EVERALDO GOMES

HIDRA: ARQUIVAMENTO DIGITAL DE ALTA-CONFIABILIDADE UTILIZANDO AUDITORIA EM REDES PEER-TO-PEER

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre. Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, Universidade Federal do Paraná.

Orientador: Prof. Dr. Luis C. E. De Bona

CURITIBA

2010

EVERALDO GOMES

HIDRA: ARQUIVAMENTO DIGITAL DE ALTA-CONFIABILIDADE UTILIZANDO AUDITORIA EM REDES PEER-TO-PEER

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre. Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, Universidade Federal do Paraná.

Orientador: Prof. Dr. Luis C. E. De Bona

CURITIBA

2010

Sumário

Lista de Figuras

Resumo

A integridade dos documentos armazenados em dispositivos eletrônicos no longo prazo é garantida pelo arquivamento digital. Comumente, as redes Peer-to-Peer (P2P) são utilizadas nessa tarefa, pois elas permitem a construção de sistemas escaláveis e tolerantes à falhas. Além disso, técnicas de replicação e auditoria dos objetos armazenados são utilizadas na contenção das ameaças à integridade dos dados. Neste trabalho é proposto o modelo de arquivamento digital confiável Hidra, que utiliza algumas premissas inicialmente propostas por Vignatti[1][2]: o arquivamento digital é distribuído, utilizando máquinas com confiabilidades distintas; uma confiabilidade desejada, informada para cada documento, é garantida através de replicação; e através das múltiplas hashes de um documento são determinadas as máquinas que podem ser selecionadas para receber réplicas. Hidra utiliza uma nova estratégia de seleção de repositórios para a inserção de réplicas e propõe um novo procedimento de auditoria periódica responsável pela verificação da integridade e recuperação das réplicas falhas com o passar do tempo. Um protótipo não-funcional foi implementado num simulador de redes P2P e os resultados experimentais da inserção e auditoria dos documentos digitais são apresentados na avaliação do modelo.

Abstract

The long-term integrity of documents storaged in electronical devices is guaranteed by digital archiving.

Commonly, the Peer-to-Peer networks are used in this task, because they allow the design of scalable and fault-tolerant systems. Moreover, replication and audit techniques of digital objects are used in contention of threats to data integrity.

In this work, we propose a model of reliable digital archiving called Hidra, that uses some assumptions initially proposed by Vignatti[1][2]: the digital archiving is distributed, using machines with distinct reliabilities; a desired reliability is defined for each document, guaranteed by replication; and through the use of multiple hashes of a document a set of machines is selected to receive replicas.

Hidra introduces a new strategy in repository selection to the insertion of replicas, and proposes a new auditory procedure responsible for verifying the integrity and to recover damaged replicas with the past of the time. A non-functional prototype was implemented using a P2P simulator and the experimental results of the insertion and audit of the documents are presented in the evaluation of the model.

Capítulo 1

Introdução

O século XX foi marcado por uma grande evolução nas telecomunicações e na eletrônica digital. Esse progresso permanece no início do século XXI e espera-se que até 2010, somente nos EUA, os dados armazenados em sistemas comerciais e do governo ultrapassem 27 Exabytes[3]. Esse volume de informações revela uma grande dependência existente nas instituições em relação aos documentos digitais. Consequentemente, o pleno funcionamento da sociedade hodierna está diretamente relacionado à integridade desses documentos.

A preservação digital é o conjunto de atividades que garante a integridade e o acesso dos dados armazenados no longo prazo, compreendendo períodos de tempo superiores a um século[4][5]. Um dos grandes desafios dessa tarefa é a rápida obsolescência e o curto tempo de vida das mídias digitais. Por exemplo, o livro do Apocalipse pertencente a William, O Conquistador, escrito em couro no ano de 1086, durou mais de 900 anos, chegando até nós. Mas uma versão digitalizada dessa obra, gravada em 1986 numa mídia removível, não podia mais ser lida apenas 20 anos depois[6]. Dessa forma, compreende-se que o principal requisito da preservação digital é a interoperabilidade temporal: a habilidade dos sistemas atuais ou legados de interoperar com novos sistemas que utilizarão novos formatos de arquivos, modelos de dados, linguagens de programação, protocolos de comunicação, e *hardware*[7].

A acessibilidade dos documentos digitais no longo prazo é garantida com o uso de

emuladores, ou pela conversão do conteúdo do arquivo para novos formatos que venham a surgir com o tempo. Ela não faz parte do escopo deste trabalho, que possui foco na integridade dos *bits* armazenados com o passar do tempo. Denominamos como arquivamento digital essa tarefa da preservação digital responsável somente pela integridade dos dados armazenados, sem se ocupar com a acessibilidade do conteúdo.

No projeto de sistemas que realizam o arquivamento digital, uma das estratégias amplamente adotadas consiste no uso de redes P2P[8, 9, 10, 11, 12] como parte da infraestrutura para o armazenamento dos dados. As redes P2P utilizam técnicas de redundância, comumente a replicação de dados, para permitir que o conteúdo armazenado esteja disponível mesmo com a ocorrência de falhas simultâneas em inúmeros dispositivos de armazenamento.

Porém, a preservação fica comprometida se todas as réplicas de um documento forem perdidas. Uma solução possível é a definição de uma estratégia na escolha das máquinas que receberão as réplicas de um documento e de um procedimento de auditoria para a monitoração da integridade dos dados, evitando que a saída dos *peers* ou a danificação das réplicas provoquem a perda de um documento digital[1].

Alguns exemplos de sistemas P2P relacionados com a integridade dos dados são: Kosha[13], Pastis[14] CFS[15], Glacier[16], Farsite[17] e OceanStore[18]. Esses são sistemas de arquivos distribuídos, que utilizam replicação para aumentar a disponibilidade do conteúdo da rede.

Na preservação de informações culturais e científicas, no ramo de bibliotecas digitais, existem os sistemas LOCKSS[19], Sierra[20], DSpace[21] e EPrints[22]. LOCKSS e Sierra focam na preservação de periódicos científicos, utilizando mecanismos de auditoria que garantem a integridade do conteúdo armazenado com o passar do tempo. Os sistemas DSpace e EPrints não implementam auditoria e nem preservação distribuída. Eles tem como objetivo principal a organização do conteúdo armazenado.

Neste trabalho, propomos um modelo de arquivamento digital de longo prazo distribuído denominado Hidra. Esse modelo utiliza algumas premissas inicialmente propostas por Vignatti[1][2]: cada máquina, ou repositório, possui uma confiabilidade e capacidade

de armazenamento distintas; os documentos são armazenados com uma confiabilidade definida pelo usuário utilizando replicação; e múltiplas hashes são utilizadas no processo de escolha das máquinas que arquivarão as réplicas de um documento.

Essa dissertação apresenta uma nova estratégia de seleção de repositórios para o armazenamento de réplicas e um procedimento de auditoria periódica, que permite o monitoramento da confiabilidade dos documentos digitais com o passar do tempo e a geração de novas réplicas, em substituição às que foram danificadas por falhas nos repositórios. Uma avaliação do desempenho de Hidra foi realizada, com a implementação de um protótipo não-funcional no simulador de redes P2P Peersim[23]. Nesse protótipo, foram realizados experimentos que avaliaram o protocolo de seleção de repositórios e o mecanismo de auditoria.

1.1 Organização do trabalho

Esta dissertação de mestrado tem a seguinte organização: o capítulo 2 explana sobre os desafios e características dos sistemas de arquivamento digital. As redes P2P, comumente utilizadas como substrato no desenvolvimento desses sistemas são apresentadas no capítulo 3. Em seguida, o capítulo 4 apresenta Hidra: o modelo de arquivamento digital com os algoritmos desenvolvidos. Os experimentos realizados e seus resultados são expostos no capítulo 5. Finalizando a dissertação, o capítulo 6 contém as conclusões e os trabalhos futuros.

Capítulo 2

Arquivamento Digital

Uma significativa parcela dos documentos necessita ser preservada no longo prazo, devido ao valor cultural, econômico e histórico inerente a eles, ou mesmo por exigências jurídicas que obrigam a disponibilidade de alguns desses registros por longos períodos de tempo[8]. No arquivamento desses documentos, é mais vantajoso que eles sejam armazenados em meio digital, porque isso proporciona um menor custo e, concomitantemente pelas características dos dispositivos eletrônicos, numa maior velocidade no acesso às informações, comparando com o meio tradicional, utilizando papel. Neste capítulo são apresentados os principais objetivos e desafios do arquivamento digital, responsável pela integridade dos documentos em períodos de tempo superiores a um século. O capítulo se encontra dividido da seguinte forma: a seção 2.1 expõe as características do arquivamento digital; após, a seção 2.2 discorre em torno dos diferentes tipos de ameaças existentes nesses sistemas e finalizando o capítulo, a seção 2.3 mostra as principais estratégias utilizadas na preservação dos documentos.

2.1 Características do Arquivamento Digital

O arquivamento digital de longo prazo possui características, entre as aplicações de computador, que o tornam uma tarefa quase única[5]. Ele necessita ser: distribuído, porque não pode ter pontos únicos de falhas; de baixo custo, para ser viável no longo prazo e com operações que não se propagam rapidamente. Além disso, a diversidade dos

componentes de *software* e *hardware* é muito importante, porque permite a sobrevivência do sistema em caso de *bugs* no sistema operacional e aplicativos ou defeitos de fabricação em discos, processadores etc.

As operações do arquivamento digital não precisam ser finalizadas rapidamente[5]. A inserção de um documento ou uma operação de verificação de integridade pode levar horas ou dias para se propagar completamente pelo sistema. Isso permite a redução de custo do hardware, do consumo de energia e consequentemente dos gastos com o resfriamento dos dispositivos. Esse intervalo também permite a diminuição do custo de administração do sistema, aumentando a janela de tempo necessário nas intervenções do administrador para o reparo de dispositivos.

Além disso, a razão mais importante para a "indolência" no arquivamento digital é que a ocorrência de operações constantes que se propagam rapidamente pelo sistema tendem a causar um aumento no número de falhas, especialmente em ocasiões em que o sistema se encontra sob ataque. Enquanto isso, as falhas que ocorrem lentamente, com muitos avisos durante o processo, são atributos importantes no arquivamento digital, porque permitem a implementação de políticas de recuperação antes da ocorrência de uma falha total no sistema[5].

O arquivamento digital prioriza a confiabilidade e durabilidade dos dados, abdicando de alta disponibilidade no acesso. A confiabilidade é definida como a probabilidade dos dados não serem perdidos ou danificados em um determinado período de tempo[1]. A disponibilidade é a probabilidade de acesso imediato aos dados do arquivo num período de tempo, no momento em que são requisitados[17]. E, a durabilidade é definida como a probabilidade de sobrevivência de um objeto de dados específico após a ocorrência de falhas catastróficas no sistema[16].

A prática mais utilizada na preservação de documentos digitais no longo prazo é o arquivamento dos dados em discos rígidos ou mídias removíveis. A maior parte das instituições adquire esses dispositivos pelo baixo custo que apresentam. Mas suas limitações tecnológicas requerem que alguns cuidados sejam tomados.

Quando as informações digitais são arquivadas para a posteridade, a probabilidade

do usuário acessá-las novamente é infinitamente pequena[24]. Mas os discos rígidos (*hard disks* HDs) dependem de leituras frequentes nos dados para mantê-los íntegros[4]. Essas operações constantes evitam que os campos magnéticos que representam os *bits* dos dados sejam afetados com o passar do tempo e causem as falhas latentes[17]. Essas falhas acontecem quando os dados são danificados com o passar do tempo e elas são descobertas somente no momento da leitura do disco. Uma das alternativas para evitar que o baixo padrão de acesso dos usuários possa danificar os dados é a leitura periódica de todo o conteúdo do disco rígido.

Entretanto, a velocidade das operações de leitura e escrita dos HDs não cresce linearmente com o aumento de suas capacidades. A cada ano, a leitura completa de um disco é uma operação que demora mais tempo. Por exemplo, um HD com capacidade de 1 Petabyte e velocidade contínua de leitura de 600 Megabytes por segundo demoraria em torno de 20 dias para ser lido completamente[25].

Além disso, leituras excessivas desgastam os discos rígidos. Se um HD "comum", projetado para funcionar 2400 horas por ano for utilizado por 4308 horas no mesmo período, seu tempo médio entre duas falhas será de apenas três quartos do esperado[26]. Portanto, há mais vantagem na redução do tempo utilizado nas operações de leitura dos discos para evitar seu desgaste prematuro[4].

Outro aspecto a ser considerado é a impossibilidade ou inconveniência do armazenamento dos documentos digitais em um único HD. Quando a coleção de dados é maior que a capacidade do disco não é possível armazená-la em apenas uma mídia. E a permanência de todos os documentos em um único dispositivo é indesejável, porque sua falha ou destruição implica na perda de todo o conteúdo digital[5].

Outra limitação que as mídias removíveis apresentam em relação aos HDs é o armazenamento *offline* dos dados. Essa prática torna necessária a intervenção humana nas tarefas de *backup* e restauração dos dados. Porém, é desejável que o processo seja totalmente automatizado, porque a intervenção humana aumenta os custos de manutenção e eventuais falhas causadas pelas pessoas comprometem o processo de arquivamento[8].

A norma ISO 14721:2003, *Open Archival Information System (OAIS)*, define um mo-

delo de referência para a preservação digital. Esse modelo foi desenvolvido pelo *Consultative Committee on Space Data Systems* visando a preservação no longo prazo dos dados provenientes das missões espaciais e de instrumentos científicos[27]. As contribuições das comunidades de bibliotecas e arquivamento digital ajudaram a tornar o modelo abrangente o suficiente para ser aplicado em organizações que desejam garantir a preservação de suas informações no longo prazo. O modelo de referência OAIS provê uma definição em alto-nível do ambiente de um sistema de informação para arquivamento, com a definição de suas funções e a apresentação de um modelo lógico das informações armazenadas. Embora o modelo não especifique, em particular, nenhum projeto ou implementação de um sistema de arquivamento, ele provê recomendações no gerenciamento de dados para que eles estejam de acordo com padrões estabelecidos e possam ser periodicamente migrados das mídias de armazenamento e formatos originais, garantindo a sua preservação no longo prazo[7].

O modelo[27] é composto por seis entidades: ingestor, armazenamento de arquivos, gerenciamento de dados, administração, plano de preservação e acesso. O arquivamento digital compreende a entidade armazenamento de arquivos, do modelo OAIS. Essa entidade é responsável pelo armazenando dos dados no longo prazo, mas suas funcionalidades não incluem o tratamento da obsolescência de formatos, permissões de acesso e segurança, gerenciamento de meta-dados, planos de migração e outras atividades que compreendem a preservação digital e são realizadas pelas outras cinco entidades do modelo.

2.2 Ameaças ao Arquivamento Digital

A integridade dos documentos digitais pode ser comprometida por diversas ameaças. Por conta disso, o projeto de sistemas de arquivamento digital precisa considerar um conjunto de riscos. As principais ameaças ao arquivamento digital[5] são: falhas nas mídias de armazenamento, no *hardware*, no software, nos serviços de rede, organizacionais, econômicas; erros de comunicação ou do operador; ataques internos ou externos; obsolescência de mídia, *hardware* e *software* e desastres naturais.

As falhas nas mídias removíveis são provocadas pela degradação com o passar tempo.

Elas causam erros de leitura irrecuperáveis. Além disso, acidentes como quedas também danificam as mídias. Enquanto isso, as falhas de *hardware* ocorrem com a falta de energia elétrica, constituindo falhas recuperáveis de curta duração. Entretanto, com o tempo a vida útil dos discos se esgota (exaustão), acarretando em falhas irrecuperáveis.

As falhas de *software* são causadas por erros nos programas (*bugs*) que podem colocar em risco os os dados armazenados. Os erros de comunicação ocorrem porque não há garantia absoluta de que o conteúdo enviado pela rede seja entregue inalterado. Estudos sugerem que em média, de 16 milhões de pacotes de dados, um possui erro e dentre 10 bilhões de pacotes de rede, um possui erro que não pode ser detectado por *checksum*.

As falhas nos serviços de Rede, em sua maioria, são irrecuperáveis. Por exemplo, a perda de um domínio de rede, por falta de pagamento, ou a falha de uma URL persistente, por falhas no site que presta o serviço podem comprometer permanentemente os dados. A obsolescência de mídia ou de *hardware* ocorre quando, antes de falharem, uma mídia removível ou dispositivo de *hardware* se tornam incapazes de se comunicarem com outros componentes do sistema ou de serem substituídos em caso de falhas, pelo término da fabricação de peças de reposição. Há um grande histórico de mídias removíveis que não podem ser lidas pela ausência do leitor (*drive*) adequado. Exemplo: um disquete de $5\frac{1}{2}$ polegadas em funcionamento não pode ser lido porque os computadores atuais não acompanham mais drives de leitura para esse tipo de mídia.

Enquanto isso, a obsolescência de software ocorre quando os bits dos dados permanecem íntegros e acessíveis, mas a informação não pode mais ser recuperada pela ausência de software adequado. Isso é muito comum quando formatos de arquivos não padronizados são utilizados, ou mesmo com o abandono de antigos padrões sem migração com o surgimento de novos formatos.

Os erros do operador são divididos em recuperáveis e irrecuperáveis. Eles não ocorrem necessariamente no sistema de arquivamento digital. Esses erros podem ser feitos no sistema operacional em que o software executa, em outras aplicações que compartilham o mesmo ambiente, no hardware subjacente ou até mesmo na rede de comunicação.

Os desastres naturais se manifestam tipicamente através de outros tipos de ameaças:

de mídia, hardware e falhas na infra-estrutura. Os ataques externos só podem ser evitados através do isolamento do sistema à rede externa. Porém, muitas vezes os ataques ao sistema são internos e partem de pessoas que tiveram ou costumaram ter autorização de acesso a ele. Nessas situações, as medidas de isolamento não surtem efeitos.

Os custos com energia, resfriamento, rede, administração do sistema, registro de domínios, entre outros, tornam as informações digitais mais suscetíveis à interrupção do orçamento do que as informações preservadas em papel. Essas ameaçãs constituem as falhas econômicas.

As falhas organizacionais ocorrem com a falência ou mesmo mudança de objetivos das instituições que realizam o arquivamento digital. Para que o conteúdo permaneça preservado ele deve ser transferido para uma organização sucessora, ou de outra forma, descartado de forma segura.

Os gestores das organizações são os responsáveis pela contenção das falhas econômicas, organizacionais e ataques internos. As estratégias utilizadas para conter as outras ameaças são apresentadas na seção 2.3.

A Tabela 2.1 apresenta quais as características do arquivamento digital que mitigam os diversos tipos de ameaças existentes.

Tipos de falhas	Características do Arquivamento Digital			
x	Distribuído	Baixo-custo	Indolência	Diversidade
Mídia	x	x		x
Hardware	x	x		x
Software	x	x		x
Rede	x	x		
Organizacionais		x		
Econômicas	x	x		
Erros de Comunicação			x	
Erros do Operador	x		x	
Ataques Internos			x	
Ataques Externos	x		x	
Obsolescência				x
Desastres Naturais	x			

Tabela 2.1: Características do Arquivamento Digital e tipos de falhas mitigadas

2.3 Estratégias de Arquivamento

As falhas causadas por obsolescência de mídias, hardware ou software podem ser sanadas com migração, emulação e padronização dos dados, mas não fazem parte do escopo deste trabalho. O arquivamento digital tem como objetivo impedir a perda dos dados causadas pelas falhas humanas, de software e de hardware. As principais estratégias para isso são a replicação e a auditoria periódica dos dados[5].

A replicação é uma das técnicas mais simples de redundância dos dados e consiste na cópia dos dados em outros locais do espaço de armazenamento. Isso concorre para o aumento da confiabilidade e disponibilidade dos dados, da tolerância às falhas e na melhoria do desempenho do sistema[8]. Existem inúmeras técnicas de replicação: os dados podem ser replicados integralmente ou divididos em fragmentos; criptografia pode ou não ser utilizada; várias réplicas podem existir no mesmo disco ou em máquinas distintas e elas podem aceitar ou não atualizações no conteúdo.

Porém, estratégias de reparo são necessárias para suprir a perda de réplicas, que são muito prováveis de ocorrer quando os nós participantes deixam o sistema. Nos sistemas de arquivamento digital, a auditoria é o procedimento de leitura periódica dos discos que descobre e corrige as falhas latentes[17]. Esses erros são descobertos somente no momento da leitura dos dados pelo usuário. Entretanto, o padrão de acesso aos documentos é muito baixo[5], e no intervalo de tempo entre duas leituras do mesmo arquivo pode não ser mais possível recuperá-los. A auditoria permite que a falha latente seja descoberta num período menor de tempo, aumentando a probabilidade de recuperação dos arquivos.

A auditoria não pode ser utilizada de forma indiscriminada, porque as operações constantes de leitura nos discos apressam seu desgaste. Além disso, HDs de grande capacidade levam cada vez mais tempo para serem completamente auditados, já que a evolução da velocidade de leitura dos HDs não acompanha a evolução de seu aumento de capacidade[8].

Uma boa estratégia de auditoria não desgasta excessivamente os dispositivos de armazenamento e mantém alta confiabilidade dos dados do sistema. As estratégias costumam ser divididas em duas categorias: periódicas ou oportunísticas.

A auditoria periódica audita os dados num intervalo fixo de tempo, ou num intervalo

aleatório centrado num valor médio entre auditorias. A desvantagem dessa estratégia consiste na dificuldade em encontrar um intervalo de auditoria ideal que equilibre o desgaste dos discos com a confiabilidade dos documentos.

Enquanto isso, a auditoria oportunística realiza a leitura dos dados utilizando uma técnica de *piggybacking* sempre que possível. Essa estratégia aproveita ao máximo possível as outras operações de leitura do HD para evitar seu desgaste. Um tempo médio entre auditorias é definido para cada região do disco e quando ele é acessado a região que excedeu esse intervalo desde a última auditoria é imediatamente auditada. Essa estratégia não adiciona desgaste na vida útil dos discos que possuem acesso frequente. Porém, para discos com padrão de acesso infrequente aos dados, não é possível depender desse tipo de auditoria, já que o intervalo entre duas operações seguidas pode exceder o período de tempo no qual uma falha latente ocorre, corrompendo os dados. No Capítulo 4 são apresentadas as técnicas de replicação e auditoria escolhidas na implementação de Hidra, o modelo de arquivamento digital proposto.

As redes P2P, apresentadas no Capítulo 3 são amplamente utilizadas no armazenamento e compartilhamento de dados entre usuários de PCs. Técnicas de replicação são amplamente utilizadas para prover disponibilidade dos dados em face à alta probabilidade de falha das máquinas, mas nesses sistemas, usualmente, a durabilidade dos dados no longo prazo não é uma prioridade. Mas, se essas redes incluírem auditoria periódica, elas podem ser utilizadas no arquivamento digital de longo prazo.

Capítulo 3

Redes P2P

As redes P2P são sistemas distribuídos formados por redes de sobreposição (*overlay*), que usualmente executam serviços na Internet. Nas redes de sobreposição, a topologia das máquinas é diferente da topologia real dos nós. Portanto, numa rede P2P, máquinas que estão localizadas fisicamente em continentes distintos podem ser vizinhas.

As redes P2P se tornaram populares na década de 1990, em contra-posição ao paradigma de programação mais utilizado na Internet: o modelo cliente-servidor. Seu uso foi motivado pelo crescimento da capacidade de computação, armazenamento e largura de banda dos nós-folha da Internet.

No modelo cliente-servidor, os servidores são máquinas de grande capacidade encarregadas da maior parte do processamento da aplicação. Elas permanecem on-line a maior parte do tempo e são assinaladas com um endereço IP fixo. Esse endereço raramente muda nos servidores, e seu valor é obtido através da tradução realizada pelo serviço *Domain Name System* (DNS)[28]. Enquanto isso, o papel dos clientes restringe-se ao envio de requisições para o servidor e a apresentação dos resultados ao usuário. As máquinas clientes, geralmente, são os nós-folha da Internet e possuem um endereço IP dinâmico (possivelmente diferente a cada conexão), sua capacidade de processamento, armazenamento, disponibilidade e largura de banda são inferiores a dos servidores[29].

A principal limitação do modelo cliente-servidor consiste na incapacidade dos servidores atenderem todas as requisições do sistema, quando o número de clientes aumenta

muito, enquanto os nós-folha permanecem com capacidade ociosa e dependem da disponibilidade do servidor para o funcionamento pleno do sistema: quando o servidor fica *off-line*, todo o sistema para de funcionar. Enquanto isso, nas redes P2P, o modelo de computação é descentralizado: as máquinas se comportam como clientes e servidores simultaneamente. Não há pontos únicos de falhas, mas inúmeras precauções devem ser tomadas em relação à segurança do sistema e sincronia das operações.

Neste capítulo, apresentamos as redes P2P: suas características, aplicações e alguns trabalhos relacionados na área de arquivamento digital que utilizam redes P2P. O capítulo está dividido da seguinte forma: A seção 3.1 apresenta as principais características das redes P2P. As redes P2P baseadas em Tabelas Hash Distribuídas (DHTs) são expostas na seção 3.2. A Seção 3.3 discorre sobre as técnicas de replicação utilizadas nas redes P2P e finalizando o capítulo, a seção 3.4 apresenta alguns trabalhos relacionados.

3.1 Características das Redes P2P

As redes P2P são sistemas distribuídos que utilizam um modelo descentralizado. Nesses sistemas, as máquinas são chamadas *peers*[1], porque elas se comportam ora como clientes, ora como servidores. Neste trabalho, consideramos como um *peer* uma máquina *desktop* comum conectada à rede mundial de computadores e com capacidade de armazenamento comum. As tarefas do sistema são realizadas através da colaboração direta dos peers que se conectam diretamente uns com os outros. Eles podem se conectar com quantas máquinas desejarem, de acordo com suas capacidades de conexão. Trocas de mensagens são utilizadas no processo de descoberta dos *peers*.

As primeiras aplicações que tornaram as redes P2P muito populares surgiram na década de 1990. Os sistemas SETI@HOME[30] e Napster[31] foram muito populares, com a participação de centenas de milhares de usuários. SETI@HOME, proposto em 1995, iniciou seu funcionamento em 1999. Ele utiliza o tempo ocioso de desktops conectados à Internet para o processamento de dados de telescópios de radiofrequência que procuram

[1]em português, a tradução de *peer* é par, mas neste trabalho consideramos o termo ambíguo e optamos por não utilizar o termo vernacular.

por vida inteligente fora de Terra. Ele é um projeto da universidade de Berkeley, sucessor do projeto *Search for Extraterrestrial Intelligence* (SETI) , que processava seus dados utilizando super-computadores, acarretando num alto custo.

O sistema Napster, também lançado em 1999, permitia que seus usuários compartilhassem músicas, o que o tornou muito famoso no mundo. Como não havia pagamento dos usuários às gravadoras pelo compartilhamento das músicas, Napster foi alvo de uma guerra judicial com os sindicatos que representam as gravadoras de CDs[31].

Após o surgimento desses dois sistemas, inúmeras aplicações surgiram utilizando de alguma forma redes P2P. Elas possuem propósitos e arquiteturas muito diferentes entre si, mas são consideradas redes P2P[28], porque o conceito que define essas redes é abrangente. As principais características que permitem identificar um sistema como P2P[29] são: a existência de *peers*, escalabilidade, capacidade de auto-organização, tolerância à falhas e redundância de recursos.

Ao contrário do modelo cliente-servidor, o desempenho de uma P2P melhora com o aumento do número de nós participantes. Essa característica é a escalabilidade. As redes P2P também são auto-organizáveis: enquanto no paradigma centralizado a adição de novos servidores comumente obriga a reconfiguração do sistema, nas P2P a entrada e saída de nós não afetam o comportamento e nem o desempenho do sistema, descartando a necessidade de intervenção humana em sua organização. A redundância garante a tolerância à falhas. Numa rede P2P os serviços são distribuídos de forma redundante entre os *peers* e a saída de alguns deles não provoca falhas no sistema[29].

Quanto ao comportamento dos nós, uma rede P2P pode ser classificada como pura ou híbrida[29]. Ela é pura quando todos os nós agem como clientes e servidores, sem nenhum ponto central na rede. Mas nos casos em que alguns *peers* realizam mais tarefas que outros, a rede P2P é denominada híbrida. Os nós que possuem mais funcionalidades são chamados de *super-peers* e sua existência não descaracteriza o sistema como P2P. Mas, é importante lembrar que os *super-peers* constituem um ponto único de falhas no sistema.

O sistema de compartilhamento de arquivos Gnutella[32] é um exemplo de P2P pura, pois todos os nós agem como clientes e servidores. Enquanto o sistema de compartilha-

mento Napster[31], o comunicador de mensagens instantâneas e de ligações de voz sobre IP (VoIP) Skype[33], e os comunicadores MSN[34] e ICQ[35] são redes híbridas. Nesses sistemas, os *super-peers* são servidores centrais, com alta capacidade de processamento, que se comportam como diretórios de localização de *peers*.

Quanto à topologia, as redes P2P são classificadas em estruturadas e não-estruturadas[29]. Uma rede P2P não-estruturada é composta de *peers* que possuem poucas regras para participação na rede. *A priori*, a topologia da rede não é conhecida pelos *peers* no momento em que eles ingressam no sistema. Gnutella[32] é um exemplo de uma rede não-estruturada.

Ao contrário das redes P2P não-estruturadas, a topologia das P2P estruturadas é fortemente controlada e seu conteúdo não pode ser disposto em nós aleatórios, mas apenas em determinados nós, de acordo com sua topologia[29]. Assim, as trocas de mensagens tornam-se mais eficientes. Dentre algumas topologias utilizadas nas P2P estruturadas, destacam-se o hipercubo e as tabelas hash distribuídas (DHTs).

O roteamento de mensagens é uma das tarefas mais importantes nas redes P2P. Ele consiste na troca de mensagens entre *peers* para que seja possível que dois deles se descubram e estabeleçam uma conexão direta. Ele é parcialmente centralizado nas P2P híbridas e totalmente distribuído nas redes puras. Nas P2P não-estruturadas, as técnicas mais utilizadas de roteamento são a inundação (*flooding*) e caminhos aleatórios (*random-walks*). Enquanto isso, as redes estruturadas possuem mecanismos próprios de roteamento, de acordo com sua topologia.

Nas mensagens enviadas utilizando inundação, quando um *peer* deseja procurar outro, ele envia mensagens para todos os seus vizinhos, que por sua vez, repassam a mensagem recursivamente para seus próprios vizinhos até que o *peer* destinatário a receba. Quando isso acontece, uma conexão direta é estabelecida com o remetente da mensagem.

As técnicas baseadas em inundação são resilientes à entrada e saída de *peers* na rede e efetivas na localização de items com alta taxa de replicação. Porém, elas são impróprias na localização de itens raros. Essa abordagem não é escalável, porque a carga em cada *peer* cresce linearmente com o aumento do número de mensagens enviadas e o tamanho da rede[29].

No entanto, o roteamento de mensagens em redes não-estruturadas pode ser implementado com técnicas de inundação com uso de *Time to Live (TTL)* ou caminhos aleatórios (*random-walks*). Na inundação com uso de TTL, esse campo da mensagem tem seu valor decrementado cada vez que é repassado e as mensagens são descartadas quando o TTL atinge o valor zero. Nas técnicas de caminhos aleatórios, a mensagem é enviada por um caminho qualquer, sem inundar toda a rede.

Nas redes estruturadas, o roteamento é feito de acordo com a topologia da rede ou outras regras pré-estabelecidas. Na próxima seção, são apresentadas as redes P2P baseadas em DHTs.

3.2 Redes que utilizam Tabelas Hash Distribuídas (DHTs)

As redes P2P que utilizam Tabela Hash Distribuída[9, 10, 11, 12] (*Distributed Hash Tables - DHT*) são redes estruturadas que armazenam suas informações em locais definidos deterministicamente por uma função hash. Esse tipo de P2P fornece uma interface de localização de objetos às aplicações que a utilizam. Sistemas de compartilhamento de arquivos como BitTorrent[36], LimeWire[37] e e-Mule[38] utilizam DHTs. A rede de distribuição de conteúdo CORAL[39] utiliza DHTs na formação de um conjunto de *proxies web*. Esses nós permitem que um site seja acessado mesmo que ele esteja *offline*, através do acréscimo da extensão *.nyud.net* à sua URL.

Cada nó da rede armazena uma parte da tabela hash, e conseqüentemente, uma parte do conteúdo da DHT. Os *peers* também mantém uma pequena tabela de roteamento com o endereço IP e o identificador da rede (ID) de seus vizinhos. Os objetos ou valores a serem armazenados recebem um identificador único, definido pela função de *hash*, denominado chave. Nas DHTs, há somente um *peer* que corresponde e armazena a chave de um objeto.

A busca de objetos é feita com o envio de mensagens que são encaminhadas recursivamente às máquinas mais próximas da chave buscada. O processo é finalizado quando a mensagem alcança o *peer* de destino. Então, uma conexão direta é estabelecidade entre os *peers*[29]. Em teoria, todas as DHTs garantem que o custo da localização de qualquer objeto, em *hops*, é, em média, numa rede com N *peers*, da ordem do produto de uma

pequena constante por $O(log\ N)$.

A escolha da função de *hash* é essencial para o bom funcionamento da DHT. Ela deve evitar colisões e, preferencialmente, deve ser consistente[40] (*consistent hashing*). Uma função hash consistente distribui o conteúdo de maneira uniforme na DHT. Isso permite que a saída de um *peer* provoque no máximo o deslocamento de $\frac{1}{N}$ chaves do espaço de identificadores[9] [40]. SHA-1[41] é uma função de *hash* consistente muito utilizada nas DHTs.

Dentre as DHTs existentes, destacamos: Chord[9], CAN[10], Pastry[11] e Tapestry[12]. Cada uma delas possui um esquema distinto de roteamento de mensagens, de organização dos objetos de dados e do mapeamento do espaço de chaves nos *peers*.

Chord[9] não foi projetada com a finalidade específica de compartilhamento de arquivos. Seu objetivo é a alocação de dados de uma maneira mais genérica[42]. Essa DHT organiza todos os *peers* num anel. Cada peer possui uma tabela de tamanho $log\ N$, utilizada no roteamenteo de mensagens, onde cada elemento, chamado de "finger", contém o endereço de outro *peer*. Os "fingers" são escolhidos para que o sistema adquira características de *small-world*: há várias entradas referentes aos *peers* próximos (vizinhos) e poucas apontando nós distantes.

Na DHT CAN[10] (*Content Addressable Network*) os dados são organizados em vetores com D dimensões. Uma função hash distinta é utilizada em cada dimensão. As mensagens são roteadas para todos os vizinhos de um *peer* em cada uma das dimensões. A complexidade do roteamento é da ordem de $(D\ \sqrt[D]{N})$ e o custo de armazenamento é constante para cada um dos *peers*[42].

Pastry[11] utiliza roteamento baseado em prefixos. Cada *peer* recebe um identificador de 128 bits (*NodeID*) escolhido aleatoriamente quando o *peer* entra na rede. As máquinas são organizadas num espaço circular variando de 0 a $2^{128}-1$ e assume-se que a distribuição dos NodeIds é uniforme. As mensagens são roteadas para o *peer* com o NodeID mais próximo da chave desejada. Para isso, são utilizados os prefixos dos identificadores. Numa rede com N *peers*, o custo do roteamento é menor que $log_B N$ *hops* numa situação de funcionamento normal, com $B = 2^b$ (O valor de b é um parâmetro de configuração com

valor típico de 4)[29].

Tapestry[12] é uma rede que possui muitas similaridades com Pastry. Ela utiliza aleatoriedade descentralizada para obter balanceamento de carga e localidade no roteamento de mensagens. A diferença entre Pastry e Tapestry consiste no gerenciamento de localidade dos *peers* e na replicação dos objetos de dados[29].

Nas aplicações que utilizam redes P2P, a ausência de controle centralizado, a transiência e colaboração dos *peers* e o roteamento de mensagens impõem muitos desafios em seu desenvolvimento. Por exemplo, as DHTs não consideram as características heterogêneas de capacidade dos nós. Isso significa que o espaço de chaves designado a cada *peer* não representa sua capacidade real. Outro problema similar é a assertiva de que todos os objetos armazenados possuem a mesma importância. Objetos muito populares congestionam a rede e sobrecarregam seus *peers*. Essa limitação pode ser resolvida com o uso de técnicas de replicação dos objetos mais populares e sua distribuição (balanceamento de carga) nos *peers* adjacentes aos itens mais populares.

A ocorrência de *churn*, a taxa de entrada ou saída de *peers* no sistema dada uma unidade de tempo, também deve ser tratada pelas DHTs. Apesar das redes P2Ps serem auto-organizáveis, a partida brusca de várias máquinas pode desestabilizar a rede. Na próxima seção, são apresentadas algumas técnicas de replicação como solução do problema da disponibilidade dos objetos nas redes P2P.

3.3 Técnicas de Replicação nas redes P2P

Nas redes P2P, a tolerância às falhas com a saída de *peers* pode ser obtida através da redundância dos recursos. A técnica mais utilizada de redundância é a replicação dos dados. Para o arquivamento digital de longo prazo, a replicação com fragmentação não é recomendada[5], porque dificulta a recuperação dos documentos no longo prazo. Essa técnica faz uso de *erasuring code* para que um fragmento perdido possa ser recuperado com o uso da redundância presente nas outros partes[43]. No longo prazo, é recomendável que as réplicas não sejam criptografadas e nem divididas em fragmentos[19].

As técnicas de replicação podem ser divididas utilizando-se dois critérios que definem

como são feitas as atualizações das réplicas[44]. O primeiro critério é o local (*peers*) onde pode ocorrer replicação. O segundo, consiste nos momentos em que as réplicas são criadas.

De acordo com o primeiro critério, as réplicas de um objeto são classificadas como cópias primárias ou secundárias. Uma *cópia primária* é uma réplica mantida por um *peer* denominado *mestre* que aceita operações de leitura e escrita. Enquanto isso, as réplicas que apenas podem ser lidas são as *cópias secundárias*, que são mantidas por máquinas denominadas *escravos*. Quando as réplicas possuem apenas um mestre são categorizadas como *single-master*, senão são denominadas *multi-master*.

A escolha entre o uso de réplicas com um ou mais mestres influencia nos mecanismos de sincronia dos dados e no desempenho do sistema. Com a abordagem *multi-master*, várias réplicas podem sofrer atualizações (operações de escrita). Nessa abordagem, é possível que o usuário acesse uma réplica desatualizada, se a propagação de uma operação de atualização (escrita) ainda não estiver concretizada. O desempenho do sistema sofre influencia do número de operações de escrita: um sistema single-master com muitas operações de escrita torna-se sobrecarregado, pois há somente um nó que pode ser acessado. Além disso, quando o nó mestre sai da rede, deve haver um procedimento de definição de um novo mestre, entre as máquinas escravas. A escolha de um novo mestre representa um custo ao sistema.

Segundo o critério do momento em que as réplicas são criadas, a replicação é dividida em síncrona e assíncrona. A replicação é síncrona quando as atualizações das réplicas são propagadas antes do fim da transação da operação de escrita. E assíncrona quando as atualizações não são propagadas até o término da atualização dos dados..

As transações síncronas são mais seguras, pois nenhum objeto pode ser lido enquanto as atualizações não forem concluídas. Elas não são tão eficientes quanto as transações assíncronas e também são inviáveis em sistemas que necessitam da disponibilidade dos dados como requisito fundamental.

Para resolver o problema da baixa disponibilidade dos dados em transações síncronas, são utilizadas transações que são variações do modelo assíncrono. Essas operações são divididas em abordagens otimistas e não-otimistas.

As transações assíncronas otimistas supõem que os conflitos de atualizações de réplicas acontecem raramente. Elas são ideais nos sistemas com operações de escrita escassas. Enquanto isso, as transações assíncronas não-otimistas assumem que os conflitos de escrita são comuns. Elas implementam mecanismos de propagação das atualizações para prevenir eventuais conflitos.

As DHTs não são responsáveis pela replicação dos objetos. Isso é tarefa da aplicação que as utilizam. No entanto, há duas técnicas principais de replicação[43] utilizadas em DHT. Na primeira, as réplicas são criadas no caminho do objeto original utilizando os endereços guardados na tabela de roteamento. Essa técnica utiliza a noção de vizinhança. A outra, responsabiliza múltiplos *peers* para um mesmo intervalo da DHT. Uma réplica do objeto é feita em cada um desses *peers*.

3.4 Trabalhos Relacionados

A seguir, são apresentados alguns trabalhos relacionados que utilizam redes P2P para arquivamento digital, ou como espaço de armazenamento em sistemas de arquivos distribuídos: Farsite[17], Glacier[16] e CFS[15]. Um outro trabalho sobre uso de leilão de espaço em disco em redes P2P também é apresentado[45]. Finalizando o capítulo, o sistema LOCKSS é apresentado.

Farsite[17] é um sistema de arquivos seguro num ambiente não confiável. Projetado para executar em estações de trabalho de grandes corporações ou universidades, assume que a rede seja de uma escala de 10^5 *desktops*, que a capacidade de armazenamento dos discos cresce mais rapidamente que suas taxas de uso e de que com as velocidades de computação atuais é possível utilizar criptografia sem comprometer o desempenho do sistema. Dessa forma, a abundância de espaço não utilizado permite o uso de replicação para a confiabilidade dos dados e o baixo custo nas operações de criptografia permite segurança em sua distribuição pela rede. Por exemplo, a codificação e decodificação de 32Kb de dados adiciona a latência de somente 1 milisegundo nessas operações. E o computo de uma assinatura RSA com uma chave de 1024 bits leva apenas 6,5 milisegundos, um tempo menor do que a rotação dos pratos em um disco com 7200 rotações por minuto.

Os experimentos com um protótipo demonstraram que o desempenho de Farsite é relativamente semelhante a de um sistema de arquivos NFS numa rede local. Porém, o sistema não foi projetado para muitas operações concorrentes de leitura e escrita num mesmo arquivo e não há suporte para falhas correlacionadas.

Glacier[16] é um sistema de armazenamento distribuído com redundância massiva para encobrir os efeitos de correlação de falhas em larga escala. O sistema foi projetado de forma a minimizar agressivamente o custo da redundância de dados no espaço e no tempo: *Erasure coding* e coleta de lixo reduzem o custo de armazenamento; enquanto que agregação de pequenos objetos e um protocolo de manutenção flexível para fragmentos redundantes minimizam o custo das trocas de mensagens.

Na realização de um teste, Glacier atingiu durabilidade de $99,999999\%$ apesar das falhas correlatas em 60% dos nós da rede, a um custo de 11 vezes o espaço de armazenamento com *overhead*, em média, de 4 mensagens por minuto por nó, durante funcionamento normal. Glacier tem sido utilizado como a camada de armazenamento num sistema experimental de e-mails descentralizado.

CFS[15] é um sistema de armazenamento, somente-leitura, com uma arquitetura totalmente distribuída e escalável, que utiliza um sistema de armazenamento de blocos chamado DHash[46]. Esse sistema é construído sobre a DHT Chord[9]. A escalabilidade e tolerância a falhas são obtidas com técnicas de replicação. Num experimento realizado com um protótipo, os resultados mostraram que CFS é capaz de entregar dados mais rapidamente que um servidor FTP. Além disso, utilizando 4096 servidores, a procura de um dado necessita contatar apenas 7 *peers*. Os testes também mostram que mesmo com a falha de quase metade dos servidores, o serviço continua em execução praticamente ileso.

Uma técnica de leilão do espaço em disco dos *peers* para construir um sistema de arquivamento digital foi desenvolvida num outro trabalho[45]. Seu funcionamento é o seguinte: uma máquina que deseja fazer uma cópia de sua coleção anuncia quanto espaço precisa, e aceita lances de quanto do seu próprio espaço ela deve oferecer para adquirir o espaço remoto. O melhor lance é aceito e a troca realizada. Experimentos foram realizados com diferentes cenários: quando há confiança entre os *peers* e com máquinas

agindo maliciosamente. Simulações dos leilões e da troca de espaço indicam que essa técnica permite que os sítios alcancem uma confiabilidade mais alta do que em sistemas em que as máquinas trocam a mesma quantidade de espaço em disco, sem lances para aquisição de espaço em disco.

LOCKSS [19] é um Sistema de Preservação Digital, de código aberto e marca registrada da Universidade de Stanford. Seu nome é o acrônimo para *Lots Of Copies Keep Stuff Safe*, que em português significa: várias cópias mantém as coisas a salvo. Ele foi projetado como um sistema P2P para preservar o acesso a periódicos (*journals*) e às informações arquivadas, previamente publicadas na WEB.

O sistema utiliza um conjunto de princípios que garantem a preservação digital: não são utilizados segredos de longo prazo, como chaves públicas e privadas e nem a reputação de *peers* por terceiros. O princípio da inércia, é utilizado para reduzir a velocidade de propagação de mudanças no sistema e facilitar a detecção de intrusos.

LOCKSS utiliza uma rede P2P não-estruturada. A comunicação ocorre somente com *peers* confiáveis, chamados de *amigos* e que são adicionados manualmente pelo administrador, e por nós indicados pelos amigos.

O sistema utiliza a replicação dos dados para garantir o acesso ao longo prazo dos documentos digitais. Esse modelo é análogo ao utilizado pelas bibliotecas na preservação dos documentos em papel.

O protocolo de pesquisa de opinião desenvolvido para o LOCKSS é o componente responsável pela preservação dos documentos digitais. Ele é encarregado da identificação das falhas latentes e da operação de reparo nos arquivos corrompidos. Além disso, é responsável pela proteção contra roubo de informações e a ocorrência de *free-loading*.

Em um intervalo de tempo aleatório, centrado num valor definido por um parâmetro que representa o tempo provável de falhas, o protocolo de pesquisa de opinião realiza votações onde os *peers* avaliam a integridade dos objetos de dados.

Além disso, o protocolo detecta e recupera *peers* corrompidos e evita a propagação de erros do conteúdo arquivado, mesmo quando há a presença de *peers* maliciosos.

Capítulo 4

Hidra: Modelo de Replicação Confiável

Este capítulo apresenta Hidra[1], um modelo de replicação confiável para arquivamento digital de longo prazo. Hidra se baseia num modelo de replicação confiável inicialmente proposto por Vignatti[1][2], mas apresenta algumas modificações e uma nova abordagem na seleção de repositórios, com a introdução da funcionalidade de auditoria. Hidra é um modelo que pode ser implementado em qualquer tipo de rede P2P estruturada. Essa rede pode ser híbrida e não necessariamente totalmente distribuída. Neste capítulo é apresentado como implementar Hidra em uma rede P2P estruturada.

O capítulo segue dividido da seguinte forma: a seção 4.1 descreve as características que Hidra compartilha com o modelo de Vignatti. As novas estratégias desenvolvidas nesse trabalho, de seleção de repositórios e auditoria, são descritas nas seções 4.2 e 4.3. Após, a implementação de Hidra em uma rede P2P é apresentada na seção 4.4.

4.1 Descrição do Modelo

Hidra compartilha várias características com o modelo de replicação confiável proposto por Vignatti. Os dois modelos são formados por uma rede de máquinas, também

[1]O nome é uma referência à Hidra de Lerna, um ser mitológico com corpo de dragão e cabeças de serpente com capacidade de regeneração. As cabeças da Hidra representam as réplicas de um documento digital e a capacidade de regeneração é uma analogia à auditoria periódica de suas réplicas.

chamadas repositórios, independentemente administradas e com características heterogêneas, que colaboram na preservação de uma coleção de documentos no longo prazo. Esses documentos são compostos por apenas um arquivo ou uma coletânea deles, sendo que cada um possui um tamanho distinto e uma prioridade específica de preservação que é denominada confiabilidade desejada. Esse valor é o complementar da probabilidade de perda do documento com o passar do tempo e garante que documentos mais relevantes tenham menores probabilidades de perda sobre os menos relevantes.

A preservação de cada documento é garantida por meio de sua replicação nos repositórios, que podem armazenar no máximo uma réplica de cada documento. Após a inserção na rede, os documentos também são chamados de itens ou objetos. Cada objeto possui a mesma confiabilidade do repositório que o hospeda e a probabilidade de ao menos uma réplica estar íntegra em qualquer instante de tempo é definida como confiabilidade associada.

As réplicas são criadas no momento em que o documento é inserido pela primeira vez, e se necessário, durante os procedimentos de auditoria periódica. Os modelos não consideram que elas podem ser atualizadas: os documentos são arquivados em modo somente-leitura e qualquer mudança num documento transforma-o em um novo objeto na rede. Os dados não são criptografados, o documento não é fragmentado no momento da replicação e as permissões de acesso não são levadas em consideração.

Um item está corretamente preservado se, em qualquer instante, houver pelo menos uma de suas réplicas armazenada de forma íntegra. A quantidade de réplicas geradas para um objeto digital depende da confiabilidade desejada para ele e da confiabilidade disponível nos repositórios. A auditoria periódica verifica a integridade das réplicas, recuperando-as quando necessário, garantindo que a confiabilidade associada de cada item, no instante da auditoria, é maior que a confiabilidade desejada para ele.

Na inserção de réplicas de um documento digital é necessário selecionar os repositórios que receberão as réplicas. Uma boa estratégia de seleção de repositórios faz bom uso dos recursos da rede, evitando que réplicas sejam criadas sem necessidade. Essa estratégia também é responsável pela definição do intervalo de auditoria dos documentos. Também

é importante que os documentos sejam distribuídos uniformemente na rede, para que eles não fiquem concentradom em apenas alguns repositórios.

A Figura 4.1 representa Hidra: o usuário insere uma coleção de documentos num conjunto de repositórios com confiabilidades distintas. Por exemplo, se uma réplica for inserida no repositório 37 e outra no 535 (Figura 4.1) , que possuem, respectivamente, confiabilidade de 75% e 90%, a confiabilidade associada desses repositórios será igual a 97,5%. Isso porque a probabilidade dos dois repositórios falharem ao mesmo tempo é de $(1-0,75)*(1-0,9) = 0,025$ e a confiabilidade é o complementar da probabilidade de falha desses repositórios $(1-0,025 = 0,975)$. Se o documento possuísse confiabilidade desejada de 99,9%, mais réplicas deveriam ser adicionadas, até que a confiabilidade desejada fosse satisfeita.

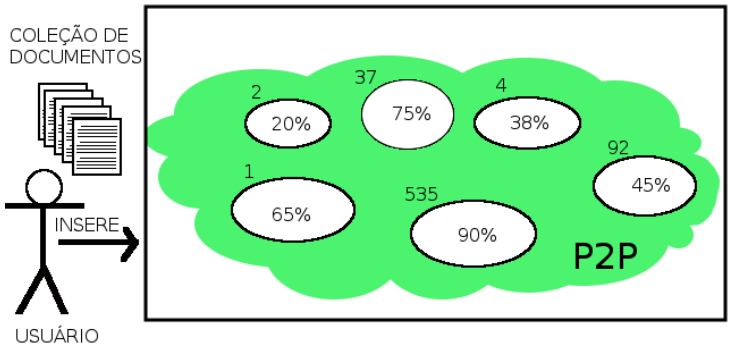

Figura 4.1: Hidra: modelo de arquivamento digital

Hidra introduz as seguintes características no modelo: (i) cada repositório possui uma capacidade distinta de auditoria; (ii) todo documento precisa manter um número mínimo de réplicas; (iii) uma nova estratégia é utilizada na seleção de repositórios e na auditoria.

Dentre as estratégias de seleção de repositórios utilizadas por Vignatti, a melhor delas, denominada de IdealSubSet, soluciona o problema da sacola (*knapsack problem*) na definição dos repositórios que serão escolhidos. IdealSubset utiliza um algoritmo de programação linear inteira, baseado em programação dinâmica, capaz de encontrar o melhor subconjunto de repositórios em tempo pseudo-polinomial. Esse critério minimiza a confiabilidade associada para que ela não seja muito maior que a confiabilidade desejada[2].

Enquanto isso, Hidra adota uma estratégia em que os repositórios são escolhidos de forma a garantir um mesmo intervalo de auditoria para todas as réplicas de um documento, através de múltiplas hashes, tornando Hidra independente do tipo de DHT utilizada. Ao contrário do modelo de Vignatti, Hidra utiliza intervalos de auditoria menores. Eles garantem mais agilidade na verificação e recuperação de réplicas falhas. Consequentemente, é possível atingir uma confiabilidade maior para os documentos gerando um número de réplicas menor que o necessário em IdealSubSet.

No trabalho de Vignatti, apesar dele ter apresentado um algoritmo de auditoria, nenhum intervalo era definido para elas, além de que as auditorias não tinham influência sobre a confiabilidade dos repositórios. Em seus experimentos, a auditoria não era realizada e consequentemente eram necessárias muitas réplicas para se atingir a confiabilidade desejada. A definição do procedimento de auditoria, e sua influência sobre a seleção e confiabilidade dos repositórios são grandes mudanças introduzidas por Hidra.

As próximas subseções descrevem detalhadamente as entidades do modelo Hidra. Primeiro são descritas as características dos repositórios e posteriormente são apresentados os itens, as réplicas com o cálculo da confiabilidade associada, a seleção de repositórios e a auditoria.

4.1.1 Repositórios

Hidra é composto por uma rede de repositórios $M = \{l_1 \dots l_m\}$ que garantem o arquivamento no longo prazo através de replicação e auditoria. Cada repositório l é responsável pelo armazenamento de um conjunto de réplicas R_l, onde toda réplica $r_j \in R_l$ corresponde a um item distinto $r_i \neq r_j \forall i, j \in R_l$. As máquinas possuem uma confiabilidade $0 < p_l < 1$, com um prazo t_l, capacidade de armazenamento $c_l \geq 0$ e um intervalo mínimo entre auditorias $0 < a_l < 1$.

A confiabilidade p_l de um repositório representa a probabilidade de arquivamento confiável (sem perdas) de suas réplicas no prazo de confiabilidade do repositório t_l. O valor de p_l depende das características de cada máquina, como a confiabilidade dos discos, a infra-estrutura de rede, o sistema operacional utilizado, aplicativos instalados e outros

fatores responsáveis pela ocorrência de falhas. Hidra considera que a confiabilidade do repositório no período de 1 ano é conhecida *a priori* e o método pelo qual ela é obtida não pertence ao escopo desse trabalho. O prazo da confiabilidade não é indefinido, pois t_l representa o tempo em que a confiabilidade é garantida, antes de sofrer degradação com o passar do tempo.

A confiabilidade de um repositório é considerada como um processo de Poisson[47], Equação 4.1. Essa equação representa a quantidade de eventos independentes passíveis de ocorrerem num determinado período de tempo. Neste caso, os eventos são as falhas irrecuperáveis nas réplicas dos repositórios e o período de tempo é o prazo da confiabilidade t. Portanto, a confiabilidade de um repositório é uma distribuição de Poisson com parâmetro λt.

$$P[K] = \frac{e^{-\lambda t}(\lambda t)^k}{k!} \tag{4.1}$$

A Equação 4.2 mostra a distribuição de Poisson para nenhuma ocorrência de falhas ($k = 0$) e prazo de confiabilidade de 1 ano $t = 1$. Essa equação é igual a confiabilidade do repositório, expressa por p, logo $p = e^{-\lambda}$.

$$P(k = 0) = e^{-\lambda} \tag{4.2}$$

Como a confiabilidade dos repositórios no período t já é conhecida *a priori*, é possível encontrar o valor de λ, isolando-o na equação: $\lambda = -\ln(p)$. Exponencial e logaritmo são operações inversas, isso significa que podemos encontrar a confiabilidade do repositório p' para qualquer período de tempo distinto t', através da Equação 4.3.

$$p'_l = p^{t'} \tag{4.3}$$

Por exemplo, uma máquina com confiabilidade de 30% em 1 ano ($p = 0,3$ *e* $t = 1$) possui uma confiabilidade de $0,3^{6/12} = 54,77\%$ em 6 meses ($t' = 6/12$) e $0,3^{7/365} = 97,71\%$ em 1 semana ($t' = 7/365$) .

A confiabilidade de um repositório num período de tempo diferente de t é chamada de

confiabilidade efetiva. A confiabilidade máxima de um repositório é $p'_l = p^{a_l}$, pois a_l é o intervalo mínimo entre duas auditorias.

Cada réplica $r_j \in R_l$ pode ter um intervalo de auditoria distinto a_r, desde que ele seja maior ou igual ao intervalo mínimo a_l. Assim, se o intervalo mínimo de um repositório com 30% de confiabilidade fosse de uma semana, então a confiabilidade máxima de uma réplica seria de 97,71%. Para isso, seriam necessárias 53 auditorias por ano. E se seu intervalo de auditoria fosse definido como 1 mês, então sua confiabilidade efetiva seria de $90,45\% = 0,3^{1/12}$, ao custo de 12 auditorias por ano.

4.1.2 Itens, réplicas e confiabilidade associada

Os documentos digitais, ou itens i, são os elementos que serão armazenados. A confiabilidade desejada é o complementar da probabilidade máxima de perda permitida para um documento e é garantida pelo arquivamento de um conjunto de réplicas.

Quando um item é inserido um conjunto de réplicas totalmente idênticas R_i é gerado. Esse conjunto tem tamanho mínimo α (alfa) e máximo ω (ômega) ($\alpha \leq |R| \leq \omega$). Se, em qualquer instante de tempo, no mínimo uma réplica desse conjunto estiver íntegra, então a confiabilidade associada A_i de R_i é maior que a confiabilidade desejada (d_i). Esse cálculo é exposto na Equação 4.4, que significa que A_i é igual a 1 menos a probabilidade de que todas as réplicas falhem ao mesmo tempo (esse valor é o complementar da probabilidade que todas as réplicas sejam íntegras). Hidra impõe a restrição do número mínimo de réplicas α para diminuir a probabilidade de perda de itens, porque mesmo a confiabilidade desejada sendo satisfeita, no caso de haver apenas uma réplica, a sua falha implicaria na perda do documento. Logo, com α réplicas essa probabilidade é muito menor.

$$A_i = 1 - \prod_{k=1}^{|R|}(1 - p_k^{t_k}) \tag{4.4}$$

4.2 Seleção de Repositórios

A estratégia de seleção de repositórios de Hidra é responsável pela escolha dos repositórios que receberão réplicas. Essa estratégia deve evitar que os repositórios de maior confiabilidade sejam sobrecarregados com muitas réplicas, que os com confiabilidades menores sejam ignorados ou mesmo que um número excessivo de réplicas seja gerado na preservação de um documento. A seleção de repositórios também define o intervalo de auditoria necessário para que a confiabilidade desejada seja garantida com o passar do tempo.

Os repositórios que receberão as réplicas de um documento não são selecionados dentre todos os repositórios da rede. Cada documento indexa um conjunto de repositórios escolhido aleatoriamente. Desse conjunto, serão definidas as máquinas que armazenarão as réplicas. A Figura 4.2 ilustra um conjunto de repositórios candidatos que foram selecionados de forma aleatoriamente uniforme entre todas as máquinas da rede. Somente as máquinas candidatas podem participar do processo de seleção de repositórios.

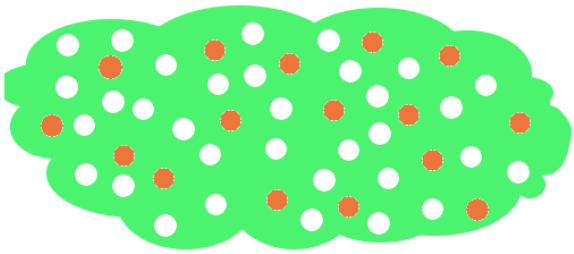

Figura 4.2: Conjunto de candidatos selecionados de forma aleatoriamente uniforme na rede

Os repositórios selecionados são denominados eleitos e: (i) possuem um tamanho mínimo α, $|E_i| \geq \alpha$; (ii) confiabilidade associada maior ou igual à confiabilidade desejada $A_i \geq d_i$; (iii) e mesmo intervalo de auditoria para todas as réplicas $t_j = t_k,\ \forall\ j, k \in |E_i|$. Os repositórios eleitos são escolhidos a partir de um subconjunto maior de repositórios candidatos, que possuem tamanho ω, $|C_i| \leq \omega$..

O Algoritmo 1 descreve passo-a-passo a estratégia adotada: primeiramente, α repositórios inicializam o conjunto de eleitos (linha 2). Se esse conjunto não superar a restrição da confiabilidade associada (linha 7) um outro repositório será escolhido (linha 8), até que seja encontrada uma solução ou até que o conjunto de eleitos atinja seu tamanho máximo, que é o número de candidatos disponíveis (linha 7).

Para satisfazer a restrição de um mesmo intervalo de auditoria para todos os eleitos, é necessário encontrar a raiz da equação de primeiro grau, com variável t (Equação 4.5). Essa igualdade utiliza a Equação 4.4, de confiabilidade associada, para encontrar um intervalo de auditoria t, igual para todos os repositórios e capaz de obter uma confiabilidade associada igual à confiabilidade desejada. Se o valor encontrado de t for maior ou igual que o intervalo mínimo entre auditorias de todos os repositórios eleitos, $t \geq a_k \; \forall \; k \in E$, isso significa que a confiabilidade associada é igual à confiabilidade desejada.

$$1 - (\prod_{j=1}^{|E|}(1 - p_j^t)) - d_i = 0 \tag{4.5}$$

Algoritmo 1: selecionarRepositorios(candidatos, d_i)

Entrada: candidatos C, confiabilidade desejada d_i
Saída: Conjunto de Eleitos E

1 $E \leftarrow \phi$;
2 **para** *j=1* to α **faça**
3 $\quad E \leftarrow E \bigcup \{l_j \in \mathsf{C}\}$;
4 numReplicas $\leftarrow \alpha$;
5 maxTam $\leftarrow$ maior($|\mathsf{C}|, \omega$);
6 t $\leftarrow$ resolva t para $(1 - (\prod_{j=1}^{|E|}(1 - p_j^t)) - d_i) = 0$;
7 **enquanto** (t $< a_k \; \forall \; k \in E$) **e** (numReplicas $<$ maxTam) **faça**
8 $\quad E \leftarrow E \bigcup \{l_{\mathsf{numReplicas}} \in \mathsf{C}\}$;
9 $\quad$ t $\leftarrow$ resolva t para $(1 - (\prod_{j=1}^{|E|}(1 - p_j^t)) - d_i) = 0$;
10 $\quad$ **para cada** l *in* E **faça**
11 $\quad\quad$ t$_l \leftarrow maior(t, a_l)$;
12 $\quad$ numReplicas $\leftarrow$ numReplicas $+ 1$;
13 **retorna** E;

Após a eleição do conjunto de repositórios eleitos realizada pelo algoritmo de seleção, as réplicas são inseridas nessas máquinas. Mas esse algoritmo também é utilizado no

procedimento de auditoria, para a migração das réplicas. A Figura 4.3 ilustra o conjunto de repositórios candidatos e o subconjunto de repositórios eleitos, ao término da seleção de repositórios.

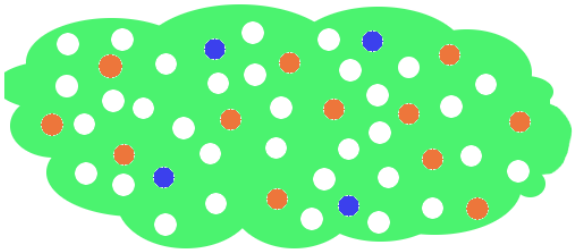

Figura 4.3: Subconjunto de repositórios eleitos entre as máquinas candidatas após a seleção de repositórios

4.3 Auditoria

A auditoria é o procedimento que garante o arquivamento digital com o passar do tempo através da criação, remoção ou migração de réplicas entre os repositórios. As restrições do arquivamento digital consistem em garantir que cada item possua o número mínimo de réplicas $|R_i| \geq \alpha$ e que a confiabilidade associada das réplicas seja maior do que a confiabilidade desejada para o item ($A_i \geq d_i$). Além disso, ela checa se todas as réplicas estão íntegras $check(k) = check(i), \forall\, k \in R_i$.

A auditoria é necessária, porque com o passar do tempo a confiabilidade dos repositórios pode mudar, suas réplicas podem ser danificadas, ou eles podem sair da rede ou falhar definitivamente. Dessa forma, a auditoria permite que novas réplicas sejam criadas ou as antigas recuperadas, de forma a atender as restrições impostas pelo modelo.

A coleção de documentos não é auditada integralmente num mesmo instante. Cada conjunto de réplicas que representam um documento digital R_i são auditados independentemente, de acordo com seu intervalo de auditoria previamente definido pelo algoritmo de seleção. Qualquer repositório que possua uma réplica de i pode invocar o procedi-

mento que a audita. A auditoria deve ser invocada sempre que o intervalo de tempo da última verificação for maior que o intervalo entre auditorias definido para as réplicas de i, $last(a_{R_i}) \geq a_{R_i}$.

O Algoritmo 2 detalha o procedimento de auditoria. Primeiramente, buscam-se todos os repositórios que contém réplicas de um objeto digital i (linha 1). Eles formam o conjunto V (máquinas que foram eleitas previamente pela inserção ou numa auditoria anterior). Após, a integridade das réplicas deve ser conferida, e as danificadas excluídas (linha 2). Posteriormente, são definidos os repositórios candidatos para uma nova eleição. Isso é muito semelhante ao processo de inserção. Na inserção, os candidatos são formados por um conjunto aleatório de repositórios. A diferença é que na auditoria as máquinas com disco cheio podem participar, desde que possuam uma réplica do item (linha 3), porque elas não precisam de espaço em disco, já que nenhum item será inserido. Os novos repositórios são escolhidos pelo algoritmo de seleção de repositórios, com os candidatos definidos no passo anterior.

As K máquinas que foram reeleitas são aquelas que formam a interseção do conjunto V, de eleitas anteriores, com o conjunto de eleitas recentemente (linha 5). Os nós que terão suas réplicas removidas são aqueles que não foram reeleitos, formando o conjunto T. Eles podem ser obtidos pela subtração de V de K (linha 6).

Após esses cálculos, novas réplicas são inseridas nos repositórios eleitos que ainda não possuem réplicas (linha 8) e removidas dos repositórios que não foram reeleitos (linha 10).

4.4 Implementação de Hidra

Hidra é um modelo de arquivamento confiável flexível que pode ser implementado como um sistema distribuído utilizando-se diversas arquiteturas. Mas para facilitar o entendimento, independentemente da arquitetura utilizada o sistema também é chamado de Hidra. Nesta seção apresentamos uma implementação de Hidra utilizando múltiplas hashes e uma rede P2P.

As próximas subseções descrevem o mecanismo de múltiplas hashes e as operações de obtenção e inserção de itens, necessárias na implementação de Hidra. Finalizando o

Algoritmo 2: Auditoria(Item i)

input: *item* com confiabilidade desejada d_i
1 V ← Conjunto de todos os repositórios que contém réplicas de i;
2 V ← check(V); `/* Cheque a integridade de todas as réplicas */`
3 CA ← conjunto de repositórios candidatos ao arquivamento de réplicas de i ou repositórios que já contenham réplicas de i;
4 eleitos ← selecionarRepositorios(CA, d_i);
5 K ← eleitos ∩ V ; `/* Repositórios que manterão suas réplicas */`
6 T ← V − K ; `/* Repositórios que perderão suas réplicas */`
7 **para cada** l *in* eleitos **faça**
8 ⌊ insira(i) em l ou atualize (i) se i ∈ l;
9 **para cada** l *in* T **faça**
10 ⌊ remova(i) de l;

capítulo, são descritos os detalhes da implementação do critério de seleção de repositórios e da auditoria.

4.4.1 Múltiplas hashes

Múltiplas funções hash são utilizadas na implementação do algoritmo de seleção de repositórios candidatos ao arquivamento de réplicas. Essa estratégia também é a mesma utilizada por Vignatti em sua implementação. Ela consiste na seleção dos n repositórios que correspondem à cada uma das h_n funções de hash utilizadas para selecionar cada repositório. Para evitar a escolha de n funções de hash distintas com características de *consistent hash*, somente SHA-1 é utilizado e cada uma das funções é obtida com o acréscimo de um "salt" ao identificador único do objeto: um número de 1 a ω (ômega) é utilizado como "salt". Por exemplo, se a chave de um documento for a string *foo*, então $h(foo1), h(foo2), \ldots, h(foo\omega)$ são as ω hash desse objeto. Dessa forma, obtém-se as h_n funções hashes necessárias para selecionar n repositórios.

As múltiplas funções hash permitem selecionar de forma aleatoriamente uniforme um subconjunto de repositórios da rede. Dessa forma, o algoritmo de seleção de repositórios não precisa escolher dentre todos os repositórios quais aqueles que podem receber réplicas. Isso aumenta seu desempenho computacional, além de espalhar os documentos pela rede. A Figura 4.4 ilustra as múltiplas hashes do identificador "mobydick", com o uso de

SHA-1, supondo uma rede com 512 repositórios com os identificadores dos repositórios uniformemente distribuídos.

identificador do documento: mobydick

Hashes	ID do Repositório
1ce4976de7c81722200fa139a85d628c76095d11	57
55a044c32e3c6a75a7d562da8ddf1074e0d9e77a	171
699f36c4dc7f5b7c0b0d2e5a68becac8558be819	211
411289e1929eeb160dd51bfb310397255af2654b	130
7461c5861e067ead11fc2d977e0f077b4fed1ab	14
b784e1935fbeffa26b7d39a7b78bb9547790be15	367
644aee8a9f964fc66c03b294beddf12fa38d6d35	200
cff885b420ad17179b7f129bab7ea4ff98216329	415
27d577c41ed37a32b55071635515d5fe5cfc4c30	79
612e3183615b67776cc751f7c42c7a5a66bc4bc1	194
666702518277756e86313603aebf996a02bbbcf2	204
c8659a15042dcc04e6512762005ef3846858ca16	400
969f92aaa9f821d8daaeb71ea52a1d153fcae53f	301
f6d8282f3c9a665179efa3797d35328ef34fe32a	493
1d5ca44ed95e209a2a76b53989f3bfeb41229766	58
7eab126545865a693b05df7cf6305d870fd8c20	143

Figura 4.4: Exemplos das múltiplas hashes SHA-1 do identificador “mobydick” e os repositórios correspondentes numa rede com 512 máquinas

4.4.2 Rede P2P

Na implementação de Hidra proposta neste trabalho foi utilizada uma DHT. As DHTs são uma categoria de rede P2P estruturadas, mas qualquer rede dessa categoria pode ser utilizada em Hidra, desde que todos os *peers* possuam identificadores únicos. Esses identificadores, ou endereços, são utilizados para o mapeamento dos repositórios que correspondem às múltiplas hashes dos documentos.

Todas as operações de Hidra: a descoberta dos candidatos, a seleção de repositórios, a inserção e auditoria de itens utilizam uma única operação da rede P2P denominada lookup(id). A função lookup() é responsável por retornar o endereço do repositório responsável pelo armazenamento do identificador do documento (id). Ela é a interface das redes P2P com Hidra.

A inserção de documentos, expressa no Algoritmo 3, é responsável pela introdução de documentos digitais em Hidra. Ela pode ocorrer em qualquer instante de tempo por qualquer repositório, mas depois que um item é inserido, ele deve ser arquivado por tempo indeterminado. A inserção também garante que a confiabilidade mínima desejada para os documentos seja respeitada, através da replicação. Além disso, ela é responsável

pela definição do primeiro intervalo de auditoria (através do algoritmo de seleção de repositórios).

O Algoritmo 3 é dividido em três passos: (i) primeiro, obtém-se todos os repositórios capazes de armazenar réplicas do item que será inserido (linha 1). Essas máquinas são chamadas de candidatos e são todos os repositórios correspondentes às múltiplas hashes do item e que possuem espaço em disco suficiente para armazenar uma de suas réplicas. O número máximo de candidatos é o parâmetro ω; (ii) após, dentre os candidatos, um subconjunto é escolhido para armazenar as réplicas (linha 2 e Algoritmo 1). Esses repositórios são denominados eleitos. Eles têm tamanho mínimo α e possuem uma confiabilidade associada maior que a confiabilidade desejada para o documento; (iii) finalmente, após a escolha, uma réplica é inserida em cada repositório selecionado (linha 4).

Algoritmo 3: insira(item)

entrada: *item* com confiabilidade desejada d_i

1 candidatos ← getAll(i);

2 eleitos ← selecionarRepositorios(candidatos, d_i);

3 **para cada** l *em* eleitos **faça**

4 insira uma replica do item em l;

A operação getAll, descrita no Algoritmo 4, retorna todos os repositórios que correspondem às múltiplas hashes da chave de um documento digital, também chamados de candidatos. Essa operação é utilizada pelo algoritmo de inserção, de seleção de repositórios e de auditoria.

Para encontrar um repositório que corresponde à hash derivada de um documento, a função lookup(id) de uma P2P estruturada é utilizada.

Algoritmo 4: getAll(id)

Entrada: identificador do item *id*

Saída: Repositórios que correspondem às múltiplas hashes de um item

```
1 repositórios ← φ ;
2 para j=1 to ω faça
3     repositórios ← repositórios ⋃ lookup(h_j(i)) ;
4 retorna repositórios
```

A operação de obtenção de itens, ou get(id) é utilizada para recuperar um objeto digital previamente armazenado. O parâmetro *id* representa o identificador único de um documento digital. Todos os repositórios podem invocar a função *get*(*id*) para acessar qualquer documento armazenado na rede.

O procedimento é descrito no Algoritmo 5 e está dividido nos seguintes passos: (i) primeiro, obtém-se todos os repositórios com chaves derivadas de id (linha 1); (ii) desses repositórios, verifique quais possuem réplicas de *id* (linha 4); (iii) escolha de maneira aleatoriamente uniforme um desses repositórios e retorne a réplica (linha 6).

Algoritmo 5: get(id)

Entrada: identificador do item *id*

Saída: uma réplica do item

```
1 candidatos ← getAll(i) ;
2 R ← φ;
3 para cada l in candidatos faça
4     if l possui id then //contém réplica
5         R ← R ⋃ l;
6 retorna escolha uniformemente qualquer l ∈ R e retorne uma réplica de id;
```

4.4.3 Critério para seleção de repositórios

Na seleção de repositórios de Hidra é necessário escolher um conjunto de repositórios eleitos que receberão réplicas. No Algoritmo 1, foi definido que esse conjunto inicia com

α repositórios, e até que ele satisfaça a confiabilidade desejada do item a ser armazenado, mais uma máquina deve ser adicionada ao conjunto. No entanto, o algoritmo de seleção de repositórios não define nenhum critério na escolha desses repositórios, indicando que pode ser uma escolha aleatória.

Na implementação de Hidra proposta um critério foi desenvolvido para a seleção de repositórios. Nesse critério, os repositórios com maior espaço livre em disco são selecionados. Apesar de não garantir o balanceamento da rede, a inserção de réplicas em repositórios com maior espaço livre em disco permite utilizar racionalmente os recursos da rede.

4.4.4 Auditoria

O algoritmo de auditoria de Hidra define que todas as réplicas danificadas devem ser removidas e que uma nova seleção de repositórios deve ser efetuada para a definição de um novo intervalo de auditoria. Os repositórios que forem escolhidos novamente (reeleitos) mantém suas réplicas, enquanto que os outros devem migrar suas réplicas para os novos repositórios escolhidos.

Na implementação de Hidra, ao invés de verificar a integridade das réplicas, a função *check*() assume que todas as réplicas de um repositório que não sofreu falhas estão íntegras. Essa assertiva parte do princípio que a confiabilidade do repositório é a probabilidade que seus documentos estejam íntegros. Logo, se a máquina não sofreu falhas a confiabilidade garante que suas réplicas estão íntegras.

No Algoritmo 2, exposto na seção 4.3, foi definido que o conjunto V contém os repositórios que já possuem réplicas e o conjunto CA é formado pelos repositórios candidatos à auditoria. Esses dois conjuntos podem ser obtidos pela operação $getAll(id)$ que foi implementada com o uso de múltiplas hashes. A função *getAll* retorna todos os repositórios que correspondem ao identificador de um documento. Iterando sobre esse conjunto, todos os repositórios que possuem uma réplica do item devem entrar no conjunto V.

Enquanto isso, os candidatos à auditoria são todos os candidatos que possuem espaço em disco para armazenar o documento, ou os repositórios que não possuem espaço em disco, mas contém réplicas. O Algoritmo 6 apresenta a auditoria utilizando as operações

getAll e *insira*, construídas sobre a rede P2P.

Algoritmo 6: Implementação da Auditoria utilizando as operações das DHTs

```
   input: item com confiabilidade desejada d_i
 1 Candidatos ← getAll(item);
 2 para cada p em Candidatos faça
 3     se p possui réplica de item então
 4         se check(r_p) então
 5             V ← V ⋃ p
 6         CA ← CA ⋃ p
 7     senão se p possui espaço em disco para item então
 8         CA ← CA ⋃ p
 9 eleitos ← selecionarRepositorios(CA, d_i);
10 K ← eleitos ∩ V ;            /* Repositórios que manterão suas réplicas */
11 T ← V − K ;                  /* Repositórios que perderão suas réplicas */
12 para cada l in eleitos faça
13     insira(i) em l
14 para cada l in T faça
15     remova(i) de l;
```

Capítulo 5

Resultados Experimentais

A avaliação de Hidra, o modelo de replicação confiável exposto no capítulo anterior, foi realizada com a implementação de um protótipo não-funcional no simulador P2P Peersim[23]. Essa avaliação visa a análise do desempenho do modelo e a comparação com o modelo proposto por Vignatti[2].

O capítulo foi estruturado da seguinte forma: a seção 5.1 descreve o simulador Peersim e os parâmetros utilizados nos experimentos. Os experimentos de inserção são expostos na seção 5.2 e os de auditoria na seção 5.3.

5.1 O simulador Peersim

Peersim[23] é um simulador de redes P2P, implementado com a linguagem de programação Java[48], altamente escalável e com suporte ao dinamismo da entrada e saída de nós, inerente às redes P2P. Dessa forma, é possível a realização de testes de novos protocolos com a capacidade de controle do ambiente de simulação. Assim, não é necessário que centenas ou milhares de máquinas sejam adquiridas no teste de novos protocolos em redes P2P.

O modelo de simulação no Peersim mais simples permite que os protocolos sejam testados com centenas ou milhares de nós, mas ele não permite a simulação da camada de transporte e nem concorrência nas operações da rede. Essa característica não afeta o teste de Hidra, porque uma das características desejadas para o arquivamento digital é a

indolência das operações.

Duas versões foram implementadas no simulador Peersim. Na primeira versão, apenas as operações de inserção existiam. Após, numa nova versão, a funcionalidade de auditoria foi incluída no protocolo de Hidra. O modelo implementado não considera a variação da confiabilidade com o passar do tempo. Apesar dele suportar isso, a variação não foi implementada para simplificar a simulação. Assume-se que são feitas manutenções periódicas nos repositórios e o valor de p é estático, a menos que sofra mudanças explícitas por parte do administrador do repositório. Por padrão, o prazo da confiabilidade t é de 1 ano, para fins de cálculo da confiabilidade efetiva utilizando o processo de Poisson.

O procotolo de auditoria implementado também não verifica a integridade das réplicas, porque a confiabilidade dos repositórios pressupõe que elas estão armazenadas corretamente. Assim, quando os repositórios falham todas as suas réplicas são descartadas, pois são consideradas inválidas. O método utilizado para o cálculo do mesmo intervalo de auditoria nas réplicas foi o da bisseção, que permite encontrar a raiz de uma equação. A margem de erro estabelecida foi de 10^{-11}.

A configuração básica da rede consistia de 2048 repositórios, cada um com capacidade de armazenamento de 4000 itens. Cada item tem o mesmo tamanho e o tamanho do disco não é expresso em *bytes*, mas nessa unidade de medida virtual expressa por quantos itens é possível armazenar. O intervalo mínimo entre auditorias foi definido como semanal, com uma quantidade mínima de 4 réplicas e máximo de 16 para cada documento digital, com o uso de 16 hashes na seleção de repositórios. A confiabilidade mínima desejada para cada item variou uniformemente de 4 noves (99,9999%) ao valor máximo de 7 noves (99,9999999%), com a confiabilidade dos repositórios variando uniformemente de 10% até 90%, no período t de 1 ano. Os valores-padrão dos parâmetros utilizados nas simulações estão presentes na Tabela 5.1.

A escolha dos parâmetros obedeceu a limitação de memória do ambiente onde as simulações foram executadas. Assim, quando foi necessário aumentar o número de repositórios

Experimento padrão		
Parâmetro		Valor padrão
Tamanho da rede	$\|M\|$	2048
Confiabilidade (repositório)	p_l	uniforme entre 10 a 90%
Confiabilidade desejada (documento)	d_i	99,9999% a 99,9999999%
Espaço em disco	c_l	4000
Tamanho de um item	s_i	1
Intervalo mínimo entre auditorias	a_l	SEMANAL
Número mínimo de réplicas	α	4
Número máximo de hashes	ω	16

Tabela 5.1: Experimento Padrão

para avaliar a escalabilidade do protocolo, foi necessário diminuir sua capacidade de armazenamento. O ambiente possuía 3GB de memória RAM e a memória máxima utilizada pela máquina virtual Java foi de aproximadamente 2650MBs.

5.2 Experimentos de Inserção

Os experimentos executados na primeira versão de Hidra foram de inserção de objetos digitais. Os documentos foram gerados de forma que cada um possuísse um identificador único, confiabilidade desejada e tamanho em disco de acordo com os limites estabelecidos pelos parâmetros de configuração. Cada experimento foi executado 5 vezes e os valores médios obtidos utilizados na composição dos gráficos.

Em cada experimento, houve a variação de um parâmetro, para que fosse possível analisar a sua importância no desempenho do sistema. O número de repositórios utilizados vairou entre 32 (2^5) a 131072 (2^{17}), enquanto que seu espaço em disco variou de 1 a 4000. A confiabilidade dos repositórios variou de 10 a 90% e o número de hashes utilizadas na seleção de repositórios foi de 4 até 20. Também foi realizada uma comparação entre o algoritmo utilizado pelo Hidra e o IdealSubSet proposto por Vignatti.

Os documentos foram gerados e inseridos na rede até o preenchimento de toda a capacidade de armazenamento de Hidra. Os gráficos a seguir analisam o espaço utilizado do total presente na rede, o total de réplicas inseridas e de documentos únicos presentes e comparam o desempenho entre Hidra e IdealSubset.

5.2.1 Espaço da rede utilizado

Em Hidra, o espaço de armazenamento é a soma da capacidade de armazenamento de todos os repositórios ativos. Analogamente, o espaço de armazenamento utilizado é a soma do espaço utilizado em todas as máquinas. Enquanto isso, o espaço mínimo utilizado representa o menor espaço em disco utilizado por algum repositório da rede. Na avaliação do espaço de armazenamento utilizado foram executados quatro experimentos com a variação da confiabilidade dos repositórios, do número de hashes, do número de repositórios e do espaço em disco, expressos no eixo x e o espaço de armazenamento utilizado representado pelo eixo y. Após, um último experimento compara o desempenho de Hidra com o algoritmo IdealSubSet proposto por Vignatti[1].

Nos experimentos, o espaço de armazenamento utilizado foi expresso em porcentagem, porque entre os diferentes experimentos foram utilizadas capacidades totais distintas. Mas, se considerarmos que cada item tem um tamanho médio de 500MB, um disco com capacidade de armazenamento de 4000 itens teria 2 TB de espaço. Assim, uma rede P2P com 2048 repositórios possui uma capacidade total de 4 PetaBytes, considerando que todos as máquinas estejam ativas.

No primeiro experimento, representado na Figura 5.1, foi testada a variação da confiabilidade dos repositórios. Em cada execução, todos os repositórios possuíam a mesma confiabilidade que variou entre 10% a 90% (eixo x). O gráfico mostra que repositórios com qualquer confiabilidade são capazes de utilizar mais de 90% do espaço de armazenamento (eixo y). Além disso, não existem diferenças significativas entre o percentual total da rede comparadas ao percentual do disco menos utilizado no sistema. Isso mostra que Hidra é capaz de utilizar bem o espaço de armazenamento.

O segundo experimento avaliou a variação do número máximo de hashes, de 4 a 20, utilizadas na escolha dos repositórios . O resultado é apresentado na Figura 5.2. Esse experimento foi o que mostrou maior variação nos valores, indicando que o número de hashes utilizadas é um parâmetro importante na inserção de objetos digitais. Quanto mais hashes, mais repositórios podem ser escolhidos, permitindo que boas escolhas sejam feitas. Com poucas hashes, não há muitas opções de repositórios, coibindo o balancea-

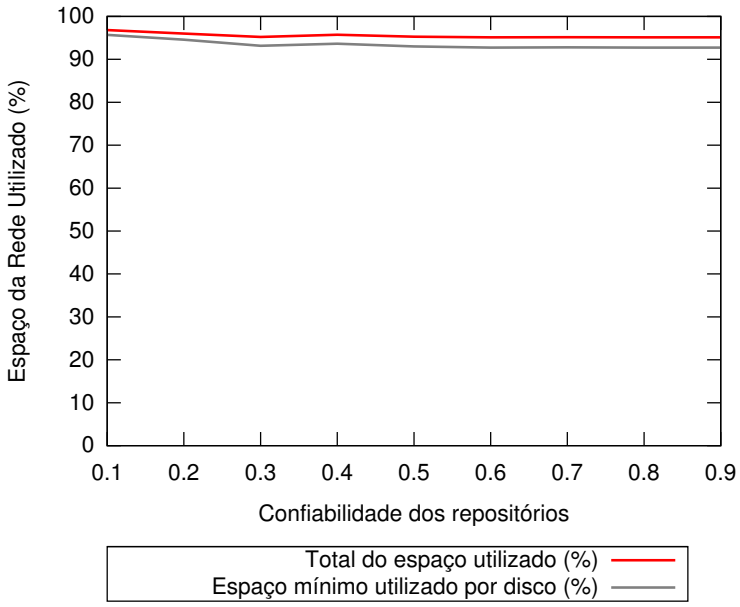

Figura 5.1: Espaço da rede utilizado com a variação da confiabilidade dos repositórios

mento do espaço utilizado na rede. É possível notar que a partir de 6 e 10 hashes, há ganhos significativos no espaço utilizado, mas que com 15 hashes o ganho deixa de ser significativo. Além disso, o uso de muitas funções hash apresenta desvantagem no desempenho dos algoritmos. Porque quanto mais repositórios forem utilizados, mais tempo levará a sincronização deles no desempenho das funções de inserção, auditoria e obtenção de repositórios.

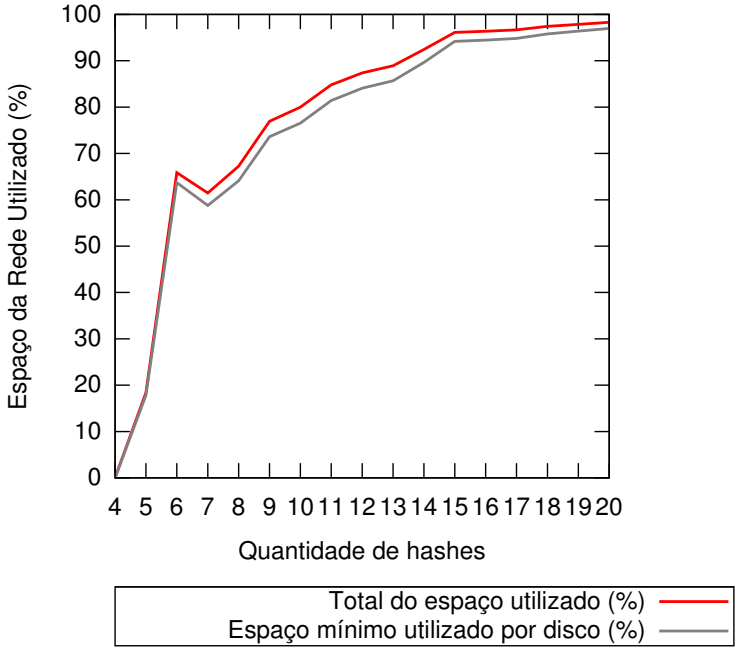

Figura 5.2: Espaço da rede utilizado com a variação do número de hashes

O terceiro experimento avalia a a variação no número de *peers* (tamanho da rede). O tamanho mínimo utilizado foi de 32 (2^5) repositórios e o máximo de 131072 (2^{17}). Na Figura 5.3 são expressos os resultados: o número de repositórios no eixo x e o espaço de armazenamento no eixo y. A partir de 64 repositórios já é possível utilizar 90% do espaço de armazenamento, mas mesmo assim, é possível observar que a variação do número de *peers* não altera drasticamente a capacidade e o balanceamento da rede. O algoritmo de seleção de repositórios provou ser escalável, mantendo um bom desempenho mesmo com uma diferença de 4096 vezes no tamanho na rede.

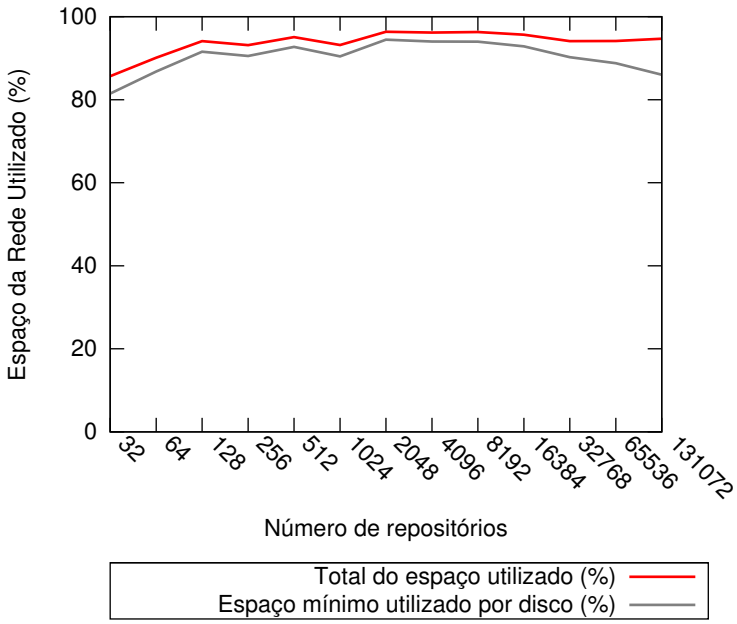

Figura 5.3: Espaço da rede utilizado com a variação do número de repositórios

A Figura 5.4, mostra a variação do espaço dos discos das máquinas. A partir de 125 unidades de espaço em disco o algoritmo estabiliza. Esse gráfico demonstra que Hidra não necessita de máquinas com grande capacidade, porque 125 itens representariam em torno de 60GB, sendo que atualmente, um disco de 1TB pode ser adquirido por menos de 300 reais. Isso demonstra que um disco rígido comum é suficiente para o arquivamento digital com Hidra.

Esses experimentos que mostraram o espaço de armazenamento utilizado com a variação da confiabilidade dos repositórios, do número máximo de hashes utilizadas, do número de repositórios da rede e do espaço em disco permitem concluir que Hidra apresentou um

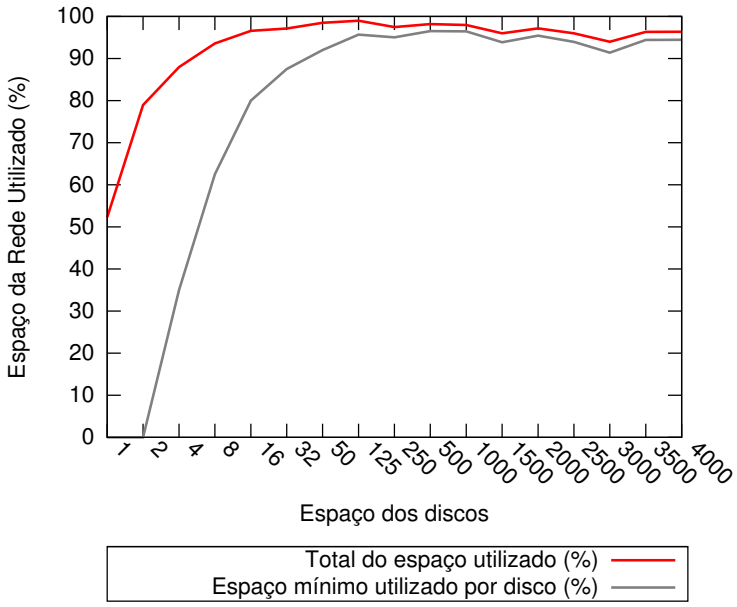

Figura 5.4: Espaço da rede utilizado com a variação do tamanho dos discos

bom desempenho na inserção de documentos digitais. Os maiores limitadores do algoritmo de seleção de repositórios foram o número de hashes e o espaço em disco dos repositórios. Para atingir um bom desempenho, ocupando mais de 90% da rede, é necessário utilizar 16 hashes e um espaço em disco mínimo para armazenar 32 itens. O tamanho da rede e a confiabilidade dos repositórios influenciam pouco o desempenho do algoritmo de seleção: porque os repositórios são selecionados de maneira uniforme pelas funções hash e a baixa confiabilidade das máquinas é contornada com mais réplicas e um intervalo de auditoria menor.

5.2.2 Total de itens e réplicas

Os experimentos anteriores mostraram que Hidra consegue utilizar bem o espaço de armazenamento, atingindo taxas superiores a 90%. No entanto, somente essa informação não é capaz de avaliar a capacidade de inserção de Hidra, porque não é possível saber quantos documentos únicos foram inseridos e qual é a quantidade de réplicas. Para avaliar esse quesito foi realizado um experimento que mediu quantos itens e réplicas foram inseridos na rede com a variação da confiabilidade dos repositórios.

A quantidade total de réplicas, expressa na Figura 5.5, mostra que independentemente

da confiabilidade dos repositórios, é possível preencher quase toda a capacidade da rede. Mas a quantidade de documentos únicos, apresentadas na Figura 5.6, são diferentes com a variação da confiabilidade dos repositórios. Isso significa que com menos confiabilidade mais réplicas precisam ser inseridas. Mesmo assim, se dividirmos o total de réplicas pelo número de documentos, é possível observar que Hidra não cria muitas réplicas: com 10% de confiabilidade o número médio de réplicas por documento é de 5,155197 e com 90% é de 4,000006. Sendo que o número mínimo estabelecido (α) nos experimentos foi 4.

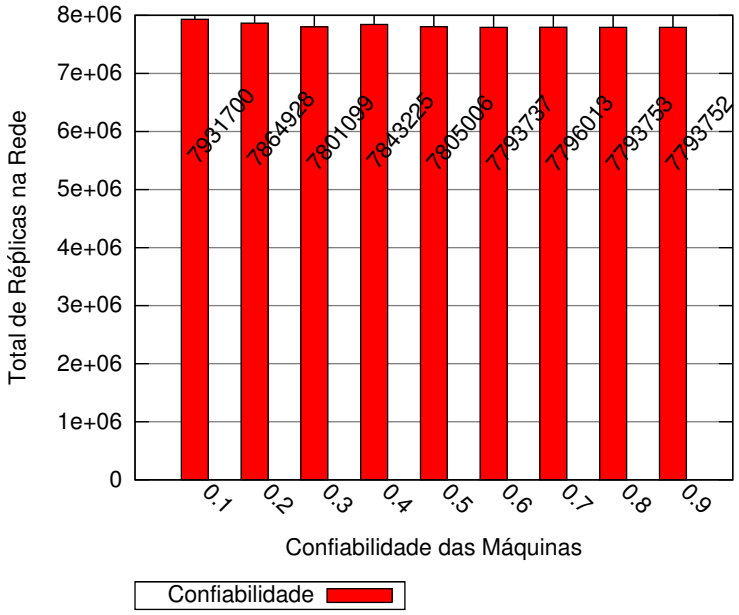

Figura 5.5: Total de réplicas inseridas com a variação da confiabilidade dos repositórios

Além disso, a partir de 60% de confiabilidade nos repositórios não são inseridos mais objetos na Rede, sendo que com 50% de confiabilidade somente 1 item foi inserido a menos. Essas informações também estão presentes na Figura 5.6. Dessa forma, é possível concluir que a confiabilidade dos documentos é garantida pelo intervalo de auditoria e a confiabilidade dos repositórios não tem muita relevância na quantidade de documentos inseridos quando o limiar de 50% é ultrapassado.

Esses experimentos demonstraram que Hidra é capaz de inserir documentos digitais utilizando quase toda a capacidade da rede e mesmo com grande variação na confiabilidade dos repositórios, o número médio de réplicas variou em pouco mais de uma unidade quando a menor confiabilidade, de 10%, foi utilizada.

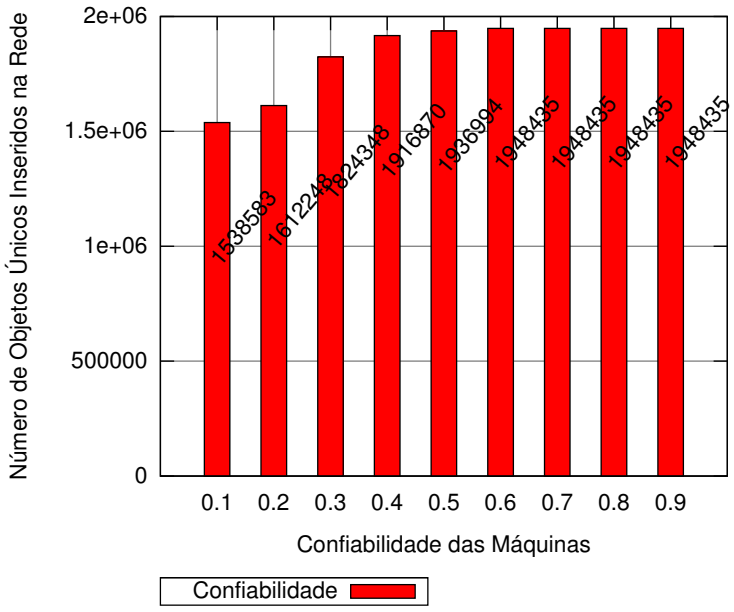

Figura 5.6: Total de itens inseridos com a variação da confiablidade dos repositórios

5.2.3 Comparação do desempenho de Hidra e IdealSubset

Para comparar o desempenho de Hidra e IdealSubset na inserção de itens, foram realizados três experimentos que mediram: o espaço de armazenamento utilizado, o total de itens inseridos e o número médio de réplicas.

No primeiro experimento Hidra foi comparado com o algoritmo IdealSubset utilizando 32 e 64 hashes. Hidra utiliza somente 16 hashes, mas com 16 Hashes IdealSubSet não conseguiu inserir itens, porque esse algoritmo não utiliza a confiabilidade efetiva dos repositórios: a confiabilidade associada é obtida pelo valor da confiabilidade p no período t. Nesse experimento, expresso na Figura 5.7, foram comparados o espaço das redes utilizados nos dois algoritmos: itens são inseridos, até que chegue um momento em que um item não pode ser inserido, porque o subconjunto escolhido não satisfaz sua confiabilidade desejada. IdealSubset com 64 hashes, alcançou em torno de 50% da rede e IdealSubset com 32 hashes não atingiu nem 10%. Enquanto isso, Hidra ultrapassou 90% do espaço de armazenamento.

O espaço de armazenamento utilizado não é capaz de fornecer informações sobre o número de itens inseridos. Então, no segundo experimento, expresso na Figura 5.8, esses valores são apresentados. Novamente Hidra apresenta um desempenho superior, pois

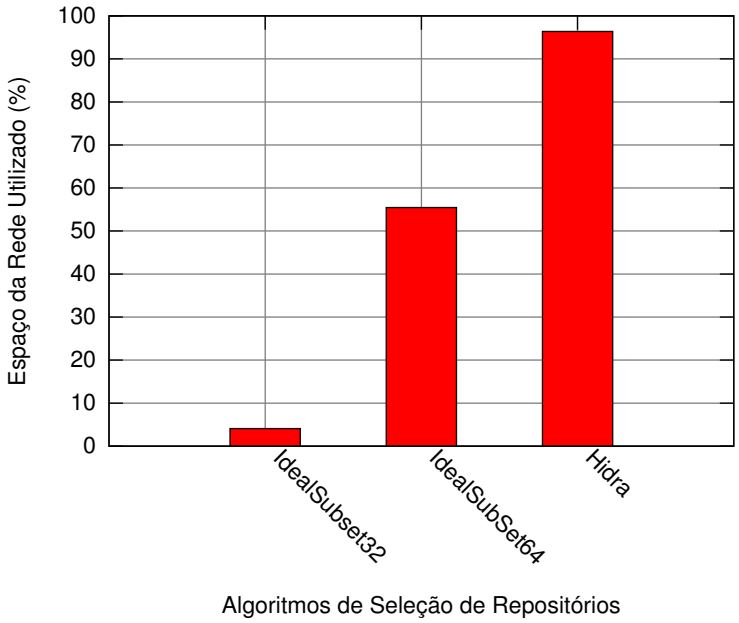

Figura 5.7: Comparativo do espaço de armazenamento utilizado entre os algoritmos Hidra e IdealSubSet

IdealSubSet com 32 hashes insere apenas 20741 itens e com 64 hashes chega a 264756, um valor 10 vezes maior. Porém, Hidra insere 1944193, um númeo de itens mais de 7 vezes maior que IdealSubSet com 64 hashes..

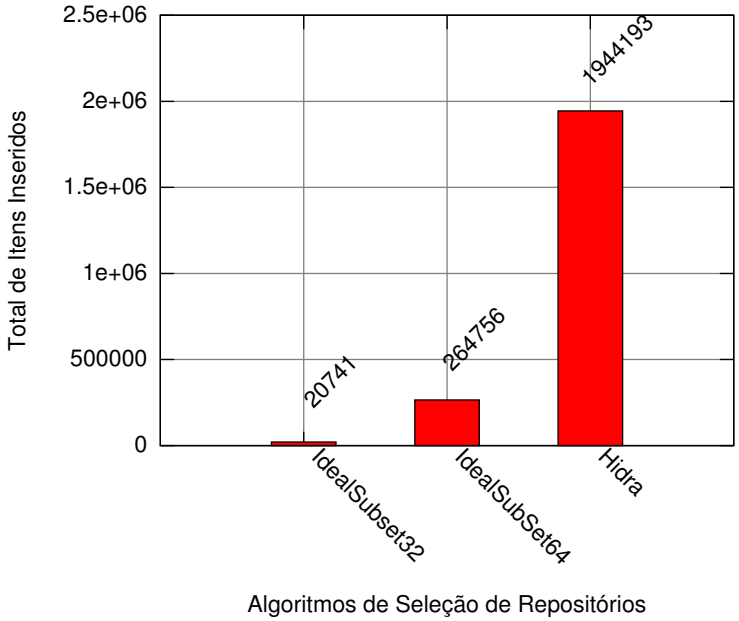

Figura 5.8: Comparativo do total de itens inseridos com a variação da confiabilidade dos repositórios entre os algoritmos Hidra e IdealSubSet

Na última comparação, apresentada na Figura 5.9, são avaliadas a quantidade de réplicas geradas pelos dois algoritmos. Hidra gera menos réplicas, porque enquanto Ide-

alSubSet com 32 e 64 Hashes geram, respectivamente, 16,19 e 17,16 réplicas em média Hidra gera em média somente 4,06 réplicas por documento.

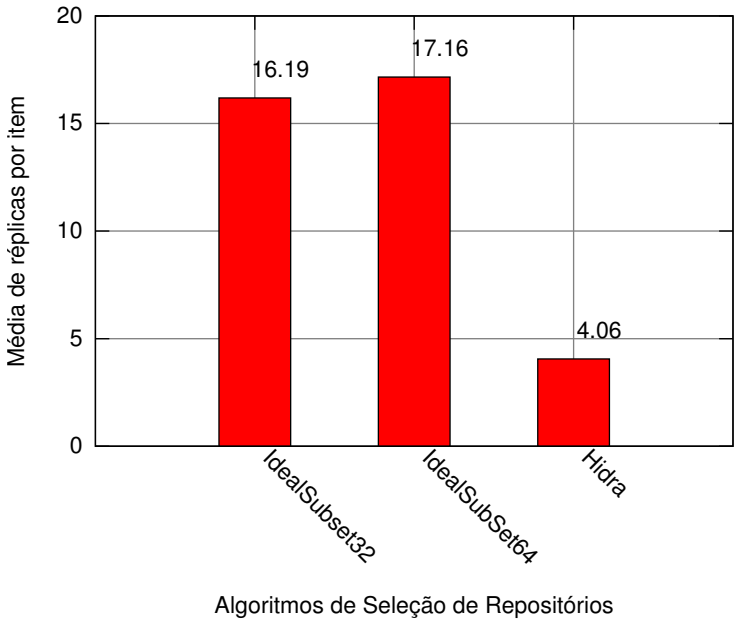

Figura 5.9: Comparativo da média de réplicas entre os algoritmos Hidra e IdealSubSet

Esses experimentos mostram que Hidra apresenta um desempenho superior a IdealSubSet em número de itens inseridos, espaço utilizado e média de réplicas. O custo dessa vantagem é a auditoria periódica, que IdealSubSet não realiza. IdealSubset utiliza um algoritmo de programação linear inteira, baseado em programação dinâmica, capaz de encontrar o melhor subconjunto de repositórios em tempo pseudo-polinomial. Mas essa solução não leva em conta que a confiabilidade dos repositórios pode ser maior quando um intervalo de auditoria é utilizado. Enquanto isso, Hidra utiliza um processo de Poisson para descobrir a confiabilidade dos repositórios em períodos de tempos menores. Então, utiliza seu algoritmo de seleção de repositórios para inserir réplicas com um mesmo intervalo de auditoria. Assim, é possível arquivar os documentos de forma confiável com um menor número de réplicas.

5.3 Experimentos de Auditoria

No arquivamento digital de longo prazo, a auditoria permite a detecção e recuperação de erros nas réplicas, devido às falhas nos repositórios. Ela não pode ser indiscrimina-

damente utilizada, porque desgasta os discos quando realizada em excesso. Além disso, gera tráfego de rede e consome processamento. Mas se o intervalo de tempo da auditoria dos itens for longo, há a possibilidade de que todas as suas réplicas sejam perdidas. As próximas subseções apresentam os experimentos realizados na avaliação do desempenho da auditoria. O primeiro experimento mostra o custo médio estimado da auditoria, no momento da inserção dos itens.

5.3.1 Custo médio estimado da auditoria

O custo médio estimado da auditoria é representado pelo número de operações de leitura das réplicas durante o período de 1 ano. Esse valor é apenas estimado, porque a entrada e saída de *peers* e a migração de réplicas realizada pela auditoria, que são processos dinâmicos, alteram esses valores.

Um experimento foi realizado para estimar o custo médio de auditoria com a variação da confiabilidade dos repositórios. Esse experimento permitiu constatar a hipótese de que os repositórios com confiabilidade menores necessitam de mais auditorias para atingir a confiabilidade desejada. Na Figura 5.10, é possível observar que o aumento da confiabilidade diminui o custo da auditoria por máquina. Isso acontece porque quanto maior é a confiabilidade dos repositórios, maior é o intervalo entre as auditorias, e consequentemente são feitas menos auditorias em um ano.

Utilizando a média de réplicas por repositório foi possível estimar a quantidade de auditorias realizadas em um ano, em cada réplica. Em média, foram realizadas 30 auditorias por ano em cada réplica, quando a confiabilidade dos repositórios é de 50%. Isso representa um intervalo médio entre auditorias de 12 dias e 4 horas, aproximadamente. Esses valores são expressos na Figura 5.11.

5.3.2 Coeficiente de variação da auditoria

Um experimento foi realizado para medir o coeficiente de variação da auditoria, que é uma medida de dispersão que se presta para a comparação de distribuições diferentes [47]. O coeficiente de variação é igual ao desvio-padrão dividido pela média.

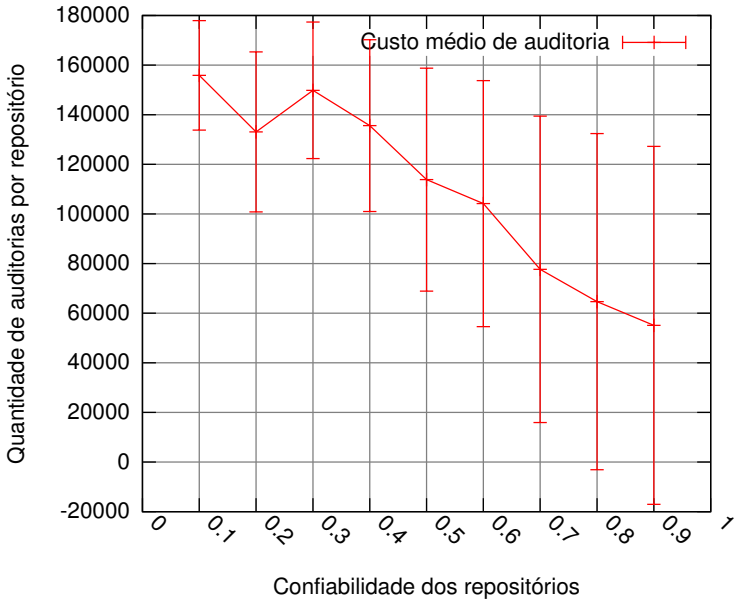

Figura 5.10: Custo médio de auditoria com a variação da confiabilidade dos repositórios

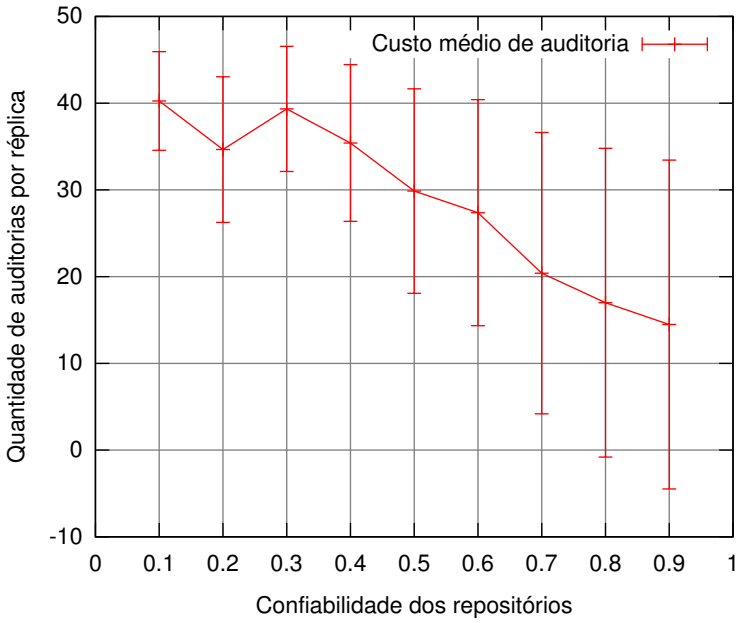

Figura 5.11: Custo médio de auditoria por réplica com a variação da confiablidade dos repositórios

Nesse experimento, expresso na Figura 5.12, é possível observar que o coeficiente de variação do custo estimado de auditoria por máquina acompanha o aumento da confiabilidade dos nós, porque quanto mais confiável o repositório, menor o número de auditorias necessárias. Essa informação corrobora as conclusões dos experimentos anteriores.

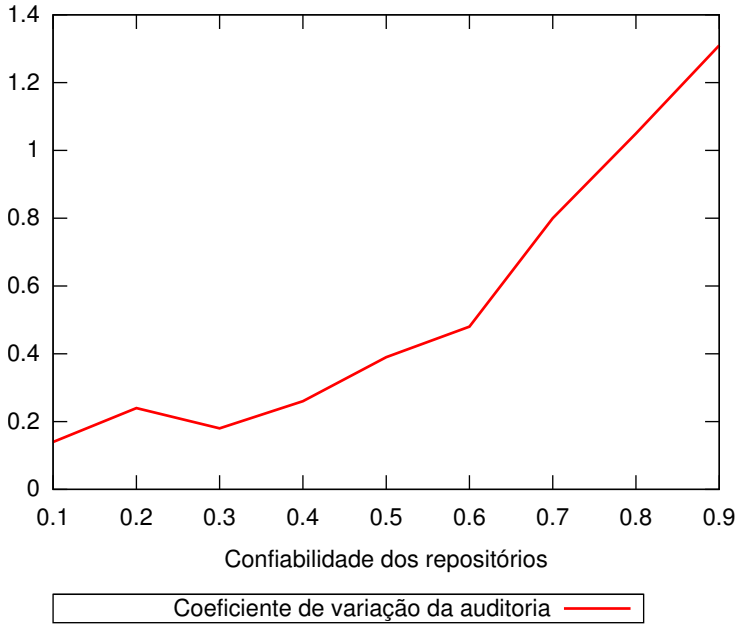

Figura 5.12: Coeficiente de variação do custo da auditoria com a variação da confiabilidade dos repositórios

5.3.3 Custo estimado de auditoria com a variação do tamanho da rede

O último experimento de estimativa do custo de auditoria variou o tamanho da rede para analisar como isso afetava o custo da auditoria. Esse experimento corrobora as informações da simulação que mediu o espaço de armazenamento utilizado com a variação do número de repositórios. Naquele experimento, o espaço utilizado não sofreu grandes variações e da mesma forma, a variação do tamanho da rede praticamente não influenciou no custo estimado da auditoria, conforme mostra a Figura 5.13. Isso se dá porque o algoritmo de múltiplas hashes possibilita um espalhamento uniforme das réplicas na rede.

5.3.4 Arquivamento digital no longo prazo

O arquivamento digital no longo prazo, num período de 100 anos, foi avaliado em uma segunda versão de Hidra implementada no simulador Peersim. Por se tratar de um experimento com alto tempo de execução, em torno de 40 horas, uma outra configuração de rede foi utilizada.

Primeiro, um experimento foi executado para avaliar o tempo de vida do sistema sem

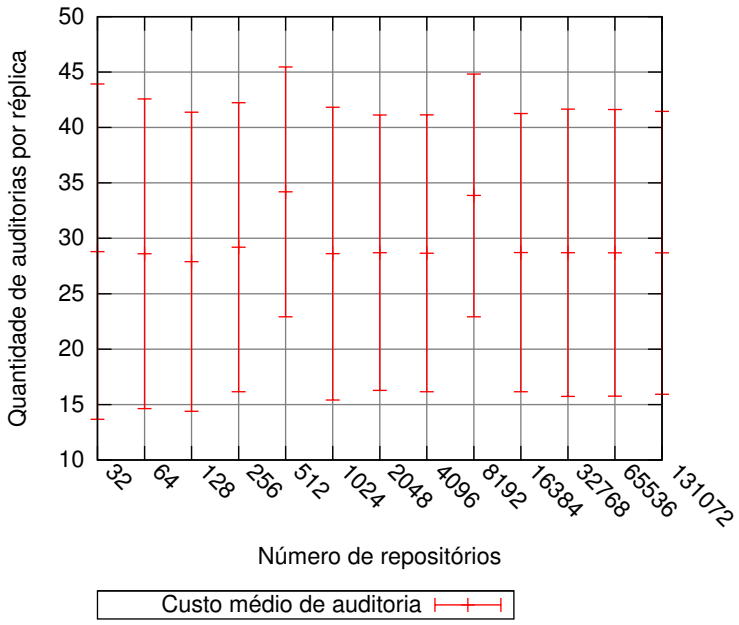

Figura 5.13: Custo médio de auditoria por réplica com a variação do número de repositórios

uso de auditoria. A rede continha 2048 repositórios, com disco com capacidade para armazenar 250 itens cada uma. O restante dos parâmetros são idênticos aos expostos na Tabela 5.1. Os itens foram inseridos até que a capacidade máxima da rede fosse atingida, sendo que foi possível inserir 125965 itens, preenchendo 99,85% da capacidade.

Em pouco mais de 37 anos todos os itens foram perdidos. Essa situação é exposta na Figura 5.16. Esse experimento comprova que sem auditoria, não é possível arquivar documentos digitais no longo prazo, porque nenhum repositório sobrevive tanto tempo sem sofrer falhas que comprometam seu conteúdo armazenado.

Além disso, é possível observar que o tempo de vida do sistema é prolongado devido à replicação. Mesmo com vários repositórios falhos, a porcentagem de itens disponíveis é maior. Até que chegue um momento em que somente uma réplica de um item está disponível e a falha de seu repositório implica na perda definitiva desse documento. Essa situação é exposta na Figura 5.15 que relaciona o percentual de repositórios falhos e de itens disponíveis. No final do experimento, todos os repositórios estavam falhos e nenhum item estava mais disponível.

O próximo experimento avalia o desempenho de Hidra no longo prazo com o uso de auditoria. A simulação ocorre num período de 100 anos ou 36500 dias e foi executada

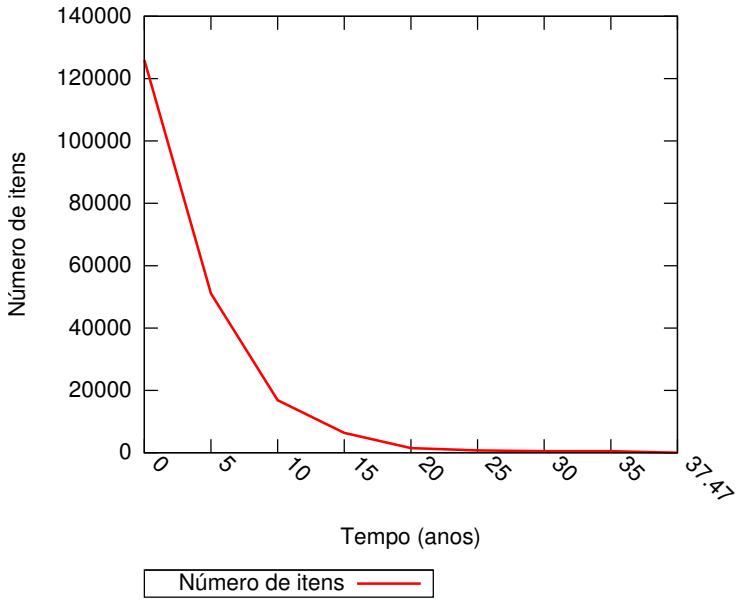

Figura 5.14: Número de itens em Hidra, durante 100 anos, sem auditoria

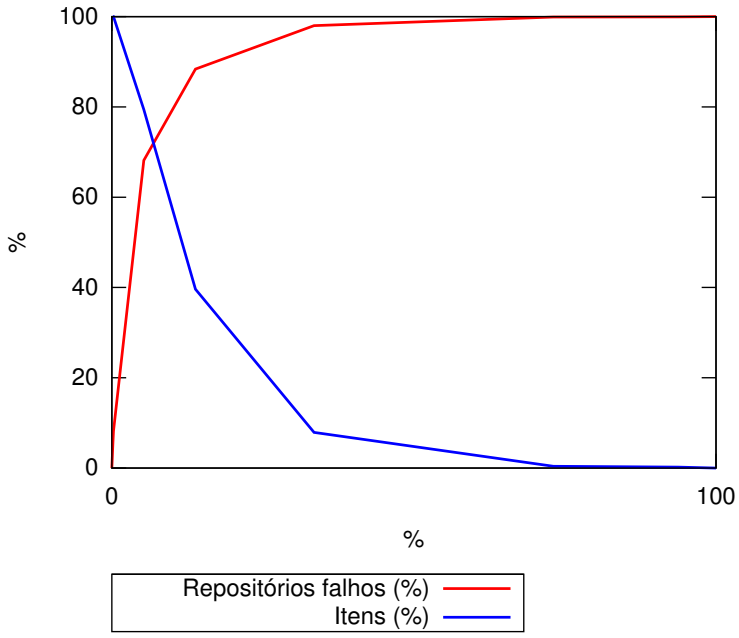

Figura 5.15: Percentual de itens em Hidra, em relação ao percentual de repositórios falhos, durante 100 anos, sem auditoria

com outra configuração no Peersim, devido ao tempo necessário para a execução do experimento. Apenas 512 repositórios foram utilizados com 256 unidades de espaço em disco cada. Mas conforme demonstraram os experimentos anteriores de inserção, a variação do número de repositórios e do espaço em disco não afeta consideravelmente o desempenho de Hidra. No início, foram inseridos itens até preencher a capacidade da rede. Foi possível inser 32222 itens, preenchendo 99,82% do espaço disponível na rede.

O experimento de auditoria provou que a auditoria é crucial no arquivamento digital. Em 100 anos, somente 36 itens foram perdidos, com um percentual de 99,888% de itens preservados. A Figura 5.16 apresenta o número de itens perdidos com o passar do tempo.

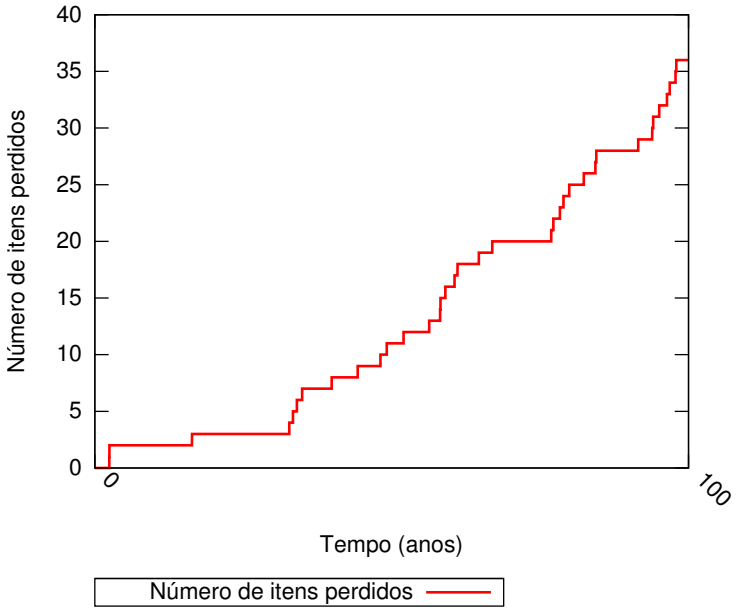

Figura 5.16: Número de itens perdidos, durante 100 anos, com procedimento de auditoria

Hidra utiliza um processo de Poisson para obter a confiabilidade efetiva de cada repositório em qualquer período de tempo. Assim, é possível saber qual a probabilidade de falha de um repositório em cada dia. Essa informação foi utilizada no experimento de auditoria, para decidir se um repositório deveria ou não falhar, baseado num número gerado aleatoriamente. Se esse valor fosse maior que a confiabilidade efetiva do repositório, ele deveria falhar. No experimento de auditoria aproximadamente 1% dos repositórios estavam falhos durante cada dia. Num mesmo dia, a quantidade máxima de repositórios falhos foi de 8 máquinas, representando apenas 1,5625% do total da rede que possuía 512 máquinas. Esses dados são expostos na Figura 5.17.

Com base nessas informações, é possível melhorar os parâmetros de Hidra para diminuir a perda de itens. Por exemplo, como o máximo de repositórios falhos foi de 8 por dia, se fosse utilizada uma quantidade mínima de 9 réplicas por documento digital, nenhum documento seria perdido no período de 100 anos.

O experimento de auditoria também permitiu analisar o custo real dessas operações. Dos 32222 itens, em média 2500 foram auditados por dia. Essas informações estão pre-

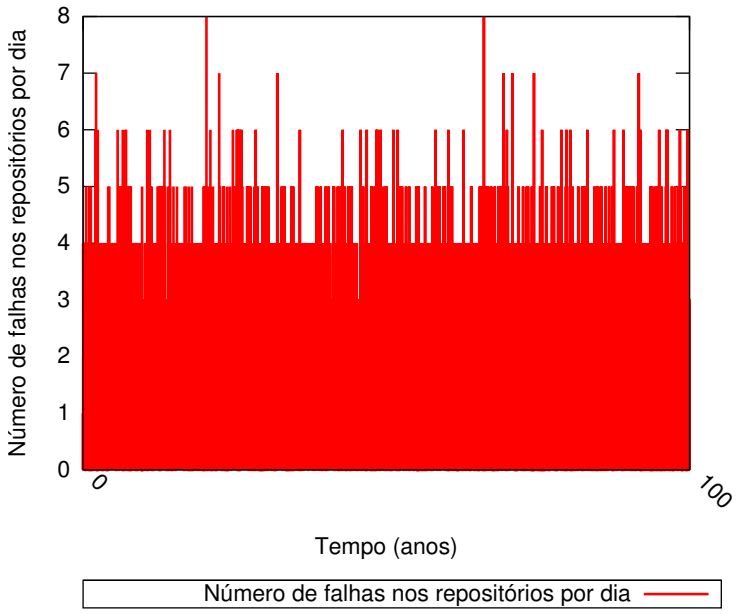

Figura 5.17: Número de repositórios falhos por dia, durante 100 anos

sentes na Figura 5.18. Isso significa que 7,75% da rede sofreu auditoria, em média, a cada dia.

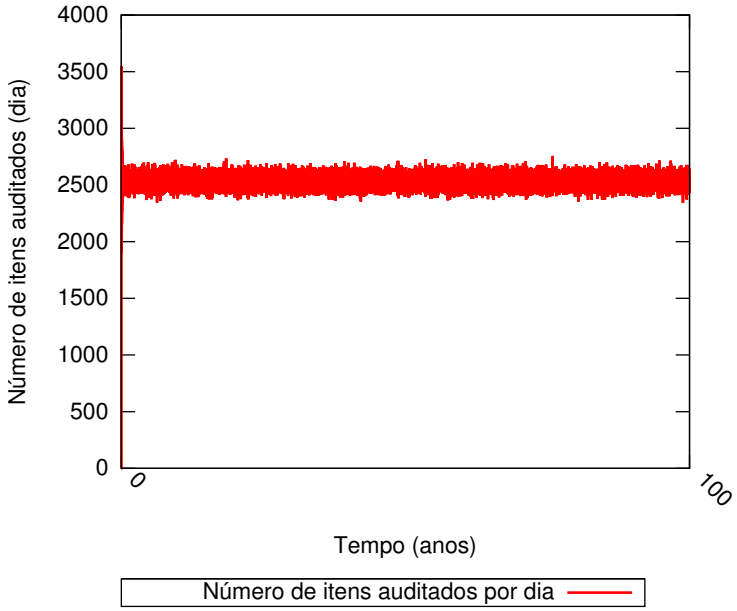

Figura 5.18: Número de itens auditados por dia, durante 100 anos

No período de 100 anos, o total de auditorias realizados nos mais de 32000 itens presentes foi de 92488448, conforme apresenta a Figura 5.19. Em média, cada item sofreu 28,7 operações de auditorias por ano, informação muito próxima do custo estimado de auditoria.

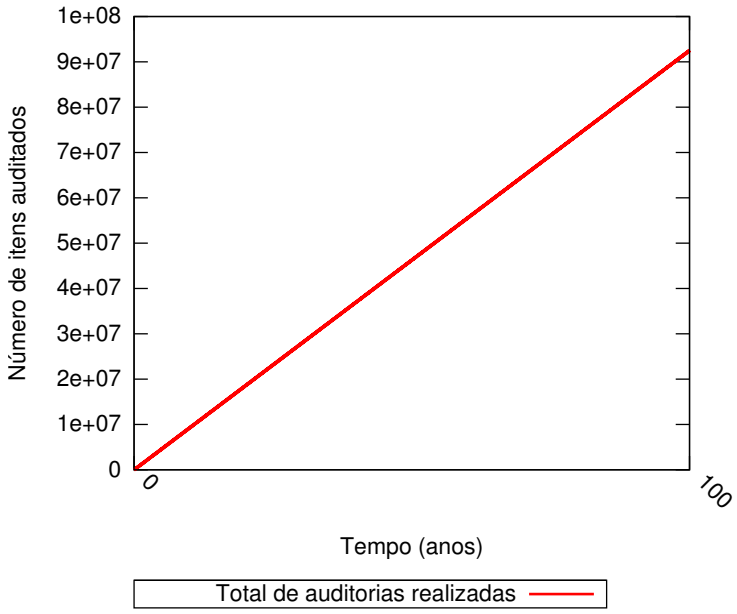

Figura 5.19: Total de auditorias realizadas durante 100 anos

O total de auditorias, exposto na Figura 5.19, cresce linearmente com o passar do tempo. Essa informação demonstra que a integridade dos documentos digitais com o passar do tempo é garantida pelas constantes operações de leitura desempenhadas pelo protocolo de auditoria.

O experimento de arquivamento digital no longo prazo com auditoria demonstra que os protocolos de seleção de repositórios e de auditoria de Hidra permitem que os documentos digitais sejam preservados utilizando uma rede P2P com máquinas de confiabilidades heterogêneas e de baixa capacidade de armazenamento. Poucos itens foram perdidos no período de 100 anos e não foi necessário utilizar muitas operações de auditoria ou réplicas na preservação dos documentos.

5.3.5 Comentários

Os experimentos realizados com o Peersim demonstram que Hidra possui um desempenho superior à abordagem utilizada por IdealSubSet. Isso se dá pelo uso de auditoria e de um processo de Poisson para o cálculo da confiabilidade dos repositórios em períodos de tempo distintos.

Hidra utiliza bem a capacidade da rede, porque escolhe uniformemente os repositórios

através de múltiplas hashes. Dessa forma, não é necessário que as máquinas tenham confiabilidade alta ou muito espaço disponível em disco, desde que sejam utilizadas uma quantidade mínima dessas funções. As múltiplas hashes também diminuem a influência do tamanho da rede no desempenho do sistema. Mesmo com redes pequenas, de 64 repositórios, é possível fazer um bom uso do espaço de armazenamento. Além disso, não são geradas muitas réplicas no processo de arquivamento. Mesmo com repositórios de baixa confiabilidade, a média de réplicas fica em torno de 5 cópias por documento.

Além das réplicas, a confiabilidade dos documentos digitais é garantida pelo protocolo de auditoria, que realiza verificações periódicas nos documentos. Os experimentos mostraram que, em média, não são necessárias mais que 30 auditorias por réplica no período de 1 ano, mesmo utilizando repositórios com confiabilidade mediana.

Capítulo 6

Conclusões e Trabalhos Futuros

A preservação das informações produzidas em meio digital é essencial para o pleno funcionamento da sociedade. O valor histórico, cultural, econômico e exigências jurídicas tornam necessário a preservação no longo prazo de muitos desses documentos, muitas vezes por períodos superiores a cem anos.

O arquivamento digital é a tarefa da preservação digital responsável pela integridade dos dados armazenados no longo prazo. O arquivamento é responsável pela contenção das principais ameaças à preservação dos dados, tais como as falhas dos discos rígidos e desastres naturais. Além disso, ele deve ser feito com baixo custo, para ser viável no longo prazo.

As redes P2P possuem características que as tornam ideais como substrato no arquivamento digital: elas são escaláveis, tolerante à falhas e de baixo custo, pois utilizam o tempo ocioso das máquinas. A redundância de recursos permite a disponibilidade dos dados mesmo com falhas em inúmeros *peers* da rede.

Neste trabalho, foi proposto Hidra: um modelo de arquivamento digital confiável em redes P2P baseado num modelo desenvolvido por Vignatti. Esse modelo utiliza uma P2P como substrato para o armazenamento de documentos através de réplicas. Ele considera que cada repositório possui uma capacidade de armazenamento e confiabilidade distintas. A confiabilidade é a probabilidade dos documentos não serem danificados num período de tempo pela ocorrência de falhas catastróficas. Os documentos digitais são armazenados

com uma confiabilidade desejada informada pelo usuário e através de replicação e auditoria são criadas réplicas em número suficiente para atender a confiabilidade desejada e um intervalo de auditoria é definido para que periodicamente a integridade das réplicas seja verificada e eventuais falhas recuperadas.

Além disso, Hidra introduziu várias modificações no modelo. A confiabilidade dos repositórios é considerada como um processo de Poisson, permitindo que uma máquina tenha mais confiabilidade num período de tempo menor. Uma estratégia de seleção de repositórios para o armazenamento de réplicas foi desenvolvida, aliada a um procedimento de auditoria periódica, que verifica a integridade das réplicas e garante a confiabilidade associada com o passar do tempo. Uma implementação de Hidra foi proposta com o uso de uma rede P2P, atendendo aos requisitos do arquivamento digital.

Experimentos realizados no simulador P2P Peersim mostram que Hidra é capaz de atingir uma confiabilidade mínima de 99,9999% para todos os documentos, gerando, em média, em torno de 5 réplicas. A porcentagem de itens perdidos, quando a auditoria foi utilizada, foi menor que 0,113% num período de 100 anos. Para a maioria dos documentos, o custo médio estimado de auditoria foi de 30 leituras por réplica no período de um ano.

O sucesso do arquivamento digital depende dos parâmetros de configuração de Hidra. Os experimentos mostraram que para ocupar mais de 90% do espaço disponível na rede foi necessário utilizar no mínimo 16 hashes e um espaço em disco mínimo capaz de armazenar 32 itens. Os resultados experimentais demonstraram que o número de repositórios e sua confiabilidade tem pouca influencia no desempenho do algoritmo de seleção de repositórios, porque eles são selecionados de maneira uniforme pelas múltiplas funções hashes e a baixa confiabilidade das máquinas é superada com mais réplicas e um intervalo de auditoria menor.

Nos trabalhos futuros, Hidra será aperfeiçoado para considerar a diversidade das características dos repositórios, como sistema operacional, *hardware* e sistema de arquivos, além da localização geográfica. Com base nessas informações, os repositórios serão selecionados de forma a minimizar a correlação de falhas, diminuindo a probabilidade de perda dos documentos.

Além disso, outras funcionalidades previstas no modelo OAIS podem ser implementadas para tornar Hidra um modelo mais completo. As funções hash dificultam a busca de documentos quando a chave informada não é exata, e novas abordagens podem ser utilizadas para resolver esse problema de busca em redes P2P.

Referências Bibliográficas

[1] T. Vignatti. Arquivamento digital a longo prazo baseado em seleção de repositórios em redes peer-to-peer. Master's thesis, Universidade Federal do Paraná, 2009.

[2] T. Vignatti, L.C.E. Bona, M.S. Sunye, and A.L. Vignatti. Long-term digital archiving based on selection of repositories over p2p networks. In *P2P '09. IEEE 9th International Conference on*.

[3] B. Babineau T. Asaro J. Mcknight. Digital achiving: End-user survey and market forecast 2006-2010. Technical report, The Enterprise Strategy Group, 2006.

[4] D. S. H. Rosenthal M. Roussopoulos P. Maniatis TJ. Giuli P. Bungale M. Baker, M. Shah. A fresh look at the reliability of long-term digital storage. In *EuroSys '06: Proceedings of the 1st ACM SIGOPS/EuroSys European Conference on Computer Systems 2006*, pages 221–234. ACM, 2006.

[5] T. Lipkis V. Reich S. Morabito D. S. H. Rosenthal, T. Robertson. Requirements for digital preservation systems: A bottom-up approach. *CoRR*, 2005.

[6] Memória eterna pode guardar dados por 1 bilhão de anos. http://www.inovacaotecnologica.com.br/noticias/noticia.php?artigo=memoria-eterna-podera-guardar-dados-por-1-bilhao-de-anos&id=010150090526, Acesso em Julho de 2009.

[7] M. Hedstrom. Exploring the concept of temporal interoperability as a framework for digital preservation. In *Third DELOS Workshop on Interoperability and Mediation in Heterogeneous Digital Libraries*, 2001.

[8] T. Chiueh M. Lu. Challenges of long-term digital archiving: A survey. Technical report, Stony Brook University.

[9] D. Liben-Nowell D. R. Karger M. F. Kaashoek F. Dabek H. Balakrishnan I. Stoica, R. Morris. Chord: a scalable peer-to-peer lookup protocol for internet applications. *IEEE/ACM Transactions Networks*, pages p. 17–32, 2003.

[10] M. Handley-R. Karp S. Schenker S. Ratnasamy, P. Francis. A scalable content-addressable network. In *SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 161–172. ACM, 2001.

[11] P. Druschel A. I. T. Rowstron. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *Middleware '01: Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg*, pages 329–350. Springer-Verlag, 2001.

[12] J. Stribling S. C. Rhea A. D. Joseph J. D. Kubiatowicz B. Y. Zhao, L. Huang. Tapestry: a resilient global-scale overlay for service deployment. *Selected Areas in Communications, IEEE Journal on*, pages 41–53, 2004.

[13] Y. Zheng Y. C. Hu A. R. Butt, T. A. Johnson. Kosha: A peer-to-peer enhancement for the network file system. *Journal of Grid Computing*, 2006.

[14] P. Sens J. Busca, F. Picconi. Pastis: A highly-scalable multi-user peer-to-peer file system. *Euro-Par 2005, LNCS*, 2005.

[15] D. Karger R. Morris I. Stoica F. Dabek, M. F. Kaashoek. Wide-area cooperative storage with cfs. In *SOSP '01: Proceedings of the 18th ACM symposium on Operating systems principles*, pages 202–215. ACM, 2001.

[16] P. Druschel A. Haeberlen, A. Mislove. Glacier: highly durable, decentralized storage despite massive correlated failures. In *NSDI'05: Proceedings of the 2nd conference on*

Symposium on Networked Systems Design & Implementation, pages 143–158. USENIX Association, 2005.

[17] M. Castro G. Cermak R. Chaiken J. R. Douceur J. Howell J. R. Lorch M. Theimer R. P. Wattenhofer A. Adya, W. J. Bolosky. Farsite: federated, available, and reliable storage for an incompletely trusted environment. In *OSDI '02: Proceedings of the 5th symposium on Operating systems design and implementation*, pages 1–14, 2002.

[18] Y. Chen S. Czerwinski P. Eaton D. Geels R. Gummadi S. Rhea H. Weatherspoon W. Weimer C. Wells B. Zhao J. Kubiatowicz, D. Bindel. Oceanstore: an architecture for global-scale persistent storage. *SIGPLAN*, pages 190–201, 2000.

[19] TJ. Giuli D. S. H. Rosenthal M. Baker P. Maniatis, M. Roussopoulos. The lockss peer-to-peer digital preservation system. *ACM Transactions on Computer Systems.*, pages 2–50, 2005.

[20] M. Roussopoulos P. P. Bungale, G. Goodell. Conservation vs. consensus in peer-to-peer preservation systems. *IPTPS, LNCS*, pages 240–251, 2005.

[21] Dspace.
http://www.dspace.org/, Acesso em Julho de 2010.

[22] Eprints.
http://www.eprints.org/, Acesso em Julho de 2010.

[23] G. P. Jesi S. Voulgaris M. Jelasity, A. Montresor. The Peersim simulator. `http://peersim.sf.net`.

[24] S. Martin M. Baker, K. Keeton. Why traditional storage systems don't help us save stuff forever. In *In Proc. 1st IEEE Workshop on Hot Topics in System Dependability*, 2005.

[25] Sata roadmap.
http://www.seagate.com/products/interface/sata/roadmap.html, Acesso em Agosto de 2009.

[26] Estimating drive reliability in desktop computers consumer eletronics. http://www.digit-life.com/articles/storagereliability/, Acesso em Agosto de 2009.

[27] Consultative Committe for Space Data Systems. Reference model for an open archival information system.
http://nost.gsfc.nasa.gob/isoas/ref_model.html,
Acesso em Julho de 2010.

[28] What is p2p ... and what isn't.
http://tim.oreilly.com/pub/a/p2p/2000/11/24/shirky1-whatisp2p.html?page=1,
Acesso em Julho de 2009.

[29] M. Pias R. Sharma S. Lim E. K. Lua, J. Crowcroft. A survey and comparison of peer-to-peer overlay network schemes. *Communications Surveys & Tutorials, IEEE*, 2005.

[30] Search for extraterrestrial intelligence (seti).
http://setiathome.berkeley.edu/, Acesso em Agosto de 2010.

[31] Napster.
http://wwww.napster.com, Acesso em Agosto de 2009.

[32] Gnutella.
http://rfcgnutella.sourceforge.net/, Acesso em Agosto de 2009.

[33] Skype.
http://wwww.skype.com/intl/pt, Acesso em Agosto de 2009.

[34] Msn.
http://wwww.msn.com, Acesso em Agosto de 2009.

[35] Icq.
http://www.icq.com, Acesso em Agosto de 2009.

[36] Bittorrent.
http://www.bittorrent.com/, Acesso em Julho de 2010.

[37] Limewire.
http://www.limewire.com/pt, Acesso em Julho de 2010.

[38] emule.
http://www.emule-project.net/, Acesso em Julho de 2010.

[39] The coral content distribution network.
http://www.coralcdn.org/, Acesso em Julho de 2010.

[40] T. Leighton R. Panigrahy M. Levine D. Lewin D. Karger, E. Lehman. Consistent hashing and random trees: distributed caching protocols for relieving hot spots on the world wide web. In *STOC '97: Proceedings of the 29th annual ACM symposium on Theory of computing*, 1997.

[41] National Institute of Standards and Technology (NIST). Secure hash standard, federal information processing standards publication. http://www.itl.nist.gov/fipspubs/fip1801.html, Acesso em Julho de 2010.

[42] C. Koppen M. Hermann. *Peer-to-Peer Systems and Applications*, volume 3485, chapter Self-Organization in Peer-to-Peer Systems, pages 247–266. 2005.

[43] S. Götz1 K. Wehrle S. Rieche, H. Niedermayer. *Peer-to-Peer Systems and Applications*, volume 3485, chapter Reliability and Load Balancing in Distributed Hash Tables, pages 119–135. Springer Berlin / Heidelberg, 2005.

[44] V. Martins E. Pacitti P. Valduriez. Survey of data replication and p2p systems. Technical report, Institut National de Recherche en Informatique et en Automatique, 2006.

[45] H. Garcia-Molina B. F. Cooper. Peer-to-peer data preservation through storage auctions. *IEEE Transactions on Parallel and Distributed Systems*, pages 246–257, 2005.

[46] F. Dabek. *A distributed hash table.* PhD thesis, 2006.

[47] *Statistics for Engineers and Scientists*, page 174;213. McGraw Hill, 2006.

[48] Linguagem de programação java. http://java.sun.com, Acesso em Julho de 2009.