Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Proposta de Implantação de uma Estrutura de Armazenamento por Objetos para Preservação Documental no Tribunal de Contas do Estado do Tocantins

Antonio Marcos Almeida Ferreira

Dissertação apresentada como requisito parcial
para conclusão do Mestrado Profissional em Computação Aplicada

Orientador
Prof. Dr. André Costa Drummond

Brasília
2014

Universidade de Brasília — UnB
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Mestrado Profissional em Computação Aplicada

Coordenador: Prof. Dr. Marcelo Ladeira

Banca examinadora composta por:

Prof. Dr. André Costa Drummond (Orientador) — CIC/UnB
Prof. Dr. Guilherme Novaes Ramos — CIC/UnB
Prof. Dr. Edison Ishikawa — CIC/UnB

CIP — Catalogação Internacional na Publicação

Ferreira, Antonio Marcos Almeida.

Proposta de Implantação de uma Estrutura de Armazenamento por Objetos para Preservação Documental no Tribunal de Contas do Estado do Tocantins / Antonio Marcos Almeida Ferreira. Brasília : UnB, 2014. 84 p. : il. ; 29,5 cm.

Dissertação (Mestrado) — Universidade de Brasília, Brasília, 2014.

1. Gerenciamento Eletrônico de Documentos, 2. Computação em Nuvem, 3. Armazenamento de Conteúdo Endereçável, 4. Deduplicação.

CDU 004.4

Endereço: Universidade de Brasília
Campus Universitário Darcy Ribeiro — Asa Norte
CEP 70910-900
Brasília–DF — Brasil

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Proposta de implantação de uma estrutura de armazenamento por objetos para preservação documental no Tribunal de Contas do Estado do Tocantins

Antonio Marcos Almeida Ferreira

Dissertação apresentada como requisito parcial para conclusão do
Mestrado Profissional em Computação Aplicada

Prof. Dr. André Costa Drummond (Orientador)
CIC/UnB

Prof. Dr. Guilherme Novaes Ramos
CIC/UnB

Prof. Dr. Edison Ishikawa
CIC/UnB

Prof. Dr. Marcelo Ladeira
Coordenador do Programa de Pós-graduação em Computação Aplicada

Brasília, 25 de junho de 2014

Dedicatória

Este trabalho é dedicado ao meu avô, que foi o meu exemplo de vida.

Agradecimentos

Inicialmente, agradeço a Deus por todas as graças em minha vida. Os agradecimentos principais são à minha família, em especial aos meus queridos avós que hoje se encontram ao lado de Deus, mas sempre me incentivaram em meus estudos, à minha admirável mãe, Maria de Fátima Almeida Ferreira, pelo seu amor, apoio e contribuição para minha formação acadêmica, à minha irmã, Amanda Carolina, pelo seu carinho, à minha tia, Raimunda Almeida, pelo seu apoio absoluto, e ao meu querido pai, Antonio.

A minha noiva, Ana Paula Siqueira Camargo, por toda paciência, compreensão, carinho e amor. Você foi a pessoa que compartilhou comigo os momentos de tristezas e alegrias, que sempre me incentivou para a realização dos meus ideais, encorajando-me a enfrentar e superar todos os obstáculos que surgiram em nossas vidas, obrigado por sempre estar ao meu lado me dando forças.

Agradeço aos meus irmãos de longa caminhada Alex Araújo, Stefano Farias, Raquidson Muniz, Adriel Barbosa, Gustavo Omizzolo, Bruno Tácito, Vinícius Miguel, Paula Miranda, Darley Passarin, Viviane Oliveira, Arylma Botelho e Ivo Sócrates por nossa amizade e nossas loucuras. E à equipe (família) do departamento de informática do Tribunal de Contas do Estado do Tocantins (Francisco de Assis, Hállison Vieira, Leonardo Sales, Fábio Castro, Gleyson Rodrigues, Vanius Girodo e Fernando Zucuni) por terem me tolerado e amparado durante todo o desenvolvimento e escrita da dissertação.

Um agradecimento muito especial aos meus amigos e companheiros (Ricardo Alves Moraes, Misael Sousa de Araújo, Rubens Ferreira dos Santos, Eduardo Henrique Ferreira Mendes Teixeira, Marcelo Monte Karam, Juvenal dos Santos Barreto e José Benedito de Souza Brito) da primeira turma do curso de Mestrado em Computação Aplicada pela UnB. Muito obrigado por me acolherem como amigo. Vocês já fazem parte da minha vida!

Agradeço de maneira especial ao meu orientador e mentor, professor Dr. André Costa Drummond pelo apoio e conhecimentos repassados, por ter me direcionado e orientado pacientemente.

Resumo

Esta dissertação descreve o uso do Gerenciamento Eletrônico de Documentos (GED) para gerenciar a inserção, compartilhamento e recuperação de informações atribuídas a um documento digital. Devido ao uso desta tecnologia no órgão alvo deste trabalho, Tribunal de Contas do Estado do Tocantins (TCE-TO), faz-se necessário que os processos de busca, indexação e armazenamento de documentos digitais sejam realizados de forma eficaz. Além do uso de ferramentas computacionais voltadas ao arquivamento de documentos digitais, é necessário que a estrutura de armazenamento de dados forneça meios que propiciem a disponibilização de documentos digitais com fidedignidade e autenticidade. Relacionado ao contexto apresentado, foi identificada a necessidade do desenvolvimento de uma estrutura de armazenamento por objetos para o arquivamento de documentos digitais direcionada a uma plataforma em nuvem. A dissertação apresenta também conceitos sobre computação em nuvem, armazenamento em nuvem, armazenamento de conteúdo endereçável, sistemas de armazenamento, deduplicação de dados e mecanismo de integridade. A implantação desta estrutura de armazenamento propicia melhorias em confiabilidade (fidedignidade e autenticidade aos documentos digitais) com a integração da deduplicação em nível de arquivos e ganhos em escalabilidade com o armazenamento de arquivos em recipientes. Assim como amplia a disponibilidade dos dados devido a replicação de objetos para os demais servidores e acrescenta eficiência no gerenciamento do acervo documental do TCE-TO.

Palavras-chave: Gerenciamento Eletrônico de Documentos, Computação em Nuvem, Armazenamento de Conteúdo Endereçável, Deduplicação.

Abstract

This dissertation describes the use of Electronic Document Management (EDM), to manage the insertion, sharing and recovery of information assigned to a digital document. Due to the implementation of this technology in the Court of Auditors of the State of Tocantins (TCE-TO), it is necessary that the processes of searching, indexing and storing are performed effectively. Besides the use of tools aimed at archiving digital documents, it is necessary that the structure of data storage provide a means to digital documents with reliability and authenticity. In the presented context, we identified the need to develop a structure of storage for objects for archiving digital documents directed to a cloud platform. The dissertation also presents concepts about cloud computing, cloud storage, content addressable storage, storage systems, data deduplication and data integrity mechanism. The implementation of this storage structure provides improvements in reliability (reliability and authenticity to digital documents) with the integration of file-level deduplication and scalability gains with file storage containers. As increases the availability of data due to replication of objects to other servers and adds efficiency in managing document archiving TCE-TO.

Keywords: Electronic Document Management, Cloud Computing, Content Addressable Storage, Deduplication.

Sumário

Lista de Figuras

Lista de Tabelas

Abreviaturas e Siglas

API Application Programming Interface

ATA Advanced Technology Attachment

CA Content Address

CAS Content Addressed Storage

CIFS Common Internet File System

DAS Direct Attached Storage

EDM Electronic Document Management

FC Fibre Channel

GED Gestão Eletrônica de Documentos/Gerenciamento Eletrônico de Documentos

HBA Host Bus Adapter

HTTP Hypertext Transfer Protocol

IaaS Infrastructure as a Service

IDE Integrated Drive Electronics

IP Internet Protocol

iSCSI Internet Small Computer Storage Interface

LAN Local Area Network

MD5 Message-Digest algorithm 5

MS Milissegundo

NAS Network-attached storage

NFS Network File System

NIST National Institute of Standards and Technology

OID Unique Object Identifier

OSD Object-based Storage Device

PaaS Platform as a Service

PHP Hypertext Preprocessor

RPC Remote Procedure Call

RTT Round-Trip Time

SaaS Software as a Service

SAN Storage Area Network

SAS Serial Attached SCSI

SATA Serial Advanced Technology Attachment

SCSI Small Computer System Interface

SGBD Sistema de Gerenciamento de Banco de Dados

SHA Secure Hash Algorithm

SIS Single Instance Storage

SSL Secure Sockets Layer

TCE-TO Tribunal de Contas do Estado do Tocantins

TCP Transmission Control Protocol

URL Uniform Resource Locator

UUIDs Universally Unique Identifiers

Capítulo 1

Introdução

O avanço tecnológico na última década em diferentes áreas computacionais como *Internet*, linguagens de programação, serviços sob demanda, dentre outras áreas gerou uma necessidade por sistemas de armazenamento com características que disponham de um melhor desempenho, serviços de redundância, escalabilidade e segurança da grande massa de dados digitais em ampla expansão [42].

A adoção de sistemas/equipamentos em órgãos públicos atribuídos à gestão de dados (documentos digitais, imagens, áudio, dentre outras formas de mídias) tem aumentado com a finalidade de solucionar problemas de espaço físico, localização de documentos, minimização de perda, agilidade, entre outros [32].

Atualmente é possível ter acesso a informações e documentos digitais independentemente da localização geográfica deles e eles podem ser preservados por períodos de tempos mais longos se comparados com a mídia impressa. De acordo com [28], *"a questão da preservação digital se apresenta como um problema real a ser solucionado pelas instituições, principalmente aquelas que têm por obrigação legal a manutenção de documentos a longo prazo [...]"*.

A adoção de sistemas voltados ao gerenciamento eletrônico de documentos (GED) torna possível manipular um documento digital de forma organizada por usuários de departamentos distintos, por isso, envolve tarefas tais como gerenciar o ciclo de vida documental, preservar os documentos originais, melhorar a segurança lógica no armazenamento, reduzir o espaço físico necessário para o armazenamento de dados, bem como otimizar a realização de consultas e análises em documentos [21].

Com foco de ampliar a eficácia da tramitação e divulgação de documentos digitais dentro dos órgãos públicos, a gestão de documentos foi institucionalizada e denominada "Lei de Arquivos" com a aprovação da Lei 8.159, de 08 de janeiro de 1991, cujo artigo 3º considera a gestão de documentos como *"[...] o conjunto de procedimentos e transações*

técnicas a sua produção, tramitação, uso, avaliação e arquivamento em fase corrente e intermediária, visando sua eliminação ou recolhimento para guarda permanente" [6].

É dever da administração pública prover a gestão documental, contudo os recursos tecnológicos como, por exemplo, o dispositivo de armazenamento em fita magnética, é propícia a problemas causados seja por poluição ou umidade atmosférica. Para os discos de armazenamento de dados, têm-se má utilização de espaço, o que acarreta custos maiores com aquisição de equipamentos.

Para [10], a preservação digital deve assegurar que o conteúdo em formato digital mantenha-se acessível ao longo do tempo e deve propiciar fidedignidade (confiabilidade de um documento como prova referente ao que se trata) e autenticidade (fidedignidade ao longo do tempo) ao documento digital. Com base neste contexto, observou-se como necessidade a estruturação de uma arquitetura de armazenamento e disponibilização de dados de forma íntegra e confiável pertinente às demandas do Tribunal de Contas do Estado do Tocantins.

Este trabalho propõe a implantação de uma infraestrutura direcionada ao armazenamento de conteúdo endereçável, com integridade e disponibilidade das informações, por meio de uma nuvem privada.

1.1 Motivação

Além de benefícios como redução do espaço físico para o armazenamento de documentos textuais e maior agilidade no atendimento ao público através de sistemas GED, ainda existem fatores (fidedignidade e autenticidade) relacionados com a segurança e o armazenamento dos documentos digitais. Publicações realizadas por [14] e [25] descrevem a importância dos sistemas GED nas organizações, mas não expõem as estruturas de armazenamento de dados viáveis para preservar um documento digital.

Devido à existência de sistemas GED dentro do Tribunal de Contas do Estado do Tocantins, organização alvo deste projeto, observou-se a necessidade de um estudo para a implantação de uma infraestrutura em uma nuvem privada e voltada ao armazenamento, que proporcione maior segurança para a salvaguarda dos documentos digitais que tramitam dentro deste órgão.

1.2 Objetivos Gerais

Elaborar uma arquitetura de armazenamento de dados adaptada às demandas do Tribunal de Contas do Estado do Tocantins que arquive e proteja informações oriundas do sistema GED de forma imutável e disponibilize os documentos através de uma nuvem

privada de forma a oferecer confiabilidade (fidedignidade e autenticidade) e melhor desempenho.

1.3 Objetivos Específicos

- Realizar uma revisão de literatura dos temas relacionados ao projeto.
- Identificar parâmetros básicos de um repositório para disponibilização dos dados com base nos requisitos do Tribunal de Contas do Estado do Tocantins que ofereçam fidedignidade e autenticidade aos documentos digitais.
- Propor uma solução voltada para a garantia da integridade dos dados armazenados e que aumente a disponibilidade destes dados em uma infraestrutura que reduza as perdas decorrentes de paradas inesperadas.
- Propor uma arquitetura de armazenamento de dados segura e escalável adaptada às demandas do Tribunal de Contas do Estado do Tocantins.

1.4 Estrutura do Trabalho

Esta dissertação de mestrado está organizada em seis capítulos, incluindo este introdutório, conforme descrito a seguir:

O Capítulo 2 apresenta os conceitos básicos sobre o gerenciamento eletrônico de documentos, computação em nuvem, armazenamento em nuvem, armazenamento de conteúdo endereçável, questões de segurança e sistemas de armazenamento.
O Capítulo 3 descreve técnicas e metodologias presentes na literatura voltadas à proposta do tema exposto neste trabalho.
O Capítulo 4 apresenta a solução proposta para o armazenamento de dados por objetos e a metodologia adotada na avaliação da solução.
O Capítulo 5 apresenta a avaliação realizada para verificar a qualidade da solução proposta com base em métricas voltadas ao desempenho, disponibilidade e escalabidade.
O Capítulo 6 apresenta as principais conclusões acerca do trabalho desenvolvido e indica sugestões de possíveis trabalhos futuros.
Apresenta também apêndices com os *scripts* utilizados no ambiente de avaliação.

Capítulo 2

Conceitos Básicos

Este capítulo apresenta os principais conceitos relacionados a este trabalho, que envolvem pesquisas temáticas ou subtemáticas com referência ao gerenciamento eletrônico de documentos, computação em nuvem, armazenamento em nuvem, armazenamento de conteúdo endereçável, questões de segurança e sistemas de armazenamento. Também é ressaltado que ao decorrer dos demais capítulos termos como *kilobytes* e *Megabytes* serão trocados pelas suas respectivas siglas KB e MB, uma vez que seja necessária sua alteração.

2.1 Gerenciamento Eletrônico de Documentos

O Tribunal de Contas do Estado do Tocantins, bem como qualquer organização pública/privada, produz documentos textuais. O gerenciamento desses documentos em mídia impressa é complexo devido aos custos com espaço de armazenamento, necessidade de pessoal qualificado, morosidade na recuperação da informação e, em determinadas situações, destruição precoce do documento.

Por meio da utilização de tecnologias da informação, é possível guardar tais documentos em mídias digitais, o que possibilita verificar a integridade, acesso e recuperação com maior eficácia e eficiência administrativa. Também ocorrem benefícios com a redução de espaço físico necessário para o arquivamento e durabilidade ao se utilizar o método de duplicação dos dados digitais.

A Gestão Eletrônica de Documentos ou Gerenciamento Eletrônico de Documentos, é a tecnologia que permite gerir eletronicamente documentos e conteúdos ligados aos processos de uma empresa. Ela facilita o armazenamento, indexação e gerenciamento de conteúdo e processos de uma instituição [11].

Para [10] e [44], um documento digital precisa ter certas particularidades para possuir um valor legal. Desta forma, a preservação digital tem como base sua gestão de dados

sobre os princípios da autenticidade e integridade. A confiabilidade é obtida por meio da fidedignidade em conjunto com a autenticidade, em que:

- Fidedignidade: refere-se à autoridade e à confiabilidade de um documento como prova referente a que se trata.
- Autenticidade: refere-se à fidedignidade ao longo do tempo. Está relacionada com a forma de transmissão e as estratégias de preservação e custódia.

Autores como [22] explicam o processo de implantação do GED e descrevem os tipos existentes de metodologias, porém não expõem a arquitetura a ser planejada para o tipo de dados que será armazenado. Fator este que deve ser considerado uma vez que a preservação dos documentos digitais, bem como a facilidade da sua recuperação no decorrer do tempo, são obtidas através de uma arquitetura com tais características físicas e lógicas.

2.2 Computação em Nuvem

O termo computação em nuvem não possui uma definição específica, muitas definições têm sido propostas por pesquisadores tanto de empresas quanto de universidades. Neste trabalho é adotada a definição proposta pelo Instituto Nacional de Padrões e Tecnologia (NIST - *National Institute of Standards and Technology*), que é uma agência direcionada à inovação tecnológica dos Estados Unidos da América.

O trabalho de [31] descreve que a computação em nuvem é um modelo que possibilita acesso a recursos computacionais (*hardware* e *software*) que podem ser rapidamente configurados e liberados proporcionalmente a demanda do usuário. Em resumo, pode-se dizer que a plataforma de computação em nuvem disponibiliza serviços que são distribuídos para os usuários/empresas de acordo com a sua respectiva necessidade.

Também é proposta pelo NIST [31], a divisão entre as características essenciais que devem existir na computação em nuvem; os tipos de modelos de serviços da qual a camada base é a Infraestrutura como serviço (Figura 1) e os demais modelos implementados sobre ele. Por fim, o modelo de implementação que descreve qual o tipo de nuvem a ser utilizada pelo usuário/empresa.

2.2.1 Características Essenciais

A computação em nuvem oferece vários tipos de recursos computacionais, como armazenamento de dados, redes, servidores e serviços, que podem ser configurados de acordo com a necessidade do usuário. As cinco características essenciais da computação em nuvem são [31]:

- Autoatendimento sob demanda - o usuário pode aumentar ou diminuir recursos computacionais sob demanda. Por exemplo, ter o espaço de armazenamento em disco ampliado no momento em que houver necessidade.
- Amplo acesso à rede – os recursos estão disponíveis através da rede e possuem suporte a mecanismos heterogêneos (exemplo: telefone, *notebook*, *tablet*, entre outros).
- Pool de recursos – recursos físicos ou virtuais de diferentes localidades (cidades, estados ou países) são atribuídos dinamicamente pela rede segundo a necessidade do consumidor.
- Elasticidade rápida – capacidade de ampliação ou redução de recursos computacionais de acordo com a necessidade.
- Serviços mensuráveis – os recursos podem ser monitorados, controlados e medidos, tanto para o provedor de serviços quanto ao consumidor.

2.2.2 Modelos de Serviços

Os modelos de serviços apresentados na Figura 1 são classificados de acordo com suas características e controle de recursos providos por eles, que são representados sob a forma de uma pirâmide, na qual o custo operacional é maior na camada destinada à disponibilização de aplicações. Equivalente à necessidade de nível de abstração computacional, é possível obter um maior nível de gerenciamento e flexibilidade aos demais tipos de serviços existentes.

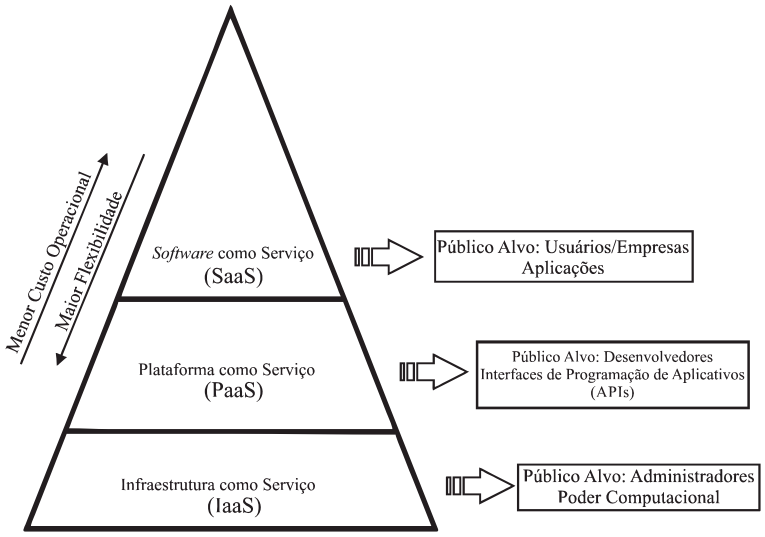


Figura 1: Modelos de Serviço.

A classificação de serviços de computação em nuvem recomendada pelo NIST é descrita como [31]:

- *Software* como Serviço (SaaS) – Possui o mais alto nível de abstração dentre os modelos de serviços. Os aplicativos que estão em execução em um servidor são disponibilizados pela nuvem e acessíveis através de um cliente como um navegador *web*. O usuário não possui controle sobre a nuvem desta infraestrutura (desde a rede, servidores, sistemas operacionais, dispositivos de armazenamento, etc), mas somente sobre os parâmetros de configuração da aplicação específicos do usuário. Exemplos de serviços SaaS: *Google Docs* [39], *Justcloud* [41] e *Salesforce* [50].
- Plataforma como serviço (PaaS) – Neste modelo de serviço, o usuário possui controle sobre linguagens de programação, bibliotecas, interfaces de programação de aplicativos (APIs) e alguns parâmetros da configuração disponibilizados pelo servidor em nuvem. Exemplos de provedores de PaaS: *Google App Engine* [36], *Microsoft Azure* [5] e *Heroku* [40].
- Infraestrutura como serviço (IaaS) – Modelo de serviço com o menor nível de abstração e alto nível de controle de recursos. É possível gerenciar sistemas operacionais, dispositivos de armazenamento, memória física, processadores e aplicativos. Os recursos físicos geralmente são compartilhados com outros servidores, desde o uso de virtualização a *middlewares*. Exemplos de provedores de IaaS: *Google Compute Engine* [38], *RackSpace* [48] e *Amazon Elastic Compute Cloud* [2].

2.2.3 Modelo de Implantação

O modelo de implantação (Figura 2) refere-se à localização e gestão da infraestrutura da nuvem, que propicia diferentes níveis de acesso e de serviços, no qual o nível de segurança é maior de acordo com o tipo de usuário, seja um cliente físico ou uma empresa. A definição para os quatro modelos de implementação é a seguinte [31]:

1. Nuvem pública: a infraestrutura de nuvem pública está disponível para uso público, contudo sua administração é restrita à empresa que a fornece.
2. Nuvem privada: a infraestrutura de nuvem privada é de uso exclusivo de uma empresa ou pessoa. Pode estar localizada dentro da própria empresa ou ser adquirida de uma empresa terceirizada com um nível de segurança mais elevado.
3. Nuvem híbrida: uma nuvem híbrida é a combinação de nuvens (pública e privada), cujas particularidades são mantidas separadas, contudo são associadas como uma

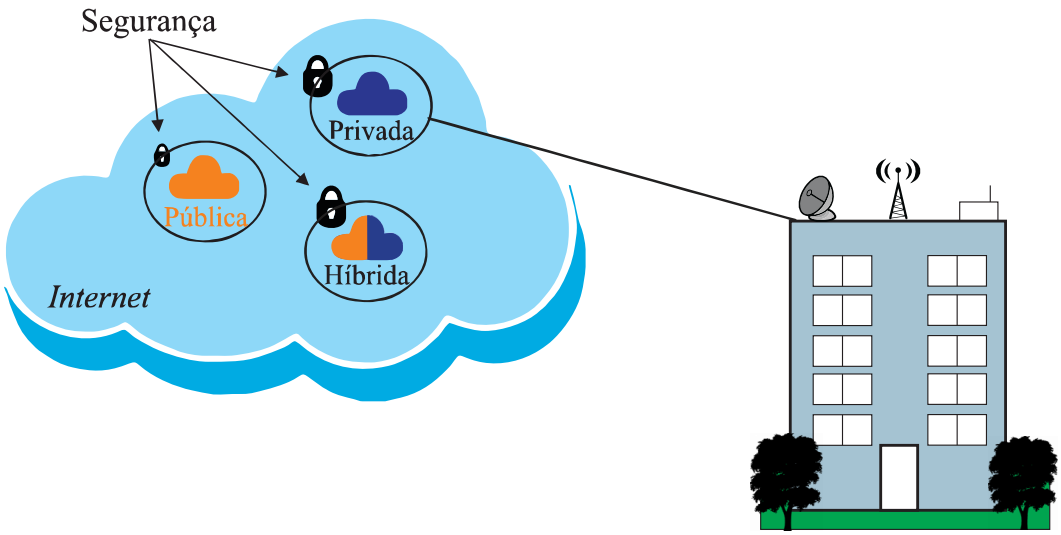


Figura 2: Tipos de Nuvens.

unidade. O que possibilita este modelo oferecer acesso padronizado de dados e aplicações, bem como a portabilidade de aplicativos.

4. Nuvem comunitária: a infraestrutura da nuvem é compartilhada por diferentes empresas que possuem preocupações em comum, por exemplo: missão, políticas de segurança e necessidades de conformidade regulamentar. Uma nuvem comunitária pode ser gerida por uma das organizações integrantes ou por uma terceira parte.

2.3 Armazenamento em Nuvem

No contexto de [35], um serviço de armazenamento de objetos em nuvem pode ser público ou privado, normalmente disponibilizados por soluções proprietárias, por exemplo *Amazon S3* [3] e *Google Cloud Storage* [37], bem como soluções com código fonte aberto, como *Eucalyptus Walrus* [33].

Os provedores de nuvem IaaS dispõem de serviços de armazenamento de diferentes tipos, que são disponibilizados de acordo com a necessidade da aplicação e dos requisitos de armazenamento. Para o entendimento sobre qual o tipo de armazenamento é recomendável para este trabalho, as subseções 2.3.1 e 2.3.2 apresentam os dois principais tipos de armazenamento: em bloco e de objetos.

2.3.1 Armazenamento em Bloco

De acordo com [51], para o acesso em nível de bloco *"[...] os dados são armazenados e recuperados de discos por meio da especificação do endereço lógico de blocos. O endereço*

dos blocos é derivado de acordo com a configuração geométrica dos discos".

Um bloco é uma sequência de *bits* ou *bytes* que possui um tamanho fixo, do qual um único arquivo, com tamanho de 64 MB, por exemplo, pode ser dividido em blocos com tamanhos de (4 KB, 8 KB, 16 KB ou 64 KB). Desta forma o tamanho do bloco utilizado, impacta na eficiência do dispositivo devido à fragmentação em disco.

As unidades de disco conectadas a um servidor podem ser acessadas por múltiplos servidores, cujos recursos de armazenamento em bloco são fornecidos por meio de uma interface de barramento como SCSI (*Small Computer System Interface*) ou ATA (*Advanced Technology Attachment*), ou também através de uma interface de comunicação conectada a uma rede de área de armazenamento como SAN (*Storage Area Network*).

Para os dispositivos em rede que necessitam de comunicação com os servidores de armazenamento baseado em bloco, é necessário criar um volume (representa o tamanho do espaço em disco a ser utilizado para armazenamento), instalar um sistema operacional e posteriormente vincular o volume ao dispositivo de armazenamento.

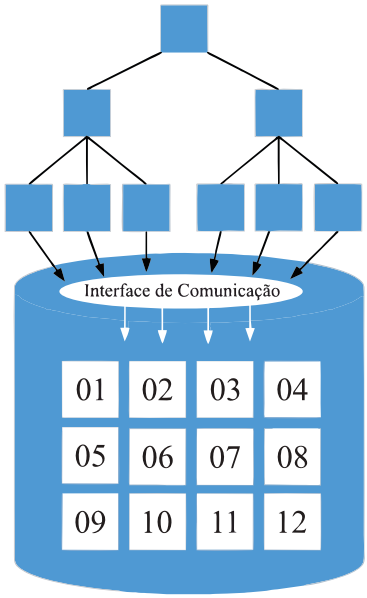


Figura 3: Sistema de Arquivos Hierárquico Tradicional.

O sistema de arquivos hierárquico, apresentado na Figura 3, ilustra o modo de acesso ao disco com base nos endereços físicos alocados nos discos, junto com estruturas de diretórios, para fins de organização. O tipo de interface de comunicação determina o desempenho do disco. Dentre as principais interfaces de comunicação tem-se ATA, SCSI, SAS (*Serial Attached SCSI*), SATA (*Serial Advanced Technology Attachment*) e iSCSI (*Internet Small Computer Storage Interface*). Este tipo de armazenamento é indicado normalmente para aplicações como, um servidor de arquivos.

2.3.2 Armazenamento de Objetos

De acordo com [46], um "objeto"é um recipiente para dados e atributos cujo protocolo de armazenamento baseado em objetos (OBS - *object-based storage*) especifica as transações (leitura/escrita/execução) que são realizadas com base no dispositivo de armazenamento baseado em objetos (OSD - *object-based storage device*). Cada dado armazenado tem um identificador de objetos (OID - *Object Identifier*) que permite que um servidor ou usuário tenha acesso a este dado sem a necessidade de saber a sua localização física.

A identificação do objeto é gerada através de um algoritmo com "função de *hash*[1] ", que determina para cada objeto uma identificação exclusiva. Não há um sistema de arquivos ou diretórios hierárquicos (Figura 3). Nesta arquitetura, o acesso ao dado armazenado é direto, assim como apresentado na Figura 4.

A comunicação com este dispositivo de armazenamento geralmente é realizado por meio de uma conexão via protocolo de transferência de *hipertexto* (HTTP – *Hypertext Transfer Protocol*). Os objetos armazenados são inseridos e recuperados por meio de um ID que é gerado para cada objeto, o que possibilita aos aplicativos armazenarem ou acessarem os dados sem a necessidade de instalação/configuração de um sistema de arquivos.

Em concordância ao ilustrado na Figura 4, cada objeto armazenado pode possuir informações tais como: identificação do objeto, atributos, dados e metadados. Exemplos de empresas que ofertam/dispõem de armazenamento direcionado aos objetos: *Amazon Simple Storage Service* (S3) e *Google Cloud Storage*.

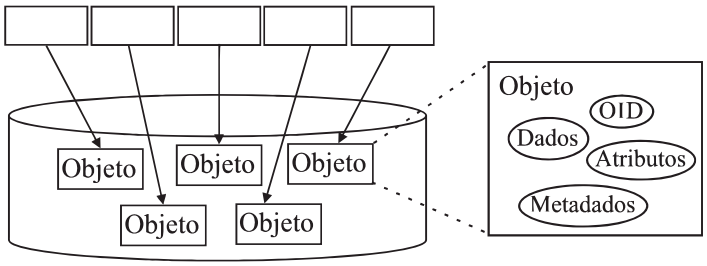


Figura 4: Arquitetura por Objeto.

[1]São usados para produzir os endereços de conteúdo chamados "*message digests*" (MD), que também são conhecidos como "*hashes*". Os MDs são funções matemáticas que processam as informações de forma a produzir uma "mensagem *digest*" diferente para cada documento [46].

2.4 Armazenamento de Conteúdo Endereçável

A necessidade de um ambiente que possibilite a preservação em diferentes tipos de mídia está correlacionada à solução tecnológica (armazenamento por bloco ou por objeto) implantada de acordo com o nível de segurança requerido pelos documentos digitais e a eficiência na recuperação dos dados. Segundo [51], embora as tecnologias (mídias de armazenamento, discos óticos e discos magnéticos) armazenem conteúdo, nenhuma delas fornece requisitos exclusivos para armazenamento e acesso ao conteúdo fixo[2].

O armazenamento de conteúdo endereçável (CAS - *Content Addressable Storage*) utiliza *hash* para reduzir os requisitos de armazenamento com base na localização de objetos de dados já armazenados. Exemplos de sistemas que utilizam armazenamento fundamentados em CAS incluem: *Deep Store* [9], *Alfresco* [1] e Presidio [20].

Segundo [52], CAS é definido como um dispositivo de armazenamento baseado em objeto, que possui um endereço único conhecido como endereço de conteúdo (CA - *Content Address*). Ao contrário dos endereços com base em localização, os endereços de conteúdo, uma vez calculados, não são alterados e sempre se referem ao mesmo conteúdo. Caso ocorra uma modificação no conteúdo de um objeto, um novo CA é calculado e atribuído ao novo conteúdo.

Com a tecnologia CAS, há a eliminação da necessidade dos aplicativos conhecerem e gerenciarem a localização física das informações no dispositivo de armazenamento. Com base nesta característica, este tipo de tecnologia é indicada para a estrutura de armazenamento proposta neste trabalho, pois possibilita acesso direto a informação armazenada.

O CAS é um sistema que armazena dados e atributos com base no seu endereço de conteúdo. Também são armazenados os metadados, *"informação acerca de informação"* [26], que descrevem o conteúdo, qualidade, condição e outras características dos dados armazenados.

A Figura 5 ilustra o servidor com API responsável pelas chamadas de função que interliga e processa toda comunicação entre um conjunto de servidores CAS e uma estação cliente. O endereço de conteúdo gerado por um servidor CAS é um identificador que aborda exclusivamente o conteúdo de um arquivo e não a sua localização.

De acordo com [51], os principais benefícios de um dispositivo CAS são:

- Autenticidade de conteúdo: para cada procedimento de leitura realizado, o dispositivo CAS utiliza um algoritmo *hashing* para recalcular o endereço do conteúdo

[2] *"À medida que ficam mais antigos, os arquivos são menos alterados e acabam se tornando "fixos", mas continuam a ser acessados por vários aplicativos e usuários"* [51].

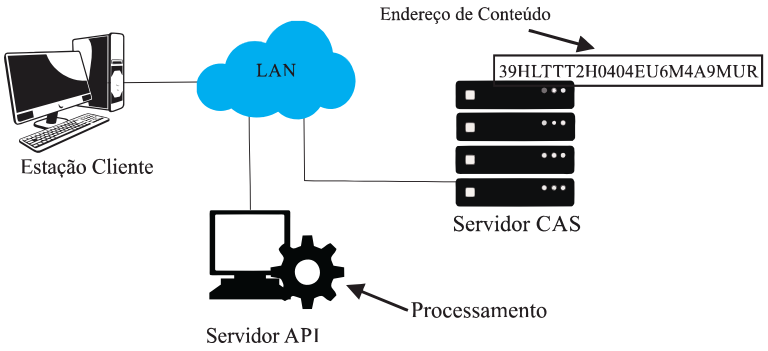


Figura 5: Arquitetura CAS.

do objeto e comparar o resultado com o endereço do conteúdo original, ou seja, possui um processo implícito de validação do conteúdo armazenado.

- Integridade de conteúdo: refere-se à garantia de que o conteúdo armazenado não tenha sido alterado. Por utilizar um algoritmo *hashing* para o procedimento de autenticidade de conteúdo, acaba também assegurando a integridade do conteúdo no CAS.

- Independência de localização: utiliza o endereço de conteúdo para acessar conteúdos fixos, torna irrelevante a localização física dos dados para o aplicativo que o solicita, o que propicia mobilidade de conteúdo independente da sua localização física.

- Armazenamento de instância única (SIS): a assinatura única é usada para garantir o armazenamento de somente uma única instância de um objeto, a qual é derivada da representação binária do mesmo.

- Imposição de retenção: o CAS institui dois componentes imutáveis: um objeto de dados e um metaobjeto (responsável por conter os atributos do objeto e as políticas de manipulação de dados) para cada objeto armazenado. O que torna possível, por exemplo, determinar por meio de sistemas que suportam recursos de retenção de objetos, o tempo de vida de um objeto armazenado.

- Proteção em nível de registro e disposição: todo conteúdo fixo é armazenado no CAS uma vez, e é feito o *backup* com um esquema de proteção. Também é possível obter um nível extra de proteção através da replicação[3] do conteúdo.

[3]De acordo com [52] *"A replicação pode ser classificada em duas grandes categorias: local e remota. A replicação local refere-se a replicação de dados dentro da mesma matriz ou o mesmo centro de dados. A replicação remota refere a replicação de dados em um local remoto".*

2.5 Sistemas de Armazenamento

Os sistemas de armazenamentos suportam uma grande quantidade de discos rígidos, o que torna possível a formação de um dispositivo com alta capacidade de armazenamento de arquivos. Dentre as suas características, destaca-se a inserção de equipamentos com dispositivos redundantes tais como fontes de energia, memória *cache*, dispositivos de rede e Conjunto Redundante de Discos Independentes (RAID - *Redundant Array of Independent Disks*).

Atualmente, as três principais arquiteturas para armazenamento em redes utilizadas em TI são DAS (*Direct Attached Storage*), NAS (*Network-attached storage*), e SAN (*Storage Area Network*).

2.5.1 Armazenamento por Conexão Direta

Segundo [12], no Armazenamento por Conexão Direta, as unidades de disco estão localizadas dentro do gabinete (caixa de servidor) conectadas diretamente a um controlador de disco, cuja conexão pode ser estabelecida, por exemplo, por meio de um Adaptador de Barramento de Hospedeiro, bem como por uma conexão externa compartilhada e acessível por meio de uma conexão SAS, iSCSI ou FC (*Fibre Channel*). A sua característica de conexão é um fator que distingue a arquitetura DAS frente às demais.

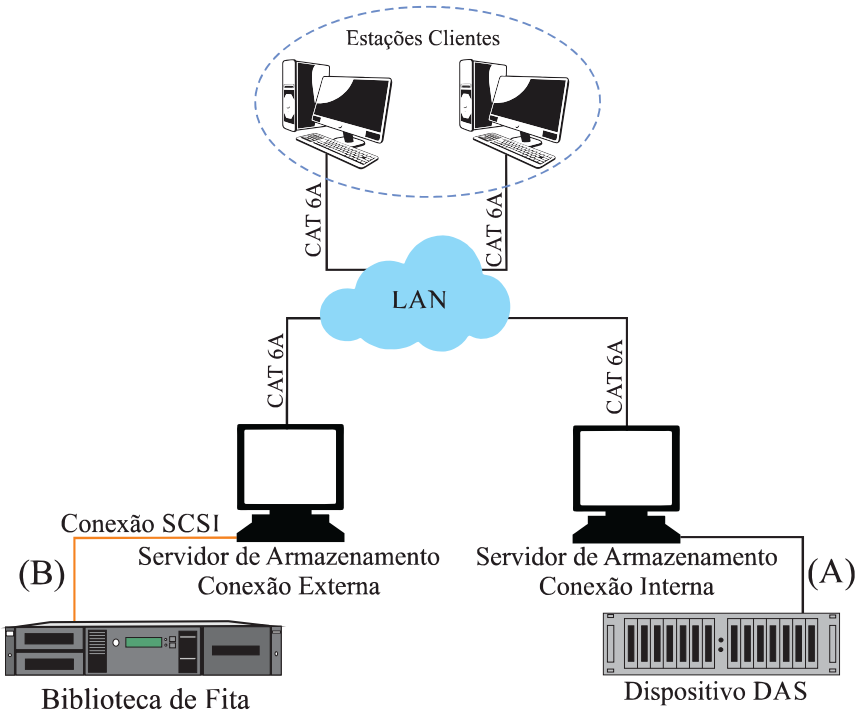


Figura 6: Arquitetura DAS Interna (A) e Externa (B).

A Figura 6 ilustra um dispositivo de armazenamento interno (A), pois mesmo sem a existência de um equipamento de rede conectado, ainda se trata de uma topologia DAS, e externo (B) em que é possível conectar equipamentos diretamente ao servidor de armazenamento através de um protocolo de comunicação local como o IDE/ATA ou o SCSI. As duas arquiteturas fornecem armazenamento/acesso de dados a estações clientes em uma rede local com cabeamento CAT6A (pode atingir velocidade de tráfego de até 10 *Gigabits*).

Como vantagem neste tipo de armazenamento, ocorre o alto desempenho para transferência de arquivos. Como pontos negativos, têm-se o baixo nível de segurança e o número limitado de 15 (valor máximo suportado) dispositivos por servidor.

2.5.2 Armazenamento Conectado à Rede

Na publicação de [29], Armazenamento Conectado à Rede é uma tecnologia da qual o sistema de armazenamento se conecta diretamente a uma rede por meio de uma interface de rede local, sejam cabos metálicos ou fibra óptica, e utiliza mensagens de protocolos de comunicação como TCP/IP para acesso orientado a arquivos.

Uma arquitetura em NAS pode conter um ou mais discos rígidos conectados a um ou mais servidores. Sua conexão em rede é realizada por meio de protocolos de acesso como NFS (*Network File System*[4]) ou CIFS (*Common Internet File System*[5]).

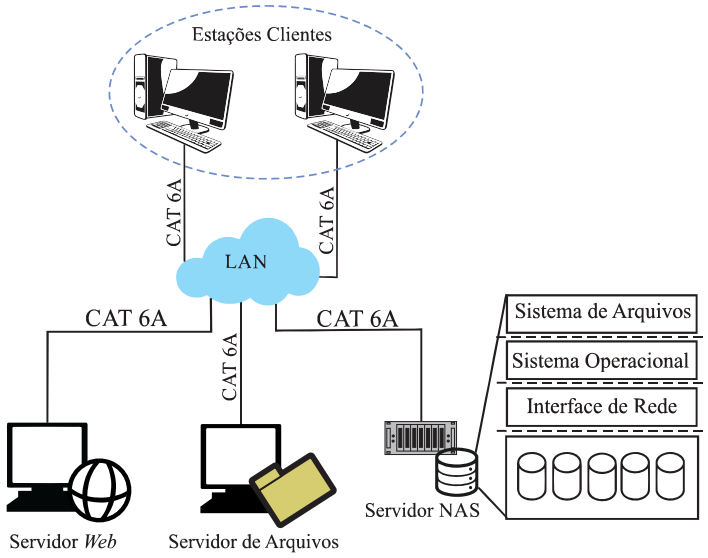


Figura 7: Armazenamento Conectado à Rede.

[4] *"NFS é um protocolo cliente-servidor para compartilhamento de arquivos que é comumente usado em um sistema UNIX"* [52].

[5] *"CIFS é protocolo de aplicação cliente-servidor que permite que programas clientes possam fazer pedidos de arquivos e serviços em computadores remotos através de TCP/IP"* [52].

A Figura 7 ilustra um servidor NAS interligado a uma rede de área local que provê armazenamento/acesso a dados para estações clientes e demais servidores. Sobre o servidor NAS, existe um sistema de arquivos, um sistema operacional, interface(s) de rede(s) e discos de armazenamento de dados, todos agregados sobre um único equipamento. Este dispositivo tem como função a gerência do protocolo de entrada/saída que será responsável pela transferência de dados entre si e os demais clientes conectados.

Em uma topologia NAS, o sistema de arquivo é compartilhado entre múltiplos clientes, ou seja, proporciona escalabilidade. Segundo [51], dentre vantagens em NAS, tem-se:

- Armazenamento centralizado: armazenamento de dados em um único ponto, cujo intuito é reduzir o número de dados duplicados;
- Gerenciamento Simplificado: possibilita gerenciar os sistemas de arquivos por meio de console centralizado;
- Alta disponibilidade: dispõe de opções de replicação e recuperação de dados.
- Segurança: permite autenticação de usuários e bloqueio de arquivos de acordo com esquemas de segurança estabelecidos.

Um ponto categórico nesta arquitetura é que devido ao número elevado de conexões que podem ser estabelecidas, a execução de tarefas pode exigir um maior poder de processamento, o que impacta nas limitações físicas por equipamento. Outro ponto a ser observado é o aumento considerável de tráfego em rede devido ao suporte a múltiplas conexões simultâneas.

2.5.3 Rede de Área de Armazenamento

Para [23], uma rede de área de armazenamento tem como objetivo principal a transferência de dados em alta velocidade através de dispositivos óticos entre os elementos de armazenamento e os computadores. Em resumo, uma rede SAN é vista como uma infraestrutura que é composta por vários hospedeiros que possuem acesso a diferentes dispositivos de armazenamentos em uma estrutura de rede exclusiva que tem como finalidade propor meios de comunicação seguro e com altas taxas de transferência de dados em rede.

A Figura 8 ilustra uma rede SAN representada por uma rede FC SAN na qual é possível interligar diferentes equipamentos (*hardware*/*software*) ao lado de uma rede de área local com estações clientes. Observa-se que as tecnologias não estão isoladas, o que ocorre é uma convergência entre os dispositivos existentes nas zonas A e B de comunicação.

Uma arquitetura FC SAN dispõe de uma melhor segurança de acesso aos dados em rede em comparação as redes DAS e NAS, pois utiliza zonas de interconexão (no qual

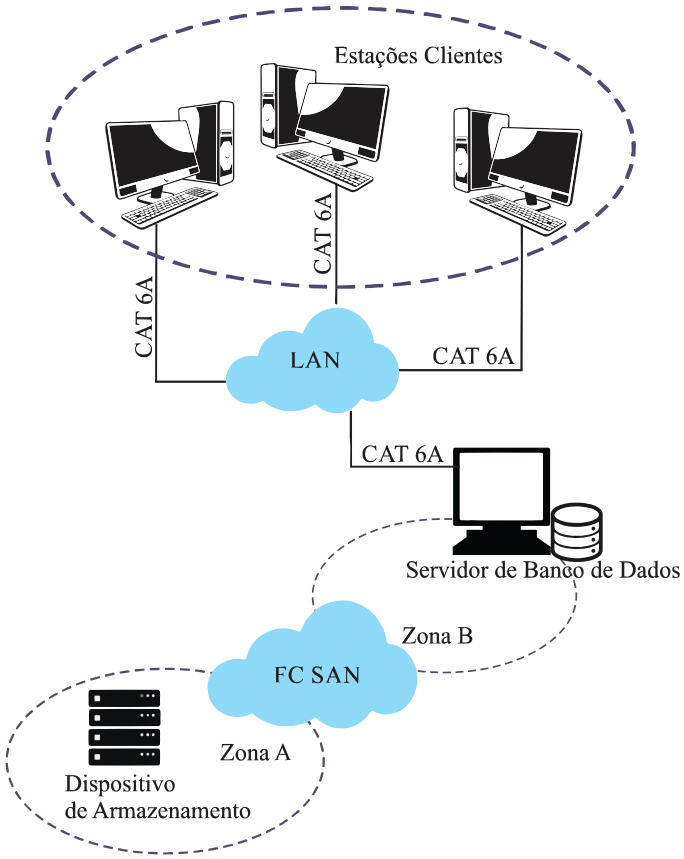


Figura 8: Rede FC SAN.

um par de chaves é utilizado para conectar servidores e dispositivos de armazenamento através de caminhos redundantes).

2.6 Deduplicação de Dados

A deduplicação [8] a deduplicação de dados utiliza algoritmos de *hash* para atribuir uma identificação (uma sequência de *bits*) aos blocos de dados de forma exclusiva. Desta forma, possibilita um melhor gerenciamento do crescimento acentuado de dados armazenados e aumenta o desempenho do tráfego de dados. Dentre as estratégias para tratar a deduplicação, tem-se a deduplicação de dados em nível de arquivos, bloco e *bytes*.

A deduplicação em nível de arquivo é mencionada como um Armazenamento de Instância Única [47], em que o índice do repositório do arquivo é examinado e os atributos dos arquivos armazenados são comparados. Se o arquivo a ser salvo for diferente de um arquivo já alocado em disco, ocorre o processo de armazenamento e atualização do índice. Caso contrário, um endereço de memória é direcionado para o arquivo já existente. Neste

nível, a deduplicação requer um menor poder de processamento, e os números de *hash* dos arquivos são gerados mais facilmente.

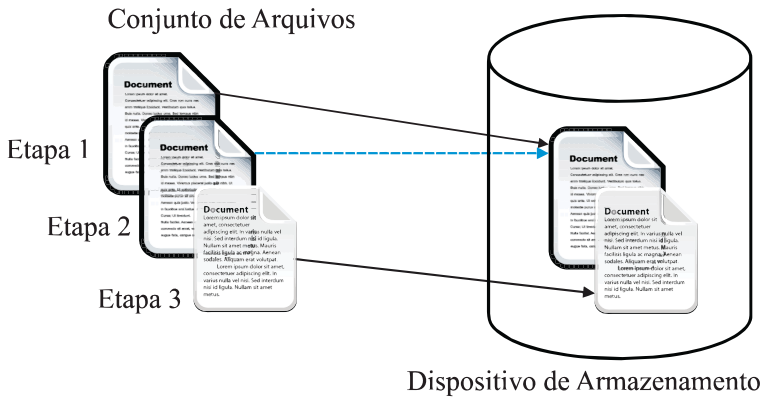


Figura 9: Processo de Deduplicação em Nível de Arquivo.

A Figura 9 exemplifica o processo de deduplicação em nível de arquivo em três etapas. Na etapa 1, tem-se o armazenamento de um documento com tamanho de 3,0 MB. Na etapa 2, ao salvar uma cópia deste mesmo documento, como se trata de uma cópia exata, o documento inteiro é deduplicado e somente uma única cópia é salva em disco, por fim tem-se um total de 3,0 MB em espaço de armazenamento.

Contudo, a etapa 3 simula que, caso um documento tenha sido modificado em apenas um *byte*, este documento será tido como novo e como resultado as duas versões do documento serão salvas, o que irá consumir mais 3,0 MB em espaço de armazenamento em disco.

De acordo com [47] e [52], a tecnologia de deduplicação de dados em nível de bloco usa um algoritmo *hash* para detectar dados redundantes dentro e através de um arquivo, e está subdividida em segmento de comprimento fixo e variável. Esta tecnologia requer um maior poder de processamento se comparada com a de arquivos, pois o número de identificadores que precisam ser processados e o tamanho do índice para acompanhar todas as iterações realizadas são maiores.

Em blocos de comprimento fixo, a checagem de blocos possui tamanhos pré-determinados (por exemplo 1 KB, 4 KB ou 256 KB) e torna possível alcançar taxas de deduplicação mais eficientes frente a redução do espaço para armazenamento se comparada com a deduplicação em nível de arquivos.

A Figura 10 apresenta o processo de eliminação de dados duplicados realizado em nível de bloco com tamanho fixo. No qual um dado é identificado na etapa 1, e dividido em blocos de tamanho fixo na etapa 2, com checagem e identificação de blocos com *hash* iguais na etapa 3, para assim realizar seu armazenamento em disco na etapa 4.

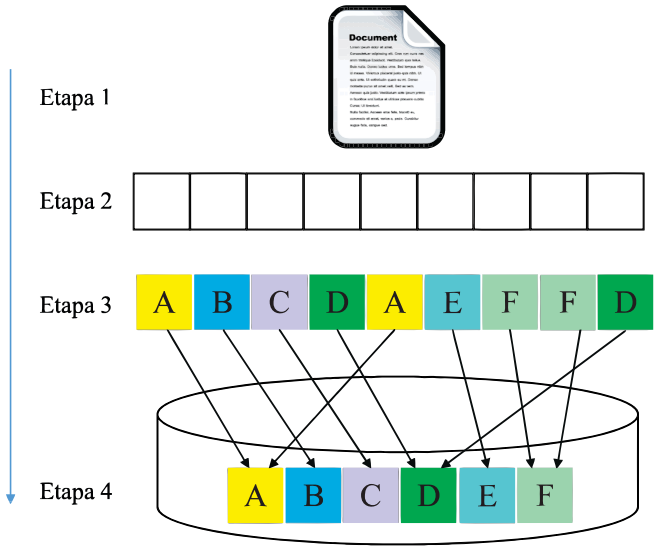


Figura 10: Processo de Deduplicação em Nível de Bloco com Tamanho Fixo.

Segundo [46], cada bloco de dado é processado e comparado com base em um algoritmo de *hash*, como MD5 ou SHA-1. No caso do uso de blocos com tamanhos mais elevados, por exemplo 1.024 MB, necessita-se de menos processamento, no entanto, às vezes, obtêm-se uma menor taxa de compressão. Com blocos de tamanho menor, por exemplo 4 KB, obtem-se uma melhor compressão, mas é demandado mais processamento.

Para o método de deduplicação com comprimento fixo para um documento com tamanho de 3,0 MB dividido em blocos de 4 KB, tem-se 75 blocos. Ao gerar uma cópia deste mesmo documento e inserir uma página ao final, haverá somente uma parte deste documento que será modificada. Esta análise determina se há a existência de blocos iguais por meio de um identificador único, um resumo *hash*.

Se o bloco a ser inserido não existir em disco, o mesmo será salvo junto com o seu identificador em um índice. Caso contrário, é inserido um endereço de memória para a localização inicial do mesmo bloco, em seguida são adicionados os blocos da página inserida. Contudo, quanto maior for o documento a ser deduplicado, maior será o tempo necessário para efetuar checagem de bloco.

De acordo com [47], para a deduplicação em nível de bloco com tamanho variável, há como parâmetro de verificação o início e término de cada bloco cujo reconhecimento de padrões repetitivos (seja uma inserção ou remoção de um arquivo) acarreta um realinhamento correlacionado com dados já armazenados. Caso exista um padrão correspondente ao dado inserido, é possível obter uma maior eficiência em redução de espaço de armazenamento, mais do que a deduplicação em nível de bloco com tamanho fixo e arquivo de instância única.

De acordo com [17], a deduplicação em nível de *byte* realiza a checagem de dados *byte*

a *byte*. Entretanto este nível requer um maior tempo e poder de processamento para realizar a comparação por *bytes*, que é tida como inviável, pois requer um alto número de operações de entrada e saída por segundo.

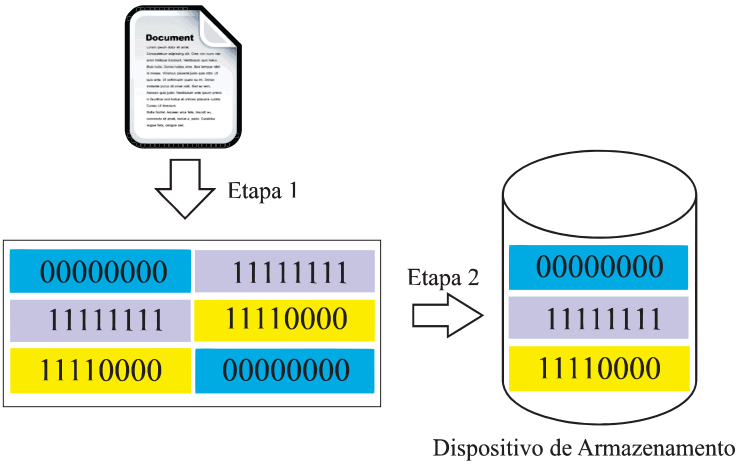


Figura 11: Processo de Deduplicação em Nível de *Bytes*.

A Figura 11 ilustra o processo de deduplicação no decorrer do envio de um documento para um dispositivo de armazenamento que é divido em *bytes* (etapa 1). Logo a seguir, realiza-se a checagem entre todos os *bytes* que representam o documento com os *bytes* já existentes no dispositivo de armazenamento. Por fim, na etapa 2, caso exista um *byte* já armazenado é adicionado um endereço de memória de ligação ao respectivo *byte*, caso contrário o novo *byte* é salvo.

Segundo [52], a deduplicação pode ser efetuada em dois modos: deduplicação em linha (síncrona) e pós-processamento (assíncrona). Na primeira, o procedimento de deduplicação dos dados ocorre antes que os dados sejam salvos no dispositivo de armazenamento remoto. Logo, pode-se dizer que os dados são checados ao serem recebidos. Alguns exemplos de aplicações que utilizam este modo de deduplicação são o *Opendedup* [45] e o S3QL [49].

Para deduplicação pós-processamento, os dados a serem salvos são deduplicados somente após o dispositivo de armazenamento remoto receber todos os dados enviados [30]. De forma resumida, este método de deduplicação requer uma capacidade de espaço de armazenamento em disco adicional, uma vez que é necessário receber os dados enviados e somente após é iniciada a deduplicação.

2.7 Mecanismo de Integridade

De acordo com [16], obtêm-se a integridade uma vez que: *"uma mensagem recebida é idêntica àquela que foi enviada"*. Para a comunicação (Figura 12) entre o servidor *web* e o

CAS, a integridade é realizada ao verificar que os dados retornados para um determinado documento são exatamente os mesmos dados que foram armazenados ao utilizar o seu nome ou UUID (*Universally Unique Identifier*) para identificação de um documento a ser salvo.

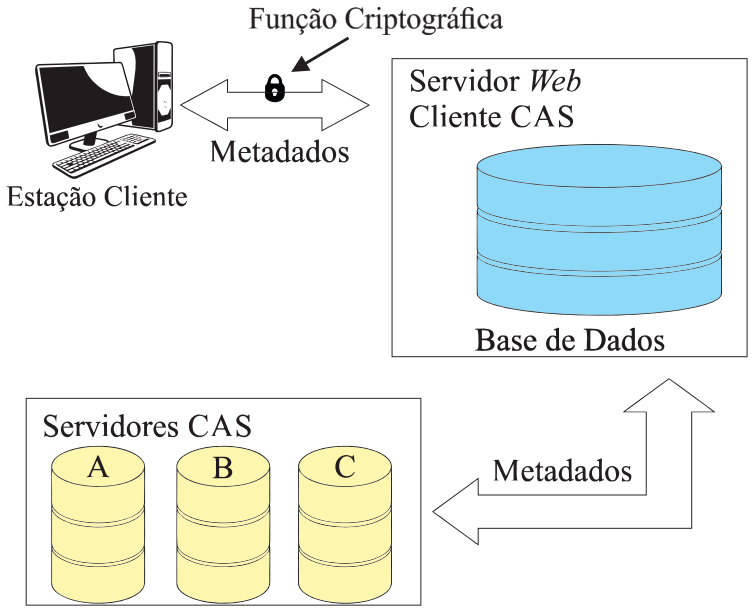


Figura 12: Conexão com um Algoritmo de Segurança.

A Figura 12 mostra o processo de comunicação entre a estação de um usuário com o servidor *web* através de um canal de comunicação que utiliza uma função criptográfica para proteger os dados e metadados transferidos a um banco de dados, que é o meio de comunicação com os múltiplos servidores de armazenamento de conteúdo endereçável. O banco de dados recebe os metadados referentes ao objeto a ser armazenado nos servidores CAS, e o objeto é identificado por um nome ou UUID.

A constatação da integridade de uma mensagem é efetuada através de uma comunicação fim-a-fim com base no *content*-MD5 (nome do cabeçalho). Por meio do *content*-MD5, o conteúdo do corpo do objeto é computado com encriptação de 128 *bits* através de um método para codificação de dados (*base*64) que é voltado a transferência de dados por uma rede.

De acordo com [7], a checagem de integridade pode ser efetuada das seguintes formas: na primeira situação, gera-se um *hash* localmente no servidor *web* do documento a ser transferido antes da transferência para o servidor CAS. Ao final da transferência, gera-se novamente o *hash* deste documento agora pelo servidor CAS e em seguida os valores *hash* de ambos são verificados para examinar se o documento foi modificado ao ser enviado.

Na segunda situação, o documento é enviado e somente ao ser armazenado no CAS é que se obtém o *hash* do documento armazenado. Neste cenário é possível que ocorra uma

captura do documento e possível modificação visto que somente obtém-se o *hash* ao final de todo o processo.

Para a estratégia de armazenamento, é possível utilizar funções criptográficas MD5 e SHA256 para demarcar de forma única os dados, o que possibilita identificar a existência de dados redundantes. Depois de criar o objeto e atribuir um nome ou UUID, o CAS apresenta um cabeçalho local com uma URL (*Uniform Resource Locator*) que pode ser utilizada posteriormente para recuperar os dados armazenados.

Além do nome ou UUID de identificação do objeto, a URL inclui os parâmetros de tipo de *hash* e o valor do *hash* computado a partir do objeto de conteúdo cuja associação é tida como um "selo" de integridade do conteúdo do objeto [7]. Um exemplo desta atribuição em um cabeçalho para localizar um objeto não identificado é:

Endereço IP do Servidor: 172.31.0.90
Porta de Comunicação: 90
UUID: 32A140B5541DC8D22JJ8D027016A0591?
Tipo de *Hash*: sha256
Hash: 6D25E6067904EAC8502498DF1BE31043

Location: http://172.31.0.90:90/32A140B5541DC8D22JJ8D027016A0591?
hashtype=sha256&hash=6D25E6067904EAC8502498DF1BE31043

2.8 Resumo Conclusivo

Este capítulo teve como objetivo apresentar conceitos básicos correlacionados à implantação de uma estrutura de armazenamento de documentos por objetos para o TCE-TO. Foram descritos e ilustrados aspectos conceituais sobre a arquitetura de sistemas em nuvem, modelos de implantação e características essenciais, assim como os principais tipos de armazenamento (bloco e objeto), os principais sistemas para armazenamentos (DAS, NAS e SAN) e as diferentes técnicas de deduplicação de dados.

No Capítulo 3, são apresentados trabalhos relacionados a esta dissertação que apresentam problemas sobre como garantir o armazenamento de dados com integridade; qual o tipo de armazenamento é viável para o armazenamento de grandes massas de dados não estruturadas e qual a arquitetura a ser adotada que proporcione escalabilidade e performance à infraestrutura de tecnologia da informação para essa dissertação.

Capítulo 3

Revisão da Literatura

O tema armazenamento de conteúdo endereçável tem sido alvo de trabalhos que tratam sobre a obtenção de melhor performance em tráfego de dados e preservação de dados digitais em dispositivos de armazenamento. O trabalho de [20] apresenta um *framework* direcionado ao armazenamento de objetos com intuito de manter os dados digitais íntegros por longos períodos de tempos e tem como fundamento o uso de um sistema de armazenamento de conteúdo endereçável escalável e eficiente.

Dentre os objetivos apresentados no trabalho [20], destaca-se o armazenamento permanente de grandes volumes de dados de forma imutável, no qual um objeto uma vez armazenado não será alterado, sendo identificado posteriormente pelo seu CA. Destaca-se também a confiabilidade para preservar os dados em dispositivos de armazenamento por objetos com acessibilidade e checagem de integridade dos dados durante o arquivamento, localização e recuperação.

No artigo de [24], é apresentada a implementação de um protótipo de rede de área de armazenamento de conteúdo endereçável direcionado a conexões iSCSI e compatível com padrão OSD. Seus resultados demonstram que a performance do seu modelo é superior a de sistemas de armazenamento por bloco.

Em ambos os trabalhos, a arquitetura para armazenamento por objetos utiliza o protocolo HTTP como meio de comunicação com o servidor CAS. A discussão anterior focalizou que é possível obter um melhor desempenho na realização de transações de leitura e escrita se comparado a uma estrutura de armazenamento por bloco.

Como um dos objetivos deste trabalho é garantir o armazenamento de documentos com fidedignidade e autenticidade, levando em consideração [20], será adotado neste trabalho uma estrutura de armazenamento de conteúdo endereçável. No estudo de [24] é possível obter um melhor desempenho na realização de transações de leitura e escrita através de uma arquitetura para armazenamento por objetos.

Ao utilizar uma estrutura de armazenamento por objetos, para cada objeto a ser salvo é computado uma função de *hashing*, por exemplo o MD5. Através deste processo é gerado um identificador único. Se um arquivo é atualizado, então se tem um novo objeto a ser salvo e gera-se um novo número de identificação para o respectivo objeto. Esta metodologia elimina e impede que objetos duplicados sejam armazenados novamente.

Os autores [17] mostram que as funções de *hashing*, como MD5 e SHA-1, são recomendadas para o uso em sistemas que necessitam identificar um arquivo, bloco ou *byte* de forma exclusiva e assim evitar o armazenamento de dados idênticos. Também é realizado uma análise comparativa entre essas duas funções de *hash*, cujos resultados demonstram que o MD5 proporciona uma melhor velocidade de cálculo para gerar o endereço de conteúdo de cada entrada de bloco de dados e assim detectar possíveis duplicações no sistema em comparação ao SHA-1. Fundamentados neste contexto, é apresentado pelos autores um sistema de armazenamento com deduplicação distribuída projetado para proporcionar ganhos com vazão e escalabilidade.

Igualmente aos estudos apresentados por [20] e [17], o enfoque desta dissertação está direcionado à preservação digital frente sua gestão sobre os princípios da autenticidade e integridade em uma estrutura de armazenamento de conteúdo endereçável na qual será aplicada uma função *hashing* através de um algoritmo MD5 para identificar similaridade e eliminar a redundância ao armazenar objetos.

Em [27], é proposto o uso do armazenamento por objeto como arquitetura de armazenamento em nuvem devido a fatores como o aumento da velocidade de conexão com a *Internet*, e a existência de características de acesso diferenciadas por usuário. Os autores discorrem sobre um modelo de replicação de dados dinâmico com base no tempo de vida de armazenamento de um objeto.

O uso de uma arquitetura de armazenamento em nuvem para o procedimento de arquivamento de documentos é recomendado [27], assim como o uso de mecanismos de replicação de documentos entre outros servidores CAS, para minimizar o tempo de indisponibilidade do serviço, a necessidade de *backups* diários e melhorar a escalabilidade em um ambiente com armazenamento de dados distribuídos.

Contudo, torna-se necessário avaliar se a estrutura de armazenamento de conteúdo endereçável proposta propicia reais ganhos ao ser comparada com a arquitetura de armazenamento por conexão direta. No artigo de [18], os autores elaboram uma ferramenta para avaliar o desempenho de serviços direcionados ao armazenamento de objetos motivados por propostas de empresas e de pesquisadores sobre o respectivo tema em ascensão. Foram tomadas para análise, a média obtida para o tempo de resposta (duração entre o início e conclusão operação), vazão (número total de transações realizadas por segundo) e a largura de banda (quantidade total de dados transferidos por segundo),

das transações de leitura em objetos com tamanho de 64 KB em 128 diferentes recipientes ("diretórios" de armazenamento) no tempo de 300 segundos, em servidores CAS.

No trabalho proposto por [43], os três principais parâmetros para as organizações que querem tirar proveito de armazenamento em nuvem são: desempenho, disponibilidade e a escalabilidade. Neste trabalho é relatado que uma arquitetura voltada ao armazenamento em nuvem deve oferecer capacidade de armazenamento "ilimitada", bem como estabelecer as bases para a eliminação da rotina diária de *backup* de dados, por meio da replicação de dados em nuvem. As medições realizadas foram direcionadas a analisar a estrutura de armazenamento em nuvem. Por este motivo, foi mensurado o tempo de resposta e a largura de banda com base nas variáveis: concorrência (múltiplas conexões simultâneas), tamanho do arquivo (desde arquivos muito pequenos a muito grandes) e o tipo de carga de trabalho (leitura, escrita e misto) de estruturas pertencentes a empresas distintas.

A metodologia utilizada por [19] descreve a implementação da ferramenta COSBench para comparar o desempenho e otimização de sistemas de armazenamento de objeto em nuvem, com foco na extensibilidade e escalabilidade. Os autores afirmam que *"o primeiro aspecto que as pessoas precisam se concentrar é sem dúvida o desempenho"*. Assim como [43] e [18] descrevem que o parâmetro de desempenho é fundamental para testes direcionados a avaliar uma estrutura de armazenamento de conteúdo endereçável.

Nesse contexto, a análise de desempenho deve mostrar a eficácia de um sistema que é expressa pelo tempo de resposta de transferência e a previsibilidade de um sistema, que está associada com a distribuição de seus tempos de resposta.

3.1 Resumo Conclusivo

O estado da arte apresentado está relacionado ao sistema de armazenamento endereçável, que conforme foi exposto nos trabalhos que tratam do problema da preservação de dados digitais em dispositivos de armazenamento, é possível armazenar grandes volumes de dados não-estruturados de forma imutável e com confiabilidade. Partindo desse pressuposto, percebe-se que a estrutura de armazenamento por objetos é a mais indicada para o arquivamento de documentos. Além disso, alguns trabalhos ressaltam que com este tipo de estrutura de armazenamento tem-se benefícios ao utilizar funções de *hashing* para identificar arquivos de maneira única e assim evitar o armazenamento de dados duplicados. Para isso, com base no conteúdo discutido, o algoritmo *hash* mais recomendado é o MD5 para gerar um endereço de conteúdo único para cada documento salvo.

A discussão anterior foca no uso de uma arquitetura de armazenamento por objetos em nuvem para propiciar ganhos tanto em desempenho quanto em escalabilidade. Levando-se

isso em consideração, foi observado a necessidade de avaliar o serviço oferecido e para tal procedimento foi encontrada na literatura uma proposta que apresenta uma metodologia para mensurar um serviço em nuvem. Considera-se interessante ressaltar assim como [18], que não foi encontrada na literatura outra metodologia de avaliação deste tipo de serviço.

No contexto geral, os estudos apresentados fundamentam que a integração desses recursos tecnológicos propicia uma arquitetura de armazenamento com um melhor nível de desempenho, disponibilidade e escalabilidade.

Capítulo 4

Proposta de Infraestrutura da Tecnologia da Informação

Este capítulo detalha a solução proposta para a implantação de uma infraestrutura direcionada ao armazenamento de conteúdo endereçável, com integridade e disponibilidade das informações por meio de uma nuvem privada.

O restante deste capítulo está estruturado da seguinte forma. A seção 4.1 expõe o cenário inicial antes do desenvolvimento deste projeto de reestruturação frente ao armazenamento de dados por bloco. Na seção 4.2 é apresentada a evolução documental dentro do TCE-TO. Logo a seguir, a seção 4.3 descreve a definição da estrutura por objetos. Por fim, a seção 4.4 apresenta a metodologia adotada para o ambiente de avaliação.

4.1 Infraestrutura de Armazenamento por Conexão Direta

O Tribunal de Contas do Estado do Tocantins dispõe de um total de 56 sistemas (internos/externos) que estão integrados em um único dispositivo de armazenamento de dados. Em conformidade com o que foi exposto no Capítulo 1 desta dissertação, existe a necessidade da implantação de uma arquitetura voltada ao armazenamento de documentos digitais salvos dentro deste órgão, bem como a melhoria no nível de segurança lógica e física dos dispositivos.

A Tabela 1 descreve o equipamento de *hardware* utilizado como servidor de arquivos: um Itautec LX201 com *Windows Server* 2008 R2 *Enterprise Edition*, conectado a um *switch* HP 5500-24G EI. Além disso, no mesmo *switch* há um servidor de banco de dados com SQL *Server* 2008 R2 *Enterprise Edition*, um servidor *web* com *Apache* [4]

Tabela 1: Recursos Tecnológicos de *Hardware* e *Software* existentes no TCE-TO até Dezembro de 2013.

Equipamento	**Descrição**	
	Hardwares	***Softwares***
Servidor	Itautec LX201 - Processador *Intel Xeon* série 5000, 3.2 GHz de 64 *bits Dual Core*, memória RAM 4 GB	• *Windows Server 2008* R2 *Enterprise Edition* • *SQL Server* 2008 R2 *Enterprise Edition*
Servidor	Itautec LX201 - Processador *Intel Xeon* série 5000, 3.2 GHz de 64 *bits Dual Core*, memória RAM 4 GB	• *Apache* 2.2.22 • *Debian* 7.2
Fita de *Backup*	*Tandberg Data StorageLibrary* T24	
Switch	*Switch* HP 5500-24G EI (JD377A)	

em execução sobre um sistema operacional *Debian* GNU/*Linux* 7.2 para o processamento de dados e execução de aplicativos pela *Intranet* e *Internet*. Há também um servidor de *backup* (*Tandberg Data StorageLibrary* T24) com armazenamento em fita.

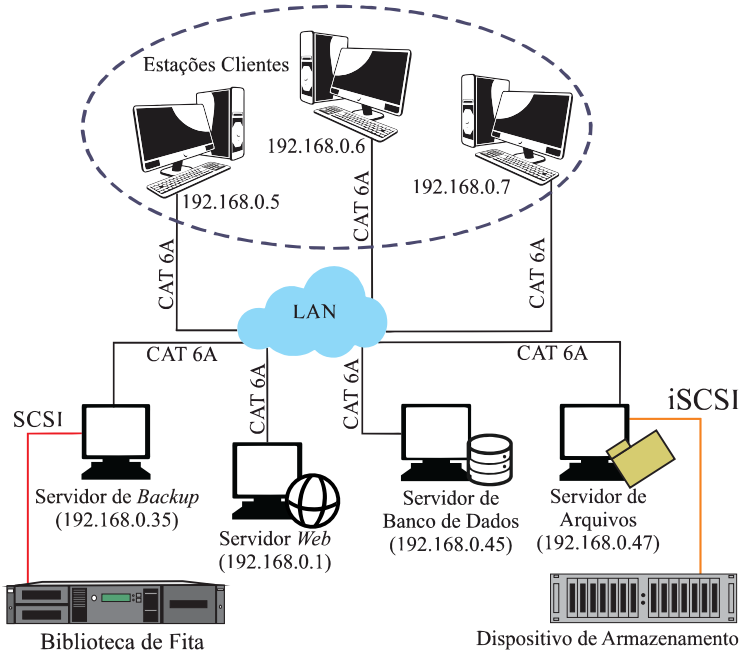


Figura 13: Topologia do Parque de Equipamentos do TCE-TO existente até Dezembro de 2013.

A Figura 13 ilustra a topologia do parque de equipamentos que dispõe de 1 servidor de arquivos composto por 12 discos rígidos de 1 *TeraByte* cada, através de uma conexão DAS com comunicação iSCSI. Tem-se 1 biblioteca de fita interligada em um servidor de *backup* por meio de uma conexão SCSI, 1 servidor *web* responsável por disponibilizar os sistemas e 1 servidor de banco de dados. Todas as demais estações clientes estão alocadas na

mesma rede de área local dos respectivos servidores, sobre uma estrutura de cabeamento *Ethernet* CAT 6A.

Os documentos armazenados no servidor de arquivos são enviados via mapeamento em rede, com velocidade de tráfego em rede *Gigabit*, por meio do protocolo NFS, e conectado a um servidor *web* e a um banco de dados. A arquitetura DAS adotada nesta infraestrutura, propicia baixo nível de segurança, tanto em nível físico como lógico dos documentos que são alocados pelo sistema atual.

Em nível físico existem implicações como baixa performance. Por exemplo, para a transferência de múltiplos documentos com tamanhos em *Megabytes*, com múltiplos usuários concorrentes em um mesmo intervalo de tempo, ocorre uma degradação no desempenho do servidor de arquivos. Para segurança em nível físico, caso um disco seja roubado é possível ter acesso às informações, que em alguns casos são de cunho sigiloso; limitação em conformidade ao número de dispositivos conectados e taxa de transferência de dados. Em nível lógico, não há controle sobre os dados armazenados que podem ser acessados e modificados.

Em conformidade ao que foi descrito nesta seção e exemplificado na Figura 13, o parque tecnológico existente no TCE-TO não proporciona uma estrutura de armazenamento que disponha de recursos que realizem a checagem da integridade dos dados armazenados tanto em nível físico quanto lógico, ou de recursos que ofereçam fidedignidade e autenticidade aos documentos digitais.

Também existem desafios sobre como expandir esta infraestrutura de armazenamento enquanto a demanda de documentos salvos aumenta (conforme é descrito na seção 4.2), uma vez que o limite suportado de discos nesta arquitetura tenha sido alcançado, não será possível adicionar mais discos de armazenamento.

4.2 Histórico Documental

No decorrer dos últimos 5 anos, o quantitativo de documentos armazenados aumentou drasticamente. A Figura 14 ilustra o volume anualmente salvo de documentos e a quantidade de documentos armazenados no período de 2013, que é muito superior aos anos anteriores. Neste período foram armazenados 157.377 documentos somente no sistema e-Contas, responsável pelo gerenciamento de processos eletrônicos do Tribunal de Contas do Estado do Tocantins desenvolvido em PHP para um quadro de 438 funcionários em exercício dentro desta corte de contas.

Este aumento ocorreu devido à obrigatoriedade da tramitação eletrônica dentro do órgão. O tamanho médio dos documentos armazenados também cresceu. Uma vez que

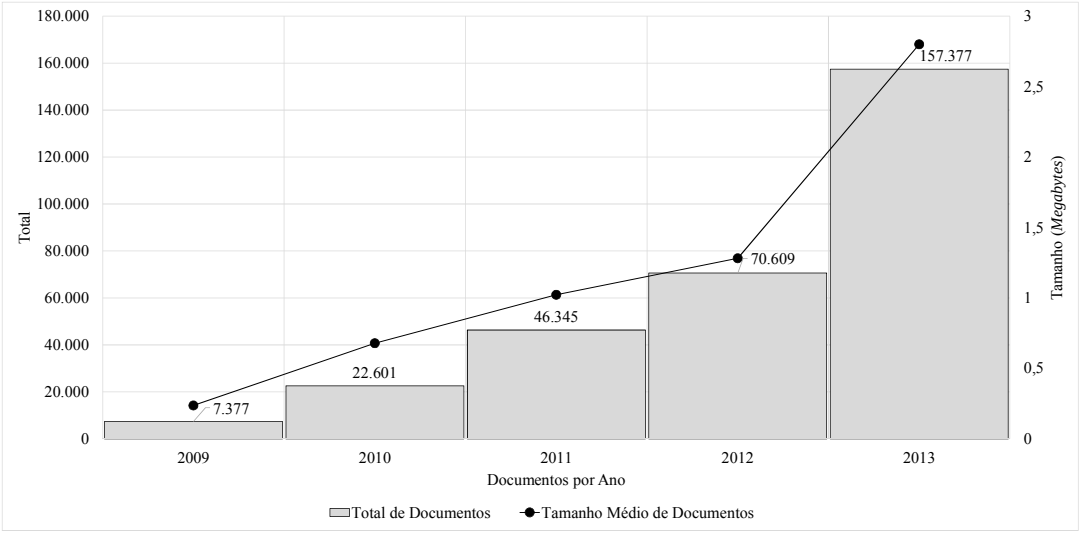


Figura 14: Evolução do Total de Documentos Armazenados entre os Períodos de 2009 a 2013 pelo Sistema e-Contas.

não existe um tamanho e formato padrão com características exclusivas para o envio de documentos, existem arquivos de diversos tamanhos armazenados.

A Figura 15 ilustra a distribuição total de documentos armazenados no período de 2013 pelo tamanho em Megabytes de cada documento. Assim, o maior documento armazenado foi de 350,96 MB, e o menor possui tamanho de 282 KB.

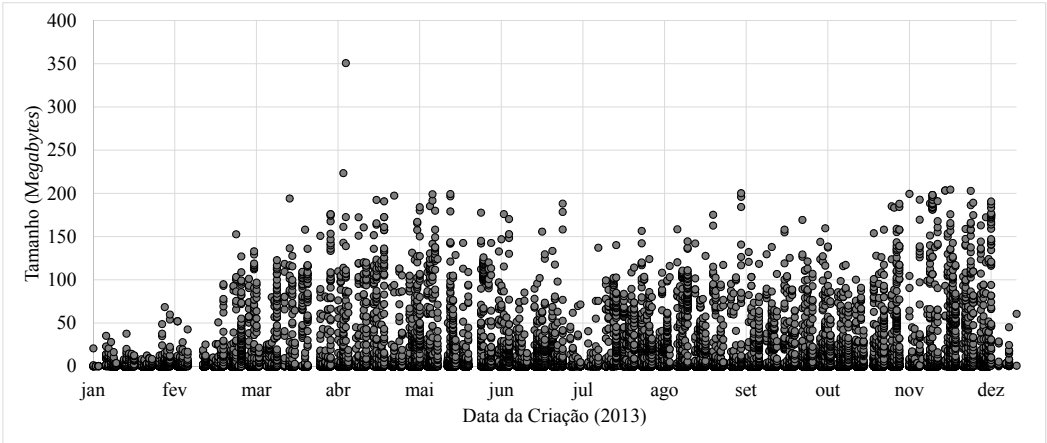


Figura 15: Distribuição dos Documentos por Tamanho no Ano de 2013.

A Figura 16 ilustra, através de três histogramas, a frequência relativa e acumulada dos documentos no ano de 2013. Logo, tem-se em (A) seis intervalos de tamanhos de arquivos. Contudo, pode-se observar que o maior percentual de dados está contido no primeiro intervalo. Por este motivo o intervalo de arquivos com tamanho entre 282 KB a 50 MB foi dividido em outro intervalo de classes, como mostra (B).

Ao gerar este novo histograma em (B), foi possível constatar que a maior concentração de documentos está distribuída nos 3 primeiros intervalos. Ao dividir novamente os 3 primeiros intervalos, foi possível concluir através do gráfico em (C), que a maior percentagem de dados acumulados está reunida entre os arquivos com tamanhos de 282 KB a 1,282 MB, e que 70% dos arquivos possuem tamanhos inferiores a 10 MB.

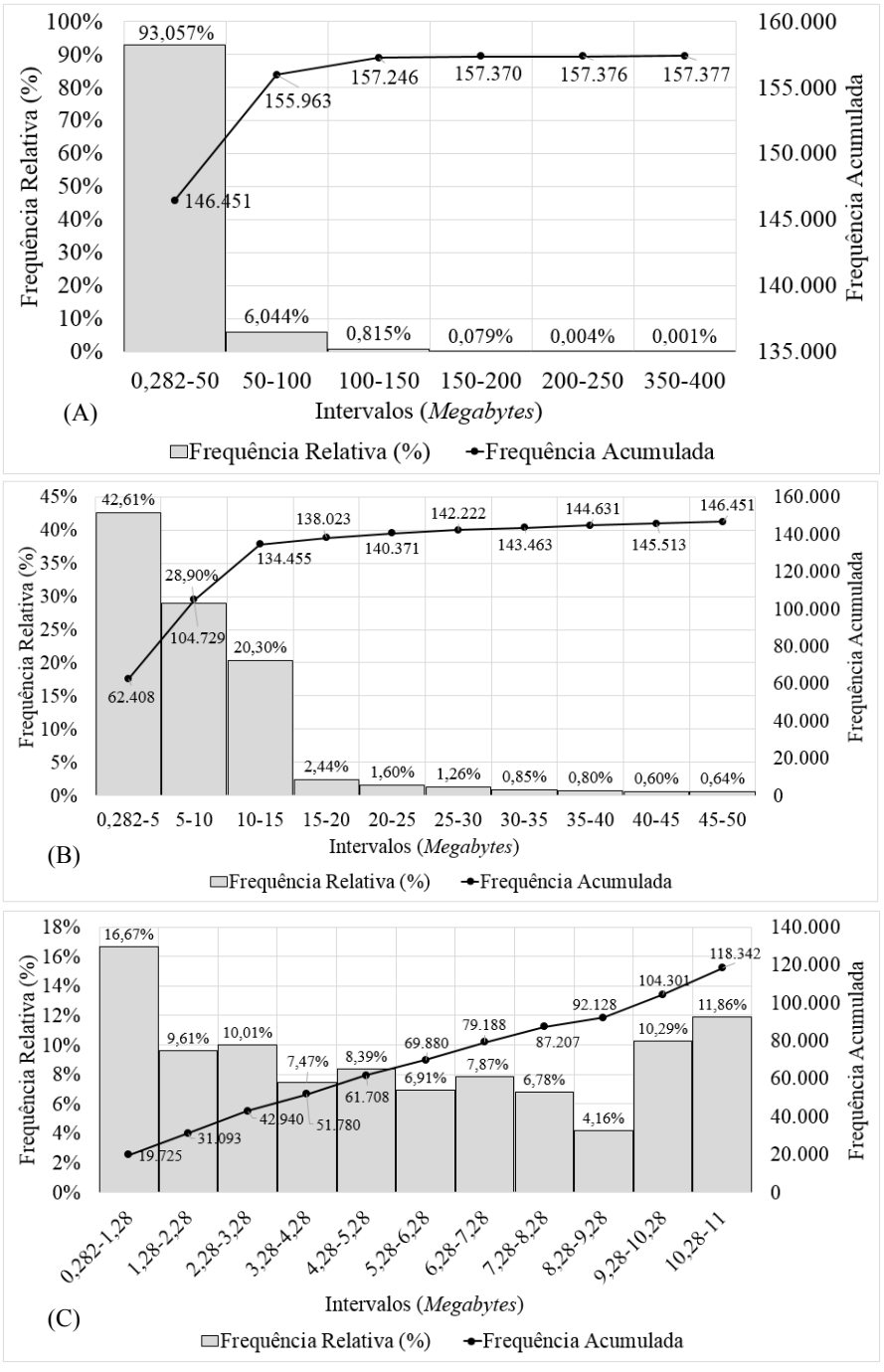


Figura 16: Histogramas de Distribuição de Frequência do Tamanho dos Documentos entre o Período de Janeiro à Dezembro de 2013.

Levando-se isso em consideração (Figura 16), tem-se na arquitetura atual problemas tanto no desempenho que sofre impacto no tempo de resposta ao efetuar múltiplas transações seja de leitura/escrita em diferentes arquivos com múltiplas conexões (número de usuários) concorrentes, quanto na disponibilidade do serviço ocasionados, por exemplo, no decorrer de paradas destinadas à manutenção (sistema operacional, antivírus, troca de disco, etc) do servidor de arquivos.

Atualizações no sistema operacional acarretam na reinicialização do servidor de arquivos, com um tempo de aproximadamente 10 minutos para inicializar seus serviços. Para uma eventual substituição de 1 disco físico com falha para esta estrutura de armazenamento, tem-se um tempo necessário de aproximadamente 8 horas de sincronização com os demais discos, o que impacta diretamente tanto na disponibilidade quanto no desempenho.

4.3 Solução Proposta

Conforme mencionado no Capítulo 1, o objetivo desta dissertação é implantar uma arquitetura para armazenamento de dados por objetos com determinadas particularidades voltadas à demanda do órgão do TCE-TO. Para isto, é proposta a migração do sistema e-Contas para esta nova arquitetura, o que tornará possível o armazenamento de documentos em uma estrutura em nuvem com maior desempenho, disponibilidade e escalabidade, bem como melhorias direcionadas à segurança física e lógica para os dados salvos.

Tabela 2: Recursos Tecnológicos de *Hardware* e *Software* utilizados na implantação da Estrutura por Objetos.

Equipamento	**Descrição**	
	Hardwares	***Softwares***
Servidor	Servidor *Dell PowerEdge* R710 - Processador: *Intel* (R) *Xeon* (R) CPU E5506 (2 processadores) 2.13GHz, Memória RAM 24 GB	• *SQL Server* 2012 *Data Center* • *Windows Server* 2012 *Data Center*
Servidor	Servidor *Dell PowerEdge* R710 - Processador: *Intel* (R) *Xeon* (R) CPU E5506 (2 processadores) 2.13GHz, Memória RAM 24 GB	• *Apache* 2.2.22 • *Debian* 7.2
Servidor	HP ProLiant DL360e Gen8 Server	• HP iLO *Management Engine*
Switch	HP *StorageWorks* 8/8 SAN (AM866A)	
Switch	HP 5500-24G EI (JD377A)	

A Tabela 2 mostra os equipamentos (*hardware* e *softwares*) recomendados para este trabalho, por estarem correlacionados com a infraestrutura (servidores e *switches*) e aos sistemas (aplicação *web*, banco de dados e sistemas operacionais) utilizados no TCE-TO. O servidor *web* a ser utilizado para disponibilizar o sistema do e-Contas terá o *Apache* em execução sobre um equipamento de *hardware Dell PowerEdge* R710 com sistema operacional *Debian* plataforma 64 *bits*.

Todo este ambiente estará conectado a uma rede SAN (HP StorageWorks 8/8 SAN *Switch*) na qual será interligado o servidor *web* com o servidor de banco de dados, que também deverá ser instalado em um *hardware Dell PowerEdge* R710 sobre um sistema operacional *Windows Server* 2012 *Data Center* e um SGBD SQL *Server* 2012 *Data Center*, ambos em plataforma 64 *bits*.

Por fim, serão utilizados 3 servidores de armazenamento de conteúdo endereçável em *hardware* da HP ProLiant DL360e Gen8 Server. Todos estes servidores CAS estarão conectados por meio de 6 *switches* HP 5500-24G EI com balanceamento de carga nas suas interfaces de conexão *GigabitEthernet*. Para cada servidor CAS, deverão ser inseridos 2 *switches* nos quais a integração entre estes equipamentos ocorrerá com respectivamente 4 interfaces de rede existentes em cada servidor.

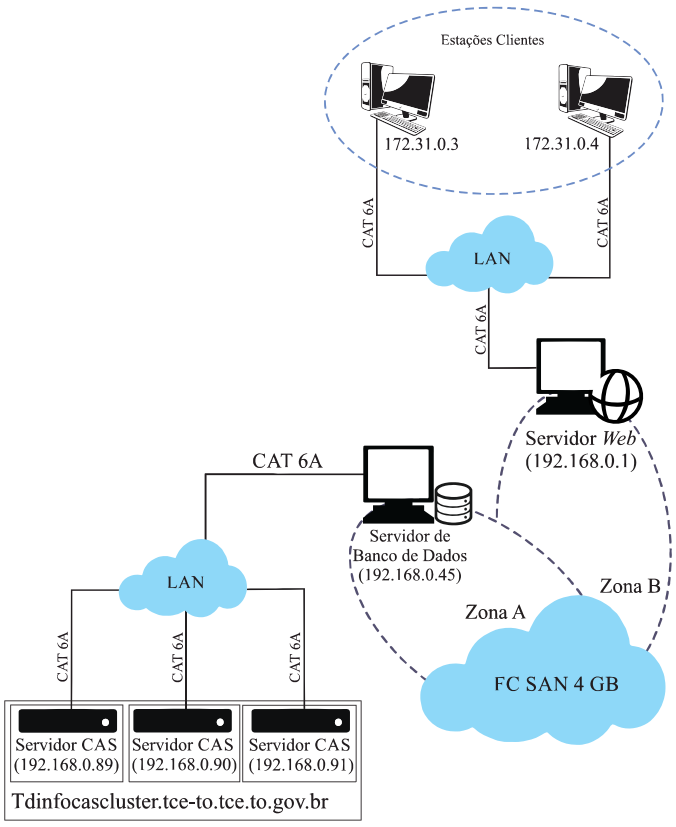


Figura 17: Topologia do Parque de Equipamentos Implantado no TCE-TO em 2014.

A Figura 17 mostra a topologia do parque de equipamentos que será implantado. Pode-se observar que o servidor de banco de dados é o responsável pela comunicação entre o servidor *web* e os 3 servidores CAS pertencentes a um *cluster* (conjunto de computadores). A comunicação entre as interfaces de rede de cada servidor CAS possui tráfego de 4 *Gigabits*, já as zonas (A e B) de acesso possibilitam ter um controle físico sobre quais os respectivos equipamentos terão acesso aos dispositivos pertencentes à rede SAN conectados através de fibras óticas.

Todas as estações clientes estão separadas por um escopo (conjunto de endereços IP em uma determinada sub-rede) de endereço IP distinto da rede de área local dos respectivos servidores, e a comunicação de dados está sobre uma estrutura de cabeamento *Ethernet* CAT 6A.

Entre os servidores *web* e de banco de dados, a comunicação será estabelecida através da técnica do zoneamento (*zonning*) definida no HP *StorageWorks* 8/8 SAN *Switch*. Por meio do uso da técnica de zoneamento, será possível determinar quais os grupos de equipamentos contidos nesta rede SAN terão comunicação entre si.

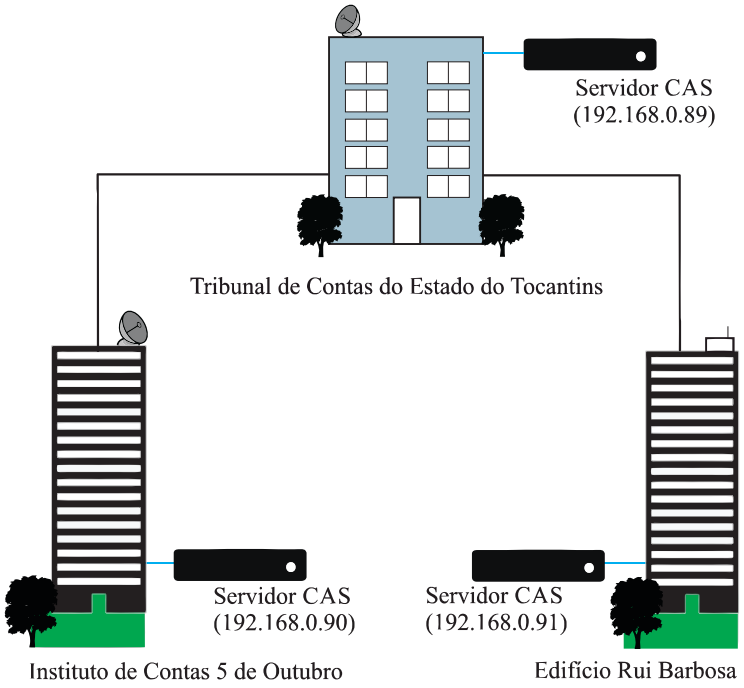


Figura 18: Replicação dos Dados Armazenados nos Servidores CAS entre os Prédios Interligados ao TCE-TO.

Pode-se observar na Figura 18 como serão interligados o prédio do Tribunal de Contas do Estado do Tocantins, o Edifício Rui Barbosa e o Instituto de Contas 5 de Outubro. Todos estes prédios estão localizados em um raio de 500 metros, o que caracteriza que o procedimento de replicação dos dados a ser implantado é tido como local.

A escolha dos prédios ocorreu devido a fatores administrativos, uma vez que todos os prédios estão sobre a administração do TCE-TO. Para comunicação entre os edifícios serão utilizados 3 *switches* com comunicação *GigabiEthernet* via fibra ótica entre todos os dispositivos de armazenamento por objetos.

4.4 Metodologia de Avaliação

Em termos gerais, a seção 4.3 apresenta uma proposta de implantação de uma infraestrutura para armazenamento de dados por objetos. Contudo, é necessário realizar a avaliação de tal solução em um ambiente de testes configurado similar ao ilustrado na Figura 19.

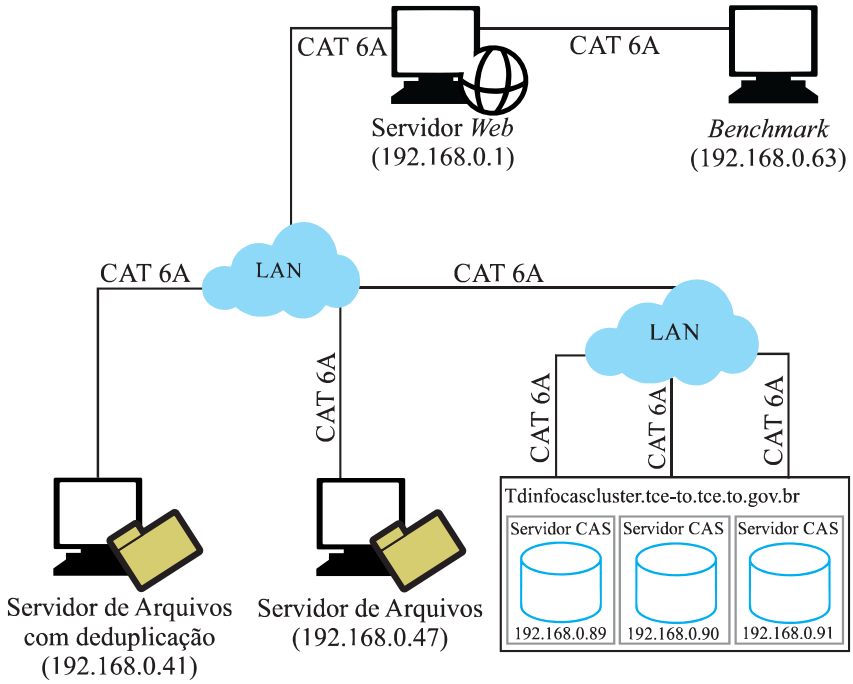


Figura 19: Ambiente de Avaliação.

A Figura 19 mostra o ambiente utilizado para avaliar as 3 estruturas de armazenamento de dados. Nesse ambiente há 1 servidor de arquivos com deduplicação em nível de bloco (com tamanho de 4 KB), no qual será utilizado o *software Opendedup*[6] para provê um volume com deduplicação em linha (síncrona), responsável pela interação entre a aplicação e o sistema operacional (*Debian* 7.4) e comunicação em rede através de 1 interface *GigabitEthernet*.

[6] *Opendedup* é um sistema de arquivos que provê deduplicação, projetado para suportar as necessidades destinadas a sistemas de armazenamento [45].

Também serão configurados 1 servidor de arquivos por bloco com sistema operacional (*Debian* 7.4) que possui 1 interface *GigabitEthernet*, e 3 (IP: 192.1680.89; IP: 192.168.0.90 e IP: 192.168.0.91) servidores CAS com 4 interfaces *GigabitEthernet* pertencentes a cada respectivo servidor, que serão agrupadas para atribuir balanceamento de carga entre as interfaces de rede e tolerância a falhas. Todos os servidores CAS serão associados em um *cluster* definido como "Tdinfocascluster.tce-to.tce.to.gov.br" e com procedimento de replicação local através da rede.

Os servidores de armazenamento de dados serão conectados sobre um mesmo escopo de endereço IP e comunicação direta com o servidor *web*, o qual está interligado ao servidor com o *benchmark* responsável pela execução dos programas de avaliação, sobre uma estrutura de cabeamento *Ethernet* CAT 6A idêntica a estrutura de cabeamento em vigor no TCE-TO.

A Tabela 3 apresenta os recursos computacionais utilizados no ambiente de testes.

Tabela 3: Equipamentos Utilizados em Testes.

Equipamento	**Descrição**	
	Hardwares	***Softwares*/Ferramentas**
Servidor	Servidor *Dell PowerEdge* R710 - Processador: *Intel*(R) *Xeon*(R) CPU E5506 (2 processadores) 2.13GHz, Memória RAM 24 GB	• *Apache* 2.2.22 • *Debian* 7.4
Servidor	Servidor *Dell PowerEdge* R710 - Processador: *Intel*(R) *Xeon*(R) CPU E5506 (2 processadores) 2.13GHz, Memória RAM 24 GB	• *Opendedup* 2.9.2
Servidor	Servidor *Dell PowerEdge* R710 - Processador: *Intel*(R) *Xeon*(R) CPU E5506 (2 processadores) 2.13GHz, Memória RAM 24 GB	• *Siege* 2.70
Servidor	Itautec LX201 - Processador *Intel Xeon* série 5000, 3.2 GHz de 64 *bits Dual Core*, memória RAM 4 GB	• *Windows Server* 2008 R2 *Enterprise Edition*
Servidor	HP *ProLiant* DL360e Gen8 *Server*	

A metodologia a ser utilizada será testada em vários casos de simulação para assim mensurar o desempenho, disponibilidade e escalabidade de todas as estruturas de armazenamento, através do *software Siege*. Este *software* é um *benchmarking* para efetuar testes de carga HTTP, para assim mensurar qual o número máximo previsto de carga (solicitações de vários recursos no máximo de tráfego em rede). As opções de configuração a serem utilizadas são [34]:

- -b : irá executar os testes sem atraso entre as requisições e os usuários (simulado);
- -c : define o número de usuários simultâneos (conexões concorrentes) no teste;
- -t : define o período de tempo de execução do teste.

A fim de verificar esta proposta de solução de armazenamento de dados, serão estudados 3 diferentes cenários de testes. Assim, serão realizados testes em:

- Cenário 1: Avaliação do Procedimento de Leitura e Escrita em um Documento com Tamanho de 282 KB.
- Cenário 2: Avaliação do Procedimento de Leitura e Escrita em um Documento com Tamanho de 10 MB.
- Cenário 3: Avaliação do Procedimento de Leitura e Escrita em Múltiplos Documentos.

De modo similar, sobre as métricas adotadas e descritas no Capítulo 3, os resultados a serem obtidos são o tempo de resposta e a largura de banda com base nas variáveis: concorrência (múltiplas conexões simultâneas), tamanho do objeto (desde objetos muito pequenos a muito grandes) e o tipo de carga de trabalho (leitura, escrita e misto) das estruturas de armazenamento estudadas nesta dissertação.

4.5 Resumo Conclusivo

Neste capítulo, foram escolhidos/abordados recursos tecnológicos para a infraestrutura da tecnologia da informação da solução proposta nessa dissertação. Conforme mencionado na seção 4.1, a arquitetura em vigor apresenta problemas relacionados à preservação documental bem como desafios a serem superados relacionados ao desempenho, disponibilidade e escalabilidade.

O servidor de arquivos da Infraestrutura de Armazenamento por Conexão Direta sofre uma degradação no seu desempenho quando há transferência de múltiplos documentos com tamanhos em *Megabytes*, com múltiplos usuários concorrentes em um mesmo intervalo de tempo. A disponibilidade sofre impacto quando há necessidade de eventuais manutenções ou falha em um dos discos físicos. Por fim, esta arquitetura não possibilita possíveis expansões devido a sua limitação de arquitetura.

Nesse caso, pode-se observar que devido à evolução documental que ocorreu nos últimos anos, torna-se necessário implantar uma estrutura de armazenamento de dados com características e particularidades para atender as demandas do TCE-TO. Levando-se

isso em consideração, foi apresentada uma solução na qual o armazenamento de dados por objetos é utilizado, assim como descrito no Capítulo 3.

Entretanto é preciso demonstrar que tal solução proposta propicia ganhos ao ser comparada com a estrutura vigente. Desta maneira, o Capítulo 5 apresentará os resultados obtidos em 3 diferentes ambientes de armazenamento de dados.

Capítulo 5

Avaliação

Neste capítulo serão avaliados o desempenho, a disponibilidade e a escalabilidade da solução proposta em relação à configuração original. As métricas utilizadas para avaliação destes parâmetros são a vazão e o tempo de resposta. Os testes realizados foram direcionados à escrita e leitura desde um único documento, até múltiplos documentos.

Ao decorrer deste capítulo, o termo transação é definido como o número de operações (leitura/escrita) concluídas com sucesso entre o *benchmark* e a respectiva estrutura de armazenamento de dados. Uma transação com erro é vista como uma operação não concluída entre o *benchmark* e um dos servidores de armazenamento analisados neste capítulo.

Uma conexão é definida como o número de usuários conectados entre o *benchmark* e a respectiva estrutura de armazenamento, ao iniciar um eventual ciclo de teste com o siege, o número estabelecido de conexões concorrentes são inicializadas ao mesmo tempo. Por fim, o limite de tráfego de dados é de 1 *Gigabit*, valor alcançado com base na velocidade de tráfego das interfaces de redes existentes no servidor *web* e no respectivo servidor de armazenamento de dados (*GigabitEthernet*).

5.1 Ambiente de Avaliação

De acordo com [13], o desempenho é medido ao analisar a sua vazão (operações por segundo), que é calculada ao somar os *bytes* recebidos (*ifInOctets*), com os *bytes* enviados (*ifOutOctets*) por um intervalo de tempo entre dois dispositivos A e B. Em notação de programação matemática o problema pode ser expresso igual ao apresentado na Equação 1.

$$Total\ de\ bytes = (ifInOctets\_B - ifInOctets\_A) + (ifOutOctets\_B - ifOutOctets\_A) \quad (1)$$

Com base na definição expressa pela Equação 1, entende-se neste capítulo que a vazão é determinada pelo número médio de *bytes* transferidos por segundo a partir do servidor *web* para cada conexão concorrente. O tempo de resposta do servidor segundo [15], é medido do momento que um usuário requisita um serviço, ao momento que é concebido. Nesse contexto, o tempo de resposta pode ser obtido por meio do RTT - *Round-Trip Time* da rede, que é expressado pela Equação 2.

$$Tempo\ de\ Resposta = (tempo\ de\ ida + tempo\ de\ volta\ de\ um\ pacote) \quad (2)$$

Conexo ao conceito apresentado na Equação 2, tem-se para este capítulo que o tempo de resposta, é o tempo obtido referente a uma solicitação e resposta de conexão. Por fim, o cálculo para a disponibilidade de uma rede ou equipamento, tem como base o tempo que a rede não se encontra operacional para o usuário, conforme apresentado na Equação 3.

$$Disponibilidade = \left(\frac{tempo\ de\ indisponibilidade * 100}{tempo\ total\ da\ operação} \right) \quad (3)$$

Desta maneira, a disponibilidade é definida como o número de transações com erros entre o servidor de arquivos com o *benchmark* incluindo o tempo total da operação dividido pela soma de todas as tentativas de conexão.

Para o teste de performance no modo de escrita/leitura com múltiplas conexões concorrentes em múltiplos documentos com tamanhos diversificados, foram criados 50 documentos para simular o cenário existente no TCE-TO. Assim, a Figura 20 (A) apresenta o tamanho dos arquivos correlacionada à distribuição dos documentos e a Figura 20 (B) ilustra através de um histograma a frequência relativa por tamanho para cada intervalo de classe.

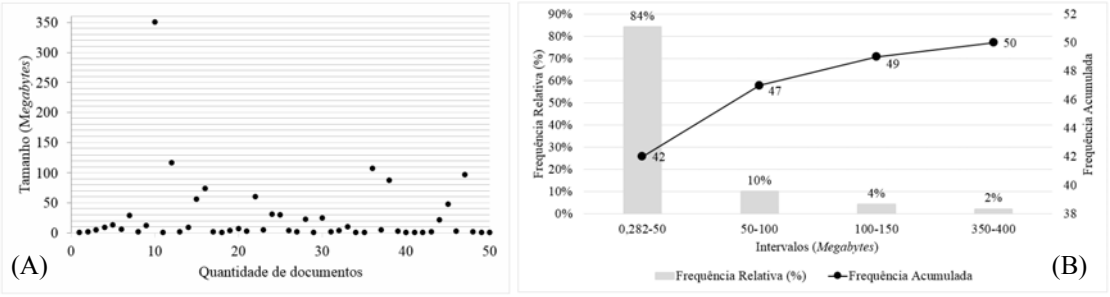


Figura 20: Distribuição dos Documentos por Tamanho.

Como o menor documento armazenado no TCE-TO possui tamanho de 282 KB e o percentual de documentos existentes neste intervalo é o mais elevado, conforme ilustrado

na Figura 16 (C), este será o tamanho representativo adotado em testes para o cenário com documentos pequenos.

Para o cenário com documentos grandes, foi utilizado o tamanho 10 MB. Este valor foi definido com base também na Figura 16 (C) referente à frequência relativa e acumulada deste tamanho de documento.

De acordo com este embasamento para distribuição de documentos, desde um documento pequeno até um grande, foi organizada e ajustada à distribuição de frequência por intervalos de classe, conforme avaliado pela Figura 16, dos documentos armazenados entre o período de janeiro a dezembro de 2013, e aqui modelado pelo conjunto de arquivos apresentado na Figura 20. O objetivo deste cenário é redistribuir os acessos para múltiplos documentos que possuem tamanhos distintos.

O formato de documento utilizado foi o PDF, por ser o padrão adotado pelo TCE-TO no sistema e-Contas para salvaguarda de documentos digitais. E os testes foram executados considerando documentos com tamanhos específicos (pequenos e grandes) ou de múltiplos tamanhos, no total 10 ciclos de execuções foram realizados. O primeiro ciclo inicia com 1 conexão, para os demais ciclos o número de conexões paralelas é duplicado para cada avaliação até 512 conexões.

Foram realizadas coletas de dados isoladas para cada teste de escrita e leitura, para assim obter a média de transações realizadas com sucesso em um nível de confiança (confiabilidade) adotado de 95% referente aos resultados obtidos em um total de 5 coletas de dados. Os resultados apresentados foram obtidos no ambiente apresentado na seção 4.4, durante o horário de serviço semanal entre 08:00 e 18:00, ao longo de um período de 60 dias, cuja duração de tempo para cada modo de operação, escrita ou leitura em cada coleta de dados foi de 60 segundos.

Uma das principais vantagens do armazenamento em nuvem é a possibilidade de aumentar a sua capacidade de atendimento. Para o teste de escalabilidade, foram analisados quantos objetos podem ser adicionados na nuvem (*cluster* de servidores de armazenamento por objetos), e se o desempenho se mantém constante à medida que o número de servidores CAS é aumentado. Também foi avaliado a disponibilidade da nuvem em diferentes taxas de operação em um determinado período de tempo.

Para a avaliação da solução de armazenamento por objetos foi utilizado somente um único recipiente ("diretório" de armazenamento) para receber todos os documentos salvos. Além disso, foi realizada uma comparação entre o armazenamento por bloco ilustrado na Figura 13, com o armazenamento por objetos ilustrado na Figura 17. Por fim, também foi considerada em testes uma estrutura de armazenamento por bloco com deduplicação em nível de bloco com tamanho de 4 KB de comprimento fixo. O objetivo desta comparação é apresentar o número de transações de leitura/escrita por segundo

suportadas e a disponibilidade do serviço, apresentada em percentagem, com base no número de requisições realizadas. Além disso, assim como discutido em [24], o objetivo é demonstrar através de resultados que a performance do modelo por objetos é comparável e em alguns casos melhor do que sistemas de armazenamento por bloco.

Todos os parâmetros adotados estão embasados em trabalhos da literatura [18], [43] e [19] já discutidos no Capítulo 3.

5.2 Avaliação do Procedimento de Leitura e Escrita em um Documento com Tamanho de 282 KB

Inicialmente foram realizados testes para avaliar a performance, a disponibilidade e a escalabidade com leitura cíclica sobre um único documento com tamanho de 282 KB, conforme apresentado na Figura 21. Nesta seção os gráficos que apresentam a média de transações no processo de leitura/escrita de um documento com tamanho de 282 KB, as colunas apresentadas no "eixo x" ilustram o número de conexões concorrentes e no "eixo y" tem-se o total de transações realizadas com sucesso (lado esquerdo) e a percentagem da disponibilidade do serviço (lado direito).

Para os gráficos que apresentam a média da vazão e tempo de resposta para leitura/escrita de um documento com tamanho de 282 KB, as colunas apresentadas no "eixo x" ilustram o número de conexões concorrentes e no "eixo y" tem-se a vazão (lado esquerdo) e tempo de resposta (lado direito).

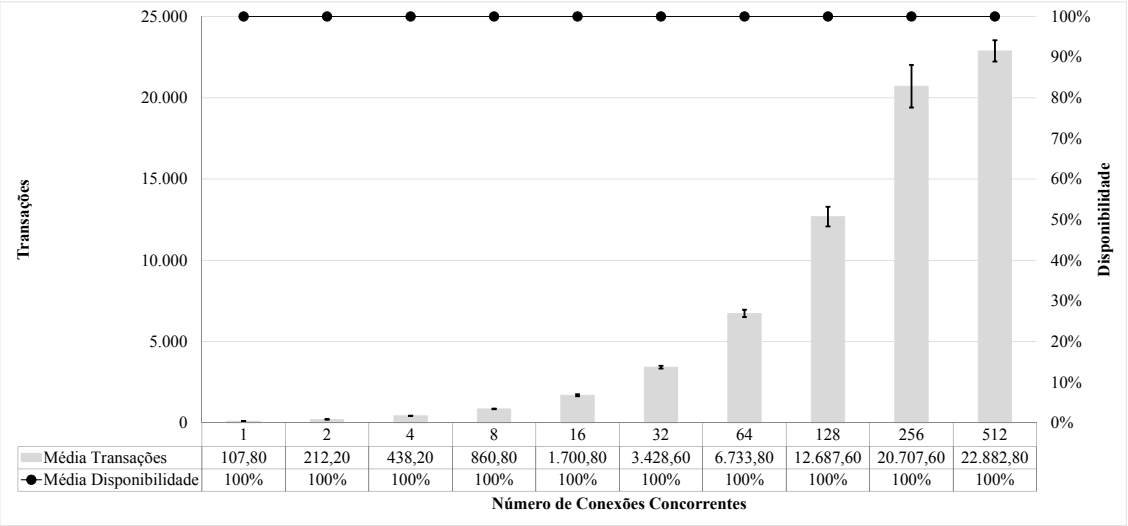


Figura 21: Média de Transações no Processo de Leitura de um Documento com Tamanho de 282 KB.

Os resultados demonstram que ao aumentar o número de conexões concorrentes, a média de transações (representada por uma barra) realizadas aumentou de forma linear e a média de disponibilidade (representada por uma linha) do serviço foi mantida em 100% das transações de leitura. Este aumento foi possível uma vez que o número de transações não ultrapassou o limite suportado de tráfego de dados pela rede referente aos documentos (282 KB) tido como pequenos neste cenário.

Com base nos resultados apresentados pode-se concluir que a estrutura para armazenamento por objetos direcionada paras requisições repetitivas sobre um único documento propicia uma boa escalabilidade para transações de leitura. De acordo com os testes realizados, ao executar 512 conexões concorrentes obteve-se uma média de transações de leitura de 22.882,80.

Pode-se observar na Figura 22 que a disponibilidade (representada por uma linha) é mantida em 100% para escrita de arquivos com até 256 conexões concorrentes. A barra representa a média de transações efetuadas, de maneira que com 512 conexões concorrentes foi possível gravar em média 4.666,80 documentos no dispositivo de armazenamento por objeto e obter uma disponibilidade no serviço de 97,29%.

Tanto o número de transações quanto a disponibilidade decaem ao atingir o limite aceitável de capacidade de tráfego de dados suportados conforme é apresentado na Figura 24 para um total de 512 conexões concorrentes.

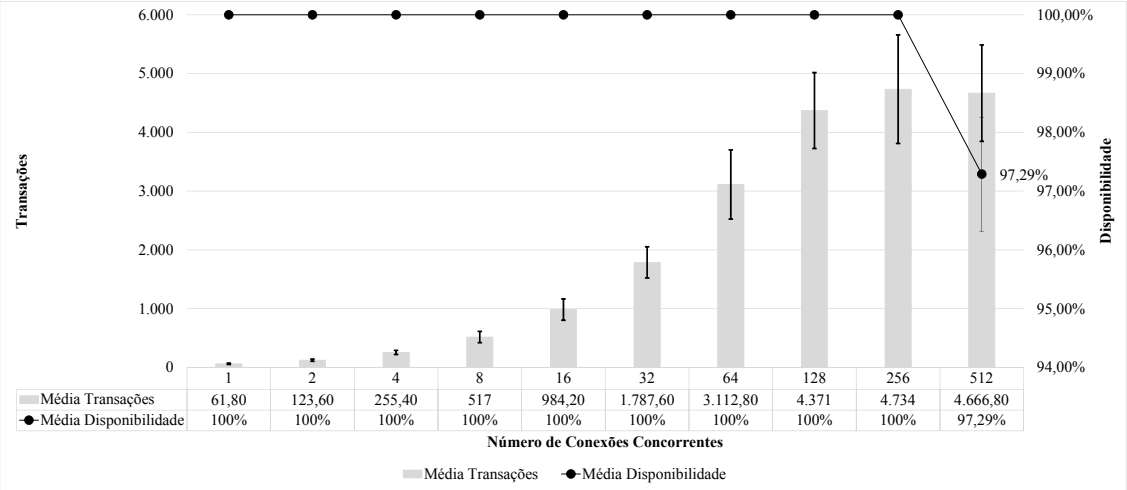


Figura 22: Média de Transações no Processo de Escrita de um Documento com Tamanho de 282 KB.

O gráfico apresentado na Figura 23 complementa o gráfico apresentado na Figura 21, pois mostra que houve um aumento na média da vazão para leitura de todos os 10 ciclos de conexões avaliados neste trabalho e média de tempo de resposta praticamente constante até 64 conexões, entre 64 a 256 conexões concorrentes a média do tempo de

resposta tende a dobrar, e com 512 conexões concorrentes o tempo de resposta aumentou de maneira quadrática.

Desta forma, recomenda-se para esta estrutura a execução de 256 conexões concorrentes para o processamento de requisições com documentos com tamanho de 282 KB, o que possibilita um tempo de resposta baixo para transações inferiores a 256 conexões concorrentes. Mais conexões neste cenário impactam em redução no número de transações com sucesso e aumento no tempo de resposta, devido ao alto tráfego de conexões concorrentes.

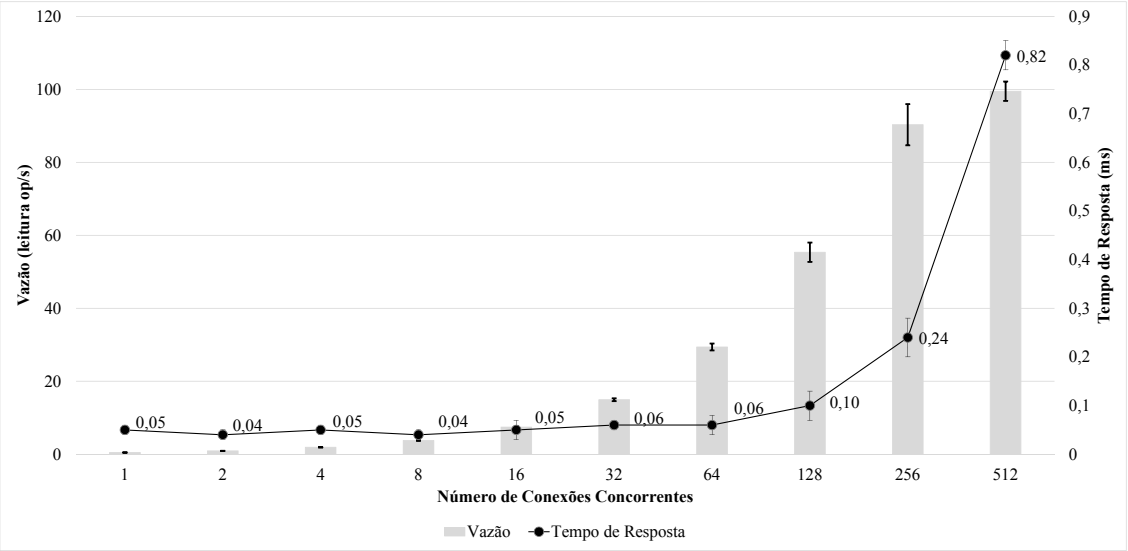


Figura 23: Média da Vazão e Tempo de Resposta para Leitura de um Documento com Tamanho de 282 KB.

Ao analisar a Figura 24, percebe-se que o aumento da vazão atinge o seu pico com 256 conexões concorrentes ao realizar 4.734 transações (Figura 22) de escrita e um crescimento linear até 64 conexões, com uma média de tempo de resposta de 2,85 ms (*Milissegundos*[7]). Do ponto de vista da capacidade de tráfego deste ambiente, mais conexões concorrentes além deste ponto apenas prolongam o tempo de resposta, contudo não aumentam a vazão, uma vez que o limite delas quanto à transferência de dados já foi alcançado.

Os resultados expostos pela Figura 25 elucidam que a estrutura de armazenamento por objetos (Média Transações CAS) dentre as demais estruturas analisadas é a que propiciou aumento no número de transações a cada novo ciclo de operação e ao mesmo tempo manteve a média da sua disponibilidade em 100% em todos os cenários. Somente foi possível obter este total de transações devido a esta estrutura de armazenamento por

[7]Unidade de medida de tempo que corresponde a um milésimo de segundo.

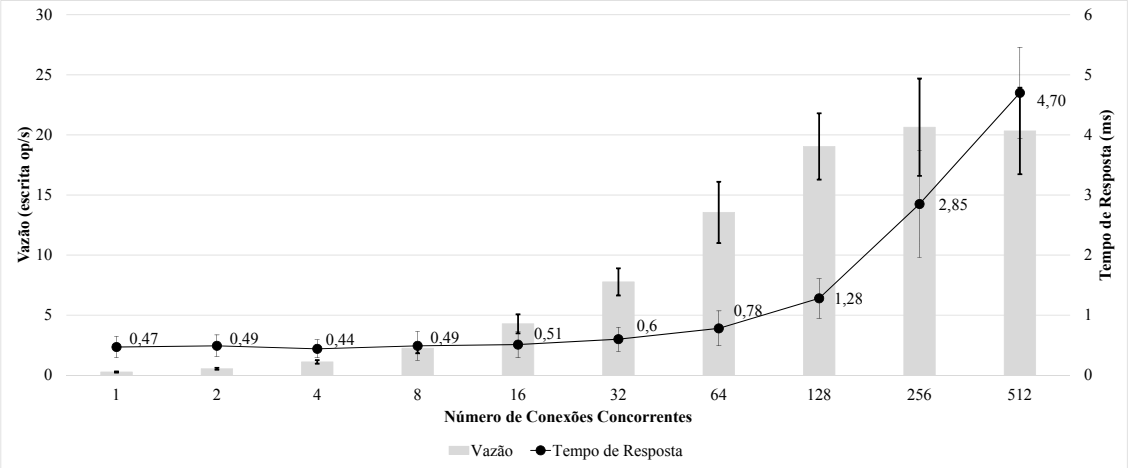


Figura 24: Média da Vazão e Tempo de Resposta para Escrita de um Documento com Tamanho de 282 KB.

objetos oferecer escalabilidade, ou seja, um crescimento horizontal referente ao número de conexões concorrentes.

O armazenamento por bloco (Média Transações por bloco) ilustrado na Figura 25, apresentou um crescimento de acordo com o número de transações executadas, cujos resultados demonstram ser a mais viável para este ambiente visto que somente com 512 conexões concorrentes é que houve uma redução para 95,42% na disponibilidade do serviço. A redução ocorre devido ao limite de tráfego de dados alcançado e processamento por parte do servidor de arquivos. Entretanto, essa solução propicia escalabilidade inferior a estrutura por objetos, bem confiabilidade reduzida conforme apresentado na seção 5.4 para um cenário com múltiplos documentos com tamanhos diversificados.

O armazenamento por bloco com deduplicação (Média Transações Deduplicação) mostrou ser viável até 64 conexões concorrentes, após este valor a disponibilidade do serviço decai ao aumentar o número de conexões concorrentes. De acordo com os testes realizados, com 512 conexões concorrentes, a média de disponibilidade de serviço reduziu para 55,88%, com isto tem-se uma taxa de transações com erro superiores as transações concluídas com sucesso, de acordo com o exposto pelo intervalo de confiança indicado sobre a barra "Média Transações Deduplicação".

É possível verificar na Figura 26 o número de transações e o percentual obtido para cada um dos 3 ambientes de armazenamento de dados avaliados neste trabalho cujos resultados demostram que o armazenamento de dados por bloco com deduplicação em nível de bloco para o procedimento de escrita de um único documento de 282 KB é o mais viável uma vez que o processo de comparação de *hashs* para um documento com este tamanho possui tempo de resposta baixo.

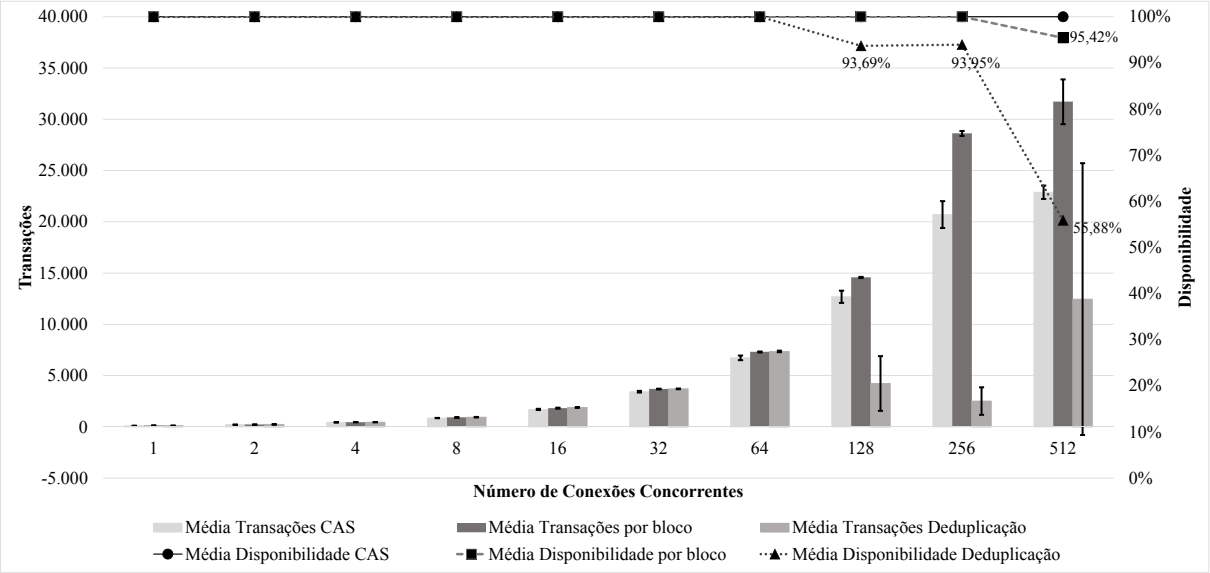


Figura 25: Média de Transações e Disponibilidade para Leitura de um Documento com Tamanho de 282 KB.

Ainda sobre a Figura 26, o ápice para a média de transações realizadas com a estrutura para armazenamento de dados por blocos com deduplicação representado pela "Média Transações Deduplicação" foi de 8.331,8 para 128 conexões concorrentes, e com uma média de disponibilidade de serviço de 100%. Neste cenário, foram gerados aproximadamente 70 blocos de dados ao utilizar deduplicação em nível de bloco com tamanho de 4 KB de comprimento fixo. Nessa abordagem, os blocos foram comparados no decorrer de sua inserção com os blocos dos *hashs* do primeiro documento já armazenado.

Como neste contexto trata-se do documento já armazenado na primeira operação realizada em um dos ciclos de testes, não há a necessidade de gravar o mesmo documento neste servidor. Por este motivo, o número de transações que foram efetuadas é muito superior às demais estruturas de armazenamento avaliadas, pois não há a necessidade de novas inserções, somente são adicionados endereços de memória do posicionamento de cada bloco.

Ao analisar o armazenamento por objeto representado no gráfico pela "Média Transações CAS" da Figura 26, com o armazenamento de dados por bloco representado no gráfico pela "Média Transações por bloco", observa-se que o número de transações realizadas tende a aumentar à medida que o número de conexões cresce. Como foi visto, a estrutura por blocos atinge o seu limiar com 64 conexões concorrentes e com 256 conexões concorrentes tem-se uma diferença de 35% de transações reduzidas em comparação com a Média de Transações CAS.

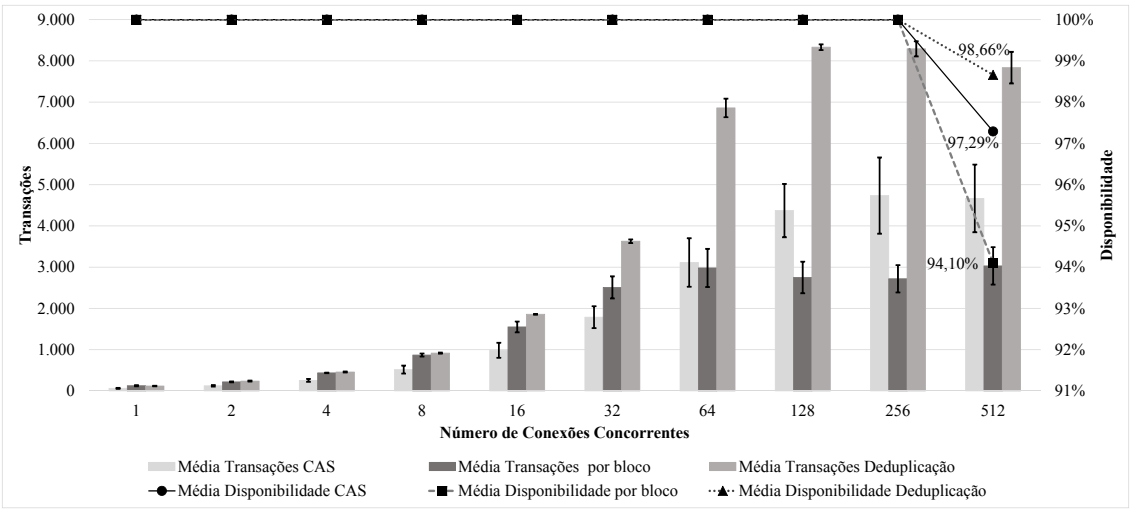


Figura 26: Média de Transações e Disponibilidade para Escrita de um Documento com Tamanho de 282 KB.

Levando isso em consideração, pode-se observar que a média de transações obtidas para 256 conexões concorrentes foi de 4.734, e média de disponibilidade de serviço em 100%. Equivalente ao ilustrado na Figura 24, mais conexões além deste ponto, não acarretaram em aumento na vazão de dados trafegados.

5.3 Avaliação do Procedimento de Leitura e Escrita em um Documento com Tamanho de 10 MB

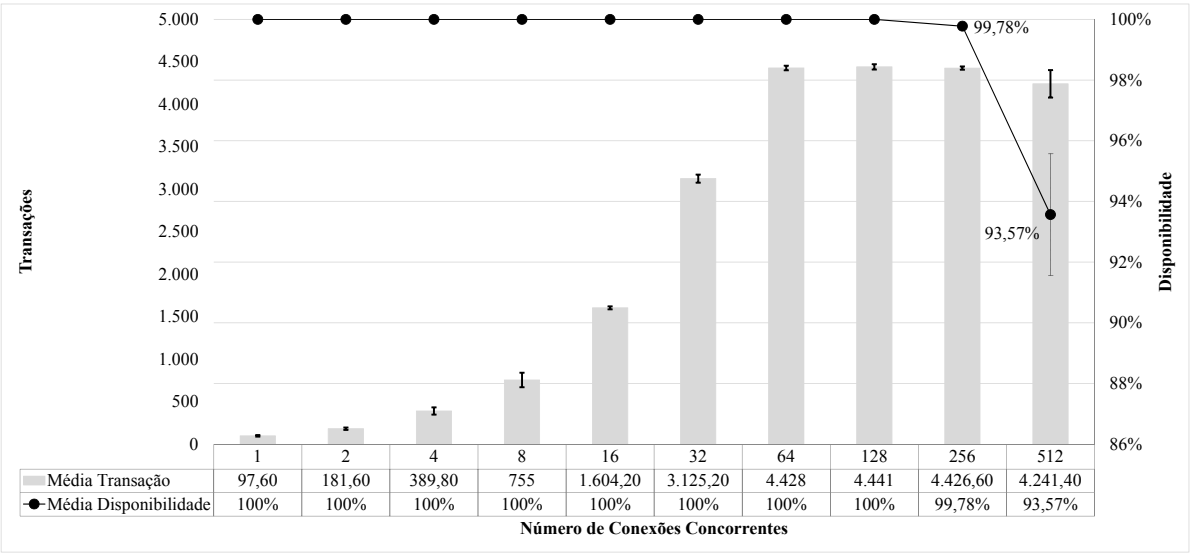
Nesta seção os gráficos que apresentam a média de transações no processo de leitura/escrita de um documento com tamanho de 10 MB, as colunas apresentadas no "eixo x" ilustram o número de conexões concorrentes e no "eixo y" tem-se o total de transações realizadas com sucesso (lado esquerdo) e a percentagem da disponibilidade do serviço (lado direito).

Para os gráficos que apresentam a média da vazão e tempo de resposta para leitura/escrita de um documento com tamanho de 10 MB, as colunas apresentadas no "eixo x" ilustram o número de conexões concorrentes e no "eixo y" tem-se a vazão (lado esquerdo) e tempo de resposta (lado direito).

Para medição na leitura cíclica de um único documento com tamanho de 10 MB, a estrutura de armazenamento por objetos (Figura 27) apresentou como resultado um crescimento no número de transações executadas até o total de 128 conexões concorrentes.

Assim como média de disponibilidade de serviço (representado por uma linha) uma taxa de 100% para esta quantidade de conexões.

O número de transações foi sustentado após 128 conexões concorrentes com base na vazão disponível para tráfego de dados e somente ao alcançar o limiar de 256 conexões concorrentes ocorre um decréscimo. Esta redução ocorre porque os servidores de armazenamento por objetos não conseguem atender todas as 512 conexões concorrentes para o procedimento de leitura cíclica em um único documento com tamanho de 10 MB. Desta forma, tanto o número de transações quanto da disponibilidade do serviço é afetado negativamente.



	1	2	4	8	16	32	64	128	256	512
Média Transação	97,60	181,60	389,80	755	1.604,20	3.125,20	4.428	4.441	4.426,60	4.241,40
Média Disponibilidade	100%	100%	100%	100%	100%	100%	100%	100%	99,78%	93,57%

Figura 27: Média de Transações no Processo de Leitura de um Documento com Tamanho de 10 MB.

Com relação à escrita de um documento com tamanho de 10 MB, os resultados obtidos elucidam o inverso ao alcançado na subseção 5.2. Conforme apresentado na Figura 28, a média de transações efetuadas que foi de 4.471,40 transações para um total de 128 conexões concorrentes. A linha representa a média da disponibilidade de serviço obtida, que foi de 100% para esta mesma quantidade de conexões concorrentes.

A disponibilidade do serviço decai para 97,57% no momento que o número de conexões concorrentes aumenta para 512, ou seja, esta infraestrutura suporta até 256 conexões concorrentes com 100% de disponibilidade, tentativas de transações com mais conexões não são concluídas. Conclui-se que, como limiar recomendável tem-se um total de 128 conexões concorrentes para transferência de dados com ganho em desempenho uma vez que ao aumentar o número de conexões, a tendência é manter a vazão para estes valores analisados e com taxa constante sem maiores ganhos como pode ser observado na Figura 30.

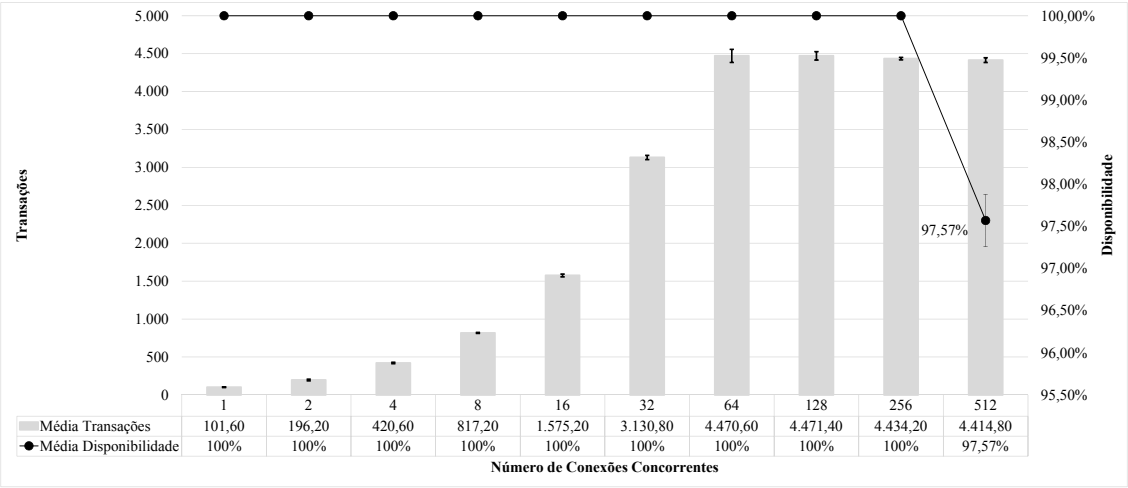


Figura 28: Média de Transações no Processo de Escrita de um Documento com Tamanho de 10 MB.

A Figura 29 mostra que o tempo de resposta se manteve abaixo de 1,20 ms com até 128 conexões concorrentes. Este resultado comprova os valores apresentados na Figura 27, pois demonstra que o limite de tráfego de dados foi alcançado. A medida que o número de conexões cresce, o tempo de resposta praticamente tende a dobrar, e o número de operações por segundo (op/s) para leitura de um documento com 10 MB é reduzido.

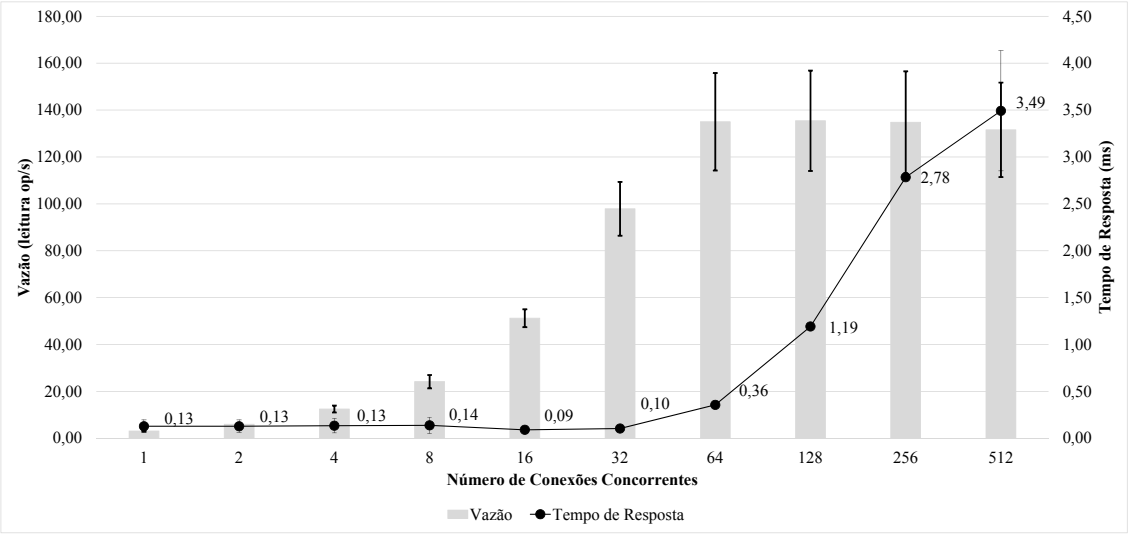


Figura 29: Média da Vazão e Tempo de Resposta para Leitura de um Documento com Tamanho de 10 MB.

De acordo com os resultados ilustrados na Figura 30, o aumento da vazão no procedimento de escrita ocorre até 128 conexões concorrentes por cliente e alcança o seu

limite ao realizar 4.471,40 transações de escrita, com uma média de tempo de resposta de 1,18 ms. Com base nesses resultados, conclui-se que mais conexões além deste ponto apenas impactam diretamente no tempo de resposta contudo a capacidade de tráfego de dados é mantida.

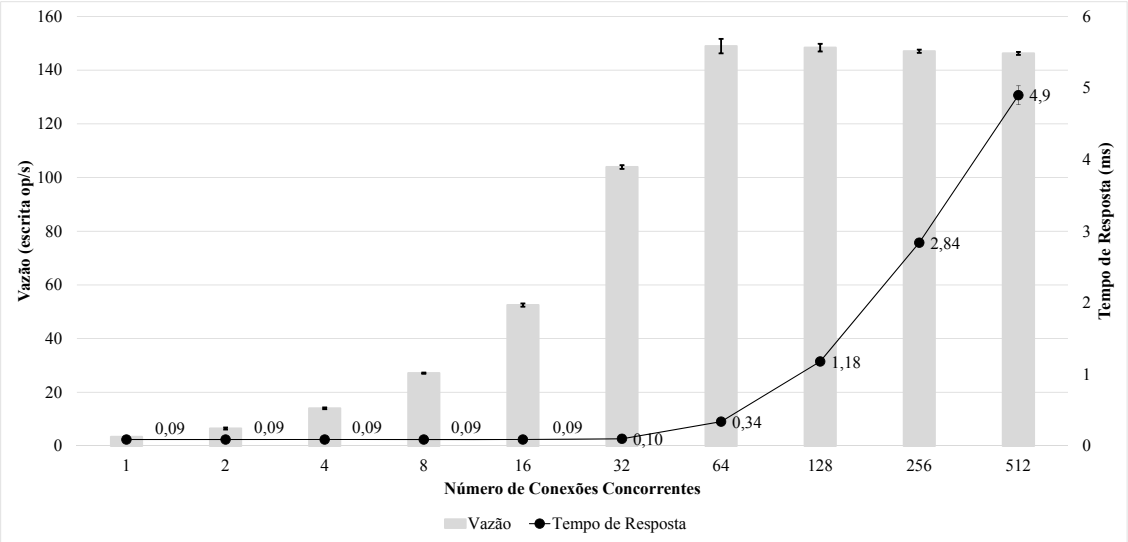


Figura 30: Média da Vazão e Tempo de Resposta para Escrita de um Documento com Tamanho de 10 MB.

A Figura 31 ilustra que para o armazenamento por objeto (representado "Média Transações CAS"), o número de transações alcançadas foi superior às demais estruturas analisadas, assim como a média da disponibilidade do serviço. É factível dizer que a percentagem sugestiva à disponibilidade do serviço com 512 conexões concorrentes é equivalente à obtida na estrutura por bloco (representado "Média Transações por bloco") uma vez que o número de transações efetuadas com a mesma quantidade de conexões e tempo de execução tenha sido superior.

A estrutura de armazenamento por bloco (representado "Média Transações por bloco") também demonstrou ser equivalente às demais estruturas com até 8 conexões concorrentes. Após esta quantidade de conexões, em quase todos os casos foi superior à estrutura com deduplicação com exceção do ciclo de operação com 64 conexões concorrentes entre as quais os resultados estão aproximados. A média de disponibilidade do serviço manteve-se em 100% até 128 conexões concorrentes, para os demais casos houve redução do número de transações suportadas. Conclui-se que esta arquitetura é equivalente a uma estrutura de armazenamento por objetos para o procedimento de leitura cíclica com um único documento com tamanho de 10MB.

De acordo com os testes realizados e ilustrados na Figura 31, a estrutura de armazenamento por bloco com deduplicação (representado "Média Transações

Deduplicação") mostrou-se proporcional às demais estruturas com até 8 conexões concorrentes. Ao elevar o número de conexões, em nenhum dos demais casos foram obtidos resultados superiores ao número de transações realizadas, bem como a sua média de disponibilidade foi reduzida após 128 conexões concorrentes.

É possível observar que através dos testes realizados com deduplicação, no pior caso, ou seja, com 512 conexões concorrentes sua média de disponibilidade caiu para 84,60%. Tal redução ocorre devido ao tempo necessário referente a checagem de *hash* para cada um dos documentos com tamanho de 10 MB aos quais efetua-se o procedimento de leitura, desta forma a medida que o número de conexões concorrentes cresce impacta negativamente no número de transações que podem ser efetuadas.

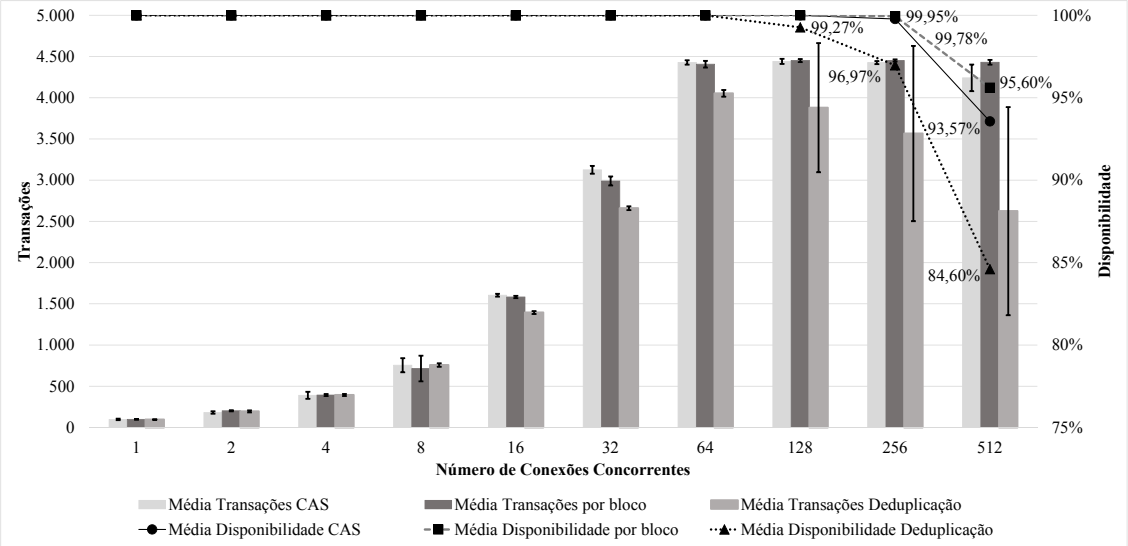


Figura 31: Média de Transações e Disponibilidade para Leitura de um Documento com Tamanho de 10 MB.

Na Figura 32 nota-se que o número de transações representado pela "Média de Transações CAS" realizadas com sucesso é superior às demais estruturas de armazenamentos analisadas, bem como a disponibilidade do serviço do número de requisições efetivadas. Neste gráfico a média de transações com o CAS foi de 4.410,80 para um total de 512 conexões concorrentes, e que somente neste ciclo de operação a disponibilidade do serviço decaiu para 97,57%, ou seja, seu limiar de transações foi alcançado.

Nenhum dos resultados obtidos para o armazenamento de dados por bloco (Figura 32) representado "Média Transações por bloco" e disponibilidade foram superiores às demais estruturas avaliadas. Entretanto, ao aumentar o número de conexões, a média obtida relacionada às transações realizadas decaiu, bem como a percentagem pertinente à disponibilidade do serviço após 128 conexões concorrentes foi reduzida em até 20,61% com

512 conexões concorrentes. Percebe-se que para o procedimento de escrita repetidas vezes com base em um único documento com tamanho de 10 MB, a estrutura de armazenamento por blocos apresenta crescimento vertical e confiabilidade inferiores a estrutura por objeto.

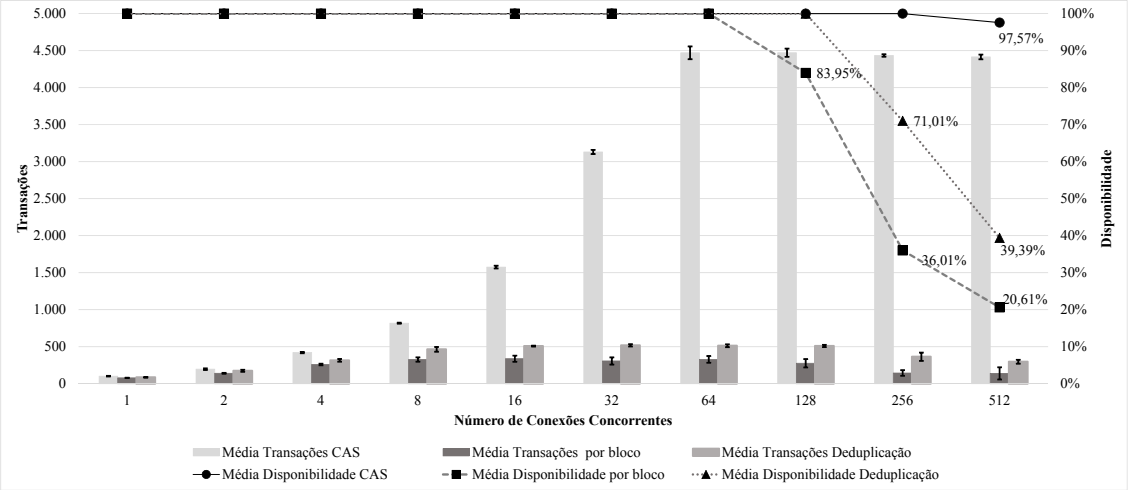


Figura 32: Média de Transações e Disponibilidade para Escrita de um Documento com Tamanho de 10 MB.

Ainda com base nesse gráfico (Figura 32), o armazenamento de dados por bloco com deduplicação representado pela "Média Transações Deduplicação", obteve um número de transações concretizadas com sucesso até um total de 128 conexões concorrentes, após este valor a disponibilidade do serviço decai para 71,01% com 256 conexões concorrentes e 39,39% com 512 conexões concorrentes. Por fim, em comparação com o armazenamento de dados por objetos, tanto o número de transações como a disponibilidade do serviço decai mais de 75%. Logo, com base nestes valores percebe-se que esta estrutura de deduplicação não é indicada para arquivos grandes (10 MB), uma vez que o tempo necessário para o processamento dos blocos com tamanhos de 4 KB é elevado.

5.4 Avaliação do Procedimento de Leitura e Escrita em Múltiplos Documentos

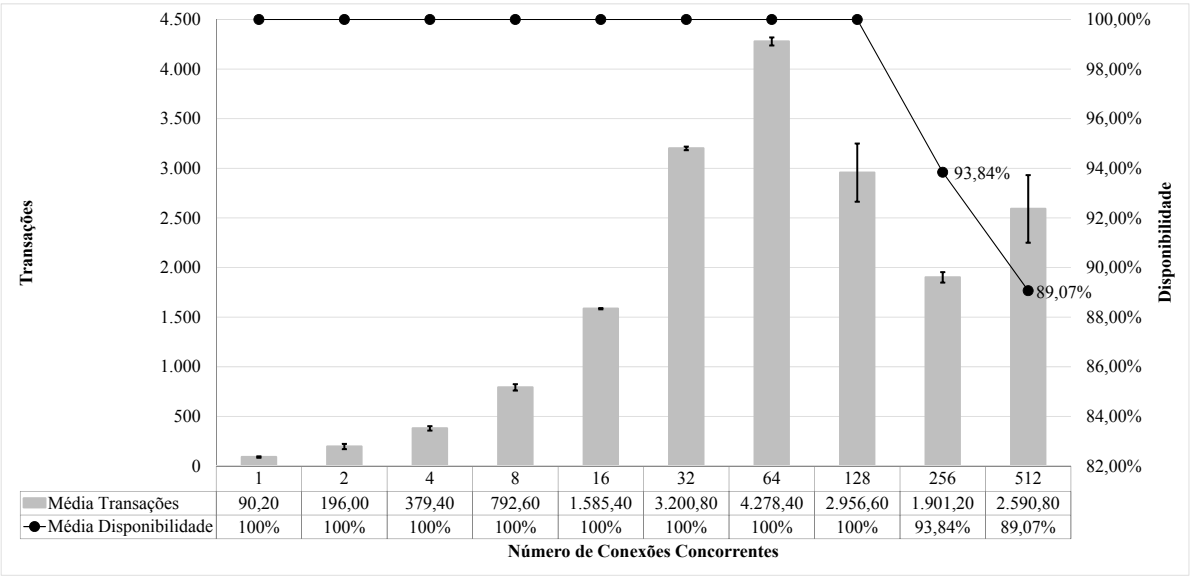
Nesta seção os gráficos que apresentam a média de transações no processo de leitura/escrita em múltiplos documentos, as colunas apresentadas no "eixo x" ilustram o número de conexões concorrentes e no "eixo y" tem-se o total de transações realizadas com sucesso (lado esquerdo) e a percentagem da disponibilidade do serviço (lado direito).

Para os gráficos que apresentam a média da vazão e tempo de resposta para leitura/escrita em múltiplos documentos, as colunas apresentadas no "eixo x" ilustram

o número de conexões concorrentes e no "eixo y" tem-se a vazão (lado esquerdo) e tempo de resposta (lado direito).

Como pode ser visto na Figura 33, os resultados obtidos de transações e da disponibilidade do serviço relativo a leitura com base em 50 documentos ilustrados na Figura 20 que possuem tamanhos variados (*kilobytes* e *Megabytes*) em um único recipiente, demonstram que o armazenamento por objetos é recomendável para até 64 conexões concorrentes com 100% de disponibilidade.

Ao analisar a Figura 33, percebe-se que a média para o limite referente ao número de transações realizadas foi de 4.278,40 para um total de 64 conexões concorrentes e que ao aumentar o número de conexões concorrentes ocorre um decréscimo na disponibilidade do serviço e no número de transações suportadas. Tal situação ocorre frente a distribuição documental (Figura 20) gerada para este ambiente de avalição, devido ao número de transações que podem ser realizadas sobre esta estrutura de armazenamento por objetos ter alcançado seu limite aceitável de tráfego de dados.



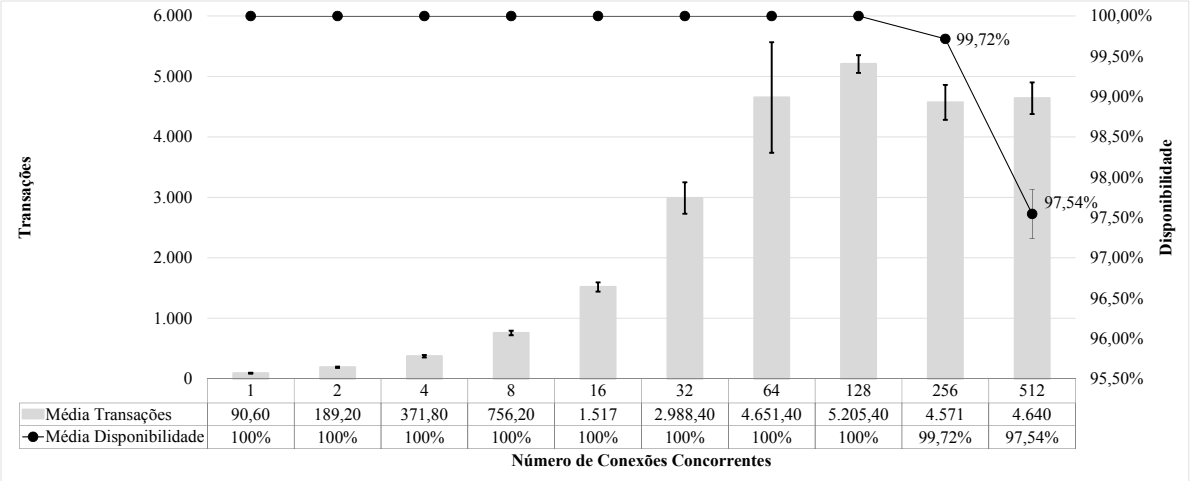
	1	2	4	8	16	32	64	128	256	512
Média Transações	90,20	196,00	379,40	792,60	1.585,40	3.200,80	4.278,40	2.956,60	1.901,20	2.590,80
Média Disponibilidade	100%	100%	100%	100%	100%	100%	100%	100%	93,84%	89,07%

Figura 33: Média de Transações no Processo de Leitura de Múltiplos Documentos com Tamanhos Variados.

Neste cenário foram realizados testes para a escrita de documentos com tamanhos distintos através de múltiplas conexões concorrentes nos quais o armazenamento por objeto também demonstrou ser o mais recomendável. A Figura 34 ilustra que à medida que o número de conexões concorrentes aumenta, o número de transações cresce até o limiar de 128 conexões concorrentes.

Desta forma, tem-se para 128 conexões concorrentes uma média de 5.205,40 transações e disponibilidade de serviço em 100% dos ciclos de execuções. Acima de 128 conexões concorrentes, ocorre um decréscimo na disponibilidade do serviço para 99,72% com

256 conexões concorrentes e 97,54% com 512 conexões concorrentes. Outro ponto negativo, é que também houve uma redução considerável no número de transações, contudo a vazão foi mantida como pode ser visto na Figura 36. A redução no número de transações ocorre devido ao limiar do tráfego de dados ter sido alcançado para este total de operações efetuadas.



	1	2	4	8	16	32	64	128	256	512
Média Transações	90,60	189,20	371,80	756,20	1.517	2.988,40	4.651,40	5.205,40	4.571	4.640
Média Disponibilidade	100%	100%	100%	100%	100%	100%	100%	100%	99,72%	97,54%

Figura 34: Média de Transações no Processo de Escrita de Múltiplos Documentos com Tamanhos Variados.

A Figura 35 mostra a vazão e o tempo de resposta para leitura de múltiplos documentos. Neste ambiente a média para o tempo de resposta é mantido abaixo de 6 ms até 128 conexões concorrentes, entre 128 e 256 esse tempo dobra. Assim, observa-se que a vazão cresce até um total de 128 conexões concorrentes e deste ponto em diante ocorre um impacto diretamente no número de transações que podem ser realizadas.

Com base nesse contexto, após este valor, o tempo de resposta dobra para 256 conexões concorrentes e com 512 conexões concorrentes o número de transações com erro cresce. Isto ocorre porque como há um elevado número de operações de leitura com documentos que possuem tamanhos diversificados em tráfego de rede, o limite suportado entre o servidor *web* e servidor CAS já foi alcançado o que impacta diretamente no número de transações que podem ser efetuadas.

O gráfico ilustrado na Figura 36 expõe que a vazão aumenta até 128 conexões concorrentes e o tempo de resposta mantém-se abaixo de 0,94 ms. Após esta quantidade de conexões concorrentes, a vazão média decai entretanto é mantida próxima ao limite e o tempo de resposta cresce linearmente. Desta forma é possível concluir que não haverá mais ganho de desempenho após 128 conexões concorrentes.

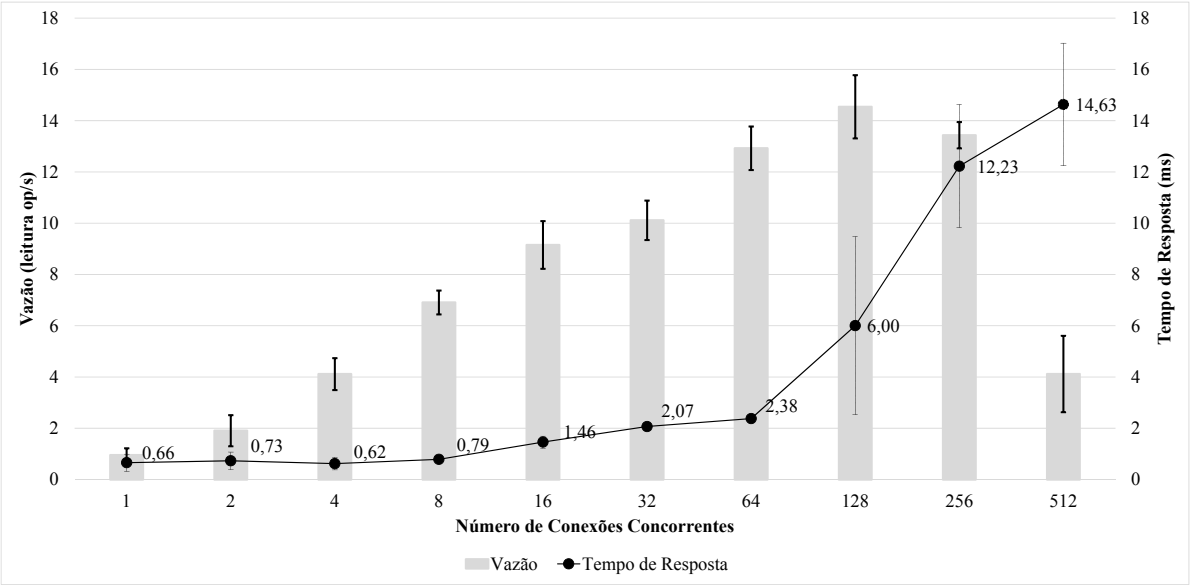


Figura 35: Média da Vazão e Tempo de Resposta para Leitura de Múltiplos Documentos com Tamanhos Variados.

O gráfico apresentado na Figura 37 ilustra que o armazenamento por objetos representado "Média Transações CAS", é similar às demais estruturas com até 16 conexões concorrentes. Após este quantitativo, o número de transações com esta estrutura aumenta até 64 conexões concorrentes e com média de 100% de disponibilidade de serviço. Esse crescimento referente ao número de transações ocorre devido ao armazenamento distribuído entre os 3 servidores CAS, ou seja, escalabilidade.

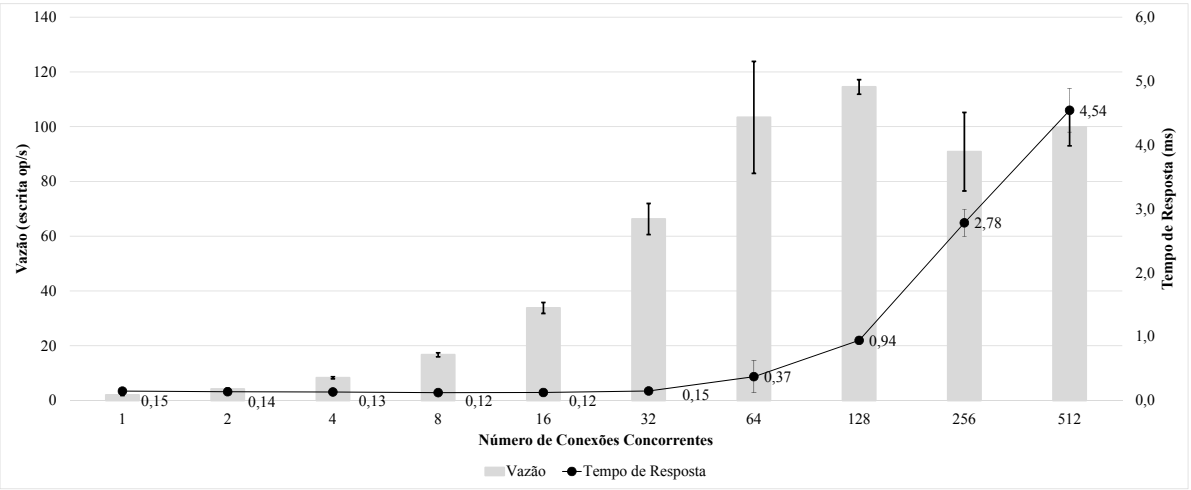


Figura 36: Média da Vazão e Tempo de Resposta para Escrita de Múltiplos Documentos com Tamanhos Variados.

A partir do gráfico ilustrado na Figura 37 pode-se concluir que tanto a estrutura por bloco representado pela "Média Transações por bloco" quanto a estrutura com

deduplicação representado pela "Média Transações Deduplicação" sofrem uma redução no número de transações de leitura de múltiplos documentos a cada ciclo após 64 conexões concorrentes. Isto ocorre porque o tempo de resposta para estes servidores referente a leitura de múltiplos documentos com tamanhos variados é mais elevado do que o servidor CAS.

Para o servidor de arquivos com deduplicação, a medida que o número de conexões concorrentes aumenta é necessário um maior tempo para o processamento de todas as transações o que acarreta em um impacto negativo referente ao número de transações suportadas e disponibilidade do serviço. Até aqui, a média de disponibilidade de serviço é mantida em 100% com até 128 conexões. Ao aumentar o número de conexões concorrentes, a disponibilidade de serviço reduz e tem-se no pior caso um decréscimo de 29,13% para a estrutura com deduplicação.

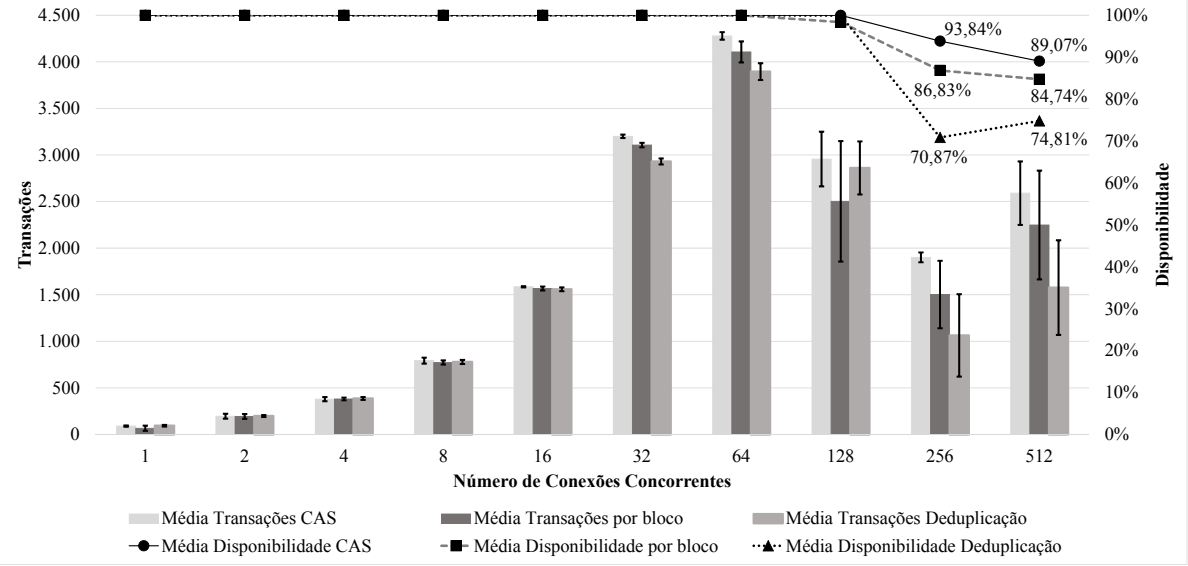


Figura 37: Média de Transações e Disponibilidade para Leitura de Múltiplos Documentos com Tamanhos Variados em Cenários Distintos.

A Figura 38 mostra a média de transações e disponibilidade dos serviços de 3 estruturas, na qual a arquitetura por objetos manteve em 100% dos casos a média de disponibilidade até 128 conexões concorrentes. Neste cenário o servidor CAS apresentou uma característica muito relevante comparado as demais estruturas ao manter o número de transações realizadas próximo do seu limite, ou seja, sua escalabilidade ao executar múltiplas conexões com múltiplos documentos.

O gráfico mostrado na Figura 38 ilustra que os resultados obtidos com o armazenamento de dados com deduplicação representado pela "Média Transações Deduplicação" em comparação com as demais estruturas analisadas possui um intervalo de confiança alto após 64 conexões concorrentes e disponibilidade do serviço baixa à medida

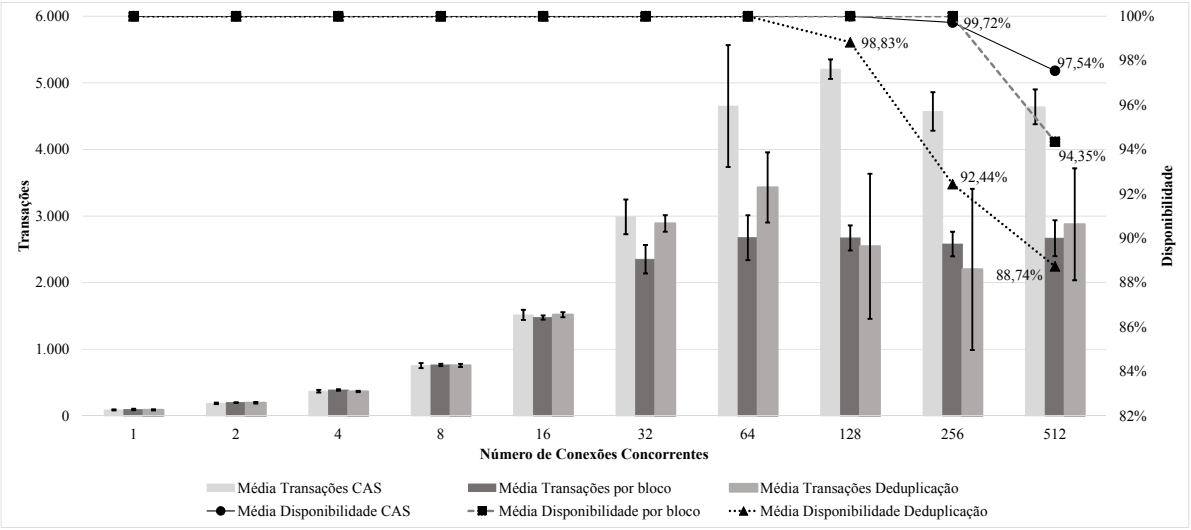


Figura 38: Média de Transações e Disponibilidade para Escrita de Múltiplos Documentos com Tamanhos Variados em Cenários Distintos.

que o número de conexões concorrentes ultrapassou o seu limite aceitável de transações, o que impactou nos resultados obtidos para cada ciclo de teste.

Por fim, vale ressaltar que nesse ambiente para cada ciclo após 64 conexões concorrentes o servidor *web* travou tanto com a estrutura por bloco quanto com a estrutura por bloco com deduplicação, cuja única solução foi reiniciar este servidor. Uma lição importante é que a arquitetura de armazenamento de conteúdo endereçável demonstrou ser a mais viável para transações de leitura com 64 conexões concorrentes e de escrita com 128 conexões concorrentes, com uma disponibilidade de serviço em 100% para cada um destes casos analisados.

5.5 Escalabilidade e Replicação

A escalabilidade é obtida devido a leitura e inserção dos documentos digitais serem efetivadas sobre uma estrutura por objetos com recipiente(s) distribuído(s) de forma horizontal. Desta forma, os procedimentos de busca, indexação e armazenamento com base no ID de um documento digital impacta na necessidade de menos informações em tráfego sobre a rede que os sistemas de arquivos hierárquico.

Foi gerado um comparativo para mostrar a capacidade suportada para cada um dos servidores de armazenamento por objetos no momento da leitura de múltiplos documentos. Com base no exemplo da Figura 39, pode-se observar que, ao acrescentar mais 2 servidores, a vazão cresce até atingir 128 conexões concorrentes, limite alcançado neste ambiente de teste. A partir deste valor pode-se concluir que, ao atribuir mais conexões, ocorre um

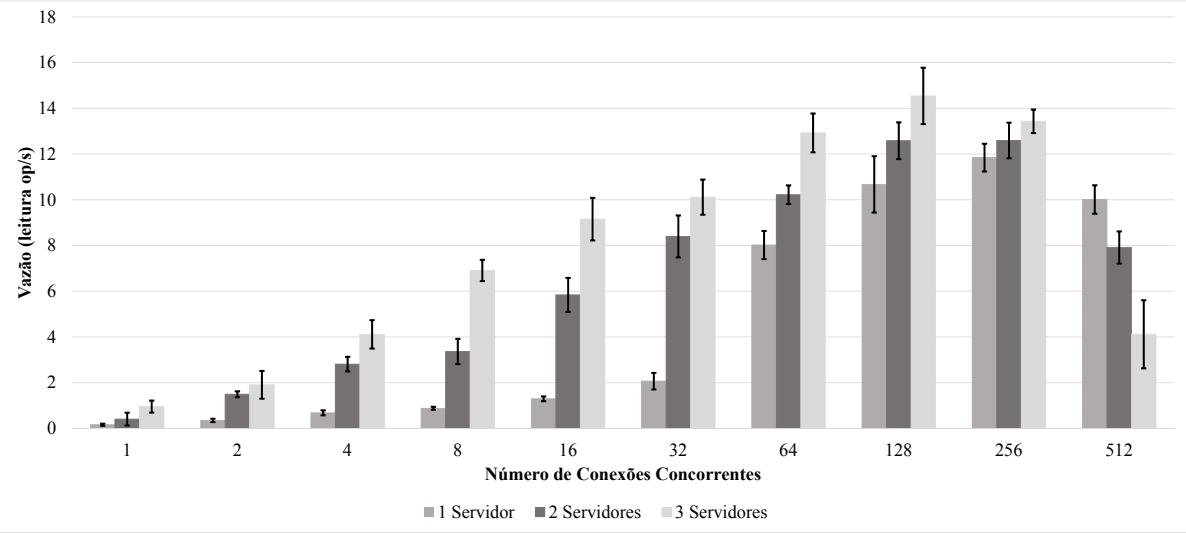


Figura 39: Vazão Média em um Sistema Distribuído CAS.

aumento no número de transações com erros uma vez que a vazão já atingiu o seu limite suportado.

Vale lembrar que a arquitetura foi implementada com replicação local. Existem mais 2 servidores CAS distribuídos fisicamente e logicamente entre os prédios interligados ao TCE-TO afim de que todo o processo de escrita seja replicado automaticamente para os demais servidores em rede.

Figura 40: Indisponibilidade de uma Unidade de Armazenamento.

A confiabilidade é obtida por meio da replicação dos documentos a serem redistribuídos para os 3 (IP: 192.1680.89; IP: 192.168.0.90 e IP: 192.168.0.91) servidores CAS. Em caso de indisponibilidade de uma unidade de armazenamento, assim como apresentado na Figura 40 com 1 servidor (IP: 192.168.0.90) CAS, os demais dispositivos pertencentes ao grupo continuam em funcionamento sem impactar na disponibilidade do serviço. Caso

ocorra falha total e substituição do respectivo servidor CAS, é iniciado um processo de recuperação das réplicas dos objetos armazenados nos 2 servidores *online*.

5.6 Checagem de Integridade Documental

A ação de escrita com checagem de integridade documental, é realizada ao adicionar um cabeçalho na requisição chamado "*Content*-MD5"que contém o *base*64 do MD5 computado do conteúdo do documento. Caso os *hashs* sejam iguais, a resposta será 201 (*Created*) e retornará um cabeçalho "*Location*"com o "selo" de integridade no tipo de *hash* especificado através do *hashtype*.

Perante à verificação online da integridade sobre os documentos armazenados, é necessário que o *framework* efetue uma validação, para que o UUID de cada documento seja obtido através do seu endereço de conteúdo e em seguida associado com os metadados. Estas informações (endereço de conteúdo e metadados) são usadas para verificar se o conteúdo de um determinado documento não foi alterado. Contudo é necessário ressaltar que os metadados de conteúdo não estão incluídos no endereço de conteúdo.

A URL contém o nome do objeto/UUID, o seu valor e o tipo de algoritmo que foi utilizado para os cálculos, refere-se aos parâmetros de identificação para leitura ou escrita de um documento com integridade. O *framework* criado para o sistema e-Contas para efetuar a checagem da integridade documental, solicita tais parâmetros da seguinte forma:

POST http:// tdinfocascluster.tce-to.tce.to.gov.br:90/?hashtype=sha256 HTTP/1.1

O "selo" de integridade pode ser utilizado em um pedido de leitura posterior para validar os dados armazenados em um *cluster*. Por exemplo, ao fornecer a URL informada no cabeçalho para uma requisição de escrita para um endereço de outro servidor CAS, o *framework* solicita a este servidor CAS para validar a leitura dos dados.

GET http:// tdinfocascluster.tce-to.tce.to.gov.br:90/41A140B5271DC8D22FF8D027176A0821?hashtype=sha256&hash=7A25E6067904EAC8002498CF1AE33023HTTP/1.1

Após receber uma requisição de leitura, o servidor CAS computa novamente o *hash* do conteúdo armazenado com o fornecido e compara ambos os valores. Se os dois valores não corresponderem, o CAS irá vai encerrar a conexão antes de enviar o conteúdo do objeto, ou seja, o objeto não será armazenado. De modo similar, se o conteúdo não foi modificado/corrompido, se ambos os valores *hash* coincidirem, o CAS então apresenta o objeto solicitado.

A Figura 41 mostra a resposta obtida durante a escrita de um documento em um dispositivo de armazenamento CAS. Levando isso em consideração, pode-se observar em

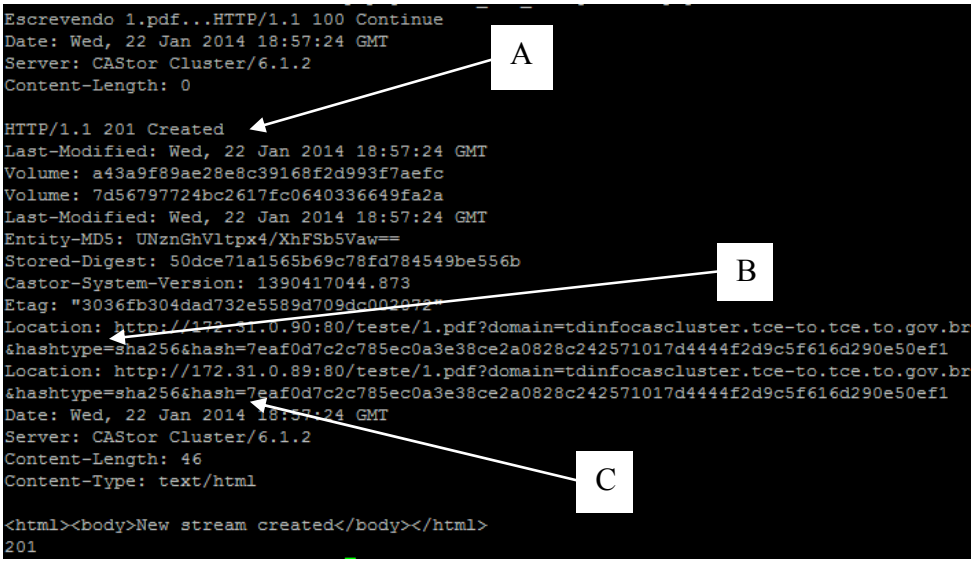


Figura 41: Resposta Obtida após a Escrita em CAS.

(A) o *status* da operação realizada com sucesso já que os *hashs* foram checados e tidos como iguais. Em (B) é demonstrado o *hashtype* (algoritmo SHA256) utilizado nessa operação e em (C) o "selo" de integridade referente à localização para um objeto não identificado.

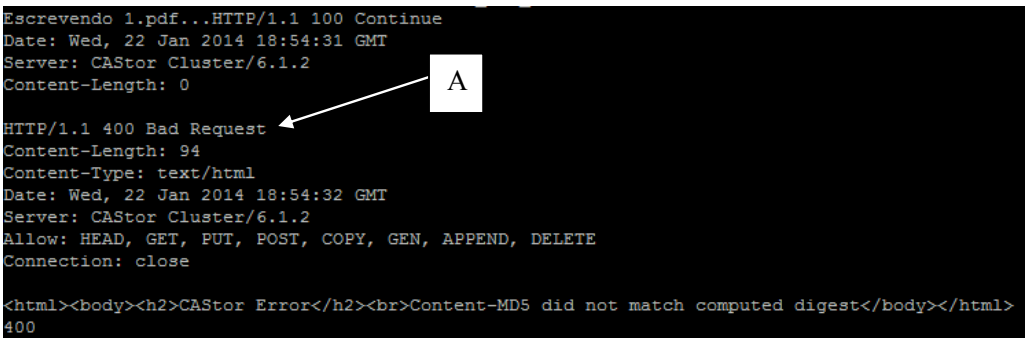


Figura 42: Resposta de Erro obtida relativo a Checagem de Integridade.

Como pode ser visto na Figura 42, ocorre uma requisição com falha no procedimento de escrita que foi constatada com a checagem de integridade. Nesse contexto, em (A) a resposta de erro (400 *Bad Request*) refere-se à operação que não foi realizada devido aos valores em *hash* serem divergentes, por isso a conexão com o servidor CAS é encerrada, e o objeto não é armazenado.

Para o processo de leitura com checagem de integridade, utiliza-se o "selo" de integridade no tipo de *hash* que é fornecido no tempo de escrita, desta forma é possível efetuar a leitura e checagem de sua integridade após o documento ter sido recebido via navegador *web*. Caso o documento seja tido como não íntegro, a conexão será finalizada

com dispositivo CAS e não há a exibição do mesmo em tela. Todos os *scripts* utilizados em ambos os métodos estão descritos no Apêndice A.

5.7 Resumo Conclusivo

O objetivo deste capítulo foi analisar a proposta deste trabalho ao realizar testes em 3 diferentes tipos de cenários. O intuito foi demonstrar os ganhos em número de transações realizadas com sucesso e o percentual obtido para cada ciclo direcionado a disponibilidade do serviço da estrutura de armazenamento por objetos em comparação com a infraestrutura existente no TCE-TO e uma infraestrutura por bloco com deduplicação em nível de bloco com tamanho de 4 KB de comprimento fixo.

No primeiro cenário, conforme apresentado na seção 5.2, de acordo com os testes de leitura e escrita realizados em um documento com tamanho de 282 KB, a estrutura por objetos apresentou ganhos para todos os ciclos, tanto no número de transações realizadas cuja variação para cada ciclo foi mantida linearmente quanto no percentual relacionado a disponibilidade do serviço, situação que não ocorreu com as demais estruturas de armazenamento. Entretanto, a estrutura de armazenamento de dados por bloco apresentou ganhos de 38,19% no percentual de transações no procedimento de leitura. E a estrutura de armazenamento de dados por blocos com deduplicação apresentou ganhos de 75,16% no percentual de transações no procedimento de escrita.

Para o cenário apresentado na seção 5.3 (documentos com tamanho de 10 MB), a estrutura por objetos demostrou ser equivalente a estrutura por bloco com ganhos de 0,57% no percentual de transações no procedimento de leitura de um único documento tido nesta dissertação como grande. Entretanto, para o procedimento de escrita, os testes demonstram ganhos com resultados 93,8% superiores as demais estruturas para o número de transações realizadas, e um percentual de 100% para disponibilidade do serviço.

O terceiro cenário mostrado na seção 5.4 (documentos de tamanhos diversos), ilustra um aumento de 15,36% no percentual de transações no procedimento de leitura e de 48,67% no percentual de transações no procedimento de escrita em relação as demais. Além disso, os resultados obtidos tanto com a vazão quanto com o tempo de resposta para o CAS mostram qual o limiar aceitável para a estrutura proposta em cada ambiente.

Por fim, foi demostrado que a infraestrutura de armazenamento de conteúdo endereçável impactou em ganhos referentes a escalabidade e que a aplicação da checagem da integridade em nível físico e lógico permite melhorar o nível de segurança para o armazenamento de dados. É necessário ter em mente que os cenários considerados foram elaborados de acordo com o formato e tamanhos de arquivos identificados no TCE-TO.

Partindo dos resultados obtidos neste Capítulo, a conclusão desta dissertação é descrita no Capítulo 6.

Capítulo 6

Conclusão

Devido a necessidade de uma infraestrutura de armazenamento documental adaptada as demandas dentro do Tribunal de Contas do Estado do Tocantins como um meio para melhorar a performance do tráfego de dados, disponibilização da informação com confiabilidade (fidedignidade e autenticidade) e ampliação do nível de segurança através da verificação da integridade pertinente aos documentos salvos. Este trabalho propõe e avalia um projeto para implantação de uma arquitetura de armazenamento de conteúdo endereçável com deduplicação em nível de arquivo para o arquivamento de documentos digitais direcionada a uma plataforma em nuvem.

Foram apresentadas as motivações relevantes a este trabalho no Capítulo 1, entre as quais destaca-se a necessidade de uma infraestrutura de armazenamento que proporcione maior nível de segurança para a salvaguarda dos documentos digitais. Com base neste contexto, foram identificados os parâmetros necessários para a implantação de um repositório que ofereça fidedignidade e autenticidade aos documentos digitais salvos.

O Capítulo 2 apresentou os conceitos básicos sobre o gerenciamento eletrônico de documentos, computação em nuvem, armazenamento em nuvem, armazenamento de conteúdo endereçável, questões de segurança e sistemas de armazenamento. Após uma visão geral, foi possível entender qual o tipo estrutura de armazenamento é mais indicado para a salvaguarda de documentos digitais e ao mesmo tempo qual propicia obter ganhos em performance. Também foi possível compreender como melhorar o nível de segurança dos dados armazenados e impossibilitar o armazenamento de documentos duplicados.

No Capítulo 3 foram descritos trabalhos existentes na literatura na qual as abordagens, estudos e metodologias estão relacionados a proposta dessa dissertação. Esse estudo na literatura encontrou obstáculos expostos pelos autores pertinentes a metodologia a ser adotada para analisar uma estrutura de armazenamento em nuvem. Contudo, foi encontrado uma única proposta sobre como avaliar serviços de armazenamento em nuvem. Com base nas características deste tipo de serviço e correlacionado a essas abordagens, a

metodologia atribuída nessa dissertação foi estabelecida de acordo com as particularidades do TCE-TO.

Os trabalhos apresentados no estado da arte contribuíram na compreensão sobre como elaborar uma análise na arquitetura de armazenamento de conteúdo endereçável. No Capítulo 4 foi analisado o histórico documental existente no TCE-TO no ano de 2013 e assim definido o tamanho dos documentos (pequenos e grandes) que foram tidos como métricas no ambiente de testes. Em seguida foi apresentado o ambiente de testes utilizado para confrontar a arquitetura atual a solução proposta de arquitetura de armazenamento, e uma arquitetura com blocos com deduplicação. Esta última, foi preparada com o objetivo de considerar se haveria ganhos ao aplicar uma técnica de deduplicação e assim continuar em uma estrutura de armazenamento por blocos.

De acordo com os resultados apresentados no Capítulo 5 chegou-se à conclusão que, a arquitetura de armazenamento por objetos possibilita reais ganhos no número de transações realizadas com sucesso tanto com o procedimento de leitura quanto com a escrita, até atingir o limiar referente a vazão para cada respectivo procedimento. Por outro lado, observou-se que ao aumentar o número de conexões além do limite obtido para cada ambiente de teste, o tempo de resposta é prolongado e a taxa de transferência de dados é conservada.

De modo similar, o teste referente a escalabilidade também demonstra que há um crescimento no número de transações à medida que mais servidores CAS são adicionados até alcançar o ápice do tráfego de dados suportado por essa infraestrutura. Por fim, a utilização de deduplicação em nível de arquivo mostrou ser uma boa alternativa tanto para impossibilitar o armazenamento de documentos duplicados quanto para melhorar o nível de segurança através da verificação da integridade pertinente aos documentos salvos.

Trabalhos futuros podem considerar a implantação da deduplicação em nível de blocos com tamanho fixo em uma arquitetura de armazenamento de conteúdo endereçável, para determinar qual o tamanho de bloco mais recomendável para o tráfego de dados em rede e assim manter um desempenho equivalente a deduplicação em nível de arquivos analisada nesse trabalho. Nesse contexto, também tem-se tanto os possíveis ganhos com redução em espaço físico de armazenamento como a produção de trabalhos acadêmicos com base nos respectivos resultados.

Referências

[1] Alfresco. Alfresco. Disponível em: http://www.alfresco.com/, abril de 2014. 11

[2] Amazon. Amazon Elastic Compute Cloud (Amazon EC2). Disponível em: https://aws.amazon.com/pt/ec2/, janeiro de 2014. 7

[3] Amazon. Amazon Simple Storage Service (Amazon S3). Disponível em: http://aws.amazon.com/pt/s3/, janeiro de 2014. 8

[4] Apache. The Apache Software Foundation. Disponível em: http://www.apache.org/, outubro de 2013. 26

[5] Azure. Microsoft Azure. Disponível em: http://www.microsoft.com/azure/, novembro de 2013. 7

[6] BRASIL. Lei 8.159, de 08 de janeiro de 1991. Dispõe sobre a política nacional de arquivos públicos e privados e dá outras providências. Disponível em: http://www.planalto.gov.br/ccivil_03/leis/L8159.htm, junho de 2013. 2

[7] Castor. *CAStor Application Guide*, volume 6.1.2. Caringo, 2012. 96. 20, 21

[8] Alex Osuna; Eva Balogh; Alexandre Ramos Galante de Carvalho; Rucel F. Javier e Zohar Mann. *Implementing IBM Storage Data Deduplication Solutions*. RedBooks, 2011. 4, 5. 16

[9] DeepStorage. Deepstorage Records Management. Disponível em: http://www.deepstore.com/, abril de 2014. 11

[10] Roniberto Morato do Amaral e Adriana Aparecida Puerta. Gerenciamento Eletrônico de Documentos (GED): justificativas para a implantação do GED e tecnologias correlatas ferramentas de hardware e software. *Portal de Congressos da FEBAB, XXIV Congresso Brasileiro de Biblioteconomia, Documentação e Ciência da Informação*, pages 1–12, 2011. 3, 8. 2, 4

[11] Ricardo Rosa dos Anjos. GED - Gestão Eletrônica de Documentos: Tecnologia Eficaz e Sustentável. *Revista Tecnologias em Projeção*, 2(1):4–6, 2011. 4, 5. 4

[12] Adam Jorgensen; Steven Wort; Ross LoForte e Brian Knight. *Professional Microsoft SQL Server 2012 Administration*. Wiley, 2012. 25. 13

[13] Larry L. Peterson e Bruce S. Davie. *Computer Networks: A Systems Approach*. Elsevier, 5 edition, 2011. 44. 38

[14] Eloi Juniti Yamaoka e Fernando Ostuni Gauthier. Ontologia de dependência tecnológica de documentos digitais: Instrumento de apoio à preservação digital. *Encontros Bibli - revista eletrônica de biblioteconomia e ciência da informação*, 17(2):211–226, 2012. 212. 2

[15] Behrouz A. Forouzan e Firouz Mosharraf. *Redes de Computadores: Uma Abordagem Top-Down*. AMGH, 2013. 697. 39

[16] George Coulouris; Jean Dollimore; Tim Kindberg e Gordon Blair. *Sistemas Distribuídos - Conceitos e Projeto*. Bookman, 5 edition, 2013. 71. 19

[17] Yang Zhang; Yongwei Wu e Guangwen Yang. Droplet: a Distributed Solution of Data Deduplication. *ACM/IEEE 13th International Conference on Grid Computing*, pages 114–121, 2012. 114. 18, 23

[18] Qing Zheng e Haopeng Chen. Cosbench: A Benchmark Tool for Cloud Object Storage Services. *IEEE 5th International Conference on*, pages 998–999, 2012. 23, 24, 25, 41

[19] Qing Zheng e Haopeng Chen. COSBench: Cloud Object Storage Benchmark. *ICPE'13 Proceedings of the 4th ACM/SPEC International Conference on Performance Engineering*, pages 199–210, 2013. 199, 201, 205. 24, 41

[20] Lawrence L. You; Kristal T. Pollack; Darrell D. E. Long e K. Gopinath. PRESIDIO: A Framework for Efficient Archival Data Storage. *ACM Transactions on Storage (TOS)*, 7(6):1–60, 2011. 1, 4. 11, 22, 23

[21] João Tiago e Leonardo Reis. *Arquivologia Facilitada. Teoria e Questões de Concursos*. Elsevier, 2011. 26. 1

[22] João Tiago Santos e Leonardo Reis. *Arquivologia facilitada: teoria e questões comentadas*. CAMPUS - RJ, 2010. 110. 5

[23] Jon Tate; Pall Beck; Hector Hugo Ibarra e Libor Miklas. *Introduction to Storage Area Networks and System Networking*. Redbooks, 2012. 11. 15

[24] Yi Jinsong; Wang Chunlu e Liu Chuanyi. Performance Comparisons of a Content-Addressable Storage Network System and Other Typical IP-SAN based Storage Systems. *Fourth International Conference on Intelligent Computation Technology and Automation*, 2:1142–1145, 2011. 1142. 22, 41

[25] Maria Cristina Diniz Caixeta e Maria Aparecida Carvalhais Cunha. Gestão documental e resgate da memória na justiça do trabalho: preservação documental é direito do cidadão e dever do estado. *Cadernos de História, Belo Horizonte*, 14(20), 2013. 33. 2

[26] Carlos Coronel; Steven A. Morris e Peter Rob. *Database Systems Design Implementation Management, 10th ed.: Design, Implementation, and Management*. Cengage Learning, 2012. 7. 11

[27] Kanatorn Jindarak e Putchong Uthayopas. Enhancing Cloud Object Storage Performanceusing Dynamic Replication Approach. *IEEE 18th International Conference on Parallel and Distributed Systems*, pages 800–803, 2012. 800. 23

[28] Maurício Barcellos Almeida; Beatriz Valadares Cendón e Renato Rocha Souza. Metodologia para implantação de programas de preservação de documentos digitais a longo prazo. *Encontros Bibli: revista eletrônica de biblioteconomia e ciência da informação*, 17(34):103–130, 2012. 103. 1

[29] Shuangbao Paul Wang e Robert S. Ledley. *Computer Architecture and Security: Fundamentals of Designing Secure Computer Systems*. Wiley, 2013. 82, 83. 14

[30] Alex Osuna; Eva Balogh; DaeSung Kim; Kai Lanzmich; Karen Orlando e Siew Chin Wong. *IBM System Storage TS7650, TS7650G, and TS7610*. RedBooks, 2011. 5, 9. 19

[31] Peter Mell e Timothy Grance. The NIST Definition of Cloud Computing (Draft). *The National Institute of Standards and Technology*, pages 1–7, 2011. 2, 3. 5, 7

[32] Ezmir Dippe Elias. GERENCIAMENTO ELETRÔNICO DE DOCUMENTOS (GED): aplicação na Universidade Federal de Santa Catarina. *Revista do Arquivo Público do Estado de Santa Catarina e do Curso de Arquivologia da UFSC, Florianópolis*, 22(45):15–30, 2012. 16. 1

[33] Eucalyptus. Eucalyptus Walrus Storage Considerations. Disponível em: http://www.eucalyptus.com, novembro de 2013. 8

[34] Jeff Fulmer. Joe Dog Software Proudly serving the Internets since 1999. Disponível em: http://www.joedog.org/siege-home/, maio de 2014. 35

[35] Paulo Ricardo Motta Gomes. Distributed Deduplication in a Cloud-based Object Storage System. Instituto Superior Técnico - Universidade Técnica de Lisboa, 2012. 9. 8

[36] Google. Google App Engine: Platform as a Service. Disponível em: https://developers.google.com/appengine/, maio de 2014. 7

[37] Google. Google Cloud Storage. Disponível em: https://cloud.google.com/products/cloud-storage, maio de 2014. 8

[38] Google. Google Compute Engine. Disponível em: https://cloud.google.com/products/compute-engine/, maio de 2014. 7

[39] Google. Google docs. Disponível em: http://docs.google.com/, maio de 2014. 7

[40] Heroku. Heroku. Disponível em: https://www.heroku.com/, maio de 2014. 7

[41] Justcloud. Justcloud. Disponível em: http://www.justcloud.com/, fevereiro de 2014. 7

[42] Lawrence C. Miller. *Object Storage for Dummies*. NetApp, 2013. 14. 1

[43] Nasuni. State of Cloud Storage Providers Industry Benchmark Report: A Comparison of Performance, Stability and Scalability. Technical report, Nasuni, 2011. 24, 41

[44] Terry Blonquist Nelson. *Managing Electronic Records*. IIMC Records Management Technical Bulletin Series, 2012. 12. 4

[45] Opendedup. Opendedup. Disponível em: http://opendedup.org/, abril de 2014. 19, 34

[46] Karl Paulsen. *Moving Media Storage Technologies. Applications & workflows for Video and Media Server Platforms*. Elsevier, 2011. 301, 302, 575. 10, 18

[47] Nigel Poulton. *Data Storage Networking: Real World Skills for the CompTIA Storage+ Certificartion and Beyond*. Sybex, 1 edition, 2014. 371, 373. 16, 17, 18

[48] Rackspace. Rackspace Cloud Files. Disponível em: http://www.rackspace.com/pt/cloud/files/, novembro de 2013. 7

[49] Nikolaus Rath. S3QL. Disponível em: https://bitbucket.org/nikratio/s3ql/overview, abril de 2014. 19

[50] Salesforce. Salesforce. Disponível em: http://www.salesforce.com/br/, maio de 2014. 7

[51] EMC Education Services. *Armazenamento e Gerenciamento de Informações. Como armazenar, gerenciar e proteger informações digitais*. Bookman, 2011. 67, 172, 173, 211, 214, 215. 8, 11, 15

[52] EMC Education Services. *Information Storage and Management, 2nd Edition: Storing, Managing, and Protecting Digital Information in Classic, Virtualized, and Cloud Environments*. Wiley, 2012. 159, 184, 185, 187, 249, 251, 263. 11, 12, 14, 17, 19

Apêndice A

Script utilizado no Ambiente de Avaliação

Este apêndice visa mostrar os *scripts* utilizados para automatizar a coleta de informações das quais os resultados foram apresentados no Capítulo 5 e o desenvolvimento do sistema de arquivos com base em deduplicação apresentado na seção 4.4. Para a execução dos ciclos de conexões concorrentes para os 3 ambientes de teste analisados nessa dissertação, tem-se o uso do *benchmark* sobre os *scripts* criados para os procedimentos de leitura e escrita dos documentos.

A.1 Script de Leitura

A Figura 43 mostra o *script* criado para o procedimento de leitura em um escopo com 50 documentos com tamanhos distintos armazenados em um único recipiente replicado para 3 servidores CAS.

```
1  <?php
2  $cont = rand(1, 50);
3  $arq = $cont . ".pdf";
4  $ch = curl_init("http://tdinfocascluster.tce-to.tce.to.gov.br/teste/$arq
   ?domain=tdinfocascluster.tce-to.tce.to.gov.br");
5  curl_setopt($ch, CURLOPT_RETURNTRANSFER, TRUE);
6  curl_setopt($ch, CURLOPT_HEADER, TRUE);
7  echo $result = curl_exec ($ch);
8  curl_close($ch);
```

Figura 43: *Script* para Leitura de Múltiplos Documentos.

Para os testes realizados no Capítulo 5 referente ao procedimento de leitura com checagem da integridade documental tem-se o *script* criado e apresentado na Figura 44. De acordo com os testes realizados, durante o recebimento do arquivo é

efetuado a comparação entre o *hash* local do arquivo enviado pelo CAS (linha 8) e o *hash* fornecido através de um metadado gerado pelo CAS (linhas 9 à 11).

```
1  <?php
2  echo "Lendo 1.pdf...";
3  $ch = curl_init("
   http://tdinfocascluster.tce-to.tce.to.gov.br/recipiente/1.pdf?domain=tdinfocascluster.tce-to.tce.to.gov.b
   r&hashtype=sha256&hash=7eaf0d7c2c785ec0a3e38ce2a0828c242571017d4444f2d9c5f616d290e50ef1");
4  curl_setopt($ch, CURLOPT_RETURNTRANSFER, TRUE);
5  curl_setopt($ch, CURLOPT_HEADER, TRUE);
6  $result = curl_exec ($ch);
7  list($header, $body) = explode("\r\n\r\n", $result, 2);
8  $hashLocal = base64_encode(hash('md5', $body, TRUE));
9  $pos = stripos($header, "Content-MD5: ", FALSE);
10 $pos1 = stripos($header, "Content-Type", TRUE);
11 $hashCas = substr($header, $pos + 13, $pos1 - $pos - 15);
12 curl_close($ch);
13 if ($hashLocal == $hashCas) {
14     echo "Íntegro!\n";
15 } else {
16     echo "Foi modificado!\n";
17 }
```

Figura 44: *Script* de Leitura com Checagem de Integridade.

A.2 Script de Escrita

A Figura 45 mostra o *script* criado para o procedimento de escrita de 50 documentos distintos em um único recipiente de forma randômica e com a aplicação da replicação automática no momento de inserção para os demais servidores CAS.

```
1  <?php
2  mt_srand();
3  $cont = mt_rand(1, 50);
4  $arq = $cont . ".pdf";
5  $nome = md5(microtime());
6  do {
7      echo $c = file_get_contents("/var/www/bc/teste/arqs/" . $arq);
8      $ch = curl_init("http://tdinfocascluster.tce-to.tce.to.gov.br/teste/" . $nome .
       "?domain=tdinfocascluster.tce-to.tce.to.gov.br&replicate=immediate");
9      curl_setopt($ch, CURLOPT_POST, TRUE);
10     curl_setopt($ch, CURLOPT_RETURNTRANSFER, TRUE);
11     curl_setopt($ch, CURLOPT_POSTFIELDS, $c);
12     curl_setopt($ch, CURLOPT_HEADER, TRUE);
13     curl_setopt($ch, CURLOPT_HTTPHEADER, array('Content-Type: application/pdf'));
14     curl_setopt($ch, CURLOPT_FOLLOWLOCATION, TRUE);
15     $r = curl_exec ($ch);
16     $http_status = curl_getinfo($ch, CURLINFO_HTTP_CODE);
17     curl_close($ch);
18 } while($http_status == 404);
```

Figura 45: *Script* para Escrita de Múltiplos Documentos.

O procedimento de escrita com checagem de integridade documental ilustrado na Figura 46 é similar ao procedimento de leitura (Figura 44). Contudo, um valor *hash* de um arquivo é gerado localmente (linhas 7 e 8) antes do envio para o servidor CAS.

Após o recebimento deste arquivo no servidor, é gerado um valor *hash* que é comparado com o valor inicial, caso ambos sejam iguais o documento é armazenado e tido como íntegro.

```
argsforeach(range(1, 1) as $cont) {
    $nome = $cont . ".pdf";
    echo "Escrevendo $nome...";
    do {
        $c = file_get_contents("/var/www/bc/teste/" . $argsr);
        $md5 = hash('md5', $c, TRUE);
        $base64 = base64_encode($md5);
        $ch = curl_init("http://tdinfocascluster.tce-to.tce.to.gov.br/teste/" . $nome .
        "?domain=tdinfocascluster.tce-to.tce.to.gov.br&replicate=immediate&hashtype=sha256");
        curl_setopt($ch, CURLOPT_POST, TRUE);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, FALSE);
        curl_setopt($ch, CURLOPT_POSTFIELDS, $c);
        curl_setopt($ch, CURLINFO_HEADER_OUT, TRUE);
        curl_setopt($ch, CURLOPT_HEADER, TRUE);
        curl_setopt($ch, CURLOPT_HTTPHEADER, array('Content-Type: application/pdf', 'Content-MD5: '.
        $base64));
        curl_setopt($ch, CURLOPT_FOLLOWLOCATION, TRUE);
        $r = curl_exec ($ch);
        $info = curl_getinfo($ch);
        echo($info['http_code']."\n");
        curl_close($ch);
    } while($info['http_code'] == 404);}
```

Figura 46: *Script* de Escrita com Checagem de Integridade.

A.3 Sistema de Arquivos com base em Deduplicação

A Figura 47 apresenta o *script* utilizado para automação da instalação do serviço *opendedup* sobre o sistema operacional *Debian* em uma plataforma de 64 *bits*. A deduplicação de documentos em nível de bloco foi configurada para processar blocos com tamanho de 4 KB em um volume com capacidade de 190 *Gigabytes*. Além disso, foi necessário efetuar a instalação do java (*openjdk*) versão 7.

```
$wget https://opendedup.googlecode.com/files/fuse_2.9.2-opendedup_amd64.tar.gz
$tar -xvf fuse_2.9.2-opendedup_amd64.tar.gz
$cd fuse_2.9.2-opendedup_amd64/
$dpkg -i fuse_2.9.2-opendedup_amd64.deb libfuse2_2.9.2-opendedup_amd64.deb
$sudo apt-get install openjdk-7-jre-headless
$wget https://opendedup.googlecode.com/files/sdfs-2.0-beta3_amd64.deb
$sudo dpkg -i sdfs-2.0-beta3_amd64.deb
$sudo echo "* hardnofile 65535" >> /etc/security/limits.conf
$sudo echo "* soft nofile 65535" >> /etc/security/limits.conf
$sudo mkfs.sdfs --volume-name=pool0 --volume-capacity=190GB
$sudo mkdir /media/pool0
$sudo mount.sdfs pool0 /media/pool0/ &
```

Figura 47: *Script* de Instalação do *Opendedup*.