A Hadoop Open Source Backup Software Solution

Heitor Faria, Rodrigo Hagstrom, Marco Reis, Breno G. S. Costa, Edward Ribeiro, Maristela Holanda, Priscila Solis Barreto and Aletéia P. F. Araújo
Department of Computer Science (CIC), University of Brasilia (UnB), Brasilia, DF, Brazil

Keywords: Hadoop Backup, Cluster, Disaster Recovery.

Abstract: Backup is a traditional and critical business service with increasing challenges, such as the snowballing of constantly increasing data. Distributed data-intensive applications, such as Hadoop, can give a false impression that they do not need backup data replicas, but most researchers agree this is still necessary for the majority of its components. A brief survey reveals several disasters that can cause data loss in Hadoop HDFS clusters, and previous studies propose having an entire second Hadoop cluster to host a backup replica. However, this method is much more expensive than using traditional backup software and media, such a tape library, a Network Attached Storage (NAS) or even a Cloud Object Storage. To address these problems, this paper introduces a cheaper and faster Hadoop backup and restore solution. It compares the traditional redundant cluster replica technique with an alternative one that consists of using Hadoop client commands to create multiple streams of data from HDFS files to Bacula – the most popular open source backup software and that can receive information from named pipes (FIFO). The new mechanism is roughly 51% faster and consumed 75% less backup storage when compared with the previous solutions.

1 INTRODUCTION

The Hadoop cluster backup is a recurrent and reasonable issue brought up by system administrators who manage big companies (Grishchenko, 2015). Their impression is that backup is the best option for protecting themselves against data loss, and this is a crucial requirement for most institutions.

As stated by (Barot et al., 2015), Hadoop backup and recovery is a relevant problem and is becoming more challenging every day, especially with the snowballing of constantly increasing data, plus an ever greater need for data security. Large Hadoop users such as Facebook (Facebook, 2017), eBay (eBay, 2017), Yahoo! (Yahoo, 2017) (the first to implement Hadoop), among others, are challenged to handle unprecedented volumes of unstructured data, something that traditional relational databases are unable to control and deliver.

The following disasters can affect a Hadoop cluster (Kothuri, 2016):

- Hardware Failures: disk corruption, node failure, and rack failure;
- Human/Application Error: logical corruption and accidental deletion of data;
- Site Failure.

There is a caveat stated by (Grishchenko, 2015), however. The backup question will be analyzed in detail because a misinterpretation might lead to substantial superfluous investments from the customer side. Since Hadoop deals with huge amount of data, there are some pitfalls and contradictions regarding what needs to be backed up to achieve data protection and service continuity levels at a reasonable cost.

The purpose of the present work is to deploy a backup protection solution for a small Hadoop cluster that is more efficient than the older Hadoop replica technique with the distcp mechanism, using well known open source solutions, such as Bacula (Bacula, 2017a) software and its universal named pipe plugin, bpipe (Bacula, 2017b). Essential backup features, such as the ability to perform Hadoop Distributed File System (HDFS) (HortonWorks, 2017) differential backup were also considered, and an experiment was carried out for each one of these approaches. They are detailed, described, their results presented, and evaluated in this article.

This paper is organized as follows: Section 2 presents the characteristics of Apache Hadoop. Section 3 presents the motivation for this paper and some im-

Faria, H., Hagstrom, R., Reis, M., G. S. Costa, B., Ribeiro, E., Holanda, M., Barreto, P. and Araújo, A.
A Hadoop Open Source Backup Solution.
DOI: 10.5220/0006809206510657
In *Proceedings of the 8th International Conference on Cloud Computing and Services Science (CLOSER 2018)*, pages 651-657
ISBN: 978-989-758-295-0

portant points to be addressed in a solution to backup large volumes of data. Section 4 details some related works. Section 5 explains a backup software solution. Section 6 presents the solution proposed. Next, Section 7 shows the results obtained with the proposed solution. Finally, Section 8 draws some conclusions and suggests future works.

2 APACHE HADOOP

Hadoop (White, 2015) is developed by Apache and consists of an open-source software project for reliable, scalable, distributed computing. As specified on the official Hadoop website (The Apache Software Foundation, 2017), it is a framework that allows distributed and scalable processing of large data sets across clusters of computers using simple programming models. The cluster can be deployed in any number of machines, even thousands, each offering local computation and storage.

According to (The Apache Software Foundation, 2017), Hadoop is composed by:

- Hadoop MapReduce: a system to parallel processing of big data;
- Hadoop Distributed File System (HDFS): a high performance and distributed file system;
- Hadoop Common: support utilities and libraries;
- Hadoop YARN (Yet Another Resource Negotiator): a framework to manage the cluster resources.

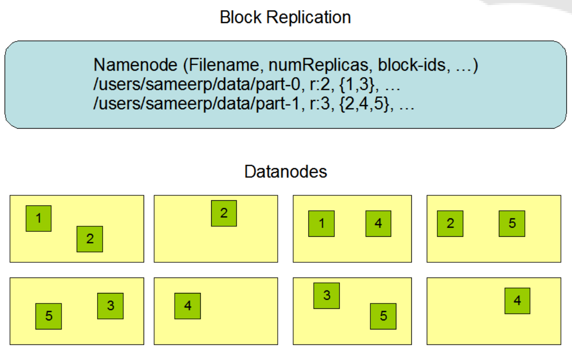
Block Replication
Namenode (Filename, numReplicas, block-ids, ...)
/users/sameerp/data/part-0, r:2, {1,3}, ...
/users/sameerp/data/part-1, r:3, {2,4,5}, ...
Datanodes
1
2
2
1
4
2
5
5
3
4
3
5
4

Figure 1: DataNodes and Block Replication in HDFS (The Apache Software Foundation, 2017).

The Hadoop's architecture for storing data is based on a master/slave technique, where the master, named NameNode, manages the file system, and the slaves, named DataNodes, store reliably the raw data. As seen in Figure 1, each file is stored in a sequence of blocks replicated between the nodes, to guarantee the fault tolerance.

The HDFS is scalable to petabytes of data and thousands of server nodes (HortonWorks, 2017), what makes the backup a challenge. Because of this, this paper proposes an efficient solution to perform backup on hadoop clusters. Some of the key challenges addressed in the proposed solution are described in Section 3.

3 HADOOP BACKUP

In the words of (Barot et al., 2015), there is a discussion about whether or not there is a need for performing Hadoop backups. Ultimately, most authors agree this is necessary (Grishchenko, 2015; Kothuri, 2016), according to them, believing that Hadoop replication shelter against data loss is a mistaken belief.

As shown in Figure 2, Hadoop datasets are written in a replication pipeline, where a block of the file is written and three different copies are made at various Data Node locations by default.

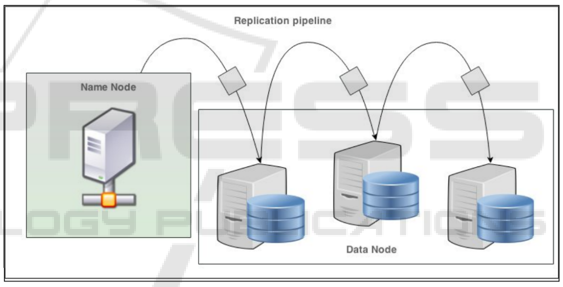

Figure 2: Hadoop Replication Pipeline (Barot et al., 2015).

As (Barot et al., 2015) explains, replication is reassuring, but it is not safe and does not guarantee failproof protection against data loss. Hadoop architecture can preserve data over some hardware failures, mostly in scenarios where a single disk, cluster or region goes down. Nevertheless, there are many scenarios where data loss may occur.

As disaster examples, Hive (Hive, 2017) storage locations, selection can be affected by human error. If the user provides a location in which data already exists, and a query is performed on the same table, the entire existing data will be deleted, regardless of its size. There are also cases when faulty applications read HDFS files and there is a metadata mismatch, triggering an unnecessary replication update of healthy blocks from other nodes.

According to (Kothuri, 2016), many datasets used by Hadoop for applications/analysis are unique - the only copy of the data. In other words, data sets cannot be regenerated if they are lost.

Another argument in favor of performing Hadoop backups is designed to work within a single datacenter (Grishchenko, 2015). In line with (Khoshkholghi et al., 2014; Hua et al., 2016; Xu et al., 2014), it is a severe threat to the system if backup data can only be stored in the same site as primary systems. In order to achieve higher fault tolerance and reliability, original data and backup replicas must be stored in geographically separated locations.

Conforming to (Grishchenko, 2015), backup target datamarts (result datasets) and aggregated reports must also be backed up. They usually represent gigabytes or terabytes of data.

That said, and in agreement with (Kothuri, 2016; Barot et al., 2015; Grishchenko, 2015), the following are the recommended Hadoop elements that need backup:

- **Data Sets:** raw data and result datasets; metadata (NameNode, Hive, HBASE and other);
- **Applications:** system and user applications;
- **Configuration:** configuration of various Hadoop components.

As shown in Figure 3, previously proposed Hadoop backup solutions (Grishchenko, 2015; Barot et al., 2015; Kothuri, 2016) use a secondary cluster to serve as a safe replica. However, having and maintaining a second cluster is much more expensive than storing data in tape-libraries, NAS or even Cloud object storages. This is clear when considering the Cloud pricing. While Storage can cost U\$0.023 per GB per Month for basic plans (Amazon Web Services, 2017b), a Hadoop cluster machine is charged U\$0.053 per Hour, which is the least expensive among them (Amazon Web Services, 2017a).

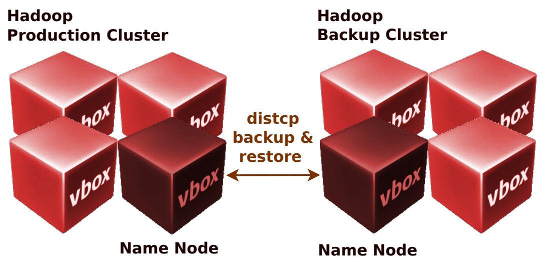

Figure 3: Previous Hadoop HDFS distcp Replica Backup Technique.

Another problem with previously proposed solutions involves inter-cluster replicas, which only save the current state of data. This does not protect against undesired modifications and former unnoticed data loss, providing a poor Recovery Point Objective (RPO).

As noted by (Grishchenko, 2015), the most prominent challenge of backing up Hadoop cluster involves HDFS datasets, which may contain petabytes of information, and therefore the backup duration is one of the most crucial comparison metrics.

Backup Hadoop application binaries and configurations are very small in size and in the number of files concerning a typical HDFS workload or even in comparison with other applications. They should be easily protected by a file level backup (e.g., using the Bacula client), and its performance impact is ignored in this study.

In the opinion of (Kothuri, 2016), there was no proper out of the box point in time recovery solution for Hadoop, at least until now.

In line (Khoshkholghi et al., 2014), disaster recovery is a persistent problem in information technology platforms, and even more crucial in distributed systems and cloud computing. Service Providers must provide services to their customers even if the data center is down (due to a disaster). Researchers have shown more interest in Disaster Recovery using cloud computing in the past few years and a considerable amount of literature has been published in this area.

As describe by (Alhazmi and Malaiya, 2013), Recovery Point Objective (RPO) and Recovery Time Objective (RTO) are the two main parameters that all recovery mechanisms should observe. If the RPO and RTO values are lower, higher business continuity can be achieved. RPO may be interpreted as the amount of lost data cost in a disaster. RTO refers to the time frame between disruption and restoration of service.

As demonstrated by Equation 1, the Recovery Point Objective value is inversely proportional to the frequency of backups completed over the time, where FB represents the *Frequency of Backup*

$$RPO \propto \frac{1}{FB} \tag{1}$$

On the other hand, as shown by Equation 2, Recovery Time Objective formula usually includes a fraction of RPO, the readiness of the backup and five fail over steps delays depending on backup capabilities.

$$\begin{aligned} RTO = fraction \quad of \quad & RPO + jmin + S1 \\ & + S2 + S3 + S4 + S5 \end{aligned} \tag{2}$$

The variables used in Equation 2 are:

fraction of RPO Computation time lost since the last backup;

jmin Depends on service readiness of the backup;

S1 Hardware setup time;

S2 Operating System initiation time;

S3 Application initiation time;

S4 Data or process state restoration time;

S5 IP address switching time.

Furthermore, as alleged by (Wood et al., 2010), disaster recovery mechanisms must meet five requirements to achieve efficient performance:

- Minimum RPO and RTO;
- Minimal footprint on the regular system operation;
- Shall be geographically separated;
- Application must be restored to a consistent state;
- DR solution shall guarantee integrity, privacy and confidentiality.

Based on this, the novel contribution in this paper is a Hadoop backup software solution, which is detailed in Section 6.

4 RELATED WORK

According to (Grishchenko, 2015), data copy or dual loading to a second Hadoop cluster using DistCP utility techniques are proposed. Data copy replicates a production cluster to a backup cluster, and Dual Loading implies storing all new data in both clusters at the same time. Dual loading data would be faster, but still faces the problem of maintaining a second expensive Hadoop backup environment and of providing a very poor RPO, since only the current version of data is available to restore.

In the same way, (Barot et al., 2015) suggests secondary cluster redundancy with DistCP or copying the incoming data to two different clusters. The study categorizes it as an Architectural Approach backup.

Finally, (Kothuri, 2016) mentions the same DistCP dual cluster solutions for HDFS backup, characterizing them as generic Hadoop backup solutions: Copying and Teeing. Basically, all the prior studies defends the same technique, but none of them established a performance benchmark to alternative approaches.

5 BACKUP SOFTWARE SOLUTION

Bacula (Bacula, 2017a) is a network distributed open source backup system (Zhang et al., 2015). According to Google Trends (Google, 2017), Bacula is the 3rd most popular enterprise multi-platform server backup system worldwide, and the leading open source solution.

Bacula supports full, differential, incremental, copy and migrate file level multiplexed jobs (Sibbald, 2011). Backups are written onto tapes or disk files, using open format bytes-sequence volumes. Compression, communication, and data encryption are other currently optional features deployed, among others.

As shown in Figure 4, Bacula follows the classic backup software modules distribution. It is comprised of: a central backup server called *Director*; specific device storage service nodes named *Storage Daemons*; and backup clients for different operating systems known as the *File Daemons*. Furthermore, there is a database titled *Catalog* where metadata is stored, e.g. job logs, termination status, a list of copied files with paths and, storage media association.

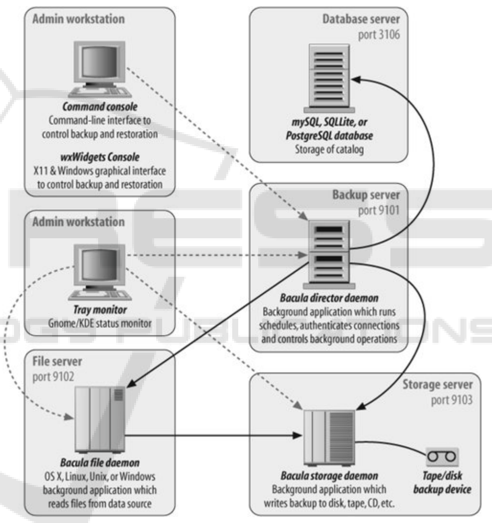
Admin workstation
Command console
Command-line interface to control backup and restoration
wxWidgets Console
X11 & Windows graphical interface to control backup and restoration
Database server
port 3106
mySQL, SQLLite, or PostgreSQL database
Storage of catalog
Admin workstation
Tray monitor
Gnome/KDE status monitor
Backup server
port 9101
Bacula director daemon
Background application which runs schedules, authenticates connections and controls background operations
File server
port 9102
Bacula file daemon
OS X, Linux, Unix, or Windows background application which reads files from data source
Storage server
port 9103
Bacula storage daemon
Background application which writes backup to disk, tape, CD, etc.
Tape/disk backup device

Figure 4: Bacula Services Distribution and Network Ports (Preston, 2007).

Bacula also has a universal backup client plugin called bpipe, which consists of an interface that reads data streams from any program to the operating system standard output when performing backups. When restoring, Bacula bpipe provides data streams to any program in order to be imported back to the original application.

6 PROPOSED BACKUP SOLUTION FOR HADOOP

As pointed out by (Jain, 1990), there are ten general steps to all performance evaluation projects that facilitate analysis accuracy, to avoid making mistakes. The first one consists of stating the goals and the boundaries of the evaluated systems. The other steps should lead to the presentation of adequate results. However, the cycle can be restarted if the results are not consistent, suggesting it is a not fixed flow. The methodology used in this research project followed these steps.

The goal of the experiment is to measure performance and determine technical parameters for executing Hadoop (version 2.7.1) HDFS backup with the latest Bacula Community (version 9.0.3), using a more efficient method than those proposed by previous studies.

As shown in Figure 5, Bacula daemons coexist with Hadoop services and their scripts are integrated using a shell script developed during this study (Faria, 2017).

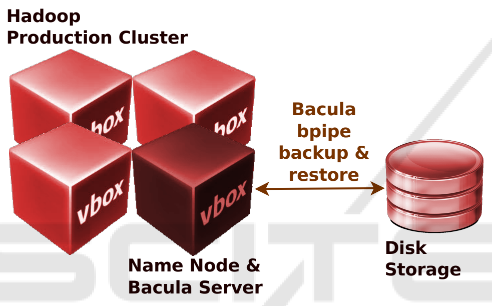

Figure 5: Hadoop New Bacula bpipe Backup Technique.

The script runs before a Bacula backup job starts and lists every HDFS files that are not in the previously full backup. Cluster file contents are streamed to the Bacula bpipe plugin during the backup job and stored in a Direct Attached Storage. The last backup date is stored in a Name Node ordinary text file, and every time a new differential backup Job runs the script uses that time to list only HDFS modified files after it. This mechanism allows Bacula to perform a file level differential backup of the cluster contents.

Other Hadoop supported protocols such as NFS (Apache Hadoop, 2017a) and FUSE-DFS (Apache Hadoop, 2017b) could be used to present HDFS contents to a backup software providing differential backup capabilities. However, as explained by (Rajgarhia and Gehani, 2010), these solutions perform extra context switches between the originating process/-kernel and the application. This indicates a limitation due to the file copy performance, which discourages its use for backups.

In comparison to our proposed technique and as displayed in Figure 3, the DistCp (distributed copy) is the other approach we tested. It is a native Hadoop client tool used for large inter/intra-cluster copying (Apache, 2017) and is used to provide replication to the secondary Hadoop backup cluster. DistCp also supports differential copies if they have access to the previously stored replica, and the analysis is mostly I/O bound.

The replication factor used by the secondary cluster in the distcp experiment is the Hadoop default, which aims to store the same data three times in the backup cluster. This leads to a significant demand for backup storage in this architecture.

7 RESULTS

Both Bacula and Hadoop cluster support common compression algorithms such as LZO and GZIP (Cloudera, 2017), but these are disabled in both experiments to provide a more transparent comparative and with faster operations. The performance metrics are:

- the full backup average duration;
- the full backup average throughput;
- the full backup storage occupancy;
- the restore duration of a full backup.

An experiment was conducted using i5-6400 Intel CPU hosts at 2.70GHz, 32GB RAM desktops. A Samsung EVO 850 SSD hosts the Ubuntu 17.04 operating system and applications, and another SSD of the same model host the cluster nodes VirtualBox (Oracle, 2017) Virtual Machines (VM). The Hadoop Cluster VMs had 4GB RAM each and one virtual CPU.

Real workload open data was used (Federal Government of Brazil, 2017), corresponding to Bolsa Família information from 2015 until 2017, a total of 52 GB of information.

All the results have a 95% confidence interval, with ten times the presented experiment unit results executed for each full copy job using different access protocols.

As shown in Figure 6, the different backup techniques represent significant differences in the duration of a full backup job. The newly proposed model using Bacula was 51% faster than the distcp experiment, significantly improving performance and capacity to back up massive data sets, such as typical HDFS workloads. This is attributed to the multiplexed data streams that are created from HDFS (one per file), which makes better use of available resources.

As displayed in Figure 7, faster backups occur because of faster throughput. Bacula bpipe backup

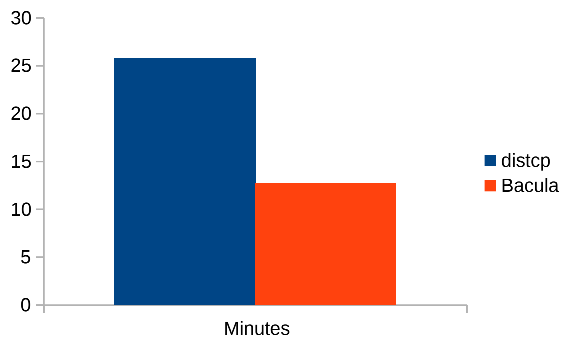

Figure 6: The Full Backup Duration.

achieved 67.97 MB/s of average efficiency when performing backups, while distcp only produced 33.55 MB/s.

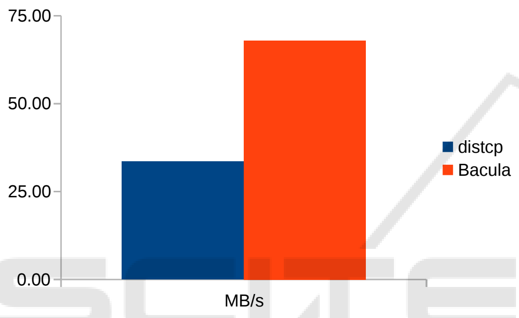

Figure 7: Backup Throughput.

Figure 8 clearly illustrates that Bacula was notably better in consuming less backup storage space (75%). As previously explained, a default HDFS replication factor of three consumes backup storage in the same proportion.

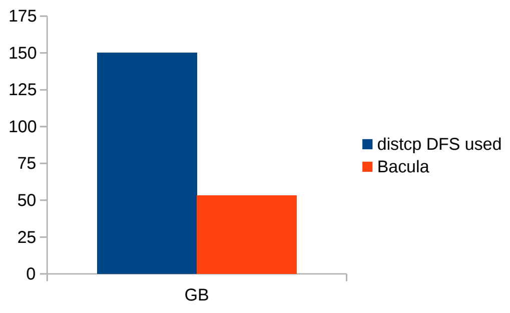

Figure 8: Full Backup Storage Occupation.

In Figure 9, results show that Bacula bpipe full restore to the HDFS cluster was 19% faster than using the former distcp inter-cluster solution. This has a direct impact on one of the most critical parameters, Restore Time Objective (RTO). Faster restores provide better business continuity, significantly minimizing the adverse effects of eventual data loss.

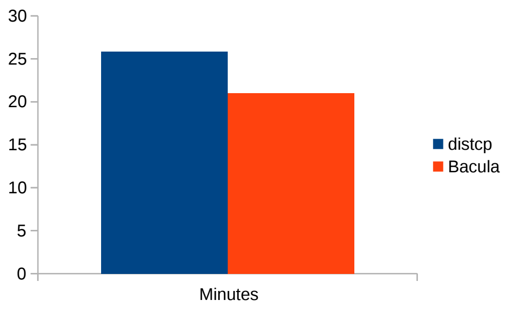

Figure 9: Restore Duration of a Full Backup.

8 CONCLUSION

The majority of current Hadoop backup research defends the need for performing secure copies of service information, such as configuration, results and even raw datasets if they are only available in HDFS.

The proposed method, Bacula backup software and its bpipe plugin, performs significantly better than solutions previously implemented, which are intercluster based. Furthermore, the new solution is by definition less expensive because it only needs storage to hold backup replicas, instead of requiring another fully configured Hadoop Cluster.

The proposed solution offers a better Recovery Point Objective, since it allows for the preservation of different versions of backup in the backup storage, generated over time. The solution also provides better Recovery Time Objectives of Hadoop clusters, which is also considered more effective than the others for providing increased HDFS availability.

In future research, we plan to extend the Hadoop cluster *ls* command to offer an option for listing files modified at a given time, similar to the *ls* from the Linux shell command. This change would make our developed script smaller, more efficient and less prone to errors in switching or processing. We also intend to execute this experiment in the cloud, testing technologies such as Swift, Amazon S3, Google and Azure Object Storage, aiming at an eventual migration.

REFERENCES

Alhazmi, O. and Malaiya, Y. (2013). Evaluating Disaster Recovery Plans Using the Cloud.

Amazon Web Services (2017a). AWS Amazon EMR Amazon EMR with the MapR Distribution for Hadoop Pricing.

Amazon Web Services (2017b). Cloud Storage Pricing – Amazon Simple Storage Service (S3) – AWS.

Apache (2017). Apache Hadoop Distributed Copy – DistCp Guide.

Apache Hadoop (2017a). HDFS NFS Gateway.

Apache Hadoop (2017b). Mounting HDFS.

Bacula (2017a). Bacula | Open Source Backup, Enterprise ready, Network Backup Tool for Linux, Unix, Mac, and Windows.

Bacula (2017b). Released Version 3.0.3 and 3.0.3a.

Barot, G., Mehta, C., and Patel, A. (2015). *Hadoop Backup and Recovery Solutions*. Packt Publishing Ltd. Google-Books-ID: _GlGCgAAQBAJ.

Cloudera (2017). Choosing a Data Compression Format.

eBay (2017). Electronics, Cars, Fashion, Collectibles, Coupons and More.

Facebook (2017). Facebook.

Faria, H. (2017). Script Backup Bacula bpipe Hadoop HDFS.

Federal Government of Brazil (2017). Portal da Transparência - Download de Dados | Despesas – Transferências – Programas Sociais – Bolsa Família - Pagamentos.

Google (2017). Google Trends.

Grishchenko, A. (2015). Hadoop Cluster Backup.

Hive, A. (2017). Apache Hive.

HortonWorks (2017). Apache Hadoop HDFS.

Hua, Y., Liu, X., and Feng, D. (2016). Cost-Efficient Remote Backup Services for Enterprise Clouds. *IEEE Transactions on Industrial Informatics*, 12(5):1650–1657.

Jain, R. (1990). *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*. Wiley. Google-Books-ID: eOR0kJjgMqkC.

Khoshkholghi, M. A., Abdullah, A., Latip, R., Subramaniam, S., and Othman, M. (2014). Disaster Recovery in Cloud Computing: A Survey. *Computer and Information Science*, 7(4):39.

Kothuri, P. (2016). Backup and Recovery for Hadoop: Plans, Survey and User Inputs.

Oracle (2017). Oracle VM VirtualBox.

Preston, C. (2007). *Backup and Recovery: Inexpensive Backup Solutions for Open Systems*. "O'Reilly Media, Inc.". Google-Books-ID: M9mbAgAAQBAJ.

Rajgarhia, A. and Gehani, A. (2010). Performance and extension of user space file systems. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, pages 206–213. ACM.

Sibbald, K. (2011). Main Reference.

The Apache Software Foundation (2017). Apache Hadoop.

White, T. (2015). *Hadoop: The definitive guide*. "O'Reilly Media, Inc.".

Wood, T., Cecchet, E., Ramakrishnan, K. K., Shenoy, P. J., van der Merwe, J. E., and Venkataramani, A. (2010). Disaster Recovery as a Cloud Service: Economic Benefits & Deployment Challenges. *HotCloud*, 10:8–15.

Xu, J., Zhang, W., Ye, S., Wei, J., and Huang, T. (2014). A Lightweight Virtual Machine Image Deduplication Backup Approach in Cloud Environment. pages 503–508. IEEE.

Yahoo (2017). Yahoo.

Zhang, X., Tan, Z., and Fan, S. (2015). NSBS: Design of a Network Storage Backup System.