Big Data e a implementação de um Sistema Distribuído com Apache Hadoop: um estudo exploratório

Márcio Estevão Petry[1]

[1]Programa Interdisciplinar de Pós-graduação em Computação Aplicada – Universidade do Vale do Rio dos Sinos (UNISINOS)
São Leopoldo – RS – Brasil

marciopetry@gmail.com

Abstract. *This article presents the concepts of Big Data paradigm and its relationship towards distributed computing systems. As a Proof of Concept (POC) to sustain this experiment, an Apache Hadoop 2.6 in 'fully-distributed mode' has been set up. The environment was tested loading some files to its HDFS, following by processing such files through one of several Apache Hadoop built-in apps. Result was the creation of a brand new file inside Hadoop Distributed File System.*

Resumo. *Este artigo descreve alguns conceitos do Big Data em um ambiente de sistema distribuído. Para a implementação de prova de conceito foi utilizado o produto Apache Hadoop 2.6. Apache Hadoop foi configurado em modo Fully-Distributed Mode, capaz de efetuar carga de arquivos de dados, distribuí-los dentre os nós via Hadoop Distributed File System, processá-los e gerar novos arquivos de saída como resultado de processamento distribuído.*

1. Introdução

A quantidade de dados produzidos e coletados nos diversos segmentos da sociedade está explodindo. Dados estão sendo coletados e armazenados em taxas sem precedentes. O desafio que se impõe traduz-se não somente em armazenar e gerenciar o vasto volume de dados (*Big Data*), mas também analisar e extrair significado e valor desse gigantesco volume de dados. Segundo [BAKSHI, 2012] há diversos modos de se coletar, armazenar, processar e analisar *Big Data*, mas o desafio maior que se manifesta é como lidar com armazenamento, recuperação e análise de dados de natureza heterogênea e não estruturados, em gigantescos volumes. Dados não estruturados dizem respeito à informação que ou não possui um modelo de dados pré-definido ou que não cabem adequadamente no paradigma de bancos de dados relacionais (*Small Data*). Ainda segundo [BAKSHI, 2012], dados não estruturados representam o maior tipo de crescimento de dados, e ilustra alguns exemplos tais como dados provenientes de imagens, sensores, telemetria, vídeo, documentos, *log files*, e-mails, dentre outros.

Segundo [KOTIYAL, KUMAR, PANT, GOUDAR, 2013] a *Big Data* representa os dados que não podem ser processados com a ajuda de ferramentas ou técnicas previamente existentes.

O cenário *Big Data* demanda ainda, segundo [SAGIROGLU, SINANC, 2013], passos revolucionários quanto à análise de dados, caracterizado por três componentes principais: variedade, velocidade e volume, conforme ilustrado na figura 1.

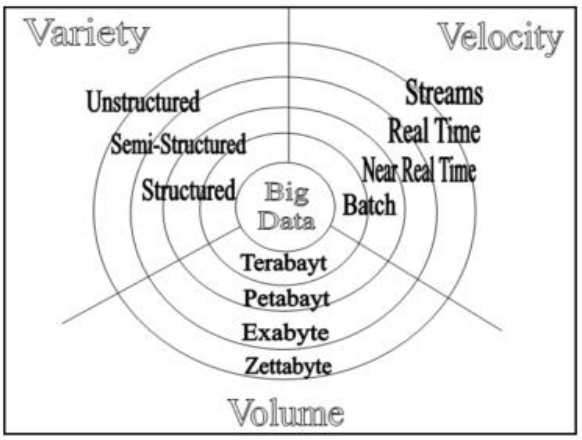

Figura 1: Os três Vs do paradigma Big Data

Nos dias atuais já há diversas tecnologias e técnicas que endereçam este espaço de problema que é a análise de dados não estruturados, oriundo de fontes heterogêneas. Segundo [BAKSHI, 2012], tais técnicas compartilham de características em comum tais como poder de escalabilidade, elasticidade e alta disponibilidade. A utilização de tecnologias como MapReduce, em conjunto com o *Hadoop Distributed File System* (HDFS) e banco de dados HBase, elementos que compõem o ecossistema Apache Hadoop, representam uma abordagem moderna e apropriada para análise e trato de dados não estruturados. *Clusters* de nós Apache Hadoop representam meios eficazes, por via do paradigma de sistemas distribuídos, capazes de processar massivos volumes de dados e podem ser modularizados com uma abordagem arquitetural apropriada [ibidem, 2012].

Este artigo está organizado da seguinte maneira: no capitulo 2 são descritos trabalhos relacionados. No capítulo 3 é descrito todo o processo de concepção e implementação de um ambiente de computação distribuída Apache Hadoop como POC e experimento deste trabalho. No capítulo 4 é apresentada a avalição de um cenário de processamento distribuído na arquitetura Apache Hadoop em ambiente totalmente distribuído. Por fim, o capítulo 5 apresenta as conclusões do presente experimento.

2. Trabalhos relacionados

O trabalho de [BAKSHI, 2012] retrata a aplicação da plataforma Apache Hadoop pelos aspectos de planejamento de uma infraestrutura em ambiente distribuído (computação, rede e sistema de armazenamento), além de rever critérios de criação de ambientes Hadoop e considerações de implementação. O autor estende seu trabalho explorando considerações de desempenho e descreve relevantes *benchmarks* com um *cluster* Hadoop. Conclui seu artigo revendo alternativas de hospedagem de um *cluster* em ambiente de *cloud computing* pública.

Já o experimento de [KOTIYAL, KUMAR, PANT, GOUDAR, 2013] propõe a criação de VMs (*Virtual Machines*) através do produto VMware, de forma a criar um setup Apache Hadoop com nós homogêneos, independentes, e realizar experimentos armazenando um *log file* de registros de acessos, o qual contém informações de *log* oriundas de um sistema cliente-servidor legado.

Por fim, o artigo proposto por [SINGH, KAUR, 2014] ilustra aspectos técnicos e de implementação da plataforma Apache Hadoop, enfatizando-a como uma plataforma de *cloud computing* aberta (*open source*) a qual provê um framework de

desenvolvimento de software para si intitulado MapReduce, concomitante com um sistema de arquivos distribuído intitulado HDFS (*Hadoop Distributed File System*). Seu artigo ainda contribui contextualizando os desafios de processamento de *Big Data* tais como os 3Vs: Variedade (de dados), Velocidade (de processamento e recuperação de informações) e Volume (quantidades gigantescas de dados, na ordem de terabytes ou petabytes).

3. Implementação

Para a implementação desse experimento foi utilizado um rol de produtos de software e hardware de baixo custo, capazes de emular um ambiente em rede cabeada e isolada de computadores, configurando-se um *cluster* de três nós para execução do framework Apache Hadoop e um *host* dedicado ao processo de interação com o sistema distribuído Hadoop, primordialmente por meio de protocolos/serviços SSH e HTTP (navegador web).

As subseções a seguir ilustrarão os elementos constituintes do ambiente estudado e desenvolvido, ilustrando os detalhes técnicos e operacionais pertinentes.

3.1. Configuração dos hosts GNU/Linux

A instalação do Apache Hadoop comumente se dá sobre sistemas operacionais GNU/Linux [SINGH, KAUR, 2014]. Para esse trabalho, portanto, foi elencada a distribuição GNU/Linux CentOS 6.8 *Minimal* versão 64bits.

Para a instalação e configuração de instâncias GNU/Linux idênticas na formação de um *cluster*, necessário para o posterior setup do Apache Hadoop em modo *Fully-Distributed Mode*, foi adotada uma abordagem análoga à utilizada por [Kotiyal, Kumar, Pant, Goudar, 2013], através da utilização de um computador *host* executando sistema operacional Microsoft Windows 10 Professional versão 64 bits (pt_BR) equipado com 6 GB de memória RAM, uma CPU Intel Core i5-3210M operando a 2.5Ghz, disco rígido 500 GB SATA e Dual Monitor (Full HD ambos).

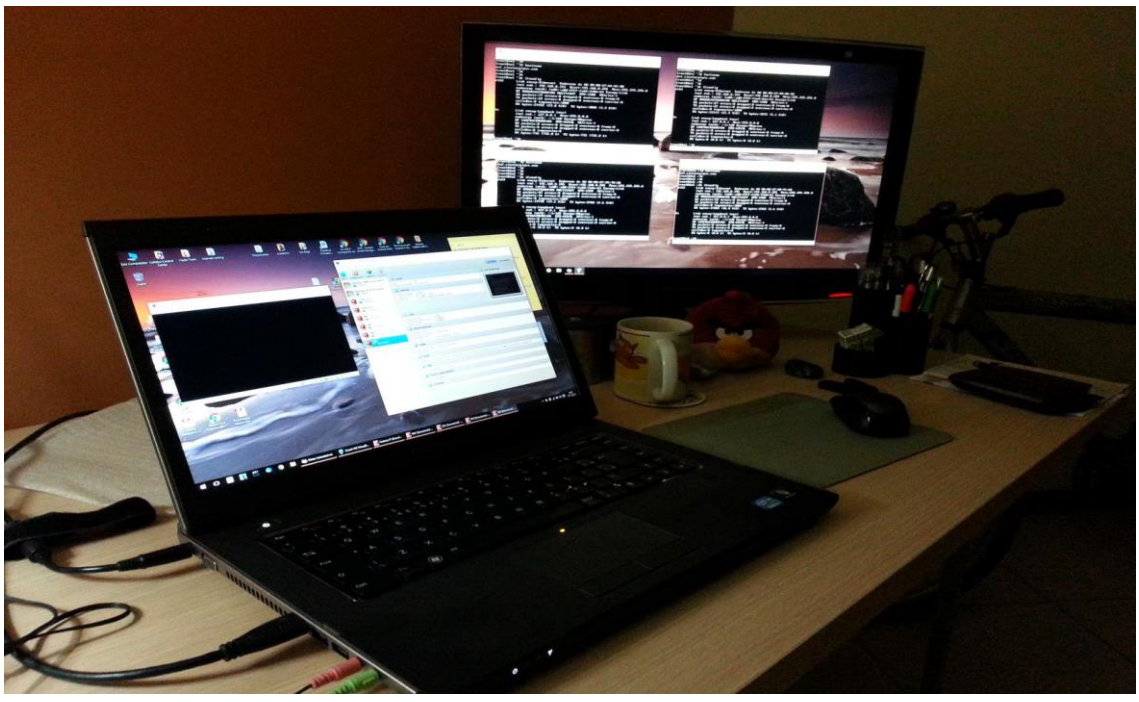

Figura 2: Ambiente de setup para experimento Big Data com Apache Hadoop

No ambiente *host* supracitado, foi instalado o software Oracle VirtualBox 5.0.22 versão 64 bits [VIRTUALBOX, 2016]. Com o produto VirtualBox instalado, foi possível realizar a criação de 03 (três) máquinas virtuais (VMs) operando todas em **modo *bridge*** – ou seja, cada qual poderia obter seu próprio endereço de empilhamento TCP/IP, de um servidor DHCP local já configurado, atendendo a subrede **192.168.1.0/24**, em rede FastEhernet. Foi configurado um domínio local **clusterpipca.com** para a resolução de nomes dentre os nós.

A instalação do GNU/Linux CentOS 6.8 *Minimal* procedeu de forma rápida e precisa, tendo ocorrido inicialmente sobre apenas 01 (uma) máquina virtual VirtualBox. A VM em questão foi configurada para operar em modo *bridge* (conforme já mencionado), com alocação de 8 GB de disco rígido (*fixed-size*) e 512 MB de memória RAM. Após a instalação do CentOS 6.8 sobre a VM, foi executada uma atualização completa da respectiva distribuição, através do comando "**sudo yum update**".

A essa instância CentOS 6.8 foi adicionado um usuário *non-root*, intitulado '**pipca'** (com senha pipca). Essa mesma instância foi nomeada de '**master.clusterpipca.com'** (*hostname*), e também recebeu um IP fixo **192.168.1.100.**

Complementando a instalação do CentOS 6.8, foi adicionado ao mesmo alguns serviços e pacotes de software adicionais, como pré-requisitos para a execução de um *cluster* Apache Hadoop. A lista abaixo relaciona os softwares que foram instalados na instância CentOS 6.8 e/ou serviços que foram adequados para preparação de **master.clusterpipca.com** e das demais instâncias CentOS 6.8 que foram clonadas, posteriormente, a partir dessa:

- Instalado Java JDK 1.8 *update* 65 versão 64 bits;
- Instalado OpenSSH versão 5.3p1-1121.e16_7, seguida da criação de chaves "*passwordless*" para usuário 'pipca' (previamente criado);
- Desabilitado 'iptables' (por tratar-se de uma rede isolada de outras subredes);
- Editado /etc/sysconfig/network configurado *hostname* **master.clusterpipca.com**
- Editado /etc/hosts de forma que pudesse resolver nomes do domínio local para endereços IPs da subrede local 192.168.1.0/24, conforme ilustra a figura 3.

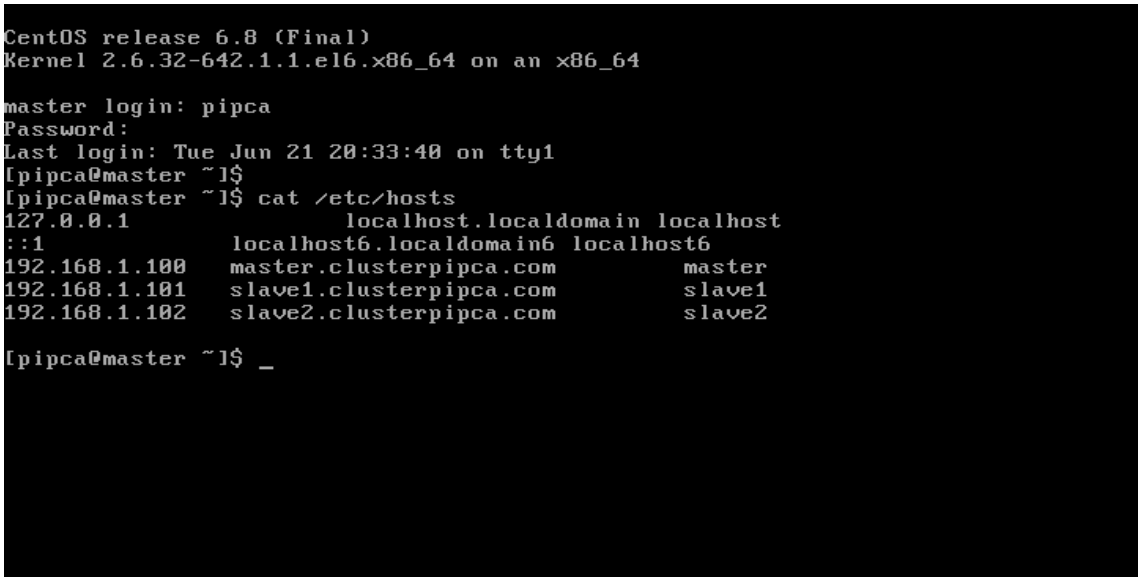

Figura 3: CentOS 6.8 - 'master.clusterpipca.com' - configurado com mapeamento estático de *Hosts* (/etc/hosts)

A partir desse momento, com a instância **master.clusterpipca.com** configurada e estabilizada, foram criadas mais duas instâncias de CentOS 6.8 *Minimal* mediante processo de 'clonagem', através do software Oracle VirtualBox [VIRTUALBOX, 2016], sendo as mesmas intituladas, respectivamente, de '**slave1**' e '**slave2**'. Adequações de *hostname* foram replicadas e aplicadas a ambas.

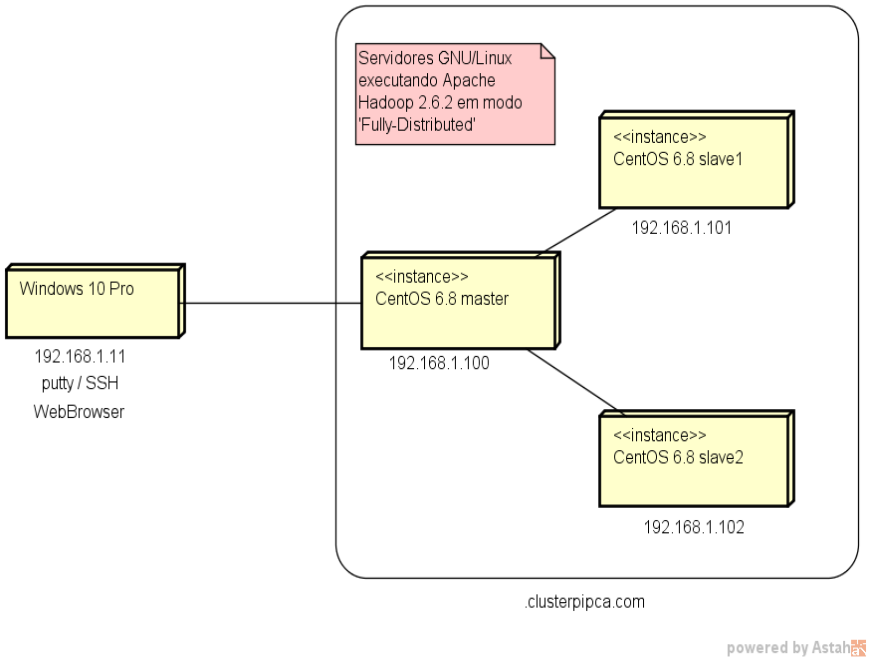

Figura 4: Topologia do ambiente Apache Hadoop em *Fully-Distributed Mode*

De igual teor, novos IPs estáticos foram atribuídos a cada uma, respectivamente **192.168.1.101** e **192.168.1.102**. A figura 4 ilustra a configuração da rede com seus respectivos nós e nomes.

3.2. Instalação do Apache Hadoop versão 2.6.2

A instalação do produto Apache Hadoop foi realizada nos 03 (três) *hosts* que compõem o *cluster* responsável pela execução do Apache Hadoop em modo *Fully-Distributed Mode*.

A instalação é relativamente simples. Foi realizado o download de um arquivo compactado [HADOOP, 2016], correspondente ao produto Apache Hadoop 2.6.2, e extraído seu conteúdo no file system **/usr/local/hadoop** de todos os *hosts* do *cluster*. Esse procedimento ocorreu nos nós **master**, **slave1** e **slave2** de forma idêntica.

O Apache Hadoop é um software escrito em linguagem Java e necessita a presença do JDK (*Java Development Kit*) 1.6 ou superior [JAVA, 2016] previamente instalado no nó. Conforme já mencionado, foi instalada previamente a versão JDK 1.8 *update* 65.

Fez-se necessário, ainda, a configuração das variáveis de ambiente para que o Java e o Hadoop pudessem ser executados apropriadamente. Essa configuração foi realizada, igualmente, nos três nós do *cluster*, no file system **/etc/profile.d/java.sh** e **/etc/profile.d/hadoop.sh**. A figura 5 ilustra os valores definidos.

```
>>> Ajustando a JVM da Oracle, tornando-a padrão para o Apache Hadoop na instância GNU/Linux
# /usr/sbin/alternatives --install /usr/bin/java java /usr/local/jdk1.8.0_65/bin/java 2
# /usr/sbin/alternatives --config java


>>> ajustando variáveis de ambienyte em /etc/profile.d/
[java.sh]
        export JAVA_HOME=/usr/local/jdk1.8.0_65/
        export JRE_HOME=/usr/local/jdk1.8.0_65/jre/
        export PATH=$JAVA_HOME/bin:$PATH

[hadoop.sh]
        export HADOOP_INSTALL=/usr/local/hadoop
        export HADOOP_BIN=/usr/local/hadoop/bin
        export HADOOP_SBIN=/usr/local/hadoop/sbin
        #export HADOOP_USERNAME="pipca"

# source java.sh
# source hadoop.sh
```

Figura 5: Configuração das variáveis de ambiente nos três *hosts* Apache Hadoop

3.3. Configuração do Apache Hadoop versão 2.6.2

Após a instalação, fez-se necessária ainda a configuração dos arquivos **core-site.xml**, **mapred-site.xml** e **hdfs-site.xml** em **todas as instâncias** CentOS 6.8 com Apache Hadoop 2.6.2 instalado, ajustando nomes dos nós e portas TCP utilizadas pelo Hadoop.

Ainda, foi necessária a edição dos arquivos intitulados 'masters' e 'slaves'. Esses arquivos ficam armazenados no filesystem **/usr/local/hadoop/etc/hadoop/**. Essa edição somente deve ser realizada no nó Hadoop **master.clusterpipca.com.**

O arquivo 'masters' contém a lista dos nós Apache Hadoop que atuarão como NameNode e JobTracker, assim sendo, contém somente a linha **master.clusterpipca.com**. O arquivos 'slaves' contém a lista dos nós Apache Hadoop que atuarão como DataNode e TaskTrackers, assim sendo, conterão as linhas **master.clusterpipca.com**, **slave1.clusterpipca.com** e **slave2.clusterpipca.com** – para esse trabalho também foi elencado o nó **master.clusterpipca.com** como membro igualmente participante no processo de distribuição de arquivos (HDFS) e tarefas (MapReduce/YARN), inerentes ao paradigma Apache Hadoop. A figura 6, adaptada de [BAKSHI, 2012] ilustra essa abordagem.

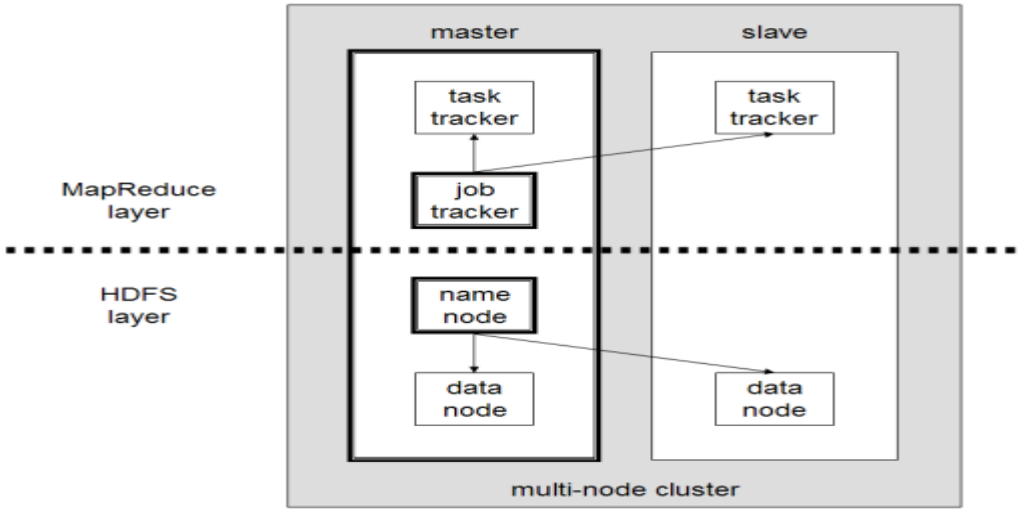

Figura 6: Arquitetura Apache Hadoop *Fully-Distributed Mode*, onde o nó master também opera como TaskTracker e como DataNode no *cluster*

Como aspecto fundamental da configuração do *cluster* Apache Hadoop, fez-se necessário configurar o SSH de todos os nós CentOS 6.8, de forma que o processo de distribuição de arquivos (**HDFS**) e de distribuição de tarefas (**MapReduce/YARN**) do framework Apache Hadoop pudesse se autenticar nas instâncias membro do seu *cluster*.

Mais precisamente, foi necessário executar os dois comandos abaixo a partir do nó **master.clusterpipca.com**, de forma a distribuir a chave-pública (criptografia assimétrica RSA) do usuário '**pipca@master**' aos nós **slave1.clusterpipca.com** e **slave2.clusterpipca.com**, garantindo assim a autenticação SSH a partir de **master.clusterpipca.com**.

```
$
$ ssh-copy-id -i $HOME/.ssh/id_rsa.pub pipca@slave1.clusterpipca.com
$ ssh-copy-id -i $HOME/.ssh/id_rsa.pub pipca@slave2.clusterpipca.com
$
```

Masters versus Slaves – Conforme ilustra [NOLL, 2016], tipicamente a uma máquina no *cluster* é designado o papel exclusivo de NameNode e a outra máquina é designado o papel exclusivo de JobTracker. Esses serão, de fato, os intitulados nós *master* de um ambiente *Fully-Distributed Mode*. O restante dos nós participantes do *cluster* Apache Hadoop atuarão tanto como DataNode quanto TaskTracker. Serão os nós intitulados *slaves*, ou ainda conhecidos na literatura como ***worker nodes***.

Uma última atividade que se fez necessária para configurar o *cluster* Apache Hadoop foi a preparação física (armazenamento) dos DataNodes nos nós **slave1.clusterpipca.com** e **slave2.clusterpipca.com**. Para tanto, foi necessário invocar uma aplicação que acompanha o Hadoop intitulada **hdfs**, para realizar a 'formatação' do HDFS nesses *workers nodes*. O comando abaixo, executado em todos os nós, finalizou o processo de *setup* do Hadoop e deixou o ambiente pronto para início de experimentos.

```
$
$ cd /usr/local/hadoop
$ bin/hdfs namenode -format
```

4. Avaliação

A avaliação e percepção do poder de distribuição de código e distribuição de dados dentre os nós, e do controle SSH exercido de forma transparente pelo Apache Hadoop 2.6.2 dentre o nó master e os nós slave1 e slave2, iniciou já a partir do momento em que se fez necessário inicializar o primeiro conjunto de processos do produto no nó **master**.

No nó **master.clusterpipca.com** foram invocadas na shell do CentOS 6.8 as duas linhas a seguir, as quais deram início a um conjunto de processos Java, correspondentes aos papeis de **NameNode/DataNode** e **JobTracker/TaskTracker**.

```
$ cd $HADOOP_HOME
$ sbin/start-dfs.sh
$ sbin/start-yarn.sh
```

Nos nós **slave1.clusterpipca.com** e **slave2.clusterpipca.com** somente os processos Java de DataNode e TaskTracker foram inicializados **automaticamente**, via SSH, por **master.clusterpipca.com**, por se tratarem de *worker nodes* [NOLL, 2016].

```
>>> O seguinte comando é utilizado para iniciar o Hadoop dfs.
$ $HADOOP_HOME/sbin/start-dfs.sh
Starting namenodes on [localhost]
localhost: starting namenode, logging to /home/hadoop/hadoop
2.4.1/logs/hadoop-hadoop-namenode-localhost.out
localhost: starting datanode, logging to /home/hadoop/hadoop
2.4.1/logs/hadoop-hadoop-datanode-localhost.out
Starting secondary namenodes [0.0.0.0]

>>> O seguinte comando é utilizado para iniciar o MapReduce/YARN.
$ $HADOOP_HOME/sbin/start-yarn.sh
starting yarn daemons
starting resourcemanager, logging to /home/hadoop/hadoop
2.4.1/logs/yarn-hadoop-resourcemanager-localhost.out
localhost: starting nodemanager, logging to /home/hadoop/hadoop
2.4.1/logs/yarn-hadoop-nodemanager-localhost.out
```

Figura 7: Acionando os componentes fundamentais do Apache Hadoop

O Apache Hadoop vem acompanhado de uma série de aplicativos MapReduce pré-compilados em seu pacote de instalação. São aplicações escritas em linguagem Java. Para efeitos de experimento e observação, foi realizada a carga de um arquivo de texto que intitulado "file1.txt" contendo alguns substantivos próprios, alguns inseridos mais de uma vez no arquivo. O propósito da criação desse arquivo foi utilizá-lo como entrada de dados para a aplicação **wordcount.jar** que acompanha o Hadoop, e que atuou como ferramenta de *Proof of Concept* para esse artigo.

Para carregar o arquivo "file1.txt" ao sistema HDFS do Hadoop, foi necessário invocar a aplicação **hdfs** que acompanha o Hadoop, responsável por salvar conteúdo (nesse caso, de forma interativa) nos DataNodes. Os seguintes comandos foram digitados na shell do CentOS 6.8, em **master.clusterpipca.com.**

$ bin/hdfs dfs -mkdir /user

$ bin/hdfs dfs -mkdir /user/pipca

$ bin/hdfs dfs -mkdir input # criará pasta "input" dentro de /user/pipca

$ bin/hdfs dfs **-put file1.txt input**

A sequência de comandos acima orientou o Hadoop a criar um sistema hierárquico de pastas **(/user/pipca/input**) dentro dos DataNodes distribuídos por **master.clusterpipca.com**, **slave1.clusterpipca.com** e **slave2.clusterpipca.com**, proporcionando a redundância dos dados.

De acordo com [CORDEIRO, 2012] alguns elementos que tornam o Apache Hadoop um ambiente distribuído tão interessante e performático deriva de sua ideia e implementação arquitetural distribuída (dados e código), onde:

- Opera um Sistema de arquivos distribuído
- Possui Replicação interna
- Recuperação de falhas automática
- Move a computação para onde estão os dados

Após efetuar a carga do arquivo **file1.txt**, foi explorado o ambiente Apache Hadoop a fim de se certificar de que o arquivo estava realmente armazenado em algum meio persistente e capaz de ser recuperado (manual ou programaticamente) em algum momento futuro. A partir do navegador *Web* sendo executado na estação Windows 10 Professional, através da interface web do Hadoop, porta **TCP 50070**, foi possível navegar (via protocolo HTTP, interface HTML) pelo seu *Hadoop Distributed File System*. A figura 8 ilustra a localização e identificação do arquivo no HDFS.

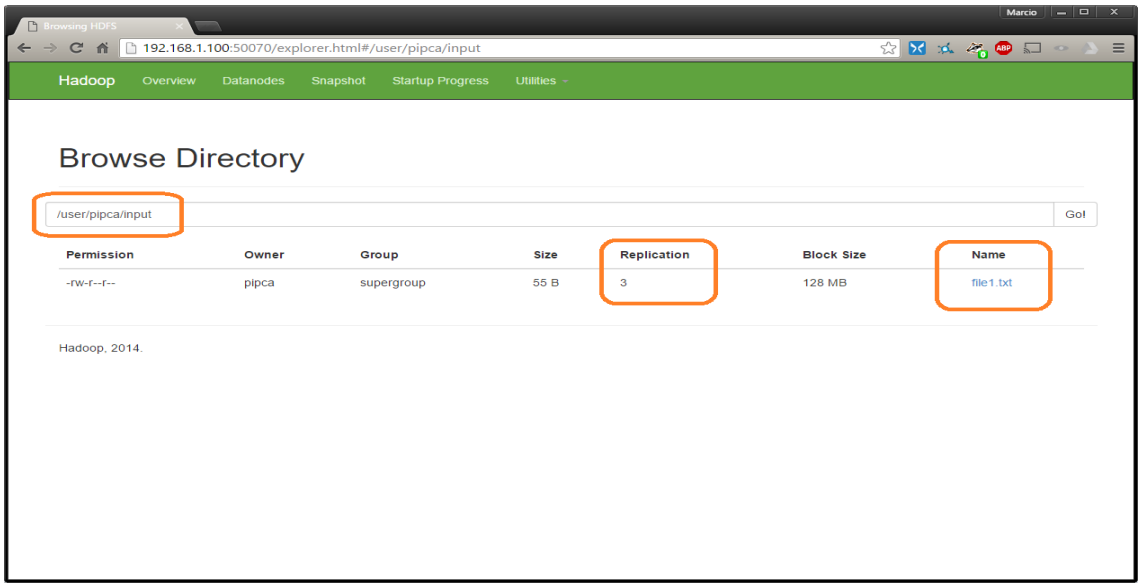

Figura 8: Explorando o HDFS e localizando o arquivo 'file1.txt'

Pela figura 8 podemos perceber que o arquivo está armazenado na pasta criada correspondente, e que há três réplicas dele, ou seja, está distribuído em três nós, no contexto deste trabalho, em **master**, **slave1** e **slave2**. Clicando sobre o hyperlink intitulado "file1.txt", nova tela se abre no navegador, conforme ilustra figura 9.

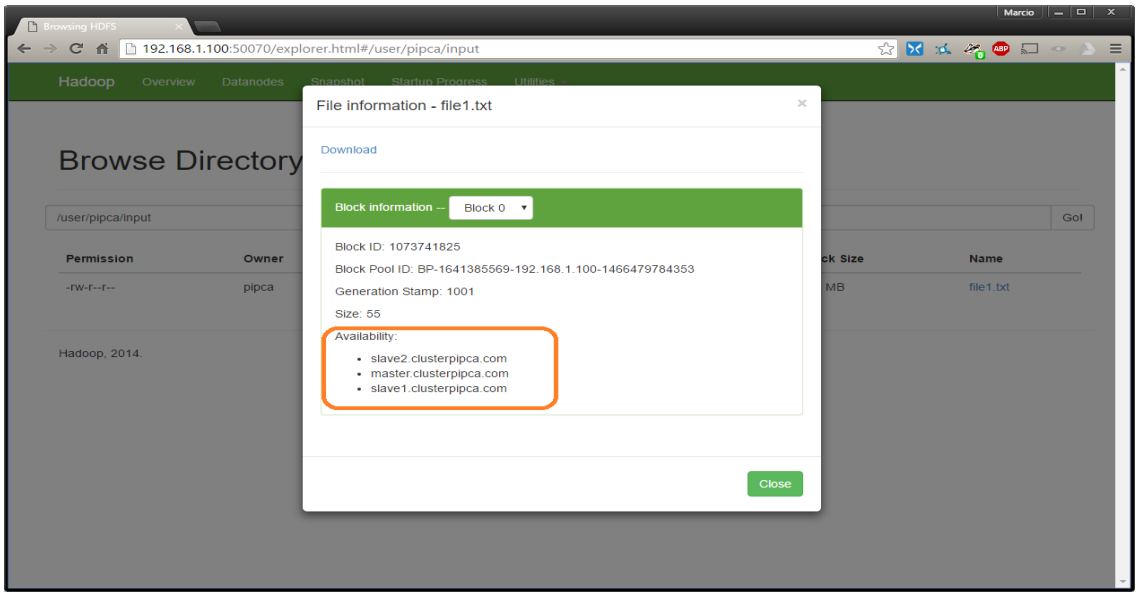

Figura 9: Explorando o HDFS e localizando o arquivo 'file1.txt' distribuído

Tendo localizado o arquivo, foi executada a aplicação *built-in* do Apache Hadoop intitulada **wordcount**, passando como parâmetro o recurso **file1.txt** para que a mesma pudesse contabilizar a quantidade de substantivos encontrados no arquivo e gerar um arquivo de saída, no cenário representado, intitulado **output**. A figura 10 ilustra a chamada da aplicação, por linha de comando, em **master.clusterpipca.com**.

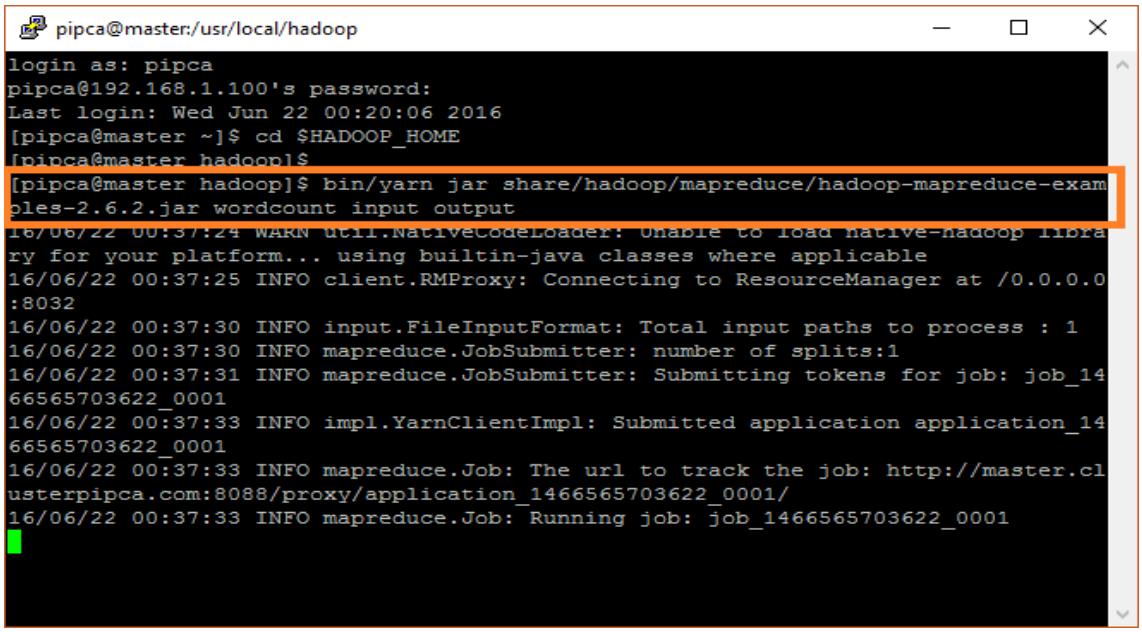

Figura 10: Executando uma aplicação Hadoop *built-in* intitulada wordcount

A partir do navegador *Web* sendo executado na estação Windows 10 Professional, novamente foi possível explorar o HDFS do ambiente de estudo. Na figura 11 é possível identificar a criação do arquivo **output** pela aplicação **wordcount**.

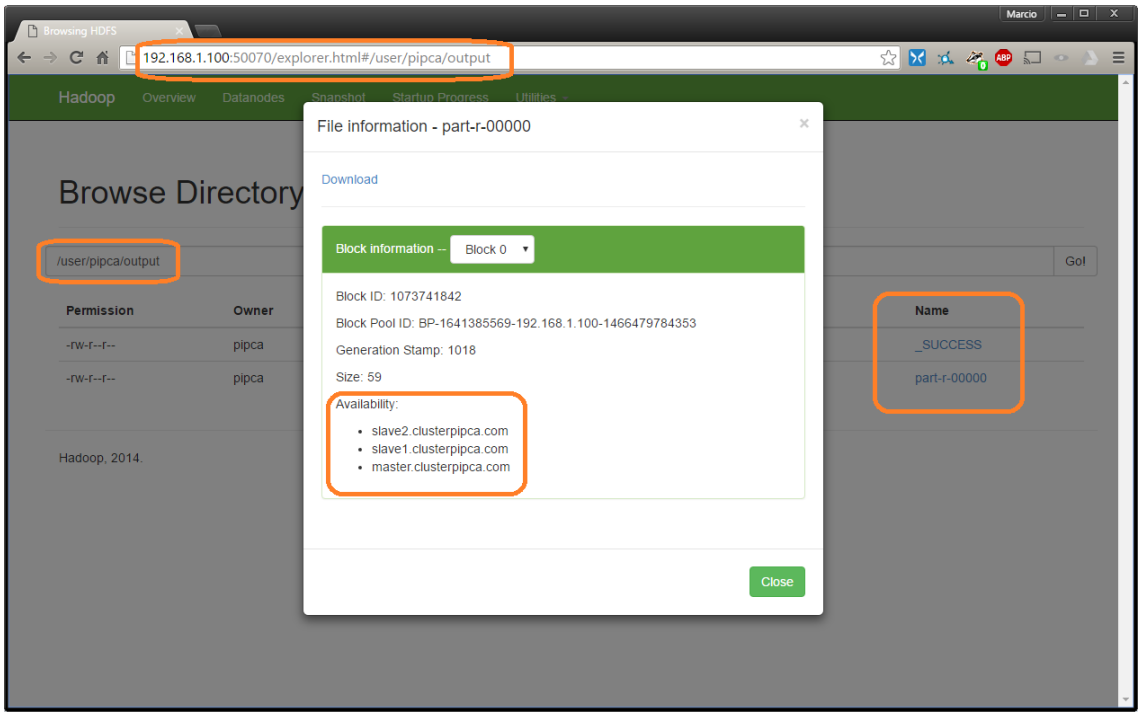

Figura 11: Explorando o HDFS e localizando o arquivo 'output' distribuído

Para encerrar o experimento do Apache Hadoop, foi acessada ainda a sua interface *Web* de gerenciamento de aplicações MapReduce/YARN, *host* **master.clusterpipca.com**, porta **TCP 8088**, conforme ilustra a figura 12.

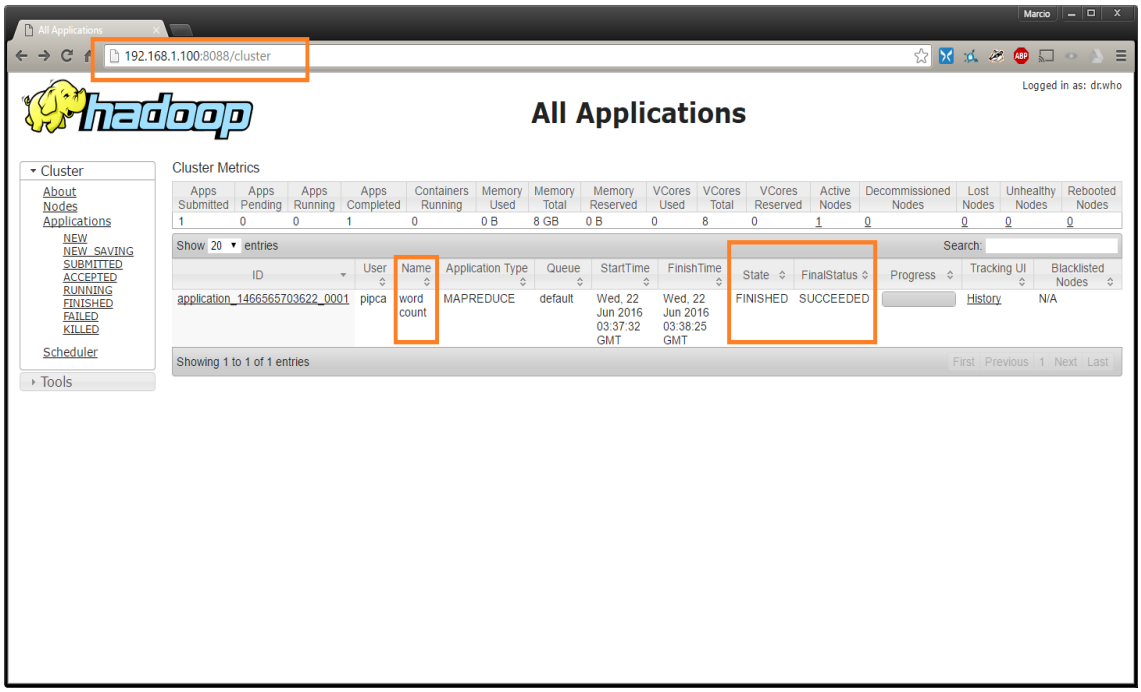

Figura 12: Identificando e localizando a aplicação 'wordcount' e seus status

Observa-se pela figura 12 que a aplicação wordcount concluiu seu processamento, de forma bem sucedida. O arquivo '**output**' criado no HDFS do *cluster* foi de sua responsabilidade. Para descobrir e verificar o resultado gravado nesse arquivo 'output', foi utilizada novamente a linha de comando do nó **master.clusterpipca.com**, invocando a aplicação Hadoop **hdfs** com a chave **-cat**, conforme ilustra a figura 13.

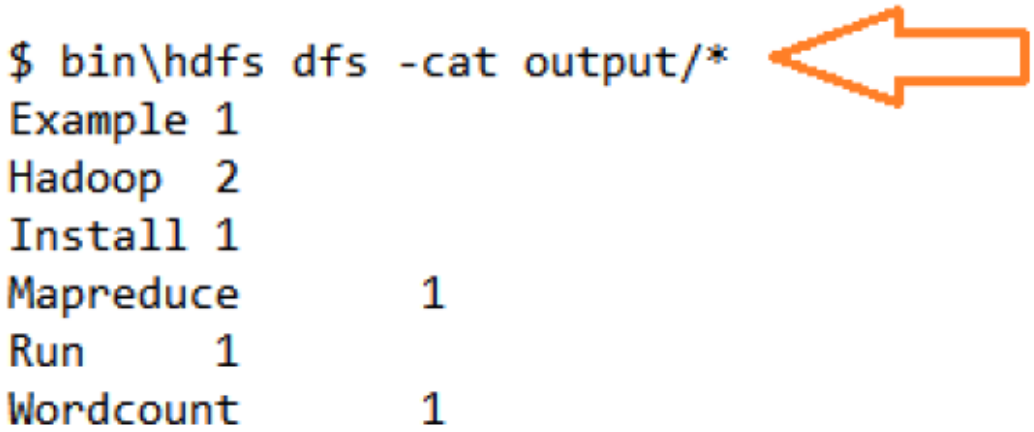

Figura 13: Imprimindo na console o conteúdo de 'output' extraído do HDFS

5. Conclusão

A experiência com o Apache Hadoop em modo *Fully-Distributed Mode* não poderia ter sido mais gratificante. Após muitas horas de pesquisa, estudo e compreensão (ainda inicial) dos papéis de cada nó em um ecossistema Hadoop distribuído, passando pelas configurações a serem contempladas no mesmo, desde o *tuning* dos sistemas operacionais (CentOS 6.8 *Minimal*), configuração de rede, Iptables, arquivos *.conf, até os detalhes mais sutis dos arquivos $HADOOP_HOME/etc/hadoop/*-site.xml, foi possível obter uma visão panorâmica, prática e melhor dimensionada da escalabilidade proporcionada por esse produto.

Sua natureza essencialmente distribuída promove escalabilidade que surpreende pela simplicidade, *efortless*, de uma maneira relativamente simples de se configurar. Por ser um produto *open-source* [SINGH, KAUR, 2014], oportuniza seu emprego em instituições várias, a baixo custo.

O conceito de dois ou mais nós que trabalham em conjunto e que se comunicam sobre uma rede de dados para atender um objetivo em comum caracteriza o produto Apache Hadoop como uma solução legítima de Sistema Distribuído, calcado, ainda, fortemente sobre nós executando o sistema operacional GNU/Linux conforme salienta [SINGH, KAUR, 2014] em seu trabalho relacionado.

Clusters de nós Apache Hadoop representam meios eficazes em sistemas distribuídos capazes de processar massivos volumes de dados e podem ser modularizados com uma abordagem arquitetural apropriada [BAKSHI, 2012], estendendo seus recursos e funcionalidades.

Conforme identificado em [BAKSHI, 2012], o poder de escalabilidade, elasticidade e alta disponibilidade são inerentes ao produto Hadoop. A utilização de tecnologias como MapReduce/YARN, em conjunto com o *Hadoop Distributed File System* (HDFS) tornam-no uma abordagem moderna e apropriada para a análise e para o trato de dados não estruturados, elementos constituintes do paradigma *Big Data*.

Considerando a pletora de dados não estruturados oriunda de fontes heterogêneas na atualidade, temos que o trato, a recuperação e a análise desses dados em tempo hábil e confiável se fazem imprescindíveis, posicionando o Apache Hadoop, enquanto uma solução de sistema distribuído, como um produto favorável e apropriado a atender tais necessidades, de forma escalável, distribuída e garantindo confiabilidade.

Referências

Bakshi, K. (2012) “Considerations for Big Data: Architeture and Approach”, IEEE Aerospace Conference Proceedings, 2012.

Cordeiro, D. (2012) “Apache Hadoop Conceitos teóricos e práticos, evolução e novas possibilidades”. Disponível em http://www.each.usp.br/dc/papers/erad-hadoop-DanielCordeiro.pdf. Acesso em 17.jun.2016

Hadoop, Apache (2016) “Apache Hadoop 2.6.2”. Disponível em https://hadoop.apache.org/docs/r2.6.2/. Acesso em 18.jun.2016.

Java, Oracle (2016) “Java SE Download”. Disponível em http://www.oracle.com/technetwork/java/javase/downloads/index.html. Acesso em 17.jun.2016.

Kitchin, R. and Lauriault, T.P. (2014) “Small data in the era of big data”, GeoJournal, Springer Science+Business Media, Dordrecht, Netherlands, 2014.

Kotiyal, B., Kumar, A., Pant, B., Goudar, R.H. (2013) “Big Data: Mining of Log File through Hadoop”, Human Computer Interactions (ICHCI), 2013, International Conference on 2013.

Noll, M. (2016) “Running Hadoop on Ubuntu Linux (Multi-Node Cluster)”. Disponível em http://www.michael-noll.com/tutorials/running-hadoop-on-ubuntu-linux-multi-node-cluster/. Acesso em 18.jun.2016.

Sagiroglu, S., Sinanc, D., “Big Data: A Review”, *IEEE*, Gazi University, Ankara, Turkey, 2013.

Singh, K. and Kaur, R. (2014) “Hadoop: Addressing Challenges of Big Data”, Souvenir of the 2014 IEEE International Advance Computing Conference, IACC 2014.

VirtualBox, Oracle (2016) “Oracle VirtualBox”. Disponível em http://www.virtualbox.org. Acesso em 18.jun.2016.