilizar experimentação ou aferição é através de *softwares benchmarks*, que são usados para coletar os dados, apresentar os resultados obtidos e compará-los com as medições anteriormente realizadas e computadas em um banco de dados para servir como referência do que foi mensurado.

Para que o estudo de avaliação inicie é necessária a escolha de métricas, que são usadas para quantificar o desempenho do sistema, ou seja, o que vai ser avaliado e comparado (JOHNSON; MARGALHO, 2011). No caso deste trabalho, que propõe uma avaliação de sistemas de arquivos distribuídos em um ambiente de nuvem, foi escolhida a experimentação ou aferição como técnica de avaliação de desempenho.

Para mensurar operações de Entrada/Saída (E/S), como leitura, escrita, re-leitura, re-escrita de arquivos, foi escolhido o *benchmark* IOZone, uma ferramenta destinada para sistema de arquivos locais e distribuídos e suporta vários sistemas operacionais (IOZONE, 2016). E para medir operações comuns em ambientes virtualizados serão utilizados *scripts* desenvolvidos durante a execução do projeto para realizar a coleta de dados na avaliação dos sistemas de arquivos distribuídos, baseado na métrica de tempo de criação e tempo de remoção de dez máquinas virtuais.

O IOZone é uma ferramenta de código aberto que realiza operações para analisar a taxa de transferência da unidade de armazenamento de um sistema de arquivo e executa suas operações com arquivos temporários (MARON *et al.*, 2014). Este trabalho utilizará esta ferramenta para coletar as taxas de transferências nas operações de escrita, reescrita, leitura e releitura.

A operação de escrita mede o desempenho para escrever um novo arquivo e a criação das informações de controle do arquivo (os metadados), o que pode tornar o desempenho desta operação inferior a operação de reescrita. O teste de reescrita mede o desempenho de gravação de um arquivo que já existe. Leitura mede o desempenho para leitura de um arquivo já existente. E releitura mede o desempenho de leitura de um arquivo que foi lido recentemente, sendo normal o desempenho mais elevado, pois o sistema operacional geralmente mantém um *cache* dos dados desse arquivo (IOZONE, 2016).

De acordo com IOzone (2016), as operações são executadas através da linha de comando no terminal do Linux, nelas são definidas quais operações serão executadas, o tamanho do arquivo e o tamanho do registro que o usuário deseja executar. Os tamanhos de

arquivos variam entre 64KB e 512MB, os tamanhos de registros variam entre 4KB e 16MB, na opção de modo automático e as taxas de transferência são medidas em KB/s. A documentação do IOZone apresenta alguns resultados que a ferramenta pode gerar e mostra que o *cache* pode ajudar nas operações de E/S.

3 TRABALHOS RELACIONADOS

No trabalho de Depardon, Mahec e Séguin (2013), é apresentada uma comparação baseada na escalabilidade, transparência e tolerância a falhas entre seis sistemas de arquivos distribuídos. Os sistemas analisados são HDFS, MooseFS, iRODS, Ceph, GlusterFS, e Lustre, e é apresentada uma introdução às suas características.

Os autores apresentam uma discussão relacionada aos benefícios e desvantagens dos diferentes sistemas de arquivos, onde compara as características que são detalhadas no trabalho, utilizando uma tabela para expor essas características.

A discussão relacionada à escalabilidade dá ênfase à arquitetura, compara as soluções de cada sistema de arquivo para lidar com armazenamento e as operações de E/S, e expõe em uma tabela os resultados do experimento realizado para analisar o desempenho, utilizando a métrica de tempo de execução de escrita e leitura de um arquivo de 20GB e mil arquivos de 1MB. A infraestrutura utilizada foi um *cluster* com nós de 8GB de memória, processador AMD Opteron 2218 2.6GHz, disco de 250GB e interface de rede Gigabit Ethernet, com um sistema operacional Debian, na plataforma grid5000. Com esses resultados, os autores concluem que o GlusterFS possui melhor desempenho na escrita de arquivos pequenos.

Em relação à transparência, o trabalho compara os diferentes recursos dos sistemas de arquivos que garantem o acesso aos arquivos, detecção de falhas e o acesso ao sistema. Segundo os autores, Ceph e GlusterFS reduzem a carga de trabalho do servidor de metadados por apenas fornecerem as informações necessárias para o cliente procurar a localização dos dados.

Na discussão sobre tolerância a falhas, o trabalho compara os diferentes recursos usados pelos sistemas de arquivos distribuídos para lidar com disponibilidade do sistema, disponibilidade e sincronização de dados e balanceamento de carga. Ceph e GlusterFS são caracterizados como altamente disponíveis, por replicarem os metadados em vários servidores. Segundo os autores, iRODS, Ceph e GlusterFS usam replicação síncrona, onde as consultas a arquivos são bloqueadas até que todas as réplicas sejam salvas no disco. Em relação ao balanceamento de carga, o artigo descreve que Ceph e GlusterFS fornecem um

algoritmo para distribuir a carga de trabalho dos servidores de metadados, armazenam dados de acordo com o espaço livre em disco e transferem dados de um servidor sobrecarregado para outro, como ocorre também com o HDFS.

O trabalho apresenta os testes realizados para analisar a acessibilidade, a disponibilidade e o desempenho de cada sistema de arquivo distribuído. Nos testes de acessibilidade, os autores tentam acessar o *cluster* em uma rede privada através de um cliente em outra, usando um comando ssh com encaminhamento de porta. Não obtiveram sucesso apenas com os sistemas de arquivos Ceph e GlusterFS. Para os testes de disponibilidade, na configuração do HDFS, foi utilizado um servidor de metadados, cinco servidores de dados e três réplicas; para o MooseFS foi utilizado um servidor de metadados, um nó de backup, cinco servidores de metadados e três réplicas; no iRODS, um servidor de metadados e quatro servidores de dados; no Ceph, dois servidores de metadados, três servidores de dados e duas réplicas; para o GlusterFS, quatro servidores e duas réplicas; e no Lustre, um servidor de metadados e quatro servidores de dados. Em cada sistema de arquivo são inseridos 34MB de dados, é observado o espaço em disco usado e depois, um nó falha. Assim, os autores analisam se os sistemas de arquivos distribuídos detectaram o nó indisponível, se a replicação foi satisfatória, se todos os dados estão disponíveis e se os sistemas são balanceados depois que o nó se torna disponível novamente.

Semelhante ao trabalho de Depardon, Mahec e Séguin (2013), o presente trabalho propõe apresentar uma avaliação dos sistemas de arquivos distribuídos Ceph e GlusterFS, porém a avaliação utiliza outro sistema de arquivo, o NFS, e faz uso de uma infraestrutura de nuvem computacional com a plataforma OpenNebula. Este trabalho utiliza a técnica de avaliação de desempenho experimentação ou aferição, onde resultados mais precisos são gerados e faz uso de uma ferramenta de *benchmark* para coletar dados e compará-los (JOHNSON; MARGALHO, 2011), além de coletar dados com códigos simples em *Shell Script* desenvolvidos no decorrer da execução do trabalho. A ferramenta *benchmark* utilizada foi o IOZone, onde foi possível fazer o uso de métricas de avaliação para obter resultados mais precisos de operações típicas de um ambiente de nuvem.

Em Barbosa Jr, Greve e Barreto (2007), são apresentadas uma avaliação quantitativa, utilizando métricas de desempenho, e uma avaliação qualitativa, relacionada a facilidade de configuração e administração dos sistemas. Os sistemas de arquivos abordados nesse trabalho são Lustre, PVFS2, GlusterFS e OCFS2 com iSCSI.

Os autores descrevem a arquitetura do Lustre e citam seus componentes, servidores de metadados (*MDS – Metadata Servers*), servidores de E/S (*OST – Object Storage*

Target) e *OBD* (*Object Based Disks*), responsáveis pelos discos físicos. O artigo assemelha o Lustre ao PVFS2, por permitir configurar mais de um servidor de metadados e cita que a arquitetura do PVFS2 permite que todos os servidores atuem como servidores de metadados e de E/S ao mesmo tempo, mas precisa de ferramentas adicionais para realizar redundância de servidores de E/S. Quanto ao OCFS2 com iSCSI, os autores explicam que adotaram essa solução mista para obter o compartilhamento dos discos em diferentes máquinas, pois o OCFS2 não provê armazenamento em diferentes servidores. O GlusterFS é descrito como um sistema que tem como principais objetivos escalabilidade, desempenho e disponibilidade.

A avaliação quantitativa é apresentada como uma avaliação experimental, que faz uso de cinco servidores, divididos em quatro servidores de E/S (com 2GB de memória, cada um) e um servidor de metadados (com 4GB de memória), e quatorze clientes com máquinas Intel Pentium 4 de 32 bits à 3GHz e 512MB de memória, todos com o sistema operacional Debian, conectados a uma rede com um switch Gigabit Ethernet. A ferramenta utilizada para realizar a avaliação da taxa de transferência foi o IOZone, executando operações de escrita e leitura de arquivos entre 16MB e 4G, em blocos de dados de tamanho entre 64KB e 16MB. Os experimentos são divididos em duas etapas, na primeira são transmitidos arquivos de tamanho diferentes, entre 16MB e 4GB, em blocos de 16MB, e na segunda, é transmitido um arquivo de 1GB em blocos de tamanhos diferentes, entre 16KB e 16MB. O artigo conclui que o Lustre obteve melhor desempenho com arquivos pequenos e o PVFS2 obteve melhores resultados nas operações de escrita de arquivos maiores. Na análise dos dados das operações com variação do tamanho de blocos, o Lustre apresentou melhor desempenho nas operações de leitura.

Na avaliação qualitativa, os autores consideram que o Lustre apresenta dificuldades na instalação, porém possui um conjunto de ferramentas de manutenção que facilita a adição de novos servidores. O OCFS2 com iSCSI é avaliado como um sistema que necessita de maiores cuidados em sua administração e na adição ou remoção de um servidor. O artigo cita que o PVFS2 possui facilidade na configuração e administração. Em relação ao GlusterFS, os autores consideram que esse sistema se destaca em sua configuração, por utilizar solução em FUSE.

Diferente do trabalho de Barbosa Jr, Greve e Barreto (2007), a avaliação presente neste trabalho é realizada utilizando os sistemas de arquivos distribuído Ceph, GlusterFS e NFS conectados à uma nuvem computacional, onde os sistemas de arquivos armazenam máquinas virtuais. Máquinas virtuais estas que executam a ferramenta IOZone para coletar a

taxa de transferência de diferentes operações com diferentes tamanhos de arquivos, como no trabalho citado anteriormente.

No trabalho de Pillon *et al.* (2013), é apresentado um comparativo entre os sistemas de arquivos distribuídos PVFS2 e GlusterFS em um ambiente de nuvem computacional. Os autores caracterizam o PVFS2 como um sistema que é utilizado geralmente em *clusters*, ressaltam que sua configuração permite a execução de servidores de E/S e metadados em uma mesma máquina. Em relação ao GlusterFS, o trabalho cita que é um sistema que depende do FUSE e os dados são armazenados em volumes distribuídos pelos nós que compõe o sistema.

Na análise comparativa, os autores utilizaram dois computadores, um atuando como cliente e outro como um servidor de máquinas virtuais, com as características sendo processador AMD Phenom II X4 B93 2,8GHz (*quad-core*), memória de 4GB e disco de 500GB SATA, conectados à um switch de velocidade de 100Mbps, e o sistema operacional usado foi o Debian Squeeze, no cliente, a versão 64 bits no servidor, com o *hypervisor* XEN. As máquinas virtuais com o sistema operacional Debian Squeeze 64 bits, uma VCPU, 256MB de memória e 4GB de disco, executaram com a técnica de paravirtualização. E a ferramenta de *benchmark* utilizada foi IOZone, executando operações de leitura, releitura, escrita e reescrita. O cliente conectado às duas máquinas virtuais executou as operações com arquivos de tamanhos entre 1MB e 128MB em 25 repetições, para cada sistema de arquivo distribuído. Os autores comparam o PVFS2 e o GlusterFS com sistema de arquivo local (ext3) e puderam concluir que o GlusterFS apresenta resultados semelhantes ao ext3, mas diminui o desempenho à medida que o tamanho do arquivo aumenta. Em relação ao PVFS2, os autores concluíram que esse sistema se mantém constante para todos os tamanhos de arquivos.

Este trabalho se assemelha ao de Pillon *et al.* (2013) ao fazer uso de uma nuvem computacional, da ferramenta *benchmarck* IOZone e do sistema de arquivo distribuído GlusterFS. Porém a nuvem computacional presente neste trabalho é criada com a plataforma OpenNebula e utiliza mais dois sistemas de arquivos distribuídos além do GlusterFS, o Ceph e o NFS. E além dos dados da ferramenta IOZone, este trabalho apresenta uma análise do tempo que cada sistema de arquivo leva para criar e remover máquinas virtuais.

4 PROCEDIMENTOS

A proposta deste trabalho é realizar uma avaliação de desempenho entre os sistemas de arquivos distribuídos Ceph, GlusterFS e NFS e pretende explorar as características de replicação de dados, consistência e eficiência em um ambiente de nuvem.

Para isso, o trabalho propõe analisar os dados coletados durante a execução da ferramenta IOZone, utilizando a métrica de taxa de transferência e tempo de execução das operações de escrita, reescrita, leitura e releitura de arquivos de tamanhos 64KB, 128KB, 64MB e 128MB. As operações com os arquivos de tamanho 64MB e 128MB foram executadas em registros de tamanho 16MB, pois é o tamanho máximo com que o IOZone executa. A execução dessas operações foi realizada, primeiramente, com uma máquina virtual e, em seguida, com duas máquinas virtuais executando ao mesmo tempo, em trinta repetições, para cada sistema de arquivos. Os sistemas de arquivos serão avaliados também pela métrica de tempo de criação e remoção de máquinas virtuais na plataforma OpenNebula.

A infraestrutura utilizada foi configurada na rede cabeada do Laboratório de Redes de Computadores da Universidade Federal do Ceará – Campus Quixadá. A plataforma OpenNebula, na versão 4.10, foi implantada em um notebook com processador Intel Core i5, memória de 4GB, disco de 330GB e sistema operacional Ubuntu Server 14.04 LTS 64bits. É necessário que a máquina possua um hipervisor, uma ferramenta responsável pela virtualização das máquinas virtuais, e este trabalho utilizou o KVM-QEMU[3], que é um *software* de código aberto e provê um serviço de virtualização completa para Linux (KVM, 2016). Esta máquina atua como *Front-end* e *Host* ao mesmo tempo, ou seja, uma mesma máquina será responsável pela administração da infraestrutura da nuvem e fornecerá os recursos para a execução das máquinas virtuais (SOTO, 2011).

Os sistemas de arquivos foram configurados separadamente para que houvesse dedicação total das máquinas na execução das operações de coleta de dados. Para o Ceph, foram utilizadas três máquinas físicas do laboratório, uma com 1TB de disco, 8GB de memória, processador Intel Core i5 e sistema operacional Ubuntu Server 14.04 LTS 64bits, configurada como sendo o MDS (*Metadata Server*) e um dos OSDs (*Object Storage Deamon*), e as outras duas com 500GB de disco, 8GB de memória, processador Intel Core i5 e sistema operacional Ubuntu Desktop 14.04 LTS 64bits, configuradas para atuarem como OSDs. Com o GlusterFS, foram utilizadas duas máquinas físicas para compor o volume replicado deste sistema de arquivo, uma com 1TB de disco, 8GB de memória, processador Intel Core i5 e sistema operacional Ubuntu Server 14.04 LTS 64bits, e outra com 500GB de

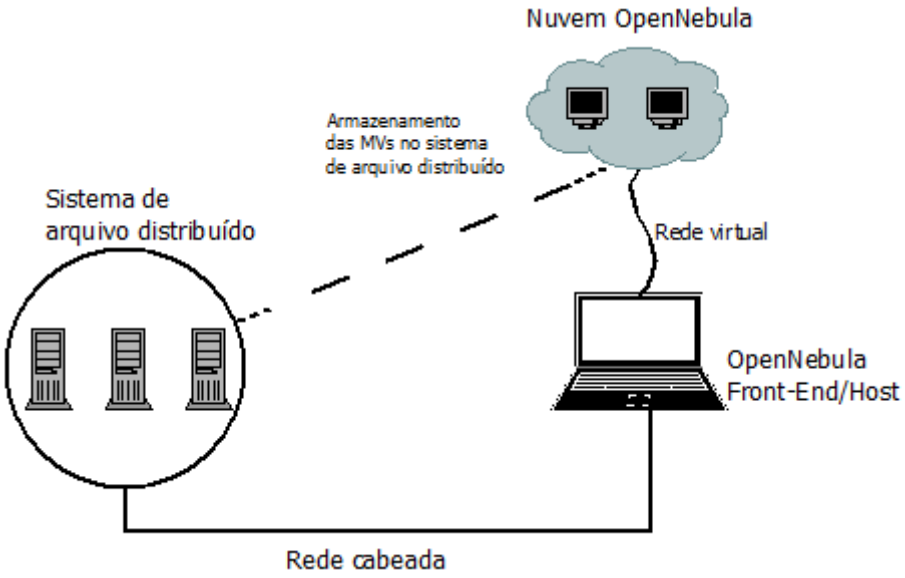
[3] http://www.linux-kvm.org/page/Main_Page

disco, 8GB de memória, processador Intel Core i5 e sistema operacional Ubuntu Desktop 14.04 LTS 64bits. E para criar um dispositivo de armazenamento com o NFS, foi configurada apenas uma máquina, com um 1TB de disco, 8GB de memória, processador Intel Core i5 e sistema operacional Ubuntu Server 14.04 LTS 64bits. Este trabalho seguiu as configurações mínimas padronizadas pelos sistemas de arquivos em suas documentações e apresenta nos Anexos, os tutoriais utilizados para criar o armazenamento.

Com os sistemas de arquivos e a nuvem computacional do OpenNebula devidamente configurados, é possível criar as máquinas virtuais com uma imagem do Ubuntu Server 14.04 LTS, com disco de 8GB, memória de 1GB, e uma VCPU, para executar as operações da ferramenta IOZone. E para as operações de criação e remoção de máquina virtual, foi utilizada a mesma configuração, mudando apenas o tamanho da memória, 512MB, considerando que o tamanho da memória não influencia na criação do arquivo da máquina virtual. As imagens são criadas com o KVM-QEMU no formato "qcow2", que é o formato nativo deste *software* e oferece flexibilidade (QEMU, 2016), depois são copiadas para o diretório de armazenamento de imagens compartilhadas do OpenNebula. Em Anexos, serão apresentados os comandos e os arquivos de configuração para a criação dessas imagens.

Para armazenar as máquinas virtuais nos sistemas de arquivos, foi necessário fazer a montagem de cada sistema de arquivo no diretório do OpenNebula que mantém as imagens de máquinas virtuais em execução, a Figura 7 ilustra a infraestrutura utilizada neste trabalho.

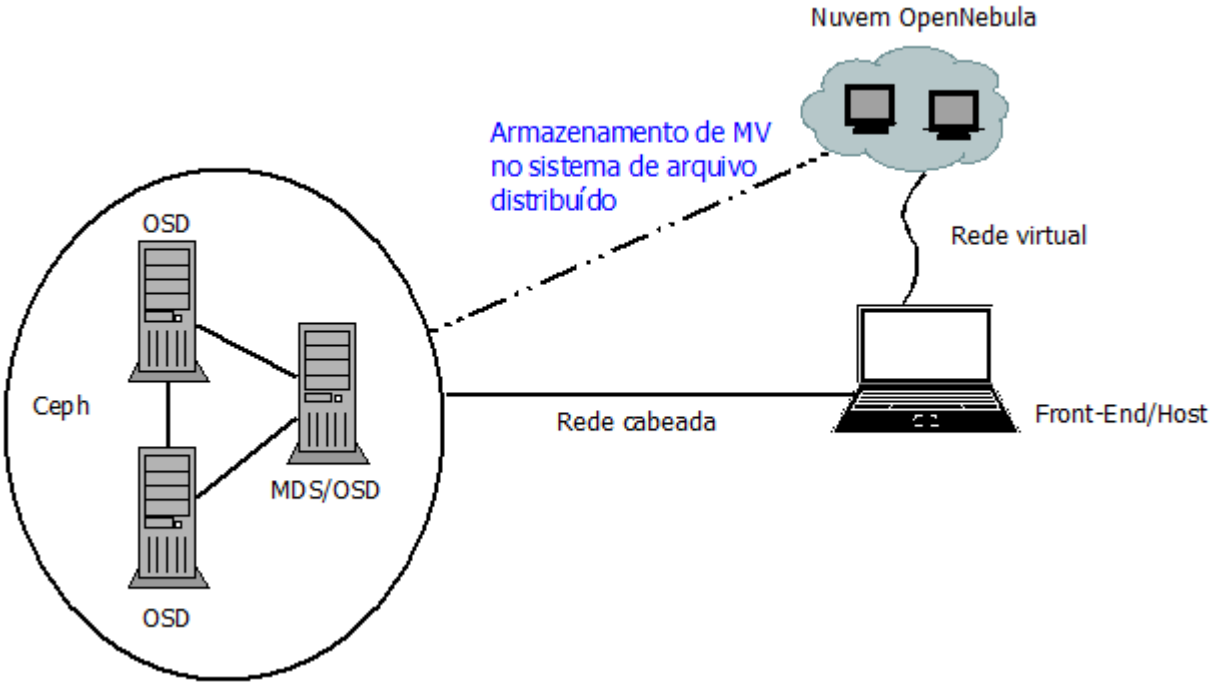
Figura 7 – Ilustração da infraestrutura



Fonte: elaborada pela autora

Ao finalizar a instalação do OpenNebula é necessário criar um novo *datastore*, onde serão armazenadas as máquinas virtuais. Este *datastore* é um diretório do sistema do OpenNebula, que por sua vez, será o local destinado para a montagem dos sistemas de arquivos distribuídos. O Ceph utilizou três máquinas físicas para realizar a distribuição de replicação dos dados, onde uma dessas máquinas foi configurada como sendo o servidor de metadados (MDS). Na máquina em que o OpenNebula foi implantado, foi montado o Ceph através do comando que utiliza o IP do MDS. Feito isso, ao criar uma máquina virtual, o OpenNebula cria um diretório para esta máquina virtual, com seu ID, dentro do diretório *datastore*, que está montado o Ceph. Assim, os arquivos necessários para a execução da máquina virtual são armazenados no sistema de arquivo distribuído. Este procedimento pode ser melhor visualizado na Figura 8.

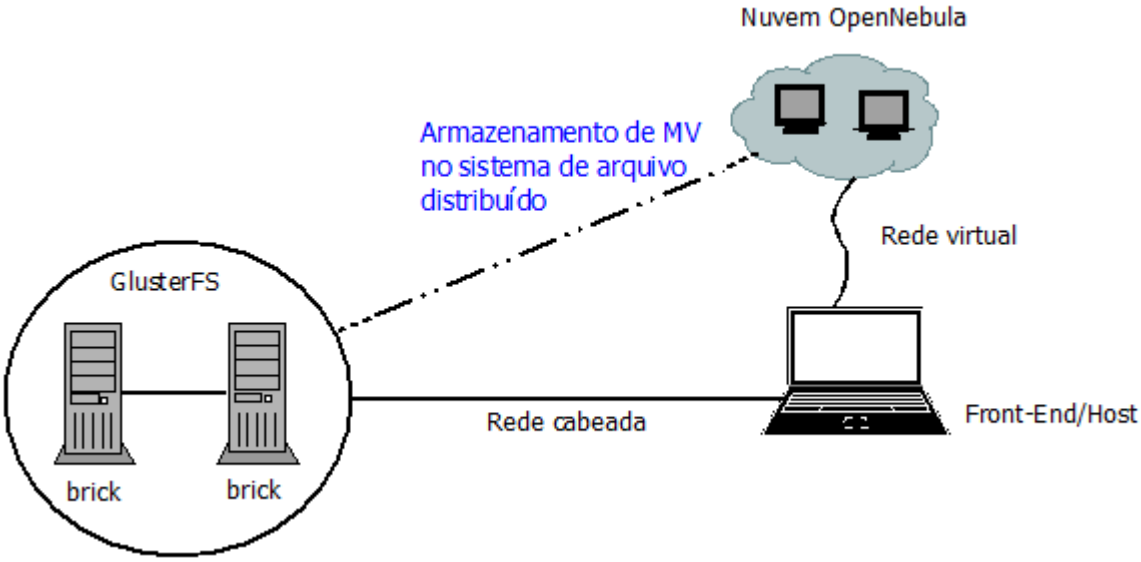
Figura 8 - Ilustração do armazenamento Ceph



Fonte: elaborada pela autora

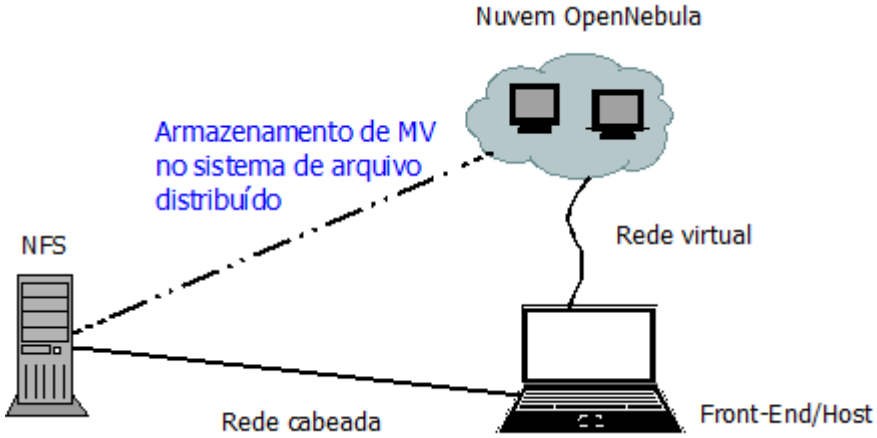
O mesmo procedimento ocorre com o GlusterFS e NFS. Porém com o GlusterFS foi utilizado duas máquinas físicas para criar um volume replicado e uma delas foi configurada para ser a máquina principal do sistema de arquivos distribuído (FIGURA 9). Ao montar no diretório *datastore*, foi utilizado o comando que especifica o IP da máquina principal e o nome do volume criado. Já com o NFS, foi utilizada uma máquina para criar o armazenamento virtualizado (FIGURA 10). A montagem no *datastore* se dá através do comando que específica o IP desta máquina e o diretório criado para ser compartilhado.

Figura 9 - Ilustração do armazenamento GlusterFS



Fonte: elaborada pela autora

Figura 10 - Ilustração do armazenamento NFS



Fonte: elaborada pela autora

5 DESENVOLVIMENTO/RESULTADOS

5.1 Coleta de dados

Para a coleta de dados da avaliação, foi necessário instalar a ferramenta IOZone nas máquinas virtuais utilizadas e foi dividida em duas etapas, em cada sistema de arquivo. Na primeira, as operações de escrita, reescrita, leitura e releitura foram executadas em uma máquina virtual do OpenNebula, com arquivos de tamanhos 64KB, 128KB, 64MB e 128MB, sendo esses dois últimos divididos em registros de 16MB; e na segunda, duas máquinas virtuais executam ao mesmo tempo as operações de escrita, reescrita, leitura e releitura, respectivamente, com os mesmos tamanhos de arquivos.

O IOZone executa as operações a partir da linha de comando do Linux, o comando "./iozone -a -i 0 -i 1 -s 64 -r 64 -b ceph1-64k.xls" é um exemplo. A opção "-a" indica que será executado o modo automático, a opção "-i" determina qual operação deverá executar ("0" indica escrita e reescrita e "1" indica leitura e releitura), "-s" indica o tamanho do arquivo, "-r" indica o tamanho do registro e "-b" cria um arquivo no formato de uma planilha. Para obter resultados mais próximos da realidade, nenhuma alteração de *cache* foi realizada. Os comandos são executados trinta vezes para cada tamanho de arquivo, para cada sistema de arquivos e nas duas etapas.

Em seguida são coletadas as taxas de transferência (KB/s) medidas e computadas nos arquivos gerados pelo IOZone. É calculada a média e o intervalo de confiança, com nível de confiança igual à 95%, para cada operação e cada tamanho de arquivo, para os três sistemas de arquivos, nas duas etapas. E a partir da coleta desses valores, são gerados os gráficos com a ferramenta Gnuplot[4]. Para obter o tempo da execução das operações, foi utilizado um cálculo que divide o tamanho do arquivo pela taxa de transferência e calculada a média e o intervalo de confiança, com nível de confiança 95%.

Com a criação de *Shell scripts* é possível coletar o tempo de criação e remoção de máquinas virtuais. O *script* que verifica o tempo de criação de máquina virtual executa um código em *loop* capturando o tempo atual durante a execução do comando do OpenNebula que cria máquinas virtuais e verifica o estado da máquina virtual até que se torne a condição satisfatória para sair do *loop*. O estado que este *script* usa para ser a condição satisfatória é o estado em que a máquina virtual esteja pronta para uso. E o *script* que coleta os dados do tempo de remoção total de máquinas virtuais dos sistemas de arquivos executa um código em *loop* que retorna o tempo atual e verifica quantos diretórios existem no diretório *datastore*, pois ao criar uma máquina virtual, o seu armazenamento em um diretório é intitulado com o seu ID. O *script* executa o comando de remoção, em seguida, inicia a verificação e para quando o número de diretório for diferente. A máquina virtual é criada a partir da imagem no formato "qcow2", onde o tamanho da máquina virtual é inicialmente de 1,9GB. Assim, com esta coleta de dados é possível analisar o comportamento dos sistemas de arquivos na criação e remoção de 1,9GB de dados.

O parâmetro usado na métrica de tempo de criação e remoção de máquinas virtuais foi de dez máquinas virtuais criadas e dez máquinas removidas, uma de cada vez, para cada sistema de arquivos. Após a execução dos *scripts*, foram calculadas a média de tempo e

[4] http://www.gnuplot.info/

o intervalo de confiança, com o nível de confiança em 95%, em seguida, foram gerados gráficos com a ferramenta Gnuplot (GNUPLOT, 2016).

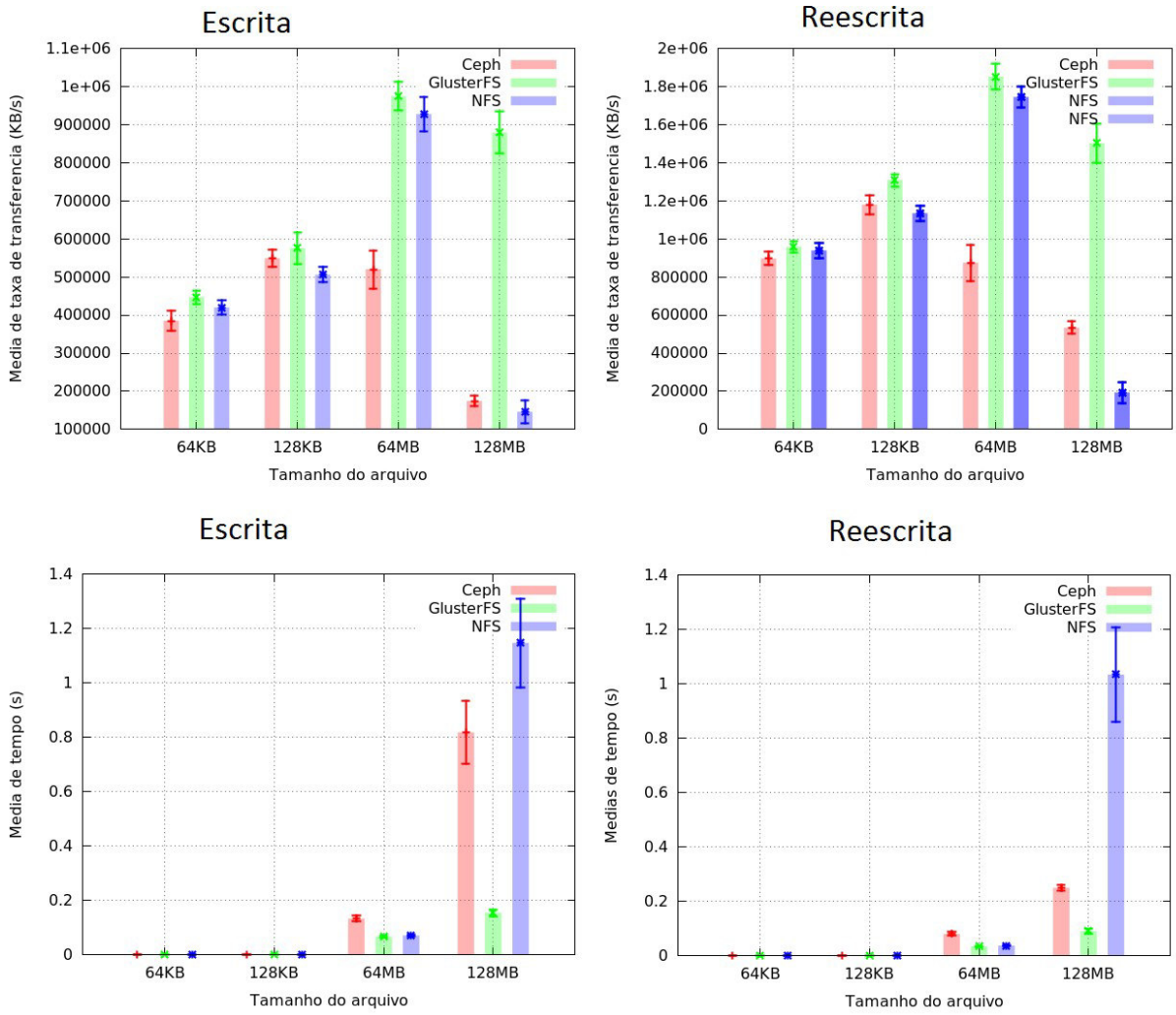
5.2 Análise de resultados

Com o intuito de analisar o comportamento de cada sistema de arquivos em um ambiente de nuvem ao executar operações comuns deste tipo de infraestrutura, são apresentados os gráficos gerados com a coleta de dados da ferramenta IOZone e com os dados gerados com os *scripts*. Os gráficos das médias de taxa de transferência estão em KB/s e o tamanho dos arquivos em KB, e os gráficos de tempo estão em segundos.

A seguir são apresentados os resultados obtidos com a coleta de dados da primeira etapa, que é a execução das operações da ferramenta IOZone por uma máquina virtual.

Em relação às operações de escrita e reescrita, pode-se observar na Figura 11 que o GlusterFS foi o sistema de arquivo que obteve o desempenho maior entre os três sistemas, isso pode estar relacionado à utilização dos tradutores de desempenho, que são responsáveis por ajustar a carga de trabalho em operações E/S. O Ceph obteve desempenho inferior, podendo ser justificado ao fato de que este sistema utiliza um algoritmo que separa os arquivos em blocos e foi configurado para os replicar em três discos. O NFS obteve as médias mais baixas com arquivos de 128MB, esse comportamento pode estar relacionado ao fato de que o IOZone pode obter auxílio de memória *cache* em arquivos de tamanho menores para alguns sistemas de arquivos e o NFS não possui um algoritmo que otimiza a sua carga de trabalho.

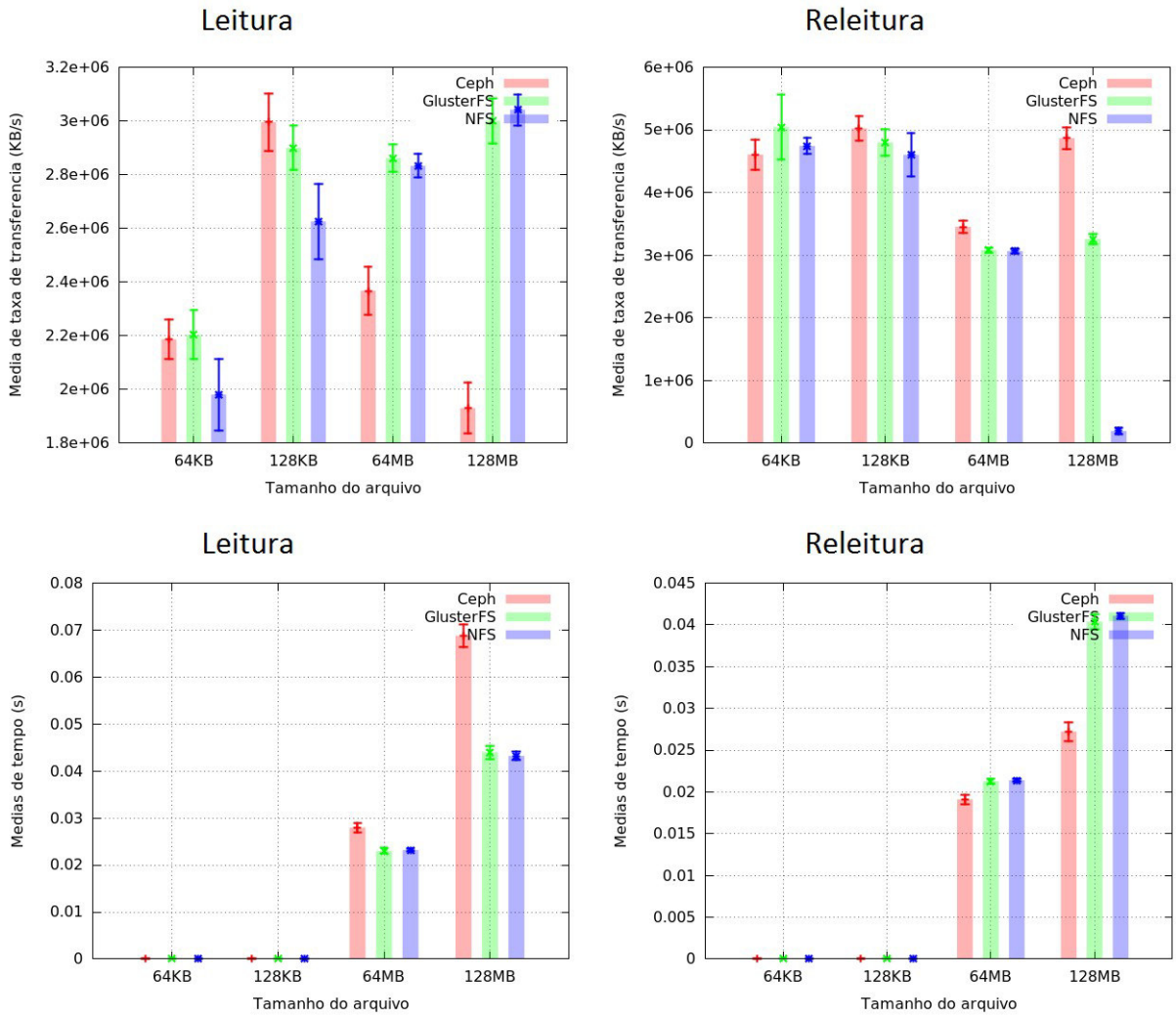
Figura 11 – Gráficos das médias de taxas de transferência e de tempos de execução das operações de escrita e reescrita com uma máquina virtual



Fonte: elaborada pela autora

Pode-se observar nos gráficos das médias de leitura (FIGURA 12) que o GlusterFS obteve um melhor desempenho com dois tamanhos de arquivo, este comportamento pode estar relacionado também ao fato desse sistema utilizar tradutores que auxiliam as operações de E/S. O alto desempenho do Ceph com um tamanho de arquivo nos gráficos de leitura e com três tamanhos de arquivos nos gráficos de releitura pode estar relacionada à utilização da função CRUSH, que facilita na busca da localização dos arquivos (FIGURA 12). Já o alto desempenho do NFS com arquivos maiores nos gráficos de leitura pode estar relacionado a utilização do cache que funciona armazenando os atributos dos arquivos na memória local, porém este sistema obtêm uma queda de desempenho com a operação de releitura, que pode estar relacionado ao fato de que este sistema não tem conhecimento dos arquivos abertos pelo sistema.

Figura 12 – Gráficos das médias de taxas de transferência e de tempos de execução das operações de leitura e releitura com uma máquina virtual



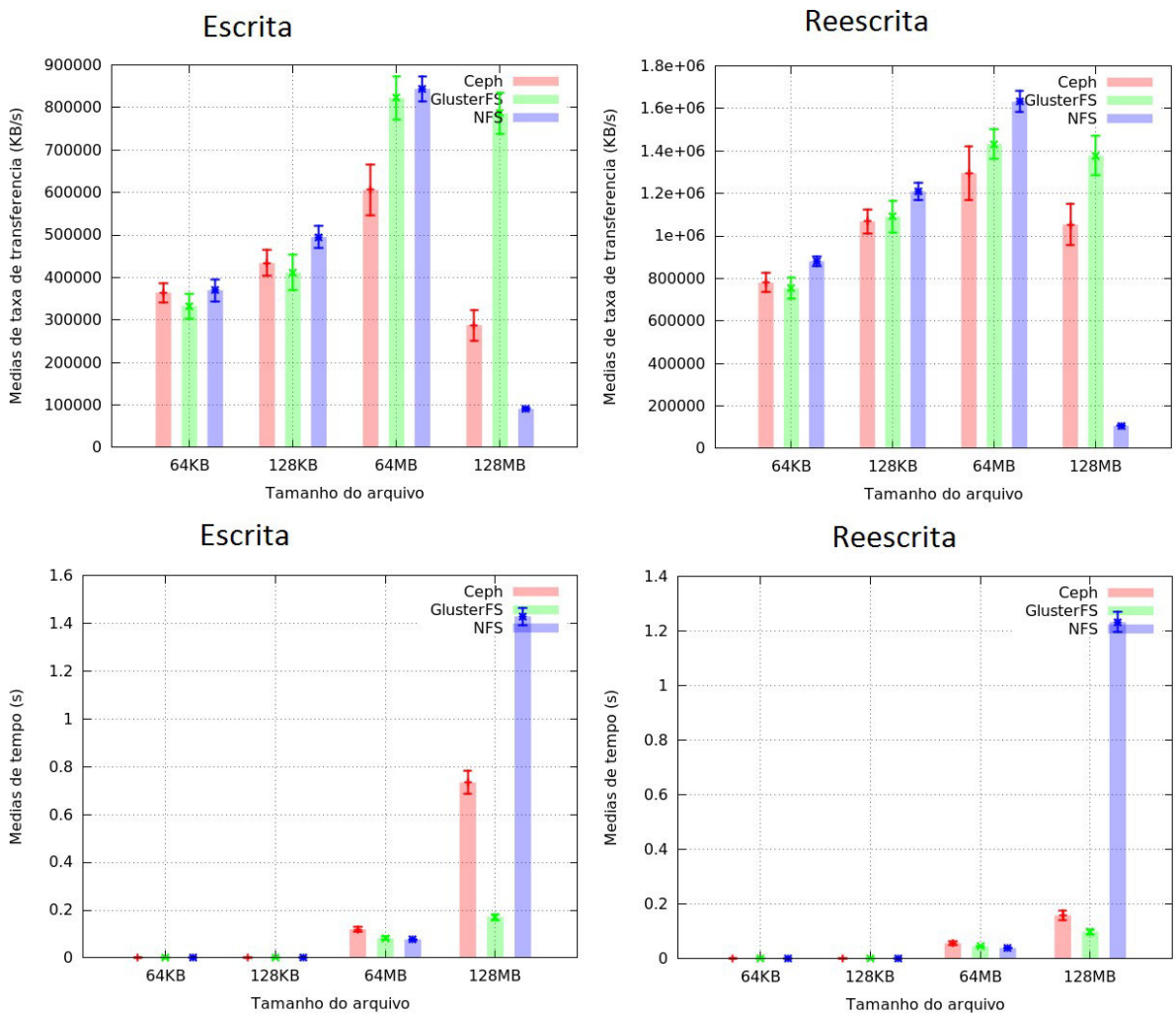
Fonte: elaborada pela autora

Na segunda etapa da coleta de dados com a ferramenta IOZone, foram utilizadas duas máquinas virtuais executando as operações ao mesmo tempo, para cada sitema de arquivo. Foram calculadas as médias de tempo de execução das operações e o intervalo de confiaça, como na primeira etapa. Os resultados serão apresentados a seguir.

Pode-se observar nos gráficos de escrita e reescrita da Figura 13 que o NFS obteve o melhor desempenho com três tamanhos de arquivos e obteve uma queda com o arquivo de tamanho 128MB, que pode ser relacionada à ausência de utilização de *cache* nesse tipo de operação do IOZone com arquivos maiores, e pode-se relacionar este fator ao desempenho do Ceph, também. O GlusterFS obteve desempenho inferior com arquivos pequenos, esse comportamento pode estar relacionado à utilização de um componente

responsável pela replicação de dados, o AFR, que mantem a consistência da replicação mesmo se estiver acontecendo alguma operação em um arquivo.

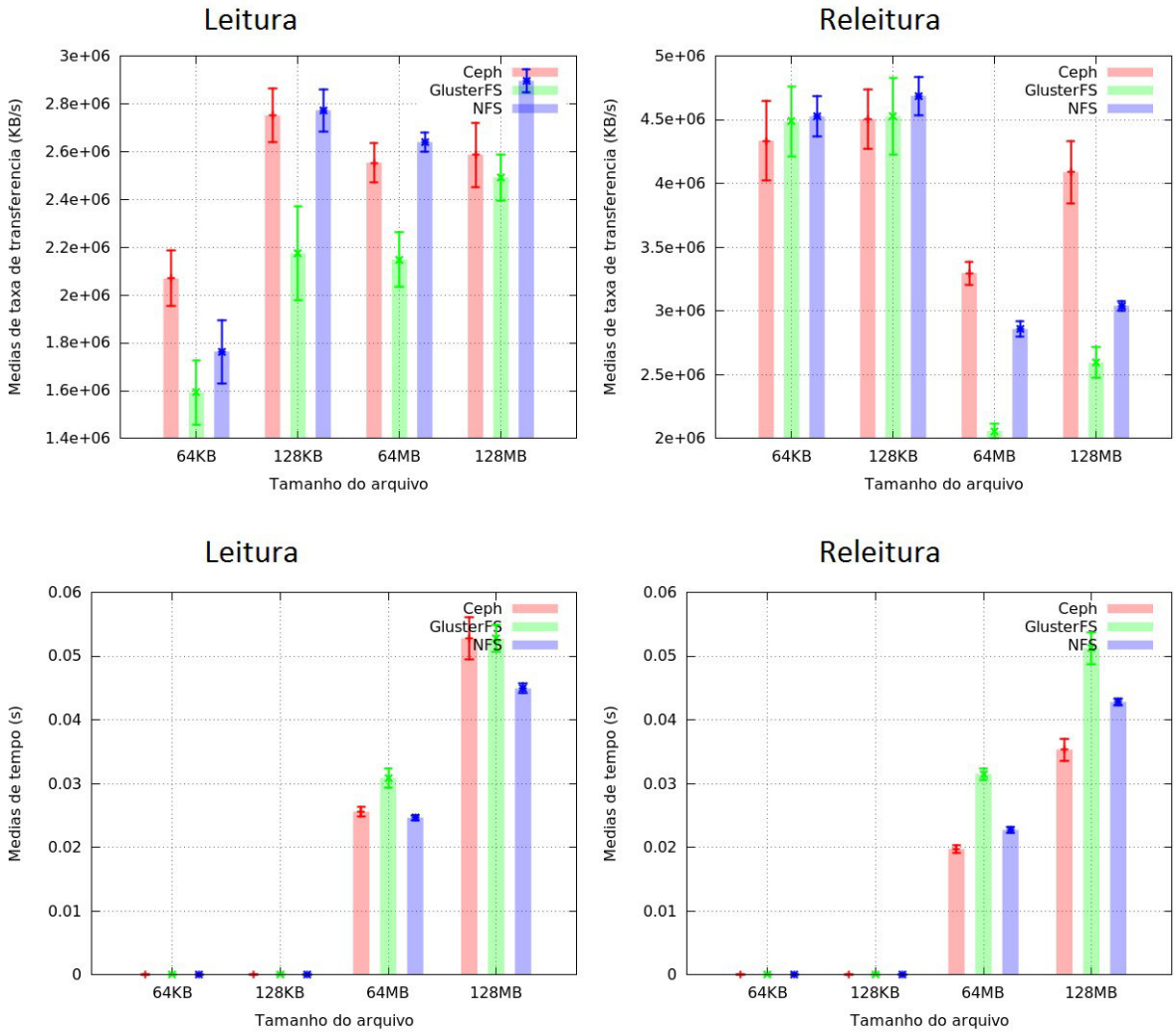
Figura 13 - Gráficos das médias de taxas de transferência e de tempos de execução das operações de escrita e reescrita com duas máquinas virtuais



Fonte: elaborada pela autora

Nos gráficos das médias de leitura da Figura 14, pode-se observar que o NFS obteve o maior desempenho com três tamanhos de arquivos e manteve as taxas na operação de releitura, porém o Ceph foi melhor com arquivos de tamanho 64MB e 128MB em releitura, isso pode estar relacionado ao componente que facilita a busca pela localização dos arquivos. O GlusterFS obteve desempenho inferior, como pode ser observado nos gráficos da operação de leitura, e com arquivos de tamanhos maiores na operação de releitura, o que pode estar relacionado à utilização do tradutor de *cluster* AFR, novamente.

Figura 14 – Gráficos das médias de taxas de transferência e de tempos de execução das operações de leitura e releitura com duas máquinas virtuais



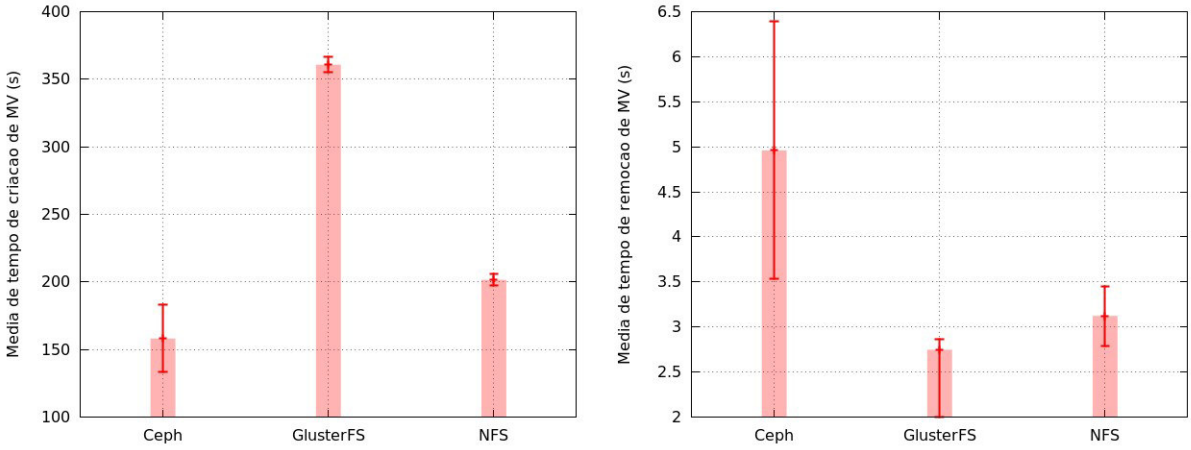
Fonte: elaborada pela autora

Com a finalidade de observar o comportamento dos sistemas de arquivos ao criar e remover dez máquinas virtuais, separadamente, e os resultados obtidos com a coleta de dados através de *scripts* desenvolvidos durante a execução do trabalho serão apresentados a seguir.

Com os resultados obtidos com as operações de criação e remoção de máquina virtual (FIGURA 15), pode-se observar que o GlusterFS obteve os maiores tempos ao armazenar 1,9GB de dados, o que pode estar relacionado à utilização do tradutor de desempenho que mantém a consistência dos dados replicados, mesmo se estiver ocorrendo alguma operação nos mesmos. Porém obteve a média mais baixa na operação de remoção de máquina virtual. O Ceph foi o sistema de arquivo distribuído que obteve o melhor resultado

na operação de criação, comportamento este que pode estar atribuído à utilização da função CRUSH, que pode auxiliar na distribuição otimizada dos dados, mas foi o que levou mais tempo para remover. O NFS obteve um resultado intermediário, sendo o sistema de arquivo que menos obteve variação de valores.

Figura 15 – Gráficos das médias de tempo de criação e de remoção de máquinas virtuais



Fonte: elaborada pela autora

Tendo como base a revisão bibliográfica dos sistemas de arquivos, pode-se analisar a consistência e replicação dos dados nos três sistemas de arquivos distribuídos. Pode-se atribuir a queda de desempenho do GlusterFS ao fato de que este sistema de arquivo distribuído possui um componente responsável pela replicação de dados, o AFR, que mantém a consistência da replicação mesmo se estiver acontecendo alguma operação em um arquivo. Essa funcionalidade é importante para a consistência do sistema de arquivos distribuído e para a recuperação de dados, mas pode acarretar em uma perda de desempenho.

Pode-se observar que o Ceph obteve melhores resultados nas operações de leitura e releitura. Este fato pode estar relacionado à função CRUSH que o Ceph implementa para distribuir os objetos dos dados, pois ela simplifica a busca pela localização dos objetos nos dispositivos de armazenamento (OSDs), e utiliza um servidor de metadados para fazer o gerenciamento das consultas dos clientes. A divisão dos dados em objetos é uma importante utilidade deste sistema de arquivo que oferece replicação, eficiência e confiabilidade aos usuários, porém isso pode ser atribuído ao fato do Ceph ter obtido um desempenho inferior na operação de remoção de máquina virtual, visto que 1,9GB de dados tiveram que ser localizados e removidos de três OSDs. A sua instalação é complexa, onde deve ser informado

qual máquina deverá hospedar o servidor de metadados e quais máquinas deverão hospedar os servidores de armazenamentos, e necessita que os servidores estejam sincronizados para o sistema de arquivo ser montado na máquina cliente.

A partir da análise dos resultados, observou-se que o NFS obteve melhor desempenho em operações que puderam ter o auxílio de um *cache*, pois este sistema de arquivo trabalha de forma semelhante ao sistema local do UNIX e utiliza o armazenamento de memória *cache* local. Além de não precisar fazer replicação de dados. O NFS é um sistema de arquivo que oferece consistência de dados através de uma técnica que copia os blocos de arquivos a cada 30 segundos. Mas na infraestrutura deste trabalho, este sistema não ofereceu replicação de dados, pois foi instalado em apenas um disco. A sua instalação e configuração são um pouco complexas, pois requer que o administrador informe quem poderá ter acesso e qual é o tipo de acesso.

6 CONCLUSÃO

Concluindo a avaliação deste trabalho, o conceito de Computação em Nuvem tem sido bastante abordado atualmente, como sendo uma solução de fácil acesso, relativamente confiável, adaptável e barata. E a virtualização oferece às organizações e instituições de estudo e pesquisa infraestruturas capazes de satisfazer os seus interesses.

Este trabalhou utilizou a virtualização de armazenamento para avaliar três sistemas de arquivos distribuídos em uma infraestrutura montada com o OpenNebula. Os sistemas de arquivos foram o Ceph, o GlusterFS e o NFS.

O Ceph, como um sistema de arquivo totalmente distribuído, possui um conjunto de funcionalidades que o torna um sistema com alto nível de replicação e obteve o melhor desempenho na criação de máquinas virtuais. Pode-se concluir que o Ceph é um sistema que se dedica a oferecer eficiência, confiabilidade e replicação de dados.

O sistema de arquivo distribuído GlusterFS possui fácil instalação e configuração, disponibiliza vários tipos de arquiteturas que podem ser adotadas como soluções para diferentes tipos de interesses e possui um sistema que trabalha para provê confiabilidade, eficiência e redundância de dados. A sua solução para prover redundância pode acarretar em uma queda de desempenho da sua performance. GlusterFS pode ser abordado como uma solução de distribuição de armazenamento em ambiente que priorizam a redundância dos dados em tempo real.

A avaliação apresentada neste trabalho conclui que o NFS é um sistema direcionado para ambiente de pequena escala, que tem foco em consistência de dados e não necessita de replicação de dados.

Com esta avaliação, o trabalho tem o intuito de acrescentar conhecimento nas pesquisas relacionadas a ambientes virtualizados e sistemas de arquivos distribuídos. Pois acredita-se que os sistemas de arquivos distribuídos podem atuar como uma solução bastante útil para as organizações que priorizam confiabilidade, replicação de dados, e eficiência em seus sistemas.

REFERÊNCIAS

AGOSTINHO, P. **Virtualização em SAP**, Universidade Lusófona de Humanidades e Tecnologias, 6º SOPCOM, Lisboa, Portugal, 2009. Disponível em: <http://conferencias.ulusofona.pt/index.php/sopcom_iberico/sopcom_iberico09/paper/viewFile/306/280>. Acesso em: 01 out. 2015.

BARBOSA JR, A. A.; GREVE, F.; BARRETO, L. P. **Avaliação de Sistemas de Arquivos Distribuídos num Ambiente de Pequena Escala.** In: 27th Congresso da Sociedade Brasileira de Computação, 6th Workshop de Sistemas Operacionais, 27, 2007. Anais...Rio de Janeiro, RJ, 2007.

BOUFLEUR, M. P. *et al.* **Avaliação do Sistema de Arquivo Distribuído GlusterFS**, Universidade Federal de Santa Maria, VIII WSCAD, Gramado, RS, 2007. Disponível em: < http://www-usr.inf.ufsm.br/~rodox/publicacoes/glusterfs-wscad2007.pdf >. Acesso em: 20 out. 2015.

BREITMAN, K. VITERBO, J. **Computação na Nuvem – Uma Visão Geral**. *In*: Congresso Internacional Software Livre e Governo Eletrônico, 3, 2010, Brasília. Amãpytuna: computação em nuvem: serviços livres para a sociedade do conhecimento. Brasília: FUNAG, 2010. p. 32-33.

CARISSIMI, A. S. **Virtualização: da teoria a soluções**. In: RUBINSTEIN, M. G. *et al.* (Org.). SBRC2008 – Livro texto dos minicursos**.** Porto Alegre: Sociedade Brasileira de Computação (SBC), 2008, v. 1, p.173-207. Disponível em: <http://www.gta.ufrj.br/ensino/CPE758/artigos-basicos/cap4-v2.pdf>. Acesso em: 20 out. 2015.

CEPH. **Ceph Documentation.** Dez 2015. Disponível em: <http://ceph.com/docs/master/cephfs/>.

COULOURIS, George; DOLLIMORE, Jean; KINDBERG, Tim. **Sistemas Distribuídos: Conceitos e Projeto**. 4.ed. Porto Alegre: Bookman. 2007. 792p.

DEPARDON, B.; MAHEC, G. L.; SÉGUIN, C. **Analysis of six distributed file systems**. HAL. 2013, p44. Disponível em: <https://hal.inria.fr/hal-00789086/>. Acesso em: 27 jan. 2016

GLUSTERFS. **GlusterFS Documentation.** Nov 2015. Disponível em: <http://www.gluster.org/>.

GNUPLOT. **Gnuplot homepage.** Jan 2016. Disponível em: <http://www.gnuplot.info/>

IOZONE. **IOZone filesystem benchmark.** Jan 2016. Disponível em: <http://www.iozone.org/>.

JOHNSON, Thienne; MARGALHO, Mauro. **Avaliação de Desempenho de Sistemas Computacionais**. Rio de Janeiro: LTC. 2011. 197p.

KVM. **Kernel Virtual Machine**. Jan 2016. Disponível em: < http://www.linux-kvm.org/page/Main_Page>.

MACHADO, D. J. **algSim – Linguagem algorítmica para simulação de redes de filas.** Monografia. Departamento de Ciência de Computação e Estatística do Instituto de Biociências, Letras e Ciências Exatas, Universidade Estadual Paulista "Júlio de Mesquita", São José do Rio Preto, 2008.

MARON, C. A. F. *et al.* **Avaliação e comparação do desempenho das ferramentas OpenStack e OpenNebula**. In: 12th Escola Regional de Redes de Computadores, 12, 2014. Canoas. RS, 2014.

MARTINS, A. **Fundamentos de Computação Nuvem para Governos**. *In*: Congresso Internacional Software Livre e Governo Eletrônico, 3, 2010, Brasília. Amãpytuna: computação em nuvem: serviços livres para a sociedade do conhecimento. Brasília: FUNAG, 2010. p. 52.

MARTINS, R. D. **Avaliação de Desempenho de Sistemas de Arquivos Distribuídos.** Monografia. Departamento de Ciência de Computação e Estatística do Instituto de Biociências, Letras e Ciências Exatas, Universidade Estadual Paulista "Júlio de Mesquita", São José do Rio Preto, 2011.

OPENNEBULA. **What do you want to know about OpenNebula?**. Out 2015. Disponível em: <http://docs.opennebula.org/4.10/administration/storage/sm.html>.

PILLON, M. A. *et al.* **Comparativo entre PVFS2 e GlusterFS em um Ambiente de Computação em Nuvem.** In: 4th Computer on the Beach, 4, 2013. Anais...Florianópolis, SC, 2013.

QEMU. **QEMU**. Jan 2016. Disponível em: <https://en.wikibooks.org/wiki/QEMU>.

RAMALHO, N. C. L. **Um estudo sobre a adoção da Computação em Nuvem no Brasil**. 2012. Tese de Doutorado. Universidade de São Pulo, São Paulo, Novembro/2012.

REGO, A. L. R. **FAIRCPU: Uma Arquitetura para Provisionamento de Máquinas Virtuais Utilizando Características de Processamento**. 2012. Dissertação (Pós-Graduação em Ciência da Computação) – Departamento de Computação, Universidade Federal do Ceará, Fortaleza, Março/2012.

SOARES, T. S. **Uma arquitetura paralela para o armazenamento de imagens médicas em sistemas de arquivos distribuídos.** Dissertação (Pós-Graduação em Ciência da Computação), Universidade Federal de Santa Catarina, Florianópolis, 2012.

SOTO, J. A. **OpenNebula: implantação de uma nuvem privada e orquestração das máquinas virtuais no paradigma da Computação em Nuvem**. Manografia (Graduação em Engenharia Teleinformática), Universidade Federal do Ceará, Fortaleza, 2011.

SOUSA, F. R. C.; MOREIRA, L. O.; MACHADO, J. C. **Computação em Nuvem: Conceitos, Tecnologias, Aplicações e Desafios**. In: MOURA, R. S.; SOUZA, F. V.;

OLIVEIRA, A. C. (Org.). III Escola Regional de Computação Ceará, Maranhão e Piauí. 1ed. Piauí: Sociedade Brasileira de Computação (SBC), 2009, v. 1, p. 150-175.

TAURION, Cezar. **Cloud computing: computação em nuvem: transformando o mundo da tecnologia da informação**. Rio de Janeiro, Brasport, 2009.

VERDI, F. L. *et al*. **Novas Arquiteturas de Data Center para Cloud Computing**. In: 28th Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos, 28, 2010. Anais...Gramado, RS, 2010.

APÊNDICE A – *Script* para a ferramenta IOZone

```
#!/bin/bash
for ((i=1; i<=30; i++))
do
./iozone -a -i 0 -i 1 -s 128 -r 128 -b gluster"$i"-128k.xls
done
```

APÊNDICE B – *Script* para o tempo de criação de máquina virtual

```
#!/bin/bash
onetemplate instantiate 0
date >> arquivo.txt
while true
do
 TAG=$(onevm list | cut -d' ' -f13 | tail -1)
 NOW=$(onevm list | cut -d' ' -f29 | tail -1)
 if [ "$TAG" != "runn" ];
 then
        echo "Uma maquina esta sendo criada. Estado: $TAG - $NOW - " `date` >>
arquivo.txt
 else
        echo "Nenhuma maquina esta sendo criada. Estado: $TAG " `date` >> arquivo.txt
        break
 fi
 sleep 1
done
```

APÊNDICE C – *Script* para o tempo de remoção de máquina virtual

```
#!/bin/bash
numVM=$1
linhas=$(ls -l /[diretório datastore do OpenNebula]/ | wc -l)
echo "$linhas" `date +%H:%M:%S.%N` >> arquivo.txt
echo "Removendo a VM $numVM"
onevm delete $numVM
#date +%H:%M:%S.%N >> arquivo.txt
while true
do
 linhasnew=$(ls -l /[diretório datastore do OpenNebula]/ | wc -l)
 if [ "$linhas" == "$linhasnew" ];
 then
        echo "A VM nao foi removida" `date +%H:%M:%S.%N` >> arquivo.txt
 else
        echo "A VM foi removida" `date +%H:%M:%S.%N` >> arquivo.txt
        echo "$linhasnew"
        break
 fi
done
```

ANEXO A – Criação do *cluster* de armazenamento Ceph

1 – Configurar todos os nós, nó administrador e nós de armazenamento, com permissão de root.

```
$ sudo aptitude -y install openssh-server
$ export EDITOR=gedit && sudo visudo
nome_de_usuário        ALL=(ALL) NOPASSWD: ALL
```

2 – No nó administrador, criar um par de chaves SSH para conectar os nós Ceph sem senha.

```
$ ssh-keygen
$ sudo vim /etc/hosts
# por exemplo
192.168.0.10   maquina1
192.168.0.11   maquina2
192.168.0.12   maquina3
$ vi ~/.ssh/config
# criar um novo arquivo "config" (definir todos os nós Ceph e seus usuários), por exemplo:
Host maquina1
        Hostname maquina1
        User usuario1
Host maquina2
        Hostname maquina2
        User usuario2
Host maquina3
        Hostname maquina3
        User usuario3
```

No nó administrador, enviar a chave pública SSH para os nós, incluindo o administrador.

```
$ ssh-copy-id usuario3@maquina3
$ ssh-copy-id usuario3@maquina3
$ ssh-copy-id usuario3@maquina3
```

3 – Configurar o nó administrador.

```
$ sudo aptitude -y install ceph-deploy ceph-common ceph-mds
```

$ mkdir ceph
$ cd ceph/
Configurar o *cluster*.
$ ceph-deploy new maquina1 maquina2 maquina3
Instalação em todos os nós Ceph.
$ ceph-deploy install maquina1 maquina2 maquina3
No nó administrador, configuração inicial para monitoramento e chaves.
$ ceph-deploy mon create-initial

4 – Configurar o armazenamento no nó administrador. Antes, recomenda-se criar os diretórios que vão armazenar os dados nos nós Ceph. Por exemplo, /storage1 no host maquina1, /storage2 no host maquina2, /storage3 no host maquina3.
Preparar os OSDs (*Object Storage Deamon*).
$ ceph-deploy osd prepare maquina1:/../storage1 maquina2:/../storage2 maquina3:/../storage3
Ativar os OSDs.
$ ceph-deploy osd activate maquina1:/../storage1 maquina2:/../storage2 maquina3:/../storage3
Configurar o servidor de metadados.
$ ceph-deploy gatherkeys maquina1
$ ceph-deploy admin maquina1
$ ceph-deploy mds create maquina1

5 – Instalação no cliente.
aptitude -y install ceph-common ceph-fs-common
Obter a chave do nó administrador gerada pelo Ceph
$ssh usuario1@maquina1 "sudo ceph-authtool -p /etc/ceph/ceph.client.admin.keyring" > admin.key
$ chmod 600 admin.key
$ sudo mount -t ceph maquina1:6789:/ /[diretório que será montado]/ -o name=admin,secretfile=/[diretório do arquivo]/admin.key

Fonte: http://www.server-world.info/en/note?os=Ubuntu_14.04&p=ceph&f=1

ANEXO B – Criação do *cluster* de armazenamento GlusterFS

1 – Editar o arquivo /etc/hosts na máquina principal, com os IPs das máquinas que serão utilizadas.

$ sudo vim /etc/hosts

por exemplo

192.168.0.10 maquina1

192.168.0.11 maquina2

2 – Instalar o *software* nas duas máquinas.

$ sudo apt-get install software-properties-common

$ sudo add-apt-repository ppa:gluster/glusterfs-3.5

$ sudo apt-get update

$ sudo apt-get install glusterfs-server

3 – Configurar o *firewall* para permitir a comunicação entre os nós em ambos.

$ iptables -I INPUT -p all -s `<ip>` -j ACCEPT

4 – Na máquina principal, configurar a *trusted pool*.

gluster peer probe maquina2

5 – Configurar o volume Gluster.

gluster volume create volume replica 2 maquina1:/diretorio maquina2:/diretorio

6 – Iniciar o volume.

gluster volume start volume

7 – Montar o volume no cliente.

sudo apt-get install glusterfs-client

mount –t glusterfs maquina1:/volume /[diretório onde vai ser montado]

Fonte: http://gluster.readthedocs.org/en/latest/Install-Guide/Install/

ANEXO C – Criação do armazenamento NFS

1 – Editar o arquivo /etc/hosts na máquina principal, adicionando o IP do cliente.
Por exemplo:
192.168.0.20 cliente

2 – Instalar o *software* no servidor.
sudo apt-get install nfs-kernel-server

3 – Criar o diretório que será compartilhado.
$ mkdir /compartilhado

4 – Editar o arquivo /etc/exports com as permissões de acesso.
$ sudo vim /etc/exports
/../compartilhado 192.168.0.20(rw,sync,no_root_squash,no_subtree_check)

5 – Criar a tabela NFS que mantém as ações de compartilhamento.
$ sudo exportfs -a

6 – Iniciar o serviço NFS.
$ sudo service nfs-kernel-server start

7 – Montar o NFS no cliente.
$ sudo apt-get update
$ sudo apt-get install nfs-common
$ sudo mount [IP do servidor]:/../compartilhado /[diretório onde vai ser montado]

ANEXO D – Criação de imagem para máquina virtual OpenNebula

1 – Criar uma imagem com o Ubuntu Server 14.04 LTS 64bits. Neste exemplo a imagem utiliza 512MB de memória.

$ qemu-img create -f qcow2 ubuntu-server14.img 8G

$ kvm -hda ubuntu-server14.img -cdrom [diretório da iso]/ubuntu-14.04-server.iso -m 512 -smp 1

2 – Criar um arquivo para criar uma imagem do OpenNebula, com o usuário do OpenNebula.

$ vim imagem1.img

NAME= "imagem1"

PATH = /[diretório da imagem]/ubuntu-server14.img

TYPE = OS DRIVER

= "qcow2"

DEV_PREFIX = "hd"

Salvar o arquivo

$ oneimage create ubuntu-server14.img -d 1

3 – Criar um arquivo para criar um *template* do OpenNebula. É necessário ter uma rede virtual criada. NETWORK_ID é o id da rede que foi criada. IMAGE_ID é o id da imagem que foi criada.

$ vim ubuntu.template

NAME="ubuntu14"

CONTEXT=[

DNS="$NETWORK[DNS, NETWORK_ID=0]",

GATEWAY="$NETWORK[GATEWAY, NETWORK_ID=0]",

IP_PUBLIC="$NIC[IP]",

TARGET="hdc"]

CPU="1"

DISK=[IMAGE_ID="0"]

NIC=[NETWORK_ID="0"]

GRAPHICS=[LISTEN="0.0.0.0", TYPE="vnc"]

VCPU="1"

MEMORY="512"

Salvar o arquivo

$ onetemplate create ubuntu.template

4 – Instanciar uma máquina virtual com o *template* criado.

$ onetamplete instantiate [id do template]