Reference: Maron, C. A. F.; Vogel, A.; Griebler, D. *Caracterizando o Desempenho de Rede e Aplicações Pipeline em Ambientes de Nuvem Privada.* Laboratory of Advanced Researches on Cloud Computing (LARCC), Technical Report, 2017.

Relatório Técnico de Pesquisa (Atividades de 2016)

Nome do Projeto:

High Performance in Cloud (HiPerfCloud)

Avaliação de Ambientes de Nuvem IaaS

RT3: Caracterizando o Desempenho de Rede e Aplicações Pipeline em Ambientes de Nuvem Privada

ID do Documento:	LARCC-HiPerfCloud-RT3
Versão:	1
Autores:	Adriano Vogel, Carlos A. F. Maron, Dalvan Griebler
Objetivo:	Avaliar o Desempenho em Rede e Aplicações Pipeline
Tarefa:	Caracterizar o desempenho de rede, nas operação de criação e deletar instâncias na nuvem, nas aplicações pipeline e na agregação de links
Hardware:	2 *clusters* isolados foram criados usando 4 máquinas idênticas, cada uma com 24 GB de RAM (1333 MHz), processador Intel Xeon X5560 (quad-core 2.80GHz), discos SATA II (7200 RPM) e conectados em uma rede Gigabit (10/1000). 2 servidores multiprocessados com AMD Opteron 2425 seis *cores* e 32 GB de RAM, discos SAS de 15k RPM.
Ambiente:	Sistema Operacional (Ubuntu Server 14.04), Virtualizador (KVM e LXC), OpenStack (vers. Kilo), OpenNebula (vers. 5.0.2), CloudStack (vers. 4.8) Benchmarks de desempenho de rede NetPIPE e Uperf, Aplicações Paralelas em multi-core (PARSEC) e TestDFSIO Hadoop benchmark. Hadoop 2.7.3 e Ethernet channel bonding driver 3.7.1
Softwares:	GNUPlot (Gráficos), Latex (Documentos).

Tarefa	Responsável	Instituição	Papel	Data
Criado por:	Dalvan Griebler	SETREM/PUCRS	Coordenador	12/12/2016
Editado por:	Adriano Vogel	SETREM/PUCRS	Pesquisador	15/02/2017
Editado por:	Carlos A. F. Maron	SETREM/PUCRS	Pesquisador	25/02/2017
Revisão de Conteúdo:	Dalvan Griebler	SETREM/PUCRS	Coordenador	04/03/2017
Revisado por:	Vera Lúcia Benedetti	SETREM	Colaboradora	13/03/2017
Aprovado por:	Dalvan Griebler Ildo Corso	SETREM/PUCRS SETREM/ABASE	Coordenador Colaborador	 04/04/2017

Log de Mudanças do Documento

Versão	Autores	Instituição	Mudança	Data
1	Dalvan Griebler	SETREM/PUCRS	Versão inicial	12/12/2016
1	Adriano Vogel e Carlos Maron	SETREM/PUCRS	Versão parcial	17/02/2017
1	Dalvan Griebler	SETREM/PUCRS	Revisão, comentários e mudanças	18/02/2017
1	Adriano Vogel e Carlos Maron	SETREM/PUCRS	Correções e versão para revisão externa	22/02/2017
1	Adriano Vogel, Carlos Maron, Dalvan Griebler	SETREM/PUCRS	Versão final	01/03/2017

Lista de colaboradores internos e externos

A baixo é listado (em ordem alfabética) as pessoas que fizeram contribuições para este relatório técnico:

- Adriano Vogel (SETREM/PUCRS)
- Carlos A. F. Maron (SETREM/PUCRS)
- Claudio Schepke (UNIPAMPA)
- Dalvan Griebler (SETREM/PUCRS)

Resumo Geral

O objetivo do **Projeto HiPerfCloud (*High Performance in Cloud*)** é avaliar o desempenho em ambientes de nuvens IaaS (*Infrastructure as a Service*) e analisar características de implantação e gerenciamento nas ferramentas disponíveis. Este documento apresenta a continuidade do RT1-2015 [7] e RT2-2016 [15] e mostra novos resultados em implantações de nuvem privada baseadas em 3 ferramentas: OpenStack, OpenNebula e CloudStack.

Contexto do Relatório

Este documento é o terceiro Relatório Técnico *Implantação, Avaliação e Análise das Ferramentas para Gerenciamento de IaaS* relativo ao **Projeto HiPerfCloud** que apresenta resultados de infraestrutura onde são avaliados o desemepnho de rede, aplicações científicas e pipeline executadas em ambientes de nuvem. Também são avaliadas aplicações *big data* em uma implantação *hadoop* em nuvem, utilizando recursos para maximizar o desempenho de rede nestes ambientes.

Estrutura do Relatório

Este documento inicialmente apresenta a estrutura geral. Posteriormente, computação em nuvem é contextualizada com as ferramentas de gerenciamento OpenStack e CloudStack, seguida pela apresentação das características dos ambientes implantados, experimentos e resultados são analisados.

Sumário

1. Introdução

Este capítulo apresenta um olhar genérico e introdutório do que será discutido nesse documento, elencando o conteúdo dos capítulos e termos relevantes desse estudo.

1.1 Visão Geral

Neste documento ferramentas de código aberto para a implantação de nuvens do tipo IaaS são analisadas através de uma taxonomia e posteriormente as mais robustas são implantadas, e o desempenho das instâncias oferecidas pelas mesmas é analisado. Além do desempenho de aplicações, tópicos especiais (alta disponibilidade, armazenamento redundante, backup) para implantações de nuvem são discutidos.

1.2 Terminologia

- **Infraestrutura:** Representa os recursos de processamento: Memória RAM, armazenamento, rede e processador.
- **Aplicações Paralelas:** Área da computação de alto desempenho.
- **Benchmark:** Programa para teste específico de determinado recurso ou serviço.
- **Cluster**: Conjunto de computadores interligados por uma rede somando recursos.
- **OpenStack**: Ferramenta de código aberto para gerenciamento de infraestrutura de nuvem IaaS.
- **CloudStack**: Ferramenta de código aberto para gerenciamento de infraestrutura de nuvem IaaS.
- **NPB-MPI**: carga de trabalho composto por diferentes *kernels* de simulação de aplicações paralelas com bibliotecas MPI.
- **NPB-OMP**: carga de trabalho composto por diferentes *kernels* de simulação de aplicações paralelas com bibliotecas OMP
- **NetPIPE**: Experimento para avaliação do desempenho de rede, abrangendo protocolos TCP e UDP e possui diversos experimentos tanto de *throughput* como latência.
- **Uperf**: experimento para avaliação de rede que simula características de aplicações reais.
- **NetPipe**: ferramenta utilizada para avaliação do desempenho de rede.
- **TestDFSIO**: aplicação MapReduce utilizada para avaliação de desemepnho de um ambiente *big data*.

1.3 Estrutura deste Documento

Este documento está organizado em cinco capítulos:

- Capítulo 1: Apresenta um visão geral deste documento.
- Capítulo 2: Nesta seção, encontra-se o referencial sobre a computação em nuvem e seus serviços, seguido pela apresentação de ferramentas e dos métodos usados para avaliar o desempenho em ambientes implantados.
- Capítulo 3: Apresenta a metodologia utilizada para a execução dos experimentos nos ambientes implantados e os resultados dos testes.
- Capítulo 4: Conclusão do estudo a partir dos resultados e trabalhos futuros.

2. Contexto

Este relatório técnico consiste em realizar novos estudos envolvendo a computação em nuvem. Dessa forma, trouxemos resultados sobre um aspecto importante para uma infraestrutura de nuvem, que é a rede. O desempenho e o aprimoramento da mesma ainda não tinha sido explorado nas pesquisas anteriores. Assim sendo, na Seção 2.1 as plataformas de nuvem privada foram brevemente contextualizadas, já que foram ferramentas que utilizamos nas pesquisas anteriores e está melhor detalhado na versão anterior do Relatório Técnico [7, 15] As otimizações em rede na camada do virtualizador são discutidas na Seção 2.2. As aplicações pipelines utilizadas nos testes são abordadas na Seção 2.3. Por fim, as tecnologias de agregação de *links* e MapReduce são contextualizadas na 2.4.

Este relatório técnico é um complemento de outras pesquisas que foram publicadas em eventos nacionais e internacionais. Em 2016 os trabalhos do LARCC foram aceitos nas seguintes conferências e revistas: PDP 2017[1] [17], ERAD 2017[2] [1, 5] e REABITC[3] [6]. Além de um artigo submetido ao ICN 2017 [4].

2.1 Plataformas de Nuvem Privada

Neste estudo foram utilizadas as ferramentas de gerenciamento de infraestrutura de nuvem OpenStack e CloudStack [16]. **OpenStack**[5] é uma ferramenta robusta com código aberto, desenvolvida em *Python* para criação de ambientes em nuvem. É composto por um núcleo de tecnologias e APIs que fornecem recursos (*e.g.* processamento, armazenamento e rede) para aplicações através de um *datacenter* [8]. É uma solução modular e distribuída, que necessita de uma conexão de rede eficiente para operar.

Apache CloudStack[6] é um software de código aberto flexível e desenvolvido em Java. Ele permite a utilização de vários recursos para implantar ambientes de nuvem [3]. A arquitetura permite implementações de redes virtuais com suporte a recursos avançados, como VLANs *(Virtual LANs)*, VPN *(Virtual Private Network)*, e GRE *(Generic Routing Encapsulation)* [10].

2.2 Otimizações de Redes em Instâncias KVM e LXC

Diversas características tornaram o paradigma de computação em nuvem amplamente aceito e explorado. Se destaca a comodidade de oferecer remotamente recursos computacionais sob demanda [13]. Dentre as diversas tecnologias (Clusters, Grids, Redes, Virtualização) que são combinadas no contexto de computação em nuvem, a Rede pode ser considerada o aspecto mais importante que propriamente possibilita a utilização e provisão de recursos. Não obstante, diversos desafios (ex: segurança, desempenho, gerenciamento, usabilidade) ainda precisam ser ajustados e superados para melhorar a qualidade dos serviços em ambientes de nuvem.

Com o advento da computação em nuvem as redes de computadores convencionais passaram a não suportar as demandas por melhor gerenciamento, isolamento e desempenho nos

[1] http://www.pdp2016.org/
[2] http://www.projetos.unijui.edu.br/eradrs2017/#in
[3] http://revistas.setrem.com.br/index.php/reabtic
[4] https://www.iaria.org/conferences2017/ICN17.html
[5] https://www.openstack.org/
[6] https://cloudstack.apache.org/

datacenters virtualizados. Por isso, surgiu o conceito de redes virtualizadas e definidas a nível de *software*.

As máquinas virtuais que são oferecidas para usuários de nuvem são interconectadas e gerenciadas através de redes internas (LAN) e acessadas de qualquer lugar através das redes geograficamente distribuídas (MAN, WAN). Especificamente nas redes internas dos *datacenters* diversos esquipamentos (hardware) são utilizados e combinados com camadas de *software* (serviços, módulos, *plugins*). As funções de redes são virtualizadas sendo um adicional integrado aos amplamente utilizados virtualizadores.

Uma melhor utilização dos servidores computacionais foi possível através da utilização de máquinas virtuais, pois diversas podem ser alocadas no mesmo ambiente físico. As máquinas virtuais podem ser criadas utilizando virtualizadores. No meio científico e de soluções de código aberto, umas das soluções de virtualização mais aceita é o KVM (*Kernel-based Virtual Machine*) [4]. Diversas máquinas virtuais baseadas em vários sistemas operacionais podem ser executadas nos mesmos servidores e ainda existem camadas de isolamento entre as VMs.

Outra solução popular nesse contexto são os ambientes baseados no LXC (*Linux Containers*) que possui contrastes com o KVM. Principalmente por utilizar um número menor de abstrações e usar o mesmo *kernel* compartilhado para todos os *containers*. O isolamento do LXC ocorre através de *namespaces* e grupos de controle. A Figura 2.1 mostra uma visão básica dos contrastes de rede entre ambientes KVM e LXC. Máquinas virtuais executadas a partir do KVM recebem um novo *kernel* que é emulado no servidor hospedeiro, enquanto contêineres LXC usam o mesmo *kernel* do ambiente nativo. Isso resulta em diversos contrastes relacionados com diversos aspectos como segurança (isolamento de recursos, usuários), gerenciamento e desempenho. No escopo desse documento, são explorados contrastes relacionados com desempenho.

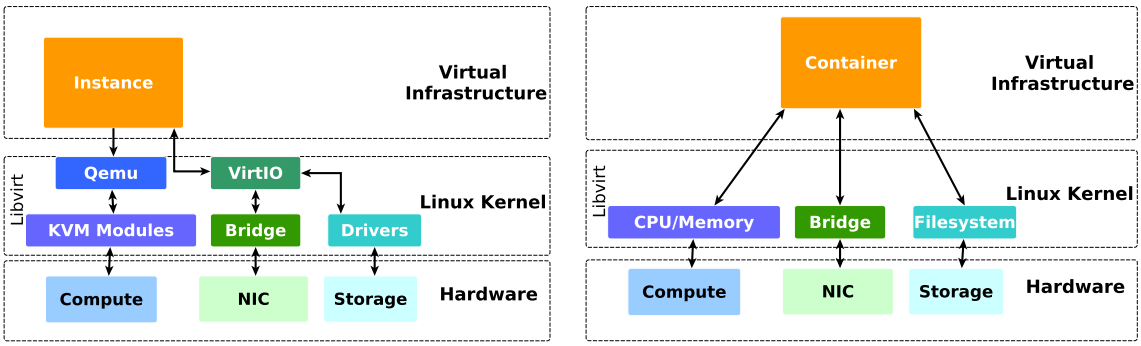


Figura 2.1: Visão Geral da Arquitetura de Rede, KVM (Esquerda) e LXC (Direita).

2.3 Aplicações Pipeline

Aplicações pipeline assemelham-se com uma linha de produção de uma indústria. A estrutura algorítmica da aplicação é composta por uma sequência ordenada de estágios e que computam elementos que percorrem através deles. Esses elementos compõem um fluxo contínuo e em cada estágio existe um produtor e um consumidor. A Figura 2.2 representa o grafo pipeline de uma aplicação com 3 estágios. O estágio **1** consome os elementos do *stream* e realiza uma primeira computação repassando (produzindo) ao estágio seguinte. O estágio **2** realiza outro tipo de computação e da mesma forma repassa ao estágio subsequente. Por fim, o estágio **3** fará a saída dos elementos já processados.

Quando uma aplicação pipeline possuí estágios que podem ser replicados, seu grafo apresenta uma estrutura um pouco diferenciada. Na Figura 2.3 tem um pipeline com o segundo

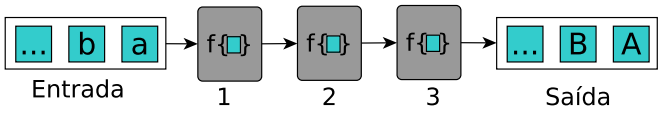


Figura 2.2: Grafo do modelo pipeline

estágio replicado. O grafo neste caso é chamado de *farm* e para que seja possível a replicação, os elementos que serão processados neste estágio não podem ter dependências com outros elementos. Desta forma, a estrutura de código nesta região trabalha de maneira independente.

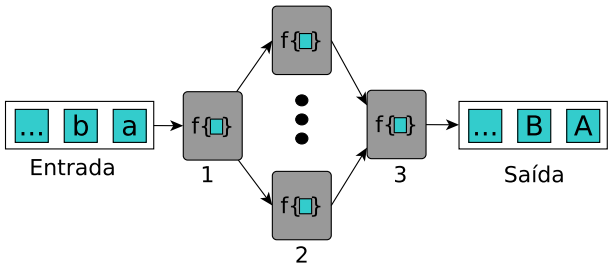


Figura 2.3: Grafo do modelo *farm*

Geralmente, uma estrutura pipeline é encontrada em aplicações de áudio, vídeo e grande volume de dados. A aplicação Ferret é um exemplo de processamento de vídeo e utiliza uma estrutura algorítmica em pipeline. Por outro lado, Dedup é uma aplicação usada para a otimização do armazenamento em grandes volumes de dados e que também utiliza a estrutura pipeline. A suíte de *benchmarks* PARSEC disponibiliza estas aplicações em conjunto com outras diversas para a avaliação de desempenho. Dedup e Ferret serão detalhados nas seções seguintes.

2.3.1 Ferret

Ferret é uma aplicação que realiza a busca por similaridade de dados em áudio, imagens, vídeos e formas 3D. Ela é paralelizada utilizando um modelo de seis estágios (Figura 2.4). O primeiro e o último são sequenciais pois envolvem a entrada e saída dos dados no pipeline. Nesta aplicação é utilizado um processo de decomposição da imagem em segmentos. No estágio 2, cada área segmentada mostra diferentes os objetos e são atribuídos pesos maiores às partes das imagens que são consideradas relevantes para a busca (semelhantes). O estágio 3 armazena em um vetor multi-dimensional uma estrutura de dados contendo a descrição matemática dos conteúdos dos segmentos analisados, as propriedades de cores, formas e área das imagens. Por fim, quando os vetores são completados, o estágio 4 fará a pesquisa e indexação das imagem em um banco de dados para obter as similares. Ao indexá-las, o estágio 5 computa e ordena de acordo com um *ranking* calculado [2].

2.3.2 Dedup

Dedup, ou *Data Deduplication*, é uma aplicação de compressão de dados que elimina dados redundantes em um certo conjunto de dados. É uma aplicação utilizada em sistemas de *backups*, pois reduz consideravelmente o tamanho dos dados processados, otimizando a utilização do armazenamento e também da largura de banda da rede.

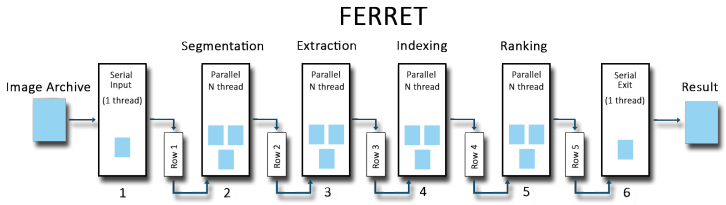


Figura 2.4: Estrutura da aplicação Ferret

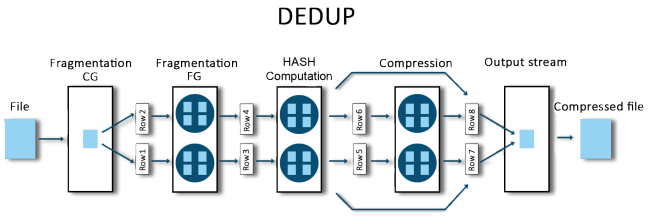


Figura 2.5: Estrutura da aplicação Dedup

Dedup é uma aplicação composta por 5 estágios (Figura 2.5). **Fragmentação com granularidade alta:** estágio sequencial que faz a leitura do *stream* de entrada e que particiona em *chunks* de granularidade alta. **Fragmentação com granularidade baixa:** estágio que processa em paralelo e utiliza o algoritmo Rabin *fingerprint* para particionar os elementos em *chunks* de baixa granularidade. **Computação Hash:** Neste estágio ocorre a identificação dos *chunks* através de uma chave HASH individual. **Compressão:** em paralelo comprime os blocos de dados utilizando o algoritmo Lempel-Ziv construindo uma uma tabela *Hash* global mapeando os valores destes dados. **Montagem do *stream* de saída:** reordena os blocos de dados e produz a saída já comprimida [2].

2.4 Agregação de Links

A rede é uma das principais características de uma nuvem. Através dela, toda a infraestrutura se torna funcional, pois interliga os recursos físicos e torna possível o seu provisionamento, o gerenciamento remoto e a comunicação interna e externa de serviços. Muitos são os papeis importantes que a rede deve desempenhar. Do mesmo modo, ela também é parte fundamental para as aplicações que executam na nuvem, visto que em boa parte dos casos um *cluster* é alocado em nuvem para unir processamento. Como é o caso de aplicações *Big Data* que processam grandes volumes de dados e precisam de um grande poder computacional para processá-los.

Por conseguinte, a rede é um dos aspectos que envolve estudos na área de tecnologia e por isso tecnologias surgem para melhorar esse aspecto, como é o caso das redes SDN (*Software Define Network*). Além disso, protocolos de rede também são desenvolvidos para que a rede consiga obter um desempenho maior, driblando algumas limitações físicas do ambiente. Por exemplo, o protocolo IEEE 802.3ad que permite a agregação de links e possibilita uma vazão maior na rede e com uma latência menor. Neste sentido, foram realizados testes com este protocolo utilizando aplicações Big Data. Na seção seguinte o protocolo será melhor detalhado.

2.4.1 IEEE 802.3ad e Apache Hadoop MapReduce

O Hadoop foi originalmente construído para usar o sistema de arquivos distribuído (HDFS) juntamente com o sistema de processamento distribuído (MapReduce) a partir de um cluster

dedicado de servidores [18]. Nesse tipo de *cluster*, cada servidor é considerado um nodo. Usualmente, o nodo mestre coordena a execução de *daemons* como Job Tracker e Name Node. Na outra extremidade, os nodos *slaves* armazenam blocos de dados e realizam a computação sobre esses dados através de *daemons* como Task Tracker e Data Node. A Figura 2.6 ilustra a arquitetura Hadoop[18, 12].

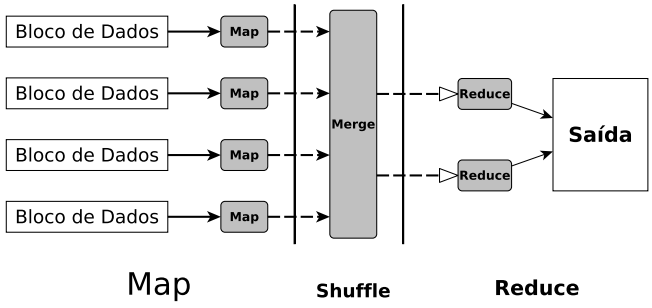


Figura 2.6: Representação dos estágios do MapReduce

Nesse sentido, a Figura 2.7 ilustra uma visão geral da arquitetura utilizando o IEEE 802.3ad. Note que Job Tracker supervisiona e coordena o processamento paralelo de dados usando MapReduce, enquanto o Name Node supervisiona e coordena a função de armazenamento de dados (HDFS), por intermédio do nodo mestre. Por outro lado, cada escravo executa um *daemon* Data Node e um Task Tracker que se comunicam e recebem instruções dos seus nodos mestres.

Assim, o mecanismo de computação distribuída do Hadoop (incluindo HDFS e MapReduce) se beneficia do uso da agregação de *links*. Utilizar esse recurso extra busca reduzir o tempo de transmissão para a entrega dos resultados computacionais entre os estágios de Map e Reduce, pois terá o aumento da largura de banda disponível com o IEEE 802.3ad.

Além desse beneficio, também é esperado a diminuição da latência. Pois nesse caso, as interfaces adicionais que formam a agregação são identificadas como uma nova rota para a informação trafegar. Dessa forma, vários tráfegos pode ser priorizados por vários *links* simultânetamente.

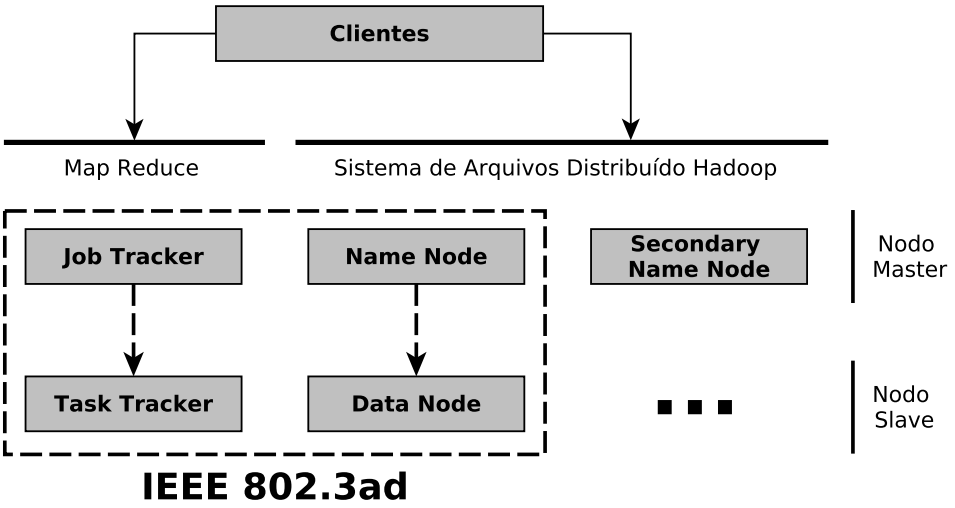


Figura 2.7: Representação do Hadoop com IEEE 802.3ad

3. Resultados

Como apresentado com Capítulo 2, as ferramentas para implantação de ambientes de nuvem suportam diversas tecnologias e configurações. Nessa seção, ambientes de nuvem usando ferramentas distintas foram implantados e o objetivo principal é comparar os resultados e apresentar um viés inovador sobre eles. Na Seção 3.1 são apresentados resultados da infraestrutura e aplicações científicas. Na Seção 3.3 são apresentados os resultados de experimentos com aplicações pipeline utilizando a biblioteca de programação paralela *pthreads*. Por fim, na Seção 3.4 são mostrados os resultados com o protocolo IEEE 802.3ad para agregação de *links*. Neste experimento, medimos o desempenho de rede (vazão e latência) no ambiente CloudStack utilizando *benchmarks* e aplicações MapReduce.

3.1 Avaliação do Desempenho de Rede em KVM e LXC

Diversas tecnologias são usadas em implantações de ambientes de nuvem, tanto em clientes quanto em provedores. Um dos aspectos mais relevantes é o desempenho da rede interna nesses ambientes. Por isso, nessa Seção são apresentados resultados de avaliação e comparação do desempenho de rede em implantações de nuvem.

As tecnologias de virtualização KVM e LXC apresentadas na Seção 2.2 foram usadas para implantação de ambientes de nuvem usando a ferramenta de gerenciamento de infraestrutura de nuvem CloudStack. Para comparar os resultados foi utilizado hardware idêntico e o desempenho de rede foi medido entre duas instâncias (máquinas virtuais) em cada um dos ambientes.

Os testes utilizados para avaliar o desempenho da rede nos ambientes de nuvem foram NetPIPE *(Network Protocol Independent Performance Evaluator)* [11] que foi utilizado para medir a vazão e latência e o teste Uperf [14] *(Unified Performance tool for networking)* que propõe ser um teste mais realista pois gera carga real que simula comportamento usual de serviços e requisições de rede, sendo uma infraestrutura de rede *gigabit* e usando o protocolo *ethernet*.

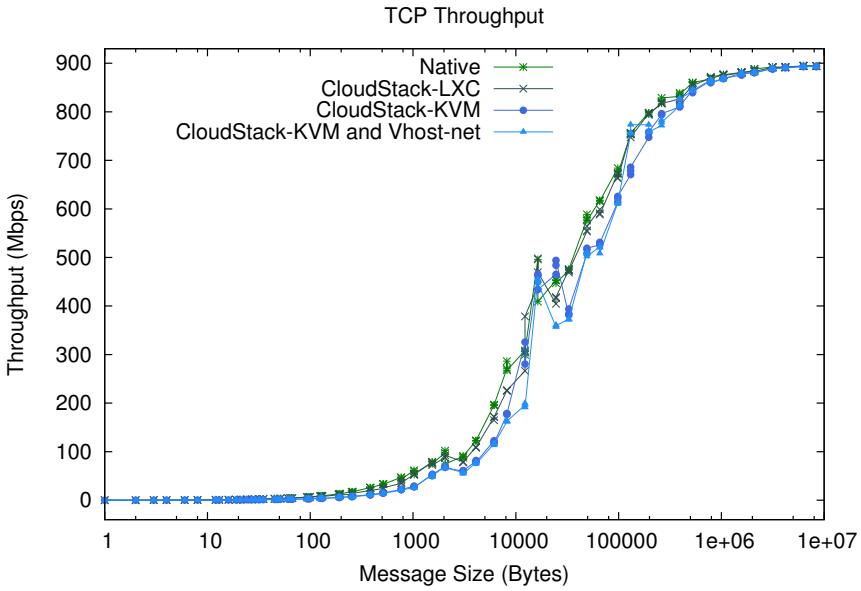


Figura 3.1: Vazão TCP entre duas instâncias.

A Figura 3.1 apresenta os resultados relacionados com a vazão de rede com diferentes ta-

manhos de pacotes. Iniciando com pacotes 1 byte e aumentando exponencialmente até 10 MB. Como esperado, o teste executado no ambiente nativo sem virtualização apresentou a melhor vazão e seguido próximo pelo ambiente LXC. As máquinas virtuais usando o KVM tiveram uma vazão um pouco menor, com pacotes menores tento mais perdas e com pacotes maiores tendo um desempenho muito próximo ao ambiente nativo.

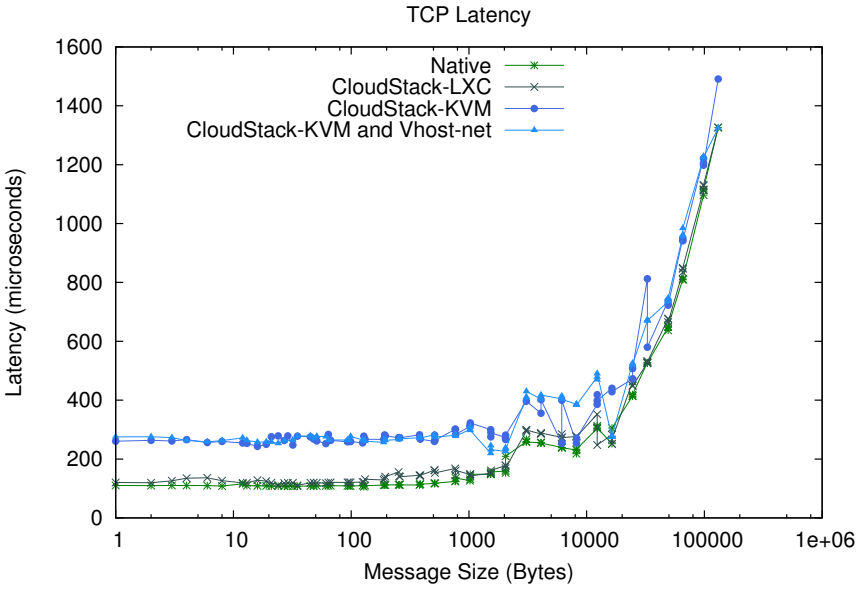


Figura 3.2: Latência TCP entre duas instâncias.

Latência também é uma métrica relevante para o desempenho de aplicações que utilizam a rede. Na Figura 3.2 são apresentados resultados dos testes nos ambientes implantados usando pacotes de 1Byte até 100 KBytes. Se percebe que o nativo teve o melhor desempenho com as menores latências seguido pelo resultados dos contêineres LXC. O desempenho no ambiente KVM com pacotes maiores foi relativamente bom, porém com pacotes menores se mostrou pobre tendo o dobro da latência comparada ao ambiente nativo.

O Uperf foi utilizados para avaliar o desempenho de tráfego TCP, o que pode ser aplicado para diversas aplicações, como servidores web e transferência de arquivos, sendo avaliado o número de conexões TCP estabelecidas por segunda como uma métrica de vazão. O desempenho foi medido em execuções durante 90 segundos, e o ambiente nativo novamente foi o melhor, seguido pelo contêineres LXC. O desempenho do ambiente usando KVM mostrou um desempenho pobre principalmente com 8, 16 e 32 *threads*.

Um dos objetivos dessa pesquisa também foi buscar soluções para otimizar desempenho em ambientes de nuvem. Portanto, como o desempenho usando KVM foi pobre em alguns aspectos, se buscou otimizar o desempenho. KVM suporta diversos módulos e serviços que podem ser customizados, o módulo vhost-net se mostra como uma alternativa para melhorar o desempenho de rede pois este diminui o número de abstrações entre a placa de rede e as máquinas virtuais, sendo separado do *Kernel* Linux como mostrado na Figura 3.4. Um ambiente adicional usando esse módulo foi implantado e os resultados foram adicionados na Figura 3.3, evidenciando uma melhora no desempenho de rede.

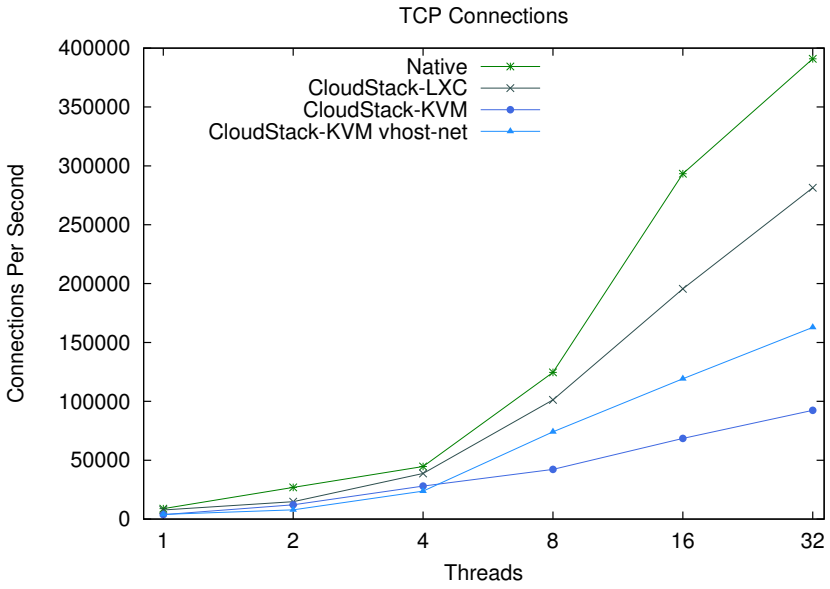


Figura 3.3: Teste de Conexões TCP usando o Uperf.

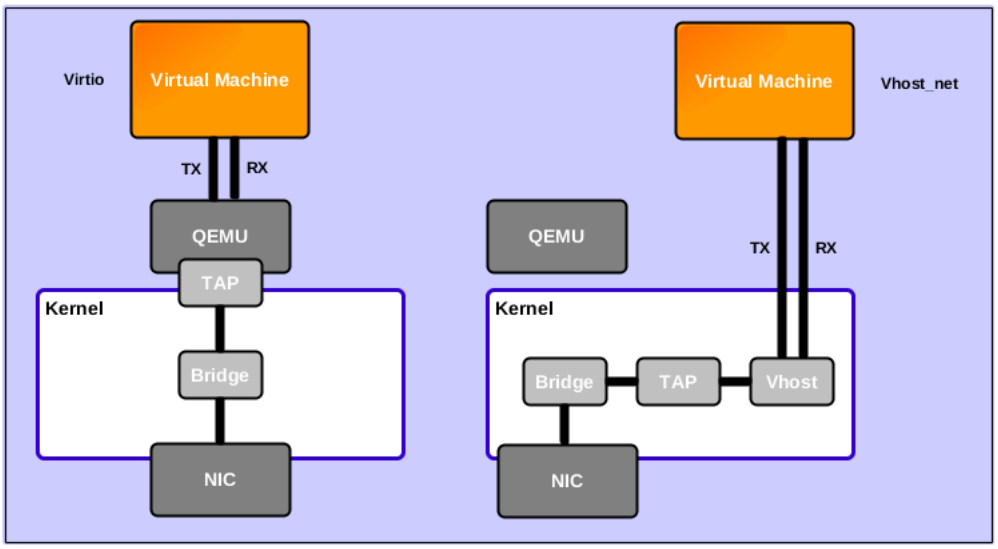


Figura 3.4: Arquitetura de rede usual do KVM e Vhost-net.

3.2 Desempenho da Operações de Gerenciamento das Plataformas de Nuvem

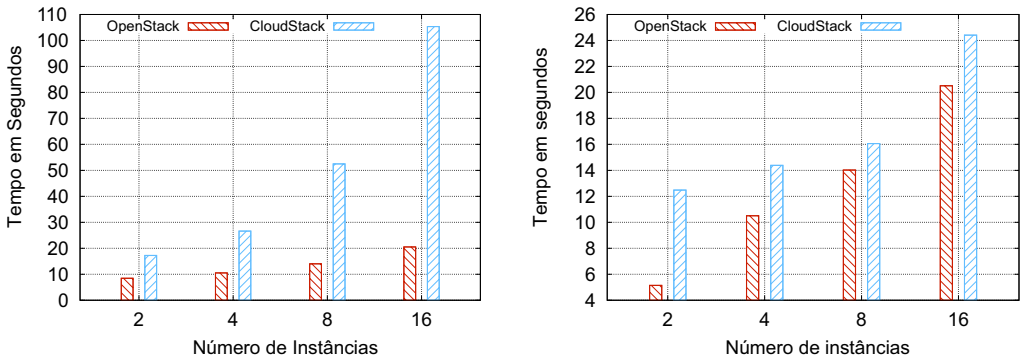
Outro aspecto relevante é o desempenho das plataformas de gerenciamento de nuvem na provisão de recursos. Por isso, foi avaliado o desempenho de criar e deletar máquinas virtuais usando as ferramentas OpenStack e CloudStack. As operações foram testadas de forma concorrente, implantando 2, 4, 8 e 16 VMs. As métricas utilizadas para comparar as plataformas foram o tempo decorrido entre a requisição que foi feita até a última requisição fosse completada.

As Figuras 3.5(a) e 3.5(b), mostram os resultados das operações nos ambientes de nuvem, para executar as requisições de criação e exclusão, variando o número de instâncias. Para a execução de experimentos, dois *clusters* isolados foram criados usando 4 máquinas idênticas, cada uma com 24 GB de RAM, processador Intel Xeon X5560, discos SATA II (7200 RPM) e conectados em uma rede Gigabit (10/1000). No ambiente foi utilizado o sistema operacional

Ubuntu Server 14.04 (*kernel* 3.19.0).

O tempo para criação das instâncias nas ferramentas de nuvem foi significativamente diferente comparando entre OpenStack e CloudStack. Por exemplo, para criar 2 instâncias OpenStack necessitou de 8,45s, enquanto que CloudStack levou 17,22s. Além disso, quanto maior o número de instâncias utilizadas, maior a diferença de desempenho entre as ferramentas. O tempo de criação usando CloudStack cresceu proporcionalmente ao número de instâncias criadas, enquanto com OpenStack não teve um aumento linear do tempo necessário.

O tempo de exclusão das instâncias também foi melhor no ambiente OpenStack, mas tento contrastes menores comparado com a criação de VMs. Outro aspecto relevante é que os dois ambientes de nuvem utilizaram cópia local das imagens usadas para criar as VMs. Portanto, o sistema apenas transferiu a imagem das instâncias na primeira requisição e manteve uma cópia local nos *hosts* que foi apenas copiada nas operações posteriores, evitando assim um tráfego excessivo pela rede.



(a) Tempo para criar instâncias. (b) Tempo para excluir instâncias.

Figura 3.5: Resultados das operações das plataformas.

3.3 Desempenho de Aplicações Pipeline

Nesta seção são discutidos os resultados obtidos com a execução das aplicações Ferret e Dedup do PARSEC em uma implantação CloudStack com instâncias KVM e LXC. Estas ferramentas foram contextualizadas no Capítulo 2. A média dos tempos de execução (10x) estão plotados nas Figuras 3.6 e 3.7, assim como o desvio padrão. O ambiente implantado utilizou Ubuntu Server 14.04 64bits em todos os testes, armazenamento distribuído com NFS (*Network File System*) em uma rede 10/1000 Mb/s. As instâncias KVM utilizaram discos no formato qcow2. As instâncias provisionadas utilizaram a capacidade total dos nodos (24GB memória RAM e processador Intel Xeon X5560 2.80 GHz).

Os resultados das aplicações pipeline do PARSEC revelaram que Dedup exige uma alta demanda de E/S e uma grande sincronização entre os estágios. Com isso, o desempenho piora em uma instância KVM à medida que o número de *threads* aumenta, enquanto que na instância LXC o desempenho se assemelha ao apresentado na máquina nativa. O fluxo e o comportamento desta aplicação pode ser visto na Figura 2.5, onde operações em disco também podem ser adiantadas no terceiro estágio, ocasionando uma concorrência com o último. Assim, além de realizar a escrita em disco, também é preciso fazer a reordenação dos dados processados.

O sistema operacional por padrão implementa mecanismos de releitura e reescrita utilizando memórias caches. Isto é, ao refazer estas operações em uma mesma região de disco feita anteriormente, o sistema operacional utiliza essas memórias para agilizar estas operações. No

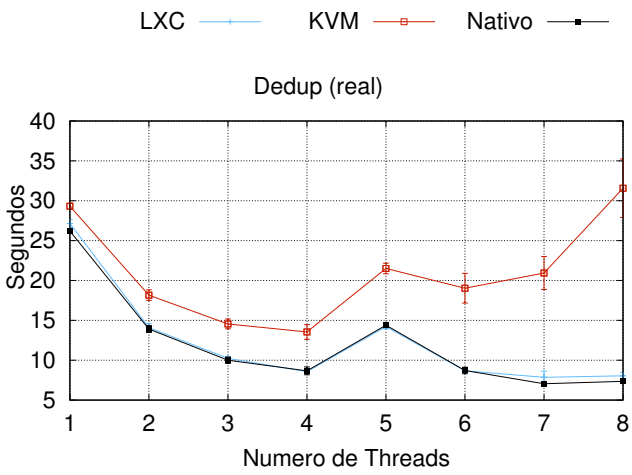


Figura 3.6: Desempenho com aplicação Dedup

virtualizador KVM este desempenho não é totalmente portado ao sistema operacional em execução na nuvem, já que o KVM usa um formato específico de disco (qcow2) [9]. Enquanto LXC realiza essas operações na mesma camada do sistema operacional e de arquivos.

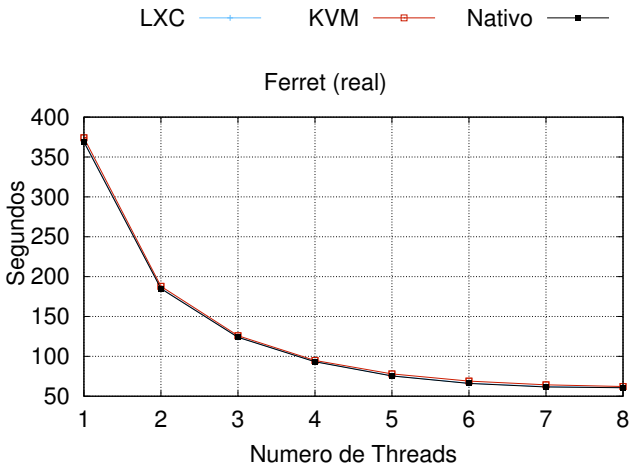


Figura 3.7: Desempenho com aplicação Ferret

Na aplicação Ferret pouca diferença é percebida, mostrando que a característica do virtualizador não impacta na aplicação. A maior parte da carga de trabalho é realizada na CPU e memória, o que demonstra que ambos os tipos de instâncias se comportam bem em relação às características em Ferret (descrito na Seção 2.3.1), atingindo desempenho semelhante ao ambiente nativo.

Portanto, ao contrastar as características das aplicações pipeline, percebemos que em Dedup a utilização dos recursos E/S são mais evidentes e em maior parte do tempo, diferente do Ferret onde o processamento concentra-se no processador e memória. Deste modo, o maior diferença entre os resultados foram obtidos com a aplicação Dedup, evidenciando a necessidade de aprimoramento das operações de E/S no virtualizador KVM.

3.4 Desempenho de Agregação de Links utilizando IEEE 802.3ad

Para testar a agregação de *links* e validar seus resultados, foi utilizado duas instâncias com Ubuntu Server 14.04 rodando em uma nuvem CloudStack, utilizando o virtualizador LXC. Para o teste de agregação, o protocolo IEEE 802.3ad foi utilizado no ambiente Nativo com duas interfaces de rede *gigabit*. Ou seja, quando utilizado o protocolo de agregação, a instância na nuvem tem disponível uma interface de 2 Gb/s teórico. As cargas de trabalho utilizadas foram o *benchmark* NetPipe e TestDFSIO disponível nos *benchmarks* do Apache Hadoop. Para isso, foram montados três cenários de testes que serão descritos a seguir utilizando a infraestrutura de *hardware*:

- Cenário 1: foram realizados experimentos para verificar essencialmente o *thoughput* e a latência. Nessa etapa, utilizou-se o *benchmark* NetPipe, primeiramente, no ambiente nativo (Regular TCP[1] e IEEE 802.3ad) e posteriormente na nuvem utilizando duas instâncias com o LXC (Regular TCP e IEEE 802.3ad).

- Cenário 2: nesse cenário foram realizados experimentos para verificar a taxa de utilização da largura de banda e o tempo de execução das aplicações *big data*. Os experimentos utilizaram o *benchmark* TestDFSIO, usando apenas a configuração das instâncias virtualizadas com o LXC sem a agregação (regular TCP). O TestDFIO foi executado com um fator de replicação de até três instâncias da aplicação em paralelo, ou seja, até três aplicações rodando ao mesmo tempo.

- Cenário 3: esse cenário apresenta as mesmas características do cenário 2, com o diferencial de utilizar em sua configuração as instâncias virtualizadas com o LXC e o protocolo IEEE 802.3ad. O TestDFSIO foi executado mantendo o mesmo padrão utilizado no cenário 2.

Os testes foram executados em 2 servidores multiprocessados com AMD Opteron 2425 seis *cores* e 32 GB de RAM, discos SAS de 15k RPM. Foi utilizado o TestDFSIO Hadoop benchmark na versão do 2.7.3 do Hadoop e Ethernet channel bonding driver 3.7.1.

O primeiro cenário consiste em analisar o *throughput* e a latência através da execução do *benchmark* NetPipe usando o TCP padrão e o IEEE 802.3ad *link aggregation* a partir de dois servidores Nativos. Em seguida, o mesmo experimento foi realizado com IEEE 802.3ad utilizando duas instâncias LXC agregando duas interfaces de rede em cada.

Ao utilizar o protocolo IEEE 802.3ad (gráfico verde) o *thoughput* mostrado na Figura 3.8 apresenta levemente melhor desempenho ao IEEE 802.3ad-LXC (grafico azul). Isso remete ao LXC, pois o *overhead* gerado por ele é baixo. Em ambos os ambientes testados, a agregação de *link* provou ser capaz de aumentar a largura de banda de maneira efetiva com os testes do NetPipe. Os resultados com protocolo tiveram sua largura de banda quase dobrada, quando comparado ao TCP regular (gráfico ciano).

Consequentemente, na latência também houve ganhos. Nesse caso, é possível observar na Figura 3.9 que a latência do TCP regular (gráfico ciano) apresentou o pior tempo possível. No entanto, o IEEE 802.3ad-LXC (gráfico azul) e IEEE 802.3ad Nativo (gráfico verde) apresentaram tempos de latência praticamente iguais. Nesta situação, mais uma vez o baixo *overhead* gerado pela camada de virtualização do LXC foi fundamental para o comportamento do *benchmark* nestes ambientes. Portanto, a agregação de *link* provou também ser capaz de reduzir a latência, algo bastante desejável e importante em um ambiente de *cloud computing*.

[1] Sem agregação de *links*. Somente com uma interface de rede com tráfego de 1Gb/s.

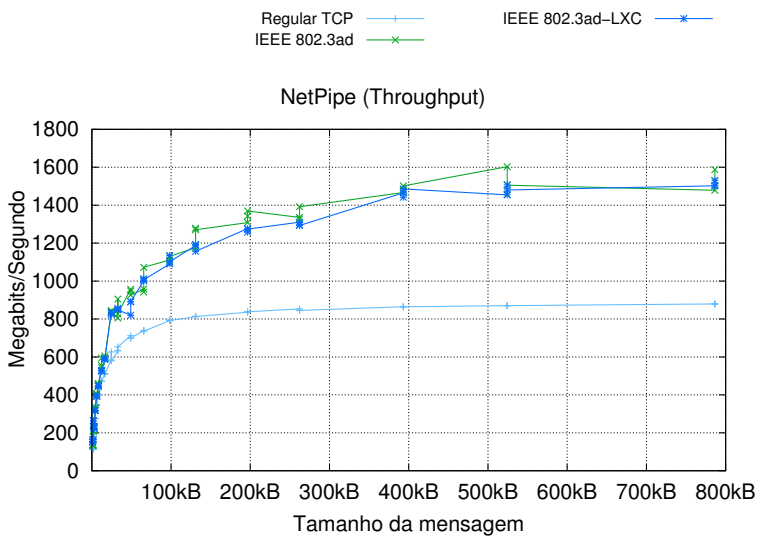


Figura 3.8: Teste de *throughput*

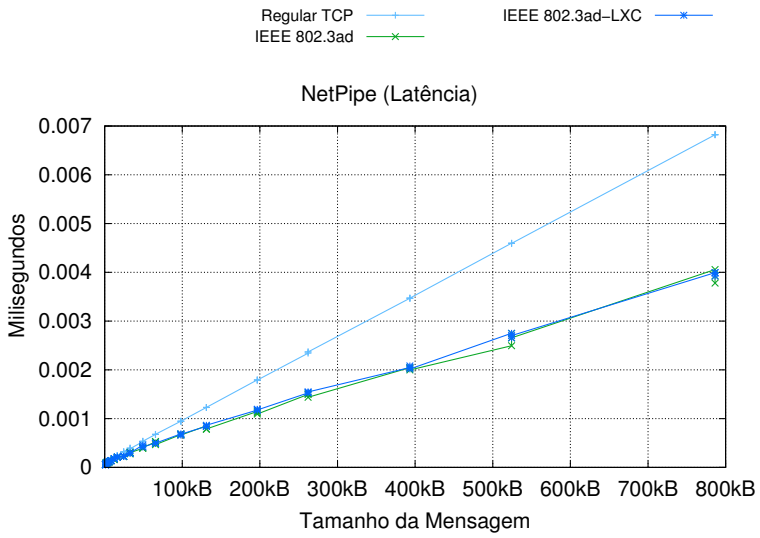


Figura 3.9: Teste de latência

Em resumo, os resultados apresentados neste primeiro cenário, demonstram claramente um desempenho equivalente do LXC em relação ao ambiente Nativo. O uso da agregação de *link* em conjunto com o ambiente virtualizado LXC demonstrou um expressivo aumento no *throughput*, além de permitir a redução da latência quando comparado com o ambiente TCP regular.

O segundo cenário consiste em avaliar o desempenho do Hadoop executando o *benchmark* TestDFSIO utilizando somente o TCP padrão. Nesse caso, o *Master node* Hadoop realiza a comunicação com o *Slave node* Hadoop a partir de uma única interface de rede em cada instância. A virtualização baseada em contêineres LXC no CloudStack foi configurada em modo bridge, de forma a aproveitar toda a largura de banda disponível, permitindo ao Hadoop uma taxa nominal de 1 Gbps para o experimento. Obviamente esse é o valor máximo teórico que o ambiente pode atingir, uma vez que o IEEE 802.3ad ainda não foi habilitado no contexto do experimento no cenário dois.

Observa-se que o comportamento do TCP regular com uma instância, ilustrado na Figura 3.10, não apresentou gargalo na rede durante a execução, apesar da sua taxa de utilização do *link* ter sido bastante elevada.

Por outro lado, os resultados apresentados na Figura 3.11 demonstram que a execução do

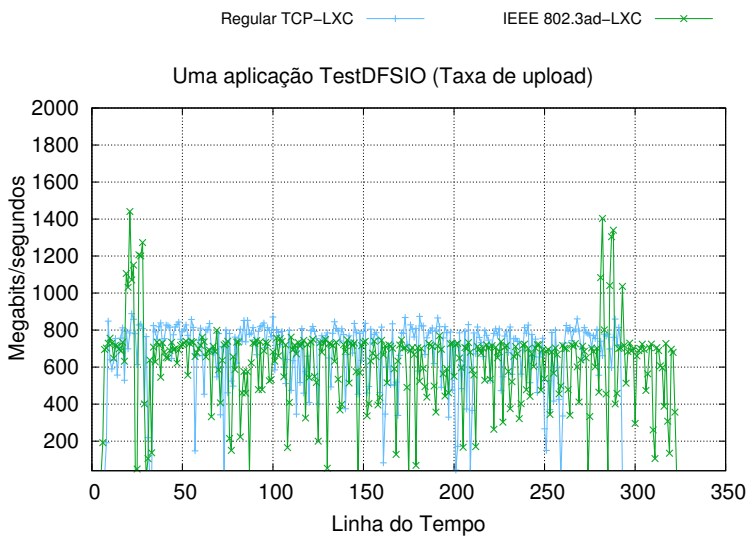


Figura 3.10: Aplicação HDFS-IO Hadoop (x1)

TCP regular com duas instâncias utiliza praticamente toda a capacidade do *link* disponível (taxa de 1 Gbps nominal). Na Figura 3.11, onde 3 instâncias da aplicação são executadas, a situação é ainda pior, pois a rede encontra-se totalmente saturada, não havendo recursos suficientes para outras aplicações. O resultado disso se reflete no tempo para a execução destas aplicações. Em outras palavras, a rede sendo o principal gargalo devido a alta latência das mensagens e alta concorrência das aplicações por esse recurso, influência diretamente no desempenho das aplicações.

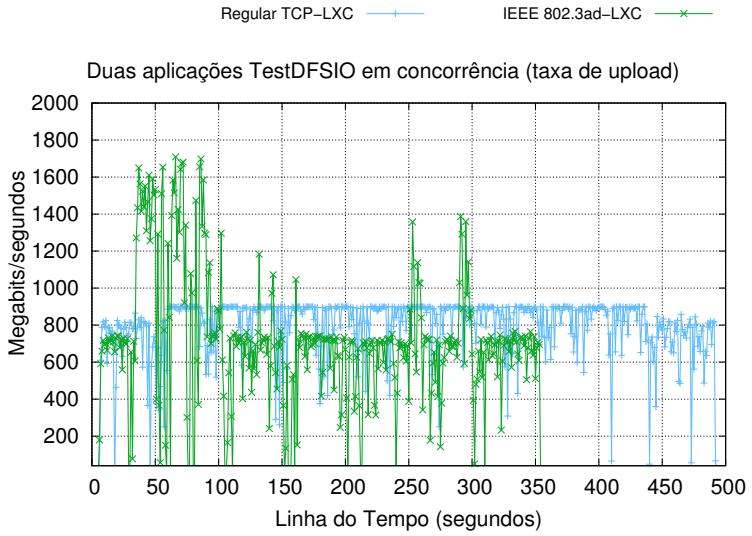


Figura 3.11: Aplicação HDFS-IO Hadoop (x2)

Nesse sentido, a configuração do cenário dois tem como objetivo principal avaliar o desempenho do Hadoop utilizando a estratégia de agregação com IEEE 802.3ad. A configuração experimental foi novamente reproduzida, no entanto, a comunicação entre o *Master* e o *Slave node* Hadoop foi montada sobre a agregação de dois *links* no ambiente Nativo. Dessa forma, o ambiente da nuvem foi configurado para habilitar uma interface de rede com uma largura de banda agregada teórica de 2 Gb/s.

Os resultados obtidos através do aprimoramento da rede para o Hadoop podem ser visualizados a partir das figuras 3.10, 3.11 e 3.12. Fica evidente, principalmente na Figura 3.12, que a taxa de utilização do IEEE 802.3ad-LXC (grafico verde) atinge em torno de 1.8 Gb/s. Em

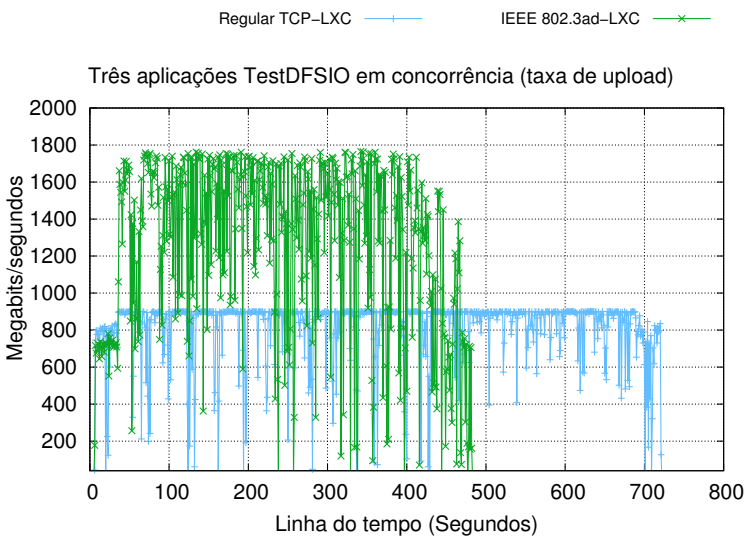


Figura 3.12: Aplicação HDFS-IO Hadoop (x3)

relação ao TCP Regular-LXC (grafico ciano), esse valor é praticamente o dobro do seu valor máximo atingido. Assim sendo, percebe-se que em ambientes com muita concorrência ao acesso a rede, as aplicações *big data* são positivamente afetadas com essa abordagem.

Além disso, a abordagem IEEE 802.3ad-LXC favorece também em momentos de picos, quando somente um *link* seria suficiente para as execuções em docorrer do tempo. Situação que pode ser vista na Figura 3.10, onde tem-se um pico inicial no uso da rede que chega até 1.4 Gbps (grafico verde).

A Tabela 3.1 apresenta um resumo dos experimentos realizados nos cenários dois e três utilizando o *benchmark* TestDFSIO. A partir desses resultados, é possível verificar os benefícios que a rede propicia em relação ao tempo de execução (em segundos) das aplicações *big data*. Na execução do TestDFSIO com três instâncias, por exemplo, a redução no tempo de execução foi de até 33.73% usando IEEE 802.3ad. Cabe destacar que essa é a situação com o maior nível de exigência concorrente para a rede.

Tabela 3.1: Tempos da aplicação TestDFSIO em Regular TCP e IEEE 802.3ad nas instâncias na nuvem.

	Número de Instâncias da aplicação TestDFSIO		
Ambiente	**1** (segundos)	**2** (segundos)	**3** (segundos)
IEEE 802.3ad (LXC)	338	375	501
Regular TCP (LXC)	315	507	756

4. Conclusão

Neste relatório técnico foram apresentados novos experimentos relacionados a computação em nuvem e computação de alto desempenho. Para isto, na Seção 2.1 foram contextualizados as ferramentas de computação em nuvem. As ferramentas de virtualização e aprimoramentos que envolvem a rede nestas ferramentas são abordados na Seção 2.2. Dedup e Ferret – as aplicações pipeline utilizadas nos experimentos – são descritas na Seção 2.3. A tecnologia Hadoop e o protocolo IEEE 802.3ad utilizado para agregação de links é descrito na Seção 2.4.

Os resultados do desempenho de rede em ambientes de nuvem mostram contrastes entre as diferentes métricas de desempenho. Considerando a latência, os resultados encontrados foram os esperados de acordo com resultados da literatura, nos quais o ambiente nativo teve o melhor desempenho por ter menos abstrações. O desempenho no ambiente usando LXC se mostrou bom, muito próximo ao ambiente nativo e levemente melhor que o desempenho do KVM (0.2% a 8%).

Por outro lado, os resultados da latência evidenciam perdas de desempenho no ambiente usando o virtualizador KVM, principalmente com pacotes pequenos transmitidos. Os resultados dos testes com Uperf mostraram novamente o ambiente de contêineres LXC tendo um desempenho próximo ao nativo e o KVM apresentando perdas significativas de desempenho. Isso motivou a tentativa de otimizações no desempenho da rede nos módulos do KVM. Dessa forma, foi possível customizar o ambiente e resultar em melhoras no desempenho.

Aspectos de desempenho do gerenciamento da infraestrutura foram avaliados nas ferramentas OpenStack e CloudStack em operações simultâneas de criar e excluir máquinas virtuais. Os resultados evidenciam que a ferramenta OpenStack foi significativamente melhor que CloudStack nesses testes.

Os resultados com as aplicações Ferret e Dedup mostram resultados importantes nos aspectos das operações de E/S na nuvem. Como pode ser observado na Seção 3.3, no KVM ainda é necessário amadurecer as operações relacionadas à disco, pois foi onde os resultados com a aplicação Dedup levaram mais tempo para serem processados. Ao contrário do LXC, onde os resultados foram melhores e semelhantes ao ambiente Nativo. As operações das aplicações que envolvem memória e processador apresentaram resultados satisfatórios, pois não houveram contrastes entre os ambientes na aplicação Ferret.

Nos testes com a agregação de *links*, o protocolo IEEE 802.3ad se mostrou muito eficaz para o desempenho da rede. Para as aplicações MapReduce que usam exaustivamente os recursos deste tipo, a melhoria foi de 33.73%. Valor considerado importante para uma infraestrutura de computação em nuvem. Pois, a redução no tempo de execução, está vinculada a uma economia de recursos, tanto para o cliente como o provedor. A aplicação executando mais rápido, traz os resultados em menos tempo para que as estratégias do negócio sejam adiantadas.

Esses resultados possibilitaram um importante avanço no entendimento da rede em uma infraestrutura de nuvem. Os testes iniciais mostrados na Seção 3.1 estudaram aspectos de desempenho e otimizações de rede. Dessa forma, esses resultados motivaram novos testes com a agregação de *links*, na qual pode comprou-se a sua eficiência (Seção 3.4). Novos testes deverão ser feitos, pois as aplicações testadas na Seção 3.3 possuem versões que utilizam a rede para comunicação de processos. Portanto, a agregação de *links* pode apresentar resultados otimistas.

Para os próximos trabalhos do LARCC, a abordagem será voltada ao método de implantação das ferramentas de nuvens. A ideia neste sentido é conseguir extrair o máximo do desempenho das aplicações que rodam em nuvens utilizando uma implantação específica. Por exemplo, alguns mecanismos de implantação podem favorecer especificamente funções de E/S (disco, rede), processamento, entre outros. Dessa forma, a implantação escolhida poderia favorecer determinadas aplicações. Para isto, um trabalho de conclusão de curso está sendo realizado,

investigando algumas maneiras de implantação das ferramentas e avaliando o desempenho de aplicações pipeline executadas em nuvem.

Referências Bibliográficas

[1] Baum, W., Maron, C.A.F., Griebler, D., Schepke, C.: Caracterização do Desempenho de Aplicações Pipeline em Instâncias KVM e LXC de uma Nuvem CloudStack. In: 17th Escola Regional de Alto Desempenho do Estado do Rio Grande do Sul (ERAD/RS). Sociedade Brasileira de Computação, Ijuí, RS, Brazil (April 2017)

[2] Chasapis, D., Casas, M., Moretó, M., Vidal, R., Ayguadé, E., Labarta, J., Valero, M.: Parsecss: Evaluating the impact of task parallelism in the parsec benchmark suite. ACM Trans. Archit. Code Optim. 12(4), 41:1–41:22 (Dec 2015), http://doi.acm.org/10.1145/2829952

[3] Cloudstack: Cloudstack (2016), https://cloudstack.apache.org/, last access Aug, 2016

[4] KVM: Kernel Virtual Machine (2016), http://www.linux-kvm.org, last access Jul, 2016

[5] Maliszewski, A.M., Vogel, A., Griebler, D., Schepke, C.: Desempenho das Operações de Criar e Deletar Instâncias KVM Simultâneas em Nuvens CloudStack e OpenStack. In: 17th Escola Regional de Alto Desempenho do Estado do Rio Grande do Sul (ERAD/RS). Sociedade Brasileira de Computação, Ijuí, RS, Brazil (April 2017)

[6] Maron, C.A.F., Griebler, D., Schepke, C., Fernandes, L.G.: Desempenho de OpenStack e OpenNebula em Estações de Trabalho: Uma Avaliação com Microbenchmarks e NPB. Revista Eletrônica Argentina-Brasil de Tecnologias da Informação e da Comunicação (RE-ABTIC) 6(1), 15 (December 2016)

[7] Maron, C.A.F., Griebler, D.: Avaliando o Desempenho das Ferramentas de Nuvem Privada OpenStack e OpenNebula. Tech. rep., Laboratory of Advanced Researches on Cloud Computing (LARCC) (2015)

[8] OpenStack: OpenStack Roadmap (2016), http://openStack.org/software/roadmap/, last access Aug, 2016

[9] Regola, N., Ducom, J.C.: Recommendations for virtualization technologies in high performance computing. In: 2010 IEEE 2º CloudCom. pp. 409–416 (Nov 2010)

[10] Sabharwal, N.: Apache CloudStack Cloud Computing. Community experience distilled, Packt Publishing (2013)

[11] Snell, Q.O., Mikler, A.R., Gustafson, J.L.: Netpipe: A network protocol independent performance evaluator. In: IASTED international conference on intelligent information management and systems. vol. 6. Washington, DC, USA) (1996)

[12] Solomon, M., Kim, D., David Kim, I., Carrell, J.: Fundamentals of Communications and Networking. Jones & Bartlett Learning (2014)

[13] Stahl, E., Corona, A., De Gilio, F., Demuro, M., Dowling, A., Duijvestijn, L., Fernandes, A., Jewell, D., Keshavamurthy, B., Markovits, S., et al.: Performance and Capacity Themes for Cloud Computing. IBM Redbooks (2013)

[14] Uperf: Unified Performance tool for networking (uperf) (2016), last access Jul, 2016

[15] Vogel, A., Griebler, D.: Implantando, Avaliando e Analisando as Ferramentas para Gerenciamento de IaaS OpenStack, OpenNebula e CloudStack. Tech. rep., Laboratory of Advanced Researches on Cloud Computing (LARCC) (2016)

[16] Vogel, A., Griebler, D., Maron, C.A.F., Schepke, C., Fernandes, L.G.: Private IaaS Clouds: A Comparative Analysis of OpenNebula, CloudStack and OpenStack. In: 24th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP). pp. 672–679. IEEE, Heraklion Crete, Greece (Febuary 2016)

[17] Vogel, A., Griebler, D., Schepke, C., Fernandes, L.G.: An Intra-Cloud Networking Performance Evaluation on CloudStack Environment. In: 25th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP). p. 5. IEEE, St. Petersburg, Russia (March 2017)

[18] White, T.: Hadoop: The Definitive Guide. O'Reilly Media, Inc., 4th edn. (2015)