Cristiano de Paula Costa

Disseminação de Conteúdo Poluído em Sistemas Par-a-Par de Compartilhamento de Arquivos

Dissertação apresentada ao Curso de Pós-Graduação em Ciência da Computação da Universidade Federal de Minas Gerais, como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

Orientadora:

Jussara Marques de Almeida

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

Belo Horizonte

1 de agosto de 2007

Agradecimentos

Meus agradecimentos especiais a minha orientadora Jussara Almeida por ter me orientado neste trabalho e em muitos outros desde o início de 2003, quando ainda era um aluno de graduação. Obrigado por estar sempre presente quando eu precisei e também pela paciência. Sua orientação me fez crescer muito como pesquisador e realizar diversos trabalhos de qualidade. Muito obrigado.

Agradeço ao professor Virgílio Almeida pelo auxílio e participação no início deste trabalho. Gostaria de agradecer também aos professores Berthier Ribeiro e Sérgio Campos, principalmente pelo auxílio no início da minha iniciação científica.

Agradeço a todo o DCC pela excelência tanto dos cursos de graduação quanto de pós graduação. Ao CNPQ pelas bolsas de estudo, permitindo assim que eu me dedicasse exclusivamente a pesquisa no departamento de computação.

Gostaria de agradecer a todos do laboratório de Vídeo sob Demanda (VoD), no qual trabalhei desde de minha iniciação científica até o final deste mestrado. Obrigado por serem tão companheiros e proporcionarem um ambiente de trabalho tão agradável durante todo esse tempo. Obrigado ao Ítalo, Alex, Marcelo, Itamar, Vanessa, Daniel, Marcus, Claudiney, e a todos outros por serem amigos e não somente colegas de trabalho.

Gostaria de agradecer a toda minha família e aos meus amigos, presentes em todo o processo.

Gostaria de agradecer aos meus pais por sempre acreditarem em mim, por sempre estarem presentes e por me apoiarem nos momentos mais difíceis. Obrigado por compreenderem o fato de eu passar mais tempo no trabalho do que em casa e obrigado por toda a ajuda que vocês me deram durante toda minha vida. Obrigado também aos meus irmãos, Guilherme e Fernanda, pelo apoio.

Obrigado a todos.

Resumo

Recentemente, sistemas Par-a-Par (P2P) de compartilhamento de arquivos vêm sendo alvo de uma nova forma de comportamento malicioso: a poluição de conteúdo. A disseminação de conteúdo poluído reduz a disponibilidade dos arquivos e conseqüentemente a confiança dos usuários no sistema. Esta dissertação apresenta um estudo desse problema, bem como o projeto e extensa avaliação de estratégias para combater esse padrão de comportamento malicioso.

O processo de disseminação de conteúdo poluído foi avaliado, via simulação, medindo a fração de *downloads* de objetos não poluídos, considerando dois mecanismos de introdução de poluição, a saber, a *inserção de versões falsas* e a *corrupção do identificador*. Este último foi apresentado nesta dissertação. Os resultados dos experimentos realizados mostraram que: (1) a *corrupção do identificador* dissemina poluição mais rapidamente e (2) a disseminação de poluição não pode ser efetivamente contida, apenas na ação voluntária dos usuários apagarem seus objetos poluídos.

Além disso, foram propostas estratégias para combater a disseminação de conteúdo poluído nos sistemas P2P. Foi apresentada uma estratégia para reduzir a poluição baseada na censura por moderadores. Também foram propostos os sistemas de reputação *Scrubber*, que atribui reputação para os pares como fontes de objetos poluídos, e o sistema *Híbrido*, que combina as funcionalidades de sistemas de reputação de pares e classificação de objetos.

A avaliação dos sistemas propostos mostrou que a estratégia baseada na censura por moderador consegue uma redução significante da poluição, mas essa estratégia pode apresentar problemas de escalabilidade. Os sistemas de reputação foram avaliados comparando a sua eficácia com o sistema de reputação *Credence*. Na avaliação foram consideradas diferentes configurações de sistema e de comportamento dos usuários, incluindo ataques de conluio, *Sybil* e *whitewashing*. Os resultados mostraram que o sistema *Híbrido* é mais efetivo e robusto que as outras estratégias avaliadas para todos os cenários.

Abstract

Recently, Peer-to-Peer (P2P) file sharing systems are experimenting a new form of malicious behavior: content pollution. The dissemination of polluted content reduces content availability and thus the confidence of user on such systems. This dissertation presents a study of this problem, as well as design and evaluates strategies to fight the dissemination of polluted content.

The dissemination of polluted content was analyzed, via simulation, measuring the fraction of unpolluted downloads, for mechanisms for pollution dissemination, namely, *decoy insertion* and *identifier corruption*. The last one identified and presented on this dissertation. The results show that: (1) the *identifier corruption* spreads polluted content faster than *decoy insertion*, and (2) the dissemination of polluted content can not be effectively prevented, only on user voluntary action of deleting its polluted objects.

In addition, new strategies were proposed to fight the dissemination of polluted content in P2P systems. A strategy based on object censorship by moderator was proposed. Furthermore, we proposed Scrubber, a peer reputation system where users assign reputation to each other as source of polluted content, and a Hybrid system that combines peer reputation and object classification.

An evaluation of the strategy based on object censorship shows a significant reduction on the dissemination of polluted content, but it may suffer from scalability problems. The reputation systems were evaluated comparing their effectiveness against the previously proposed Credence reputation system. An extensive evaluation of the three systems was performed for various system configuration and peer behavior, including collusion, Sybil and whitewashing attacks. The results show that the Hybrid system is more effective and robust than any individual strategy, for all evaluated scenarios.

Sumário

Lista de Figuras

Lista de Tabelas

1 Introdução

Ao longo das últimas décadas muitos serviços na *Internet*, que antes eram implementados em arquiteturas cliente-servidor, vêm evoluindo para uma arquitetura descentralizada na qual os clientes têm também a responsabilidade de prover serviço (ex: armazenamento, processamento, conteúdo, etc). Esses tipos de serviços com arquiteturas descentralizadas e distribuídas, chamados de sistemas Par-a-Par ou P2P, representam hoje um dos serviços mais populares na *Internet*.

As vantagens que tornam os sistemas P2P atrativos são [5]:

- a utilização de recursos (ex.: conteúdo, armazenamento, processamento, etc) desperdiçados pelos clientes da *Internet*.
- a habilidade de se auto-organizar, escalar e trabalhar em cenários com um grande número de clientes ou com um comportamento muito dinâmico.
- tolerância a falhas e robustez contra ataques, característica que foi adquirida a partir da natureza descentralizada e distribuída desses sistemas.

Diversos serviços utilizam a arquitetura dos sistemas P2P devido a essas facilidades. Apesar da terminologia recente, os primeiros sistemas P2P datam do início da *Internet* [3]. De fato, a própria *ARPANET* foi concebida para compartilhar os recursos de diferentes redes nos Estados Unidos. Os primeiros pares da *ARPANET* cooperavam independentemente e não eram organizados sobre nenhum tipo de hierarquia, como acontece na *Internet* atualmente. Outros serviços como a rede *Usenet News* [35] e o *Domain Name Service* (DNS) [55] foram criados sobre a visão descentralizada e distribuída dos sistemas atualmente conhecidos como P2P.

Porém, somente com o aumento dos recursos nos clientes, sobretudo banda de rede e espaço de armazenamento, os sistemas P2P se tornaram populares [48]. Dentre os serviços que utilizam a arquitetura P2P estão serviços de mensagens instantâneas [36], computação distribuída [59], distribuição de vídeo sob demanda e conteúdo ao vivo [85], recuperação de informação [7],

serviços de telefonia, audio e vídeo conferência [75] e compartilhamento de arquivos [11, 27, 40, 54, 68, 76].

Dentre esses serviços, os sistemas P2P de compartilhamento de arquivos (ex: BitTorrent [11], Kazaa [40] e Gnutella [68]) representam atualmente um dos serviços mais importantes da *Internet*, considerando volume de tráfego, número de usuários e objetos compartilhados. Esse serviço, no qual os pares participantes compartilham seus arquivos uns com outros, é responsável por 80% do tráfego de rede nos *backbones* [14]. Além disso, estatísticas recentes reportam que o número de usuários em alguns desses sistemas chega a oito milhões atualmente [46]. Os dados disponibilizados pelos pares nesses sistemas incluem álbuns de música, filmes e séries de televisão, documentos, *software* e jogos.

A crescente popularidade gerou problemas de desempenho nesses sistemas e diversos trabalhos de otimização foram propostos. Tentando tornar o sistemas mais escaláveis e sem servidores centrais, surgiram os sistemas baseados em super-pares (ex.: Kazaa) e os totalmente descentralizados (ex.: Gnutella). Otimizações para as pesquisas por conteúdo foram propostas através dos sistemas P2P estruturados (ex.: Kademlia [54] e Chord [76]), que mantém uma topologia controlada para que objetos possam ser encontrados de forma determinista. O BitTorrent, uma arquitetura recente que alcançou grande popularidade na Internet [14], propõe um sistema baseado somente na distribuição do conteúdo entre os pares, deixando a que pesquisa dos objetos seja realizada a partir de sítios na *Internet*.

Porém, em consequência de sua grande popularidade, os sistemas P2P de compartilhamento de arquivos também têm sido alvo de comportamento malicioso e oportunista. O comportamento mais estudado é a falta de cooperação ou comportamento egoísta (*free-riding*) [1, 72], no qual os pares egoístas não compartilham seus recursos com o sistema. Outros padrões de comportamento malicioso já observados são os ataques de negação de serviço (*Denial of Service - DOS*), que tentam tornar o sistema indisponível [46], e a disseminação de vírus e *worms* [37, 78].

Além desses, ataques mais sofisticados, como conluio [53], *Sybil* [23], *whitewashing* [29] e traidor [53], vêm sendo estudados. No ataque por conluio, os pares maliciosos agem conjuntamente para tentar dominar o sistema. No ataque *Sybil*, o par malicioso tenta dominar o sistema controlando diversas instâncias de clientes P2P. No *whitewashing*, os pares maliciosos, aproveitando da facilidade de participar de um sistema P2P, trocam suas identidades freqüentemente para não serem identificados como maliciosos. Finalmente, no ataque de traidor, os pares maliciosos se comportam adequadamente para conseguirem a confiança dos pares, e então atacam o sistema.

1.1 Disseminação de Conteúdo Poluído

Recentemente foram apresentadas evidências de uma nova forma de comportamento malicioso, a poluição de conteúdo [45]. Poluição consiste na disponibilização de objetos com o conteúdo corrompido e inútil. O conteúdo disponibilizado possui o mesmo meta-dado de um objeto não corrompido e isso acaba dificultando a localização de objetos não poluídos. Esse ataque faz com que os usuários percam confiança no sistema, além de disperdiçar recursos com tráfego indesejado.

Relatos indicam que conteúdo poluído é um problema real nos atuais sistemas P2P de compartilhamento de arquivos. Estudos apontam que o número de cópias poluídas de músicas no sistema KaZaA alcança até 80% para certos títulos [45]. A poluição tipicamente só pode ser descoberta através da inspeção manual, o que torna o problema difícil de ser tratado. Liang *et alii* tentaram criar um identificador automático de poluição, porém tal aplicação só identificava objetos de áudio, e mesmo assim dependia de dados externos (ex.: tempo de reprodução) sobre as músicas. Outro fator que cria dificuldade no tratamento da poluição são os usuários dos sistemas P2P que não apagam os objetos poluídos obtidos e acabam compartilhando-os por descuido [47]. Por esse fato, a poluição que deveria ser disseminada somente por pares maliciosos, também é distribuída passivamente pelos próprios usuários do sistema.

Pelo menos dois mecanismos de introdução e disseminação de conteúdo poluído são conhecidos. A *inserção de versões falsas* consiste na inserção de objetos poluídos que possuem o mesmo meta-dado que outro não poluído [45]. A *corrupção do identificador*, mecanismo descrito nesta dissertação no capítulo 3, consiste em explorar a fraqueza de algumas técnicas atuais de geração dos identificadores dos objetos para criar cópias poluídas de objetos não poluídos já presentes no sistema. Dessa forma, um determinado conteúdo pode ter cópias poluídas e não poluídas, todas com o mesmo identificador e portanto não distinguíveis.

Os esforços na direção de reduzir a disseminação da poluição têm sido muito tímidos. Alguns trabalhos anteriores focam na modelagem e análise da disseminação de poluição por inserção de versões falsas [42, 78]. Outros fornecem idéias gerais de como reduzir a disseminação de poluição [16, 45, 78]. Mas somente alguns [21, 39, 47, 82] propuseram e implementaram soluções práticas. Contudo, a avaliação desses sistemas foi de certa forma limitada e pouco se conhece sobre a eficácia dessas estratégias de redução da poluição.

1.2 Objetivos

Os objetivos desta dissertação são:

1. Avaliar mecanismos de introdução de poluição em sistemas P2P atuais, a saber, A *introdução de versões falsas* e a *corrupção do identificador*.
2. Avaliar a robustez dos sistemas P2P, bem como o impacto da atribuição de incentivos para os usuários apagarem seus objetos poluídos.
3. Propor novas estratégias de redução da poluição.
4. Avaliar extensivamente essas estratégias e outras já propostas na literatura. A avaliação consistirá de comparação, análise de sensibilidade dos seus parâmetros e verificação da robustez das estratégias a ataques.

1.3 Contribuições

As principais contribuições deste trabalho são:

- Apresentação de um mecanismo de introdução e disseminação de poluição, chamado *corrupção do Identificador*.
- Construção de um simulador orientado por eventos para realizar a avaliação dos mecanismos de introdução de poluição e analisar as estratégias de combate.
- Proposição de um modelo analítico para avaliar a disseminação de poluição pelo método de *inserção de versões falsas*. Tal modelo foi usado para validar os resultados obtidos por simulação para as análises realizadas com o métodos de *inserção de versões falsas*.
- Análise da disseminação do conteúdo poluído pelos mecanismos de *inserção de versões falsas* [45] e *corrupção do identificador*, bem como o impacto de atribuir incentivos aos usuários para que eles apaguem o conteúdo poluído obtido. Essa análise mostrou que os pares maliciosos conseguem disseminar poluição mais rapidamente com o mecanismo de *corrupção do identificador* do que com o mecanismo de *inserção de versões falsas*. Além disso, também é mostrado que a disseminação de poluição não pode ser efetivamente contida, apenas através da ação voluntária dos usuários apagarem seus objetos poluídos [8, 20].

- Proposição de estratégias para redução da poluição:
  - Proposição de uma estratégia na qual um moderador gerencia os objetos do sistema, censurando as cópias poluídas.
  - Proposição do *Scrubber* [18, 19], um sistema de reputação descentralizado e distribuído, no qual os pares atribuem reputação uns para os outros como fontes de objetos poluídos. O objetivo do *Scrubber* é identificar e isolar os usuários maliciosos que disseminam conteúdo poluído ativamente no sistema. Além disso, o *Scrubber* impõe uma punição severa e rápida para os poluidores (recusando servir as suas requisições), mas também inclui incentivos para a reabilitação de pares que pararam de enviar conteúdo poluído.
  - Proposição do sistema de reputação *Híbrido* [17] que, combina a reputação de pares herdada do *Scrubber* com um mecanismo que classifica os objetos.
- Realização de uma extensa avaliação dos sistemas de reputação propostos. A eficácia dos sistemas de reputação propostos foram comparados com o sistema de reputação *Credence* [80–82]. Além disso, foi realizada uma análise de sensibilidade cuidadosa dos parâmetros dos sistemas avaliados. As estratégias foram avaliadas para diversas configurações de sistema e comportamentos dos pares, incluindo cenários nos quais os pares maliciosos utilizam ataques de conluio, *sybil* e *whitewashing*.

  Nessa análise, é mostrado que o sistema *Híbrido* consegue reduzir a disseminação de conteúdo poluído muito mais rapidamente que os outros sistemas de reputação, mesmo quando os pares maliciosos utilizam ataques de conluio e *Sybil* para disseminar poluição. Além disso, o sistema *Híbrido* é menos sensível a variações dos seus parâmetros, em comparação com o *Scrubber*. Finalmente, todos os três protocolos dependem da cooperação dos usuários em atribuir opiniões corretas sobre os outros pares ou objetos do sistema. Todavia, mesmo em comunidades que não cooperam ou não são confiáveis, o sistema *Híbrido* consegue reduzir a disseminação de poluição de forma mais eficaz que as outras estratégias.

1.4 Organização da Dissertação

Esta dissertação está organizada da seguinte forma. O capítulo 2 apresenta uma revisão bibliográfica de sistemas P2P e evidências de comportamento malicioso nesses sistemas, sobretudo a disseminação de conteúdo poluído. O capítulo 3 discute os principais conceitos envolvidos nesta dissertação e introduz o modelo de simulação utilizado para modelar os problemas

apresentados. O capítulo 4 apresenta um modelo analítico utilizado para validar o simulador, além de mostrar a avaliação do impacto da disseminação de conteúdo poluído nos sistemas P2P. O capítulo 5 apresenta as estratégias para redução da poluição. O capítulo 6 apresenta uma extensa avaliação da eficácia das estratégias apresentadas no capítulo 5. Finalmente, o capítulo 7 discute as conclusões e trabalhos futuros.

2 Trabalhos Relacionados

Este capítulo discute os trabalhos da literatura mais relevantes para o estudo da introdução e disseminação de conteúdo poluído em sistemas P2P de compartilhamento de arquivos. A Seção 2.1 explica os diferentes sistemas P2P de compartilhamento de arquivos que existem atualmente. A Seção 2.2 explica os principais padrões de comportamento malicioso que ocorrem nos sistemas P2P, incluindo a poluição de conteúdo, e apresenta os principais estudos na área. Finalmente, a Seção 2.3 apresenta os trabalhos desenvolvidos para combater os padrões de comportamento maliciosos, sobretudo as estratégias criadas para reduzir a disseminação de poluição.

2.1 Sistemas Par-a-Par de Compartilhamento de Arquivos

Os sistemas P2P de compartilhamento de arquivos são serviços distribuídos para entrega de conteúdo. Os objetos disponibilizados são armazenados e servidos pelos computadores pessoais dos usuários ou pares do sistema. Os usuários provêm conteúdo ao sistema e também realizam *download*[1] dos objetos compartilhados por outros pares. As duas principais funcionalidades de um sistema P2P de compartilhamento de arquivos é a pesquisa pelos objetos presentes no sistema e a coordenação do *download* desses objetos.

O primeiro sistema P2P de compartilhamento de arquivos a se popularizar foi o Napster [57]. Neste sistema, existia um servidor central responsável por indexar todos pares e objetos presentes. A coordenação do *download* era realizada de forma descentralizada, porém a pesquisa por objetos sempre era roteada por esse servidor central. Criado por Shawn Fanning em 1999, a popularidade do sistema cresceu rapidamente alcançando 60 milhões de usuários em seus dois anos de existência [31]. O Napster fechou seu servidor em 2001 por infringir direitos autorais.

Com a imensa popularidade que o Napster alcançou, não tardou para que diversas novas

[1] Pela falta de uma tradução oficial e pela grande utilização na *Internet* atualmente, foi optado por utilizar a palavra inglesa "*download*" no seu idioma original.

propostas de sistemas P2P de compartilhamento de arquivos surgissem. O principal objetivo dessas novas propostas era descentralizar e otimizar o mecanismo de pesquisa de objetos.

Atualmente, existe um grande número de propostas e implementações de sistemas P2P de compartilhamento de arquivos. A literatura geralmente agrega os sistemas pela organização lógica formada pelos pares, classificando-os como *estruturados* e *não estruturados*. Tais sistemas também são classificados pela forma de distribuição dos objetos. A seção 2.1.1 apresenta os sistemas P2P estruturados e a seção 2.1.2 apresenta os sistemas P2P não estruturados. Finalmente, a seção 2.1.3 discute a classificação dos sistemas P2P quanto a forma de distribuição dos objetos.

2.1.1 Sistemas Par-a-Par Não Estruturados

Nos sistemas *não estruturados*, a organização lógica dos pares é criada arbitrariamente. A pesquisa por objetos não tem relação com a topologia criada e geralmente é realizada de forma não determinista. Por não existir regras que controlem a formação da topologia lógica, o custo para a manutenção desses sistemas é baixo.

Os sistemas P2P *não estruturados* geralmente são agregados em três grupos distintos, classificados de acordo com a forma como a pesquisa pelos objetos é realizada. Eles são os sistemas *totalmente descentralizados*, e *parcialmente descentralizados*. Algumas classificações [5, 49] também incluem sistemas como Napster nesse grupo e os denominam como *centralizados híbridos*.

Sistemas Totalmente Descentralizados

Nos sistemas *totalmente descentralizados* todos os pares possuem o mesmo papel, servindo pesquisas por objetos e compartilhando-os. Não existe um servidor central para realizar as pesquisas ou para qualquer outra tarefa interna do sistema. O sistema mais conhecido que usa tal topologia é o Gnutella [68].

A tentativa de evitar o uso de um servidor central gerou sérios problemas para esses sistemas. O mecanismo de pesquisa por objetos ocorre através de inundação (*flooding*). Nesse mecanismo, quando um par deseja pesquisar por um objeto, ele envia para seus vizinhos uma requisição de busca. Os pares que receberam a requisição retornam informação se possuem ou não o objeto e repassam a mensagem para seus vizinhos, *inundando* a rede com estas mensagens.

Esse mecanismo de pesquisa é ineficiente, pois gera um grande volume de tráfego de rede. Por esse motivo, diversos trabalhos propuseram modificações no mecanismo de busca do Gnutella. Dentre estas tentativas estão técnicas de pesquisa por caminhos aleatórios (*random walks*), que enviam as mensagens de pesquisa somente para alguns vizinhos escolhidos aleatoriamente [10, 32]. Foram criadas também técnicas de pesquisas baseadas em comunidades de interesse, que priorizam o envio das mensagens de pesquisa para pares que possuem um histórico de pesquisa semelhante [6, 30]. Além disso, versões mais recentes do Gnutella foram estendidas e se transformaram em sistemas Parcialmente Descentralizados, apresentados na próxima seção.

Sistemas Parcialmente Descentralizados

Os sistemas *parcialmente descentralizados* foram criados com o mesmo propósito dos *totalmente descentralizados*, tentando evitar a utilização de um servidor central. A principal diferença para os sistemas totalmente descentralizados é a hierarquia seguida pelo pares nos sistemas parcialmente descentralizados. Os representantes mais populares desse tipo de sistemas são o Kazaa [40] e o eDonkey/Overnet [27].

Os pares são divididos em dois grupos, super-pares (ou super-nós) e os pares comuns. Os super-pares são pares mais estáveis, com um maior poder de processamento e banda de rede. Esses pares são responsáveis pela indexação de todos os objetos do sistema e pelo roteamento das pesquisas. Cada super-par é responsável por indexar os objetos de um determinado grupo de pares comuns. Quando uma pesquisa é realizada, os super-pares procuram os objetos em seus índices e, se necessário, repassam a pesquisa para outros super-pares. A principal vantagem desse tipo de sistema é, além de não possuir um ponto central de falhas, a redução do tempo de resposta de uma pesquisa quando comparado com os sistemas *totalmente descentralizados*.

2.1.2 Sistemas Par-a-Par Estruturados

Os sistemas P2P *estruturados* surgiram como uma tentativa de tornar as pesquisas por objetos mais eficientes. Em tais sistemas a organização lógica dos pares é bem definida, de forma que a pesquisa por pares ou objetos seja determinista.

A principal característica desse tipo de sistema é o mapeamento entre um objeto e a sua localização. Tal funcionalidade é alcançada através de tabelas de roteamento distribuídas, que cada par do sistema é obrigado a manter. Esta tabela de roteamento distribuída determina a sequência de pares a contactar para encontrar um determinado objeto. Os sistemas mais efici-

entes garantem que em uma pesquisa o número de pares contactados é da ordem de O(log(n)), onde *n* é o número de pares no sistema.

Apesar da pesquisa eficiente, os sistemas *estruturados* precisam manter a sua topologia bem organizada. Logo, em cenários onde os pares têm um comportamento muito dinâmico, a manutenção da topologia pode ser bem dispendiosa [50].

Os principais representantes desta classe são CAN [65], Chord [76], Pastry [51], Tapestry [86] e o Kademlia [54].

2.1.3 Distribuição dos Objetos

Existem duas formas de distribuir os objetos nos sistemas P2P de compartilhamento de arquivos: (1) o *download* do objeto de um único usuário ou (2) de múltiplos usuários (*swarm download*).

Os primeiros sistemas de compartilhamento de arquivos, como o Napster, distribuíam os objetos através do *download* de um único usuário. Nesse mecanismo, assim que o usuário escolhe um objeto para *download*, um dos pares que estão compartilhando o objeto é escolhido para enviar o conteúdo. Essa forma de distribuir os objetos se mostrou ineficiente, pois o par contactado para enviar o objeto poderia ter pouca banda de rede, e portanto um ponto de contenção.

Desta forma, surgiram os sistemas que realizam a distribuição dos objetos através de múltiplos usuários. No processo de *download* desse mecanismo, os usuários que compartilham o objeto enviam segmentos de dados do arquivo. Isso torna o processo de *download* mais eficiente, pois evita que o objeto seja enviado por um único usuário com pouca banda disponível. Os sistemas mais recentes, como eMule, KaZaa e o Kademlia, utilizam esta técnica para distribuir o conteúdo.

O BitTorrent, é um sistema muito popular atualmente que se baseia somente na distribuição dos objetos. Além de permitir a distribuição do conteúdo através de múltiplos usuários, o BitTorrent possui um mecanismo para encorajar os usuários do sistema a compartilharem o conteúdo.

Além disso, o BitTorrent não cria uma topologia lógica para realizar a pesquisa por objetos. A pesquisa por um objeto é realizada a partir de sítios na *Internet*, que disponibilizam a informação sobre o objeto desejado através de arquivos indexadores (*.torrent*). Dentre as informações contidas nesses arquivos indexadores, estão meta-dados do objeto (ex.: tamanho e

nome do objeto) e o endereço do rastreador (*tracker*). O rastreador é o responsável por coordenar a distribuição do objeto. A principal função do rastreador é fornecer informação para que um determinado par possa se conectar a outros pares que possuam o objeto e assim realizar o *download*.

Com seu modo eficiente de coordenar a distribuição de objetos e delegando a pesquisa dos objetos para sítios da *Internet*, o BitTorrent é atualmente um dos sistemas P2P mais populares [14].

2.2 Comportamento Malicioso em Sistemas Par-a-Par

Esta seção apresenta os principais padrões de comportamento malicioso observados nos sistemas P2P. A Seção 2.2.1 apresenta trabalhos sobre esses padrões de comportamentos maliciosos estudados. A Seção 2.2.2 apresenta estudos realizados sobre a poluição de conteúdo, padrão de comportamento malicioso foco desta dissertação.

2.2.1 Padrões de Comportamento Malicioso

Os sistemas P2P se tornaram muito populares nos últimos anos. Porém, a grande popularidade também atraiu usuários maliciosos que tentam tirar proveito dos sistema. Dentre os padrões de comportamento malicioso observados nos sistemas P2P o problema da falta de cooperação (*free-riding*) é o mais estudado [1, 72]. Nesse padrão de comportamento os usuários não compartilham seus objetos com os outros usuários do sistema. Adar *et alii* [1] mediram a quantidade de pares egoístas no Gnutella e descobriu que 70% dos pares não cooperam. Saroiu *et alii* [72] encontraram que 25% dos usuários do sistema Gnutella não cooperam.

Também têm sido observados em sistemas reais a introdução de vírus/*worms* [37, 78] e ataques de negação de serviço (*Denial of Service - DOS*). A introdução e propagação de vírus foi analisada analiticamente por [78], enquanto Kalafut *et alii* descobriram que 68% dos objetos executáveis no Gnutella contêm *worms*. Liang *et alii* analisaram um tipo de ataque DOS no qual pares inexistentes são indexados como fonte de objetos, fazendo com que haja uma grande demora para que os *donwloads* sejam realizados.

Além desses, outros padrões de comportamentos também foram estudados. Um desses comportamentos é o conluio. O termo conluio significa a cooperação de pares para realizar alguma ação maliciosa [53]. Em particular, o termo é freqüentemente utilizado para representar ações conjuntas que tentam subverter estratégias de combate a comportamentos maliciosos [28].

No ataque *Sybil*, o par malicioso cria várias identidades (instâncias de um cliente P2P) para conseguir uma grande influência no sistema P2P. Esse é ataque é facilitado pela forma barata na qual os sistemas P2P deixam que identidades sejam criadas [23]. Esse ataque pode ser utilizado para realizar um ataque de conluio. O nome do ataque foi criado a partir do livro *Sybil* [73] que conta a história verídica de uma mulher, referenciada no livro pelo pseudônimo de *Sybil Dorset*, e sua desordem de múltiplas personalidades.

Assim como o *Sybil*, o ataque *whitewashing* se aproveita da facilidade da criação identidades nos sistemas P2P. Porém, ele usa essa facilidade para que um par possa trocar a sua identidade sempre que o identificarem como malicioso. Feldman *et alii* [29] analisaram analiticamente pares egoístas utilizando o ataque *whitewashing* para trocar suas identidades e não serem identificados como maliciosos. O estudo concluiu que a imposição de penalidades para novos pares no sistema reduz o efeito do ataque. Porém, se os usuários do sistema possuírem um comportamento muito dinâmico, o desempenho do sistema é afetado.

Finalmente, no ataque de traidor [53] um par malicioso se comporta adequadamente para que os outros pares do sistema confiem nele. Adquirindo uma boa confiança pelos outros pares, o par malicioso então aproveita desta confiança para atacar o sistema.

2.2.2 A Poluição de Conteúdo

Recentemente, uma nova forma de comportamento malicioso, conhecido como poluição de conteúdo, foi observada em sistemas P2P de compartilhamento de arquivos. Poluição de conteúdo, ou simplesmente poluição, é a inserção de objetos corrompidos nos sistemas P2P. O objetivo do ataque é tornar mais difícil achar e obter objetos não poluídos, fazendo com que os usuários percam a confiança no sistema.

Liang *et alii* [45] apresentaram o problema da poluição de conteúdo e realizou diversas medições no sistema Kazaa. Foi observado que a fração de cópias poluídas de alguns objetos populares pode ser maior que 80%. Além disso, o estudo apresentou o mecanismo de introdução de poluição por *inserção de versões falsas*. Nesse mecanismo os poluidores inserem no sistema versões de objetos com o mesmo meta-dado de objetos não poluídos já presentes no sistema. A pesquisa pelo objeto retorna as versões poluídas e não poluídas, fazendo com que os pares tenham problemas para realizar *download* de cópias não corrompidas. Os autores afirmam que para esse método ser bem sucedido os poluidores necessitam ter alta disponibilidade, uma grande quantidade de banda disponível e de clientes conectados no sistema disseminando poluição. A grande quantidade de clientes necessária para que as versões poluídas sejam populares no sistema pode ser conseguida através de um ataque *Sybil*.

Desde esse primeiro estudo, diversos foram os esforços endereçando o problema da disseminação de conteúdo poluído. Christian *et alii* [16] também mediram a poluição em sistemas P2P e mostrou indícios de que a poluição de conteúdo está presente, não somente no Kazaa, mas também nos sistemas eDonkey/Overnet e no Gnutela. Esse estudo também mostrou, através de simulação, que a disponibilização de objetos poluídos por um único par não têm impacto no sistema. Porém, a poluição intencional, chamada de *envenenamento* nesse estudo, é um problema para os sistemas P2P.

Lee *et alii* [44] conduziram uma pesquisa entre estudantes de universidades, analisando seus comportamentos ao utilizar o sistema P2P Kazaa. O estudo mostrou que 70% dos usuários pesquisados já haviam realizado *downloads* de objetos poluídos. Além disso, o estudo conclui que um grande número de usuários compartilham por descuido os objetos poluídos obtidos e que cerca de 40% dos usuários cometem erros quando tentam identificar os objetos poluídos em suas pastas de compartilhamento.

Pouwelse *et alii* [63] analisaram o impacto da inserção de arquivos *.torrent* corrompidos no extinto sítio Suprnova. Na época que o estudo foi realizado, o Suprnova era um dos maiores sítios indexadores de arquivos *.torrent*. Foi constatado nesse estudo que um pequeno número de moderadores do sítio conseguiu evitar que os objetos poluídos fossem difundidos pelos autores. Porém, o estudo não fornece detalhes da quantidade de objetos poluídos que foram inseridos e também não ouve uma medição dos objetos poluídos já presentes no sistema.

Kumar *et alii* [42] criaram um modelo analítico para analisar a velocidade da disseminação de poluição. O modelo criado é baseado em um sistema de equações diferenciais não-lineares. O modelo consegue capturar comportamentos interessantes dos usuários do sistema, incluindo o problema de falta de cooperação e o fato de os usuários deixarem o conteúdo poluído obtido em sua pasta de compartilhamento.

Thommes *et alii* [78] também criaram um modelo para analisar a velocidade da disseminação de conteúdo poluído. O modelo consiste em um sistema de equações diferencias não-lineares. Os autores concluiram que o número de atacantes que introduzem o objeto poluído têm grande impacto na disseminação de poluição a longo prazo. No capítulo 4 desta dissertação é apresentado um modelo analítico, utilizado para validar o simulador desenvolvido, que é baseado nesse modelo.

2.3 Estratégias de Combate a Comportamento Malicioso

Esta seção apresenta os estudos realizados para combater o comportamento malicioso nos sistemas P2P. A Seção 2.3.1 apresenta trabalhos que propuseram estratégias de combate aos padrões de comportamento malicioso. A Seção 2.3.2 apresenta, especificamente, as principais estratégias de combate contra o problema da disseminação de conteúdo poluído. Finalmente, a Seção 2.3.3 apresenta com detalhes o Sistema de Reputação Credence, criado para reduzir a disseminação da poluição. O Credence é apresentado com detalhes pois será avaliado e comparado com as estratégias de combate propostas nesta dissertação.

2.3.1 Combatendo Comportamento Malicioso

Nos sistemas P2P, as estratégias mais utilizadas para o combate aos padrões de comportamento malicioso são os mecanismos de reputação. Os sistemas de reputação foram criados para que os usuários de um sistema dêem sua avaliação sobre outros usuários ou componentes do sistema. De acordo com Resnick *et alii* [67], um sistema de reputação é responsável por coletar, distribuir e agregar opiniões sobre o passado dos participantes. Desta forma, os sistemas de reputação visam ajudar os usuários a descobrir quem é confiável e também encoraja os participantes agirem de forma honesta, pois eles estarão sendo classificados por seus iguais.

Um dos sistemas de reputação mais populares é o utilizado pelo sítio de leilões virtuais *eBay* [26]. No *eBay*, os participantes de um leilão (comprador e vendedor) se reputam ao fim de uma transação e estas reputações são disponibilizadas para os outros usuários do sistema. A reputação atribuída indica se o comprador/vendedor ficou satisfeito ou não com a transação. A popularidade e grande quantidade de leilões realizadas diariamente pelo eBay são atribuídas principalmente ao seu sistema de reputação [67].

Nos sistemas P2P, os principais sistemas de reputação foram desenvolvidos para minimizar o impacto da falta de cooperação dos pares. O problema da falta de cooperação existe na maioria dos serviços que usam a arquitetura P2P. Porém, pela grande popularidade dos sistemas P2P de compartilhamento de arquivos, esse problema foi evidenciado nesses sistemas. Diversos sistemas de reputação foram propostos para tratar esse problema nesse tipo de serviço P2P [13, 24, 28, 38, 77].

Apesar da grande quantidade de estratégias baseadas em reputação para sistemas P2P de compartilhamento de arquivos, algumas estratégias foram propostas para combater o comportamento egoísta em outros serviços P2P. O Lockss (*Lots of Copies Keep Stuff Safe*) [66] é um

sistema distribuído de preservação de documentos digitais. Uma das funcionalidades do Lockss é um mecanismo utilizado para verificar se uma determinada cópia local está corrompida. O mecanismo utiliza votos dos participantes do sistema para determinar a integridade da cópia. Para garantir que os participantes cooperem em uma votação, o sistema Lockss utiliza um mecanismo de reputação [52]. Nesse mecanismo cada participante tem um lista de pares que podem estar em débito, igualado ou crédito, em relação aos votos. Pares em débito têm um probabilidade menor de serem atendidos quando requisitam uma votação.

O *Ourgrid* [59] é um sistema P2P utilizado para o compartilhamento de processamento entre os pares do sistema. Para evitar a falta de cooperação dos usuários nesse sistema, Andrade *et alii* [4] propuseram o mecanismo de reputação *Network of Favors*. Nesse mecanismo, cada serviço prestado é visto como um favor. Cada par mantém um lista de pares para quem fez favores e para quem deve favores. Desta forma, um par prioriza os participantes a quem ele deve favores para compartilhar seus recursos computacionais.

Rocha *et alii* [70, 71] criaram um mecanismo de reputação para combater o comportamento egoísta em redes sobrepostas de roteamento (*Overlay Networks*). Nesse sistema de reputação distribuído e descentralizado, os pares atribuem reputação uns para os outros localmente. A reputação local dos pares é compartilhada com o resto da rede, requisito essencial para a rápida convergência de sistemas de reputação. Além disso, esse sistema pune os pares egoístas, criando incentivos para que eles se recuperem e comecem a cooperar. Esse sistema inspirou o sistema de reputação *Scrubber* proposto nesta dissertação e apresentado no capítulo 5.

2.3.2 Combatendo Poluição de Conteúdo

Vários trabalhos discutiram estratégias para reduzir a disseminação de conteúdo poluído [16, 42, 45, 74, 78]. Porém, somente algumas soluções práticas foram propostas. Liang *et alii* [47], tomando como base seu estudo anterior [45], criaram um sistema de lista negras de pares poluidores. Ao coletar objetos poluídos no sistema Kazaa, os autores observaram que os responsáveis por disseminar o conteúdo poluído possuíam algumas características em comum. Primeiramente, os pares poluidores executam diversas instâncias dos clientes P2P, tipicamente localizados em fazendas de servidores (*server farms*). Além disso, cada poluidor compartilha muitos objetos poluídos relativos a um determinado título. Tendo feito estas observações, foi proposto um método para identificar automaticamente os poluidores, colocando-os em uma lista negra. Apesar dos bons resultados em identificar os poluidores, esta estratégia necessita de um *crawler* para coletar dados do sistema constantemente.

Silva *et alii* propõem um sistema no qual os usuários votam na autenticidade dos objetos do

sistema, determinando se eles são poluídos ou não [74]. Porém, a decisão de realizar *download* de um determinado objeto não depende da vontade do usuário e sim de um sistema de contenção de *download*. O sistema limita a taxa de *download* de um determinado objeto, de acordo com os votos que os usuários atribuíram a esse objeto no passado. Uma baixa taxa de *download* é atribuída aos objetos considerados poluídos, limitando a disseminação de tal objeto. A proposta é bastante interessante por tirar do usuário a decisão de realizar *download* de um objeto corrompido.

Os sistemas de reputação também têm sido utilizados para combater a poluição de conteúdo em sistemas P2P. No sistema de reputação Eigentrust [39], os pares atribuem reputação uns para os outros como fonte de conteúdo poluído. O sistema mantém uma reputação global para cada par do sistema. A reputação global é calculada a partir das reputações locais, que são baseadas nas interações (*download*) entre os pares do sistema. O cálculo e disponibilização da reputação global dos pares é responsabilidade de um conjuntos de pares pré-confiáveis. Apesar de ser um sistema eficiente, a necessidade de um conjunto de pares pré-confiáveis para computar a reputação global levanta algumas considerações práticas que precisam ser avaliadas, por exemplo como a seleção de tais pares é realizada.

O sistema de reputação XREP [21] atribui reputação para os pares (como fontes ou não de objetos poluídos) e classifica os objetos quanto a sua autenticidade (como cópias poluídas ou não). Antes de cada *download*, um par coleta na rede votos sobre o objeto desejado. Os votos coletados são agregados e são utilizados para determinar se o objeto é poluído. Caso o objeto não seja considerado poluído, as fontes com maior reputação são então selecionadas. Depois que o *download* termina, o par vota na autenticidade do objeto e atribui reputação para as fontes que enviaram o objeto. Apesar de ser um sistema interessante, o XREP usa somente a reputação local para classificar os usuários. Porém, aprender com a experiência de outros pares é essencial na convergência rápida de sistemas de reputação nos sistemas P2P de compartilhamento de arquivos [71]. Além disso, não é claro como o sistema promove a reabilitação de usuários considerados poluidores ou como evitar *downloads* de pares com baixa reputação.

Além desses sistemas de reputação, existe o sistema Credence [80–82], o qual classifica os objetos do sistema. O Credence, por ser um dos sistemas analisados nesta dissertação, é apresentado com maiores detalhes na próxima seção.

2.3.3 O Sistema de Reputação Credence

O *Credence* [80–82] é um sistema descentralizado e distribuído, no qual os usuários classificam[2] os *objetos* obtidos quanto à sua autenticidade (poluídos ou não). Seu funcionamento é baseado em um protocolo de *pesquisa por votos*, usado para disseminar a classificação dos objetos pelo sistema e um esquema de correlação de votos que atribui mais peso para votos vindos de pares que tendem a ter a mesma opinião.

O sistema funciona da seguinte forma. Antes de um par A realizar o *download* de um objeto, ele dispara uma *pesquisa por votos* no sistema para coletar votos sobre o objeto desejado. Os votos coletados podem ser -1, se o par considerar o objeto *poluído*, ou 1, caso contrário. A classificação do objeto é computada, ponderando cada voto coletado pelo *relacionamento* que A possui com o dono do voto.

O *relacionamento* entre dois pares é expresso pela *correlação* de seus históricos de votos e captura se os pares tendem a votar de forma semelhante (correlação positiva), diferente (correlação negativa) ou se eles têm um histórico de votos não correlacionado. A correlação entre os pares i e j é computada da seguinte forma:

$$\theta(i,j) = \frac{(P - IJ)}{\sqrt{I(1-I)J(1-J)}} \tag{2.1}$$

onde I é a fração de votos positivos dados por i no passado, J a fração de votos positivos dados por j e P a fração em que ambos pares votaram positivamente.

A fim de computar a classificação de um objeto, o par i pondera o voto do par j por $r_{i(j)}$:

$$r_{i(j)} = \begin{cases} \theta(i,j) & \text{se } |\theta(i,j)| \geq 0,5 \\ \\ 0 & \text{caso contrário} \end{cases} \tag{2.2}$$

Note que o peso $r_{i(j)}$ é fixado em 0 sempre que i e j possuírem um histórico de votos não correlacionado, ou seja, nesse caso os votos de j são desconsiderados pelo par i.

Cada par sempre armazena localmente todos os votos coletados em um *banco de dados de votos*, sem levar em consideração se o objeto foi obtido ou não. Além disso, todas as *correlações*

[2] No trabalho original, os autores utilizam o termo *reputação de objetos* para categorizar o sistema Credence. Porém, como o objeto é um componente estático e não tem capacidade de mudar a sua reputação, esta dissertação utilizará o termo *classificação de objetos*. Todavia, o Credence ainda será referenciado como um sistema de reputação.

fortes ($|\theta(i,j)| \geq 0,5$) encontradas são armazenadas localmente em uma *tabela de correlação* que é atualizada periodicamente a partir do *banco de dados de votos*.

O *Credence* também executa um *protocolo de fofoca* para descobrir *correlações transitivas*. As correlações transitivas são correlações descobertas por outros pares no sistema. Periodicamente, um par *i* seleciona aleatoriamente um par j e obtém os coeficientes de correlação conhecidos por j. As correlações encontradas são então computadas multiplicando os coeficientes de correlação obtidos pelo peso $r_{i(j)}$. Todas as correlações transitivas fortes também são armazenadas na tabela de correlação.

3 Contextualiazação e Modelagem

Este capítulo apresenta a modelagem proposta para avaliação e estudo da disseminação de conteúdo poluído em sistemas P2P. A seção 3.1 introduz a terminologia utilizada. A seção 3.2 discute os mecanismos de introdução de poluição em sistemas P2P, a saber, a *inserção de versões falsas* [45] e a *corrupção do identificador*, este último descrito com detalhes neste trabalho. Finalmente, a seção 3.3 apresenta o modelo de simulação criado para modelar o problema e as soluções propostas, bem como quais são as suas principais premissas e parâmetros.

3.1 Componentes de um Sistema Par-a-Par

Esta seção apresenta uma visão geral do funcionamento de um sistema P2P de compartilhamento de arquivos, introduzindo a terminologia abordada em toda a dissertação.

Os dois principais componentes de um sistema P2P de compartilhamento de arquivos são os **pares** e os **arquivos**.

Os pares representam os **usuários** conectados ao sistema P2P. Esses usuários se conectam e fazem parte do sistema através das **aplicações** ou **clientes** P2P. Tais clientes são partes de *software* que os usuários executam em suas **máquinas** (computadores pessoais) e são responsáveis pelas tarefas de disponibilizar os arquivos locais e obter arquivos do sistema.

Arquivo é o conteúdo disponibilizado no sistema P2P. Esta dissertação assume que um arquivo específico (um documento, *software*, música ou vídeo) é chamado de **título**. Cada título possui um determinado número de **versões**, que são diferentes representações binárias do título. Por exemplo, se um determinado título representa uma música, diferentes versões poderiam ser criadas através da codificação do conteúdo em diferentes formatos (ex.: *mp3* [56] e *ogg vorbis* [79]) ou taxas (ex.: 128 Kbps ou 190 Kbps). Analogamente, versões de aplicativos poderiam ser disponibilizadas em diferentes algoritmos de compactação de dados (ex.: *gzip* [34] ou *rar* [64]), ou versões de documentos poderiam ser disponibilizados em diferentes formatos (ex.: *PDF* [61] ou *PostScript* [62]). Alguns títulos podem possuir centenas e até milhares de

versões [45].

Cada versão possui uma determinada quantidade de **cópias**. Cada cópia é mantida por um par, que para compartilhá-la inclui tal cópia em uma área em seu computador chamada de **pasta de compartilhamento** (*shared folder*). Os pares que compartilham cópias de uma mesma versão são chamados de **fontes** da versão. Nesta dissertação, o termo **objeto** será utilizado para se referir a uma versão ou sua cópia disponibilizada em um par. A diferença ficará clara no contexto em que o termo for utilizado.

Toda versão possui um **identificador**. Esse identificador é a assinatura da versão e permite às aplicações identificarem versões diferentes. Para tanto, os sistemas P2P assumem que esses identificadores são únicos. Os identificadores da versão são tipicamente gerados a partir do conteúdo da versão utilizando algoritmos de *hash*, tais como o MD5 [69] e SHA-1 [25]. Uma explicação mais detalhada sobre a geração de identificadores será dada na seção 3.2.2.

Quando um par deseja realizar o **download** de uma versão, esse primeiramente faz uma pesquisa por um título, e o sistema P2P retorna as versões encontradas, tipicamente ordenadas pelo número de fontes (e consequentemente cópias) de forma decrescente. O par escolhe a versão que deseja obter e contacta as fontes responsáveis pela versão. O *download* da versão pode ser realizada recebendo os dados a partir de uma única fonte ou recebendo **pedaços de dados** de **múltiplas fontes** (*swarm download* [9]). O recebimento a partir de **múltiplas fontes** é mais popular atualmente, pois otimiza o processo de *download* da versão evitando que uma fonte com pouca banda disponível seja responsável pelo envio de todo o objeto.

3.2 Mecanismos de Introdução de Conteúdo Poluído

Os objetos compartilhados no sistema P2P que têm o seu conteúdo corrompido são chamados de **objetos poluídos** ou simplesmente **poluição**. Os pares maliciosos que disseminam a poluição são chamados de **poluidores ativos**. Os outros pares são chamados de **pares não maliciosos**. Geralmente, as cópias poluídas obtidas pelos pares não maliciosos são mantidas por um longo tempo nas pastas de compartilhamento [16] e muitos mantêm, por descuido, essas cópias compartilhadas mesmo depois de verificarem que seu conteúdo é poluído [47]. Logo, os pares não maliciosos que obtiveram objetos poluídos e os compartilharam podem se transformar em **poluidores passivos**. Sempre que o termo **poluidor** for utilizado, ele estará se referindo aos dois tipos de pares poluidores, ativos e passivos.

Dois mecanismos são utilizados pelos poluidores ativos para introduzir seus objetos poluídos. O modelo da *inserção de versões falsas* é apresentado na seção 3.2.1. O modelo da

corrupção do identificador, mecanismo descrito em detalhes neste trabalho, é apresentado na seção 3.2.2.

3.2.1 Inserção de Versões Falsas

A *inserção de versões falsas* foi estudada e apresentada por Liang *et alii* [45]. A introdução de poluição pela *inserção de versões falsas* funciona da seguinte forma:

1. O poluidor ativo cria versões poluídas de um determinado título. Isto é realizado disponibilizando versões poluídas com o mesmo meta-dado das versões não poluídas do título. Esta dissertação assume que o meta-dado são palavras chaves utilizadas para referenciar o objeto selecionado (ex.: nome do objeto ou, no caso de uma música, nome da faixa ou artista).

2. O poluidor ativo disponibiliza as versões criadas em clientes conectados no sistema P2P.

3. Quando os pares do sistema P2P procurarem pelo título, o sistema irá retornar as versões disponíveis, incluindo tanto as versões poluídas quanto as não poluídas. Como não existe um mecanismo automático para detectar poluição, o resultado da busca não indica aos pares quais versões são poluídas e quais não são.

4. Alguns pares obtêm versões poluídas. As cópias obtidas geralmente são disponibilizadas automaticamente pelo cliente P2P. O usuário que visualiza a objeto poluído obtido pode apaga-lo. O par que não apaga o conteúdo obtido se torna um poluidor passivo.

5. As versões poluídas, com a cooperação dos poluidores passivos, se espalham pelo sistema P2P fazendo com que outros pares continuem realizando *download* das versões corrompidas.

3.2.2 Corrupção do Identificador

Como discutido na seção 3.1, os sistemas P2P assumem que os identificadores das versões são únicos. Entretanto, existe a possibilidade dessa premissa não ser verdadeira. Nesse caso, duas versões diferentes poderiam ter o mesmo identificador. Caso isto ocorra, a integridade dos dados de uma cópia obtida a partir de *múltiplas fontes* poderia estar comprometida, pois tais dados poderiam provir de versões diferentes, algumas poluídas e outras não. Logo, os poluidores ativos podem se aproveitar desse fato para introduzir versões poluídas com o mesmo

identificador de versões não poluídas já presentes no sistema. Esse método, introduzido nesta seção, é denominado de *corrupção do identificador*.

O identificador de uma versão é tipicamente criado a partir do seu conteúdo. Uma maneira de criar o identificador, utilizada pelos sistemas P2P atuais, é aplicar nos dados do objeto algoritmos de *hash* (ex.: MD5 e SHA-1). Portanto, uma maneira de introduzir objetos poluídos através da *corrupção do identificador* seria subverter os algoritmos de *hash*.

Contudo, existem maneiras mais fáceis de introduzir poluição pelo método da *corrupção do identificador*. Visando diminuir a sobrecarga de processamento dos clientes P2P, alguns sistemas [40] não utilizam **todo** o conteúdo do objeto para criar o seu identificador. Nesse caso, as partes do objetos que não são utilizadas podem ser corrompidas. Esse é o caso do Kazaa, que utiliza o algoritmo *UUHash* [83] para criar os identificadores das versões. A Figura 3.1 ilustra o funcionamento do *UUHash*. Ele utiliza os primeiros 300KB de dados do objeto e gera uma chave MD5. A partir daí novas chaves são geradas a cada 2^nMB de dados, sendo n um inteiro que é incrementado cada vez que uma chave é criada. Essas novas chaves são geradas com o algoritmo *smallhash*[1] usando 300KB de dados. A chave MD5 e todas as outras são então concatenadas para criar um identificador de 160 bits para a versão. Se o *UUHash* for utilizado para criar identificadores para objetos de áudio (ex.: 5MB) e vídeo (ex.: 700MB), a porcentagem dos dados utilizados para criar os seus identificadores com *UUHash* seria 12% e 0,5%, respectivamente. Dessa forma, seria possível corromper 88% e 99,5% dos dados desses objetos, sem alterar o identificador da versão.

Objetos poluídos podem ser introduzidos no sistema pela *corrupção do identificador* da seguinte maneira:

1. O poluidor ativo obtém uma versão não poluída, de preferência popular, do sistema P2P. O par então corrompe os dados de sua cópia de forma que o identificador da versão permaneça o mesmo.

2. Feito isso, o poluidor ativo disponibiliza as cópias alteradas da versão em máquinas conectadas no sistema P2P, assim como é feito na *inserção de versões falsas*.

3. Os usuários realizam uma pesquisa. Porém, diferentemente da *inserção de versões falsas*, as versões retornadas pelo sistema poderão possuir tanto cópias não poluídas quanto cópias poluídas.

4. Se um par decidir obter uma das versões que possui cópias poluídas, qualquer pedaço

[1] O algoritmo *smallhash* é idêntico ao algoritmo de CRC utilizado no formato digital de imagem PNG [12]

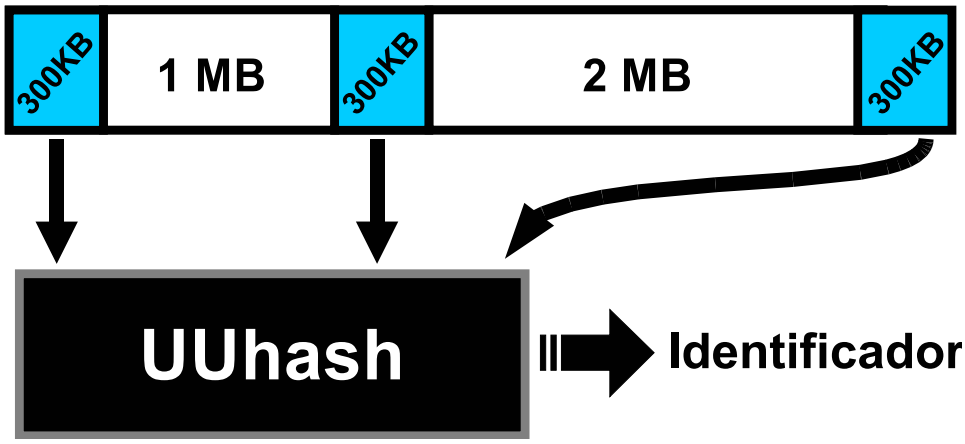


Figura 3.1: Kazaa Utilizando o UUHash para Gerar Identificadores

recebido de uma fonte poluidora poderá comprometer a cópia obtida. Arquivos binários (ex.: jogos e documentos compactados) seriam totalmente inutilizados. No caso de arquivos de música e vídeo, somente o trecho corrompido não seria reproduzido por um tocador. Porém, isso também é indesejado.

5. Assim como na *inserção de versões falsas* os pares que não apagam suas cópias poluídas obtidas se tornam poluidores passivos e os pedaços corrompidos de suas cópias podem acabar inutilizando futuras cópias obtidas por outros pares.

Com a *corrupção do identificador* os poluidores ativos não necessitam de diversos clientes para tornar as cópias poluídas populares, já que eles podem corromper versões que já são populares no sistema. Dessa forma, os poluidores ativos conseguem disseminar objetos poluídos mais facilmente com esse mecanismo do que com a *inserção de cópias falsas*, como será mostrado no capítulo 4.

Apesar de não existirem estudos caracterizando a *corrupção do identificador* em sistemas reais, diversos usuários na *Internet* reportaram que tal mecanismo é utilizado na rede do *FastTrack* [60].

3.3 Modelo de Simulação

Esta seção apresenta o modelo de simulação utilizado para modelar o problema e as soluções para a disseminação de conteúdo poluído. Foi desenvolvido e implementado um simulador orientado a eventos capaz de capturar os principais aspectos do sistema e do comportamento

dos usuários que influenciam o resultado da análise da disseminação de conteúdo poluído nos sistemas P2P de compartilhamento de arquivos. O simulador foi escrito na linguagem de programação ANSI C [41] e consta de aproximadamente 16000 linhas de código. Programas e módulos auxiliares foram escritos na linguagem *Python* [43] para sumarizar os dados obtidos pelo simulador.

O modelo foi criado para avaliar o impacto da disseminação da poluição após a introdução de conteúdo poluído, além de como as estratégias de combate conseguem reduzir a poluição do sistema. Portanto, as duas métricas principais utilizadas nessa avaliação são a *eficiência* e *convergência*, observadas através da fração de *downloads* não poluídos ao longo do tempo. A primeira está relacionada com a eficiência de uma estratégia de combate para reduzir a disseminação de poluição e é medida a partir da fração de *downloads* não poluídos ao longo do tempo. A outra métrica está relacionada com tempo necessário para que tal estratégia alcance a sua eficiência máxima.

As próximas seções descrevem os componentes do modelo de simulação.

3.3.1 Modelo do Sistema P2P

O interesse desta dissertação é avaliar o processo de disseminação dos objetos poluídos através do sistema. Portanto, o foco do modelo de sistema P2P é o *download*. Dessa forma, aspectos não cruciais puderam ser simplificados. As principais premissas desse modelo são:

- Nenhum sistema P2P específico é modelado.
- É assumido que a pesquisa por objetos e pares, assim como o roteamento dessas pesquisas, é perfeito. Isso significa que todos os pares e objetos no sistema são visíveis.
- O *download* é realizado a partir de múltiplas fontes (F) aleatoriamente selecionadas dentre os pares elegíveis, ou seja, pares que estejam presentes no sistema e que não foram banidos por alguma estratégia de combate à poluição.
- O foco do sistema é na propagação dos objetos e não no desempenho do *download*. Portanto, é assumido que o tempo de *download* é desprezível.

3.3.2 Modelo dos Objetos

Existem no sistema T títulos únicos, cada qual com V versões. No início da simulação, os objetos compartilhados por um par são escolhidos primeiramente selecionando o título, e a

seguir a versão. Ambas seleções são realizadas seguindo uma distribuição Zipf [87], na qual a probabilidade de acessar o objeto i é igual a C/i^{α}, onde $\alpha > 0$, e C é uma constante normalizadora. Esta dissertação utiliza o parâmetro $\alpha = 0,8$ tanto para a seleção dos títulos quanto para as versões [45].

Com a simulação em execução, a escolha de objetos é realizada primeiramente selecionando o título, com a distribuição Zipf, e então selecionando a versão a partir da sua popularidade atual no sistema, ou seja, o número de cópias. Para conseguir avaliar a eficácia das estratégias de combate em reduzir a poluição depois que ela é introduzida, não é modelada a adição de novos objetos durante a simulação.

Os objetos modelados possuem um identificador e a porcentagem H dos seus dados que estão poluídos. Um objeto é considerado poluído se ele possui alguma porcentagem dos seus dados poluídos ($H > 0$). A porcentagem dos dados poluídos de um objeto é um parâmetro importante para avaliar a disseminação de poluição pelo mecanismo de *corrupção do identificador*. Se um par possuir um objeto poluído, o qual possui uma certa porcentagem de dados não corrompidos, ele poderia enviar *pedaços de dados* não poluídos para outro par. Isso acaba influenciando a velocidade do mecanismo de *corrupção do identificador* disseminar poluição, como será analisado no capítulo 4. Para a *inserção de versões falsas* é considerado que a porcentagem dos dados poluídos é igual a 100%.

3.3.3 Modelo dos Pares

Foram modeladas duas classes de pares.

Pares não maliciosos: Existem N_b pares não maliciosos no sistema. No início da simulação os pares não maliciosos compartilham O_b objetos não poluídos. Esses pares podem realizar *downloads*, e deixar ou entrar no sistema. Os pares não maliciosos realizam *downloads* a uma taxa $\lambda_{\text{download}}$ seguindo uma distribuição exponencial [33]. Todos os objetos obtidos pelos pares são compartilhados e um par nunca realiza *download* de um objeto que já possua, assim como foi mostrado por Gummadi *et alii* [33]. O comportamento dinâmico dos pares, ou seja, taxa de entrada (λ_{entrada}) e saída ($\lambda_{\text{saída}}$) do sistema, também é modelado através de uma distribuição exponencial. A Função de Densidade de Probabilidade da distribuição exponencial utilizada nesta dissertação é dada por $\lambda\ e^{-\lambda\ x}$, onde λ é a taxa.

Poluidores ativos: Existem N_p poluidores ativos no sistema. No início da simulação os poluidores ativos compartilham O_p objetos poluídos. Esses pares não realizam *download*, somente

os compartilham. Eles também têm uma alta disponibilidade [47] e estão ativos no sistema durante todo o tempo de simulação. Além disso, os poluidores ativos **nunca** apagam seus objetos poluídos.

Todas as estratégias de combate à poluição apresentadas nesta dissertação necessitam que o usuário retorne sua opinião sobre o *download* realizado (poluídos ou não). Tal opinião é importante para que as estratégias possam classificar os pares do sistema (como fontes ou não de objetos poluídos) e a autenticidade dos objetos compartilhados (como cópias poluídas ou não). Portanto, todos os pares não maliciosos possuem uma probabilidade, $p_{\text{opinião}}$, de fornecer sua opinião ao sistema logo após o *download* de um objeto.

Em certas ocasiões, as opiniões retornadas pelos pares não maliciosos podem não refletir a situação real do *download* realizado (ex.: opinar que um objeto recebido é poluído quando isto não é verdade). Logo, os pares não maliciosos têm uma probabilidade p_{erro} de fornecer uma opinião que não reflete a situação real do *download*.

Além disso, algumas estratégias de combate à poluição atribuem incentivos para que os pares não maliciosos, que obtiveram objetos poluídos, apaguem tais objetos de suas pastas de compartilhamento. Portanto, seja δ_i a probabilidade de par não malicioso, *i*, reagir ao incentivo aplicado pela estratégia de combate. Foram criados três modelos de reação aos incentivos, utilizando o parâmetro δ_i. O modelo de δ_i *fixo*, o mais simples dos três, atribui uma probabilidade fixa para o par não malicioso, *i*, reagir ao incentivo. Os modelos de *crescimento linear* e *quadrático*, buscando um maior realismo, foram criados inspirados no fato que a reação dos usuários aos incentivos pode evoluir. Nesse caso, a probabilidade de reação aos incentivos aumenta de acordo com o número de incentivos recebidos. A equação 3.1 mostra a evolução do parâmetro δ_i, com relação aos incentivos recebidos:

$$\delta_i = \begin{cases} p & \text{Fixo} \\ min(\delta^0 \times n, 1) & \text{Crescimento Linear} \\ min(\delta^0 \times n^2, 1) & \text{Crescimento Quadrático} \end{cases} \tag{3.1}$$

No modelo *fixo*, δ_i permanece constante durante toda a simulação e p representa a probabilidade de reação do par *i* ao incentivo recebido. Nos modelos de *crescimento linear* e *quadrático*, o valor de δ_i evolui de acordo com o parâmetro n, onde n representa o número de incentivos recebidos no passado por *i*. Nesta dissertação é considerado que $\delta^0 = 0,1$. A implementação de mecanismos de incentivos, além de como os pares não maliciosos reagem aos incentivos

recebidos, difere para cada estratégia de combate.

3.3.4 Modelo da Introdução de Conteúdo Poluído

Foram simulados os dois tipos de introdução de conteúdo poluído: a *inserção de versões falsas* e a *corrupção do identificador*. A diferença essencial de ambos mecanismos está na forma como objetos iniciais são disponibilizados.

No mecanismo de *inserção de versões falsas*, todas as versões do sistemas são aleatoriamente divididas em dois grupos: o grupo das versões poluídas e o grupo das versões não poluídas. Os objetos compartilhados inicialmente pelos poluidores ativos são selecionados do grupo de versões poluídas e, analogamente, os objetos dos pares não maliciosos são selecionados do grupo de versões não poluídas. A escolha inicial das versões funciona como explicado na seção 3.3.2, porém a probabilidade da seleção de uma versão é condicionada pelo grupo na qual ela deve pertencer.

Na *corrupção do identificador*, como mostrado na seção 3.2.2, todas as versões podem possuir cópias poluídas. Logo, nesse mecanismo não existe uma distinção entre versões poluídas e não poluídas, ou seja, tanto os poluidores ativos quanto os pares não maliciosos utilizam o mesmo conjunto de versões para escolher as suas cópias iniciais. Porém, as cópias escolhidas pelos poluidores ativos possuem a porcentagem poluída dos dados (H) diferente de zero e as cópias escolhidas pelos pares não maliciosos possuem $H = 0\%$. Isso implica que cada versão irá possuir cópias poluídas e não poluídas.

3.3.5 Modelo de Comportamento Malicioso

Também foram modelados cenários nos quais os pares poluidores utilizam de outras estratégias para tornar a disseminação do conteúdo poluído mais difícil de combater. Foram modelados os ataques de conluio, *Sybil* e *whitewashing*.

No modelo de ataque por conluio, os pares maliciosos tentam subverter as estratégias de combate à poluição baseadas em reputação. Todas as estratégias baseadas em reputação apresentadas nessa dissertação permitem aos pares compartilharem suas reputações locais com os outros pares do sistema. Logo, os poluidores ativos podem compartilhar valores de reputações que não são verdadeiros. O modelo de conluio proposto nesse trabalho faz com que os poluidores ativos ajam conjuntamente atribuindo o valor máximo de reputação para todos os outros pares maliciosos do sistema. Além disso, eles também atribuem o nível mínimo de reputação a todos os pares não maliciosos, difamando-os. Para o Credence, apresentado no capítulo 2, esses

valores são atribuídos na *correlação*, que têm o valor máximo de $\theta = 1$ e o mínimo $\theta = -1$. Os valores máximos e mínimos das outras estratégias de reputação avaliadas nesta dissertação serão apresentados no capítulo 5.

No modelo de ataque *Sybil*, cada poluidor ativo é capaz de inserir *r* instâncias de si mesmo no sistema. Todas as instâncias compartilham os mesmos objetos. Esse modelo de ataque *Sybil* aumenta o número de cópias de uma versão poluída, fazendo assim com que elas fiquem mais populares.

Finalmente, no ataque *whitewashing*, o poluidor ativo troca a sua identidade para que os sistemas de reputação não sejam capazes de identificá-lo como malicioso. Todas as estratégias de reputação apresentadas nesta dissertação possuem níveis de reputação que classificam os pares como maliciosos ou não. No modelo do ataque *whitewashing*, quando um poluidor ativo for classificado como malicioso por *w*% dos pares do sistema, ele trocará a sua identidade. É assumido que todo poluidor ativo sabe quando um par o classifica como malicioso, modelando assim o pior cenário possível para um ataque de *whitewashing*.

Este capítulo apresentou as definições e modelagem utilizada para avaliar o processo de disseminação de poluição. Foram discutidos os mecanismos de introdução de poluição por *inserção de versões falsas* e a *corrupção de identificador*, esse último descrito em detalhes nesta dissertação. Além disso, foi apresentado o modelo de simulação utilizado para modelar o problema da disseminação de poluição e as estratégias de combate propostas. No próximo capítulo, será avaliado o impacto da disseminação de poluição nos sistemas P2P, além de apresentar uma validação do simulador apresentado nesse capítulo.

Parâmetro	Descrição
Objetos	
T	Número de títulos únicos.
V	Número de versões únicas.
F	Número máximo de fontes para realizar um *donwload*.
α	Parâmetro da distribuição *Zipf* utilizada para escolha inicial dos títulos e versões.
Pares	
N_b	# pares não maliciosos.
N_p	# pares poluidores.
O_b	# objetos/par compartilhados inicialmente pelos pares não maliciosos.
O_p	# objetos/par compartilhados inicialmente pelos poluidores ativos.
Comportamento do Par	
$p_{\text{opinião}}$	Probabilidade do par não malicioso retornar a opinião para o sistema de combate à poluição logo após realizar *download*.
p_{erro}	Probabilidade do par não malicioso retornar uma opinião que não reflete o estado do *download* realizado.
δ_i	Reação do usuário ao incentivo.
$\lambda_{\text{download}}$	taxa de *download* de um par não malicioso. Modelado com uma distribuição Exponencial.
λ_{entrada}	taxa de entrada de um par não malicioso. Modelado com uma distribuição Exponencial.
$\lambda_{\text{saída}}$	taxa de saída de um par não malicioso. Modelado com uma distribuição Exponencial.
Comportamento Malicioso	
H	Porcentagem dos dados do objeto que podem ser poluídos.
r	Número de instâncias executadas por um poluidor ativo para realizar um ataque *Sybil*.
w	Porcentagem máxima de pares não maliciosos que classificam o par malicioso *i* como tal antes que ele altere sua identidade (*whitewashing*).

Tabela 3.1: Parâmetros do simulador

4 Avaliação da Disseminação de Conteúdo Poluído

Este capítulo apresenta a avaliação da disseminação de poluição nos sistemas P2P. Será mostrado como a porcentagem de *downloads* de objetos não poluídos é afetada quando os poluidores ativos utilizam os mecanismos de introdução de poluição *inserção de versões falsas* e *corrupção do identificador*. A seção 4.1 apresenta um modelo analítico construído para analisar a disseminação de poluição pelo mecanismo de *inserção de versões falsas*. Esse modelo é usado para validar o simulador proposto no capítulo 3. A seção 4.2 avalia a velocidade de disseminação de objetos poluídos, com resultados obtidos analiticamente e por simulação. Além disso, esta seção apresenta os resultados do impacto da atribuição de incentivos para os pares não maliciosos apagarem seus objetos poluídos.

4.1 Modelo Analítico da Disseminação de Poluição

Esta seção apresenta o modelo analítico criado para representar o processo de disseminação de conteúdo poluído a partir do método de introdução de poluição por *inserção de versões falsas*, apresentado no capítulo 3. O modelo tem como principal objetivo analisar a evolução da fração de *downloads* de objetos não poluídos ao longo do tempo. O modelo analítico é simples e modela os pares não maliciosos de um sistema P2P realizando *download* de versões, que podem ser poluídas ou não, dos títulos presentes no sistema.

O modelo, baseado em um sistema de equações diferenciais não lineares, é uma extensão de um modelo criado por Thommes *et alii* [78]. O modelo criado por Thommes *et alii* tem o mesmo objetivo da modelagem proposta nesta seção. Porém, a proposta original segue as premissas simplificadoras de que (1) existe apenas **um** título no sistema e (2) apenas **um** *download* é realizado por cada par. Dessa forma, o modelo apresentado nesta dissertação estendeu a proposta original para que o sistema possua múltiplos títulos e permita a realização de mais de um *download* por par.

Parâmetro	**Definição**
M	Número de pares.
$N(0)_b$	Número de objetos não poluídos compartilhados no instante $t = 0$.
$N(0)_p$	Número de objetos poluídos compartilhados no instante $t = 0$.
T	Número de títulos.
λ	taxa de *download* por par
δ	probabilidade de um par apagar o conteúdo poluído obtido imediatamente após o *download*
p_i	Função de Probabilidade usada para descrever a probabilidade da seleção de um título i.
Função	**Definição**
$P(t)_b$	Probabilidade de obter uma cópia não poluída no instante t
$N(t)_b$	Número de objetos não poluídos compartilhados no instante t.
$N(t)_p$	Número de objetos poluídos compartilhados no instante t.

Tabela 4.1: Parâmetros e Funções do modelo

A Tabela 4.1 apresenta os parâmetros e funções do modelo[1]. Primeiramente, será apresentada uma versão preliminar do modelo no qual o sistema P2P modelado possui somente um título, embora haja múltiplas versões desse título e um par possa realizar múltiplos *downloads*.

Inicialmente, no instante $t = 0$, existem $N(0)_b$ cópias não poluídas e $N(0)_p$ cópias poluídas do título disponíveis para *download*. Existem M pares no sistema, cada um realizando *download* das versões a uma taxa λ. Além disso, os pares do sistema possuem uma probabilidade δ de apagar o objeto poluído obtido imediatamente após o *download*. Diferentemente do simulador, existe somente pares não maliciosos no modelo. No modelo não importa quem compartilha as cópias iniciais do sistema, desde que elas estejam disponíveis para os M pares considerados.

Seja $P(t)_b$ a probabilidade de um par realizar *download* de um objeto não poluído no tempo t, representado pela equação 4.1. Além disso, as funções $N(t)_p$ e $N(t)_b$ representam o número de cópias poluídas e não poluídas no tempo t, respectivamente. As equações que descrevem a disseminação dos objetos ao longo do tempo são:

[1]Os parâmetros do modelo não são os mesmos utilizados pelo simulador. Portanto, para que seja possível contrastar os resultados obtidos analiticamente e através de simulação, é necessário realizar um mapeamento dos parâmetros. Esse mapeamento está presente no Apêndice 1.

$$P(t)_b = \frac{N(t)_b}{N(t)_b + N(t)_p} \tag{4.1}$$

$$\frac{dN(t)_b}{dt} = \lambda\, P(t)_b\, M \tag{4.2}$$

$$\frac{dN(t)_p}{dt} = \lambda\, (1 - P(t)_b)\, M\, \delta \tag{4.3}$$

A equação 4.2 representa a evolução do número total de cópias de versões não poluídas ao longo do tempo. A equação 4.3 representa a evolução do número de cópias de versões poluídas. A probabilidade de um par realizar *download* de um objeto não poluído, $P(t)_b$, é representada pela fração de objetos não poluídos no sistema no tempo t. Assim como no modelo original [78], não foi possível encontrar uma fórmula fechada para as equações diferenciais. Porém, elas podem ser resolvidas através de métodos iterativos para solução de equações diferenciais ordinárias [15].

Para estender esse modelo para incluir T ($T > 0$) títulos, basta resolver as equações 4.1, 4.2 e 4.3 para cada um dos T títulos. Seja p_i a probabilidade de um par escolher uma versão do título i para realizar *download*. Além disso, para cada título i, sejam $N(0)^i_b$ a quantidade de cópias não poluídas inicialmente no sistema, $N(0)^i_p$ a quantidade de cópias poluídas iniciais e λ^i a taxa de *download* do par para versões do título i. As equações que atribuem os valores iniciais para $N(0)^i_b$, $N(0)^i_p$ e λ^i são:

$$N(0)^i_b = N(0)_b\, p_i \tag{4.4}$$

$$N(0)^i_p = N(0)_p\, p_i \tag{4.5}$$

$$\lambda^i = \lambda\, p_i \tag{4.6}$$

As equações que descrevem o modelo generalizado podem então ser reescritas como:

$$P(t)_b^i = \frac{N(t)_b^i}{N(t)_b^i + N(t)_p^i} \tag{4.7}$$

$$\frac{dN(t)_b^i}{dt} = \lambda^i \, P(t)_b^i \, M \tag{4.8}$$

$$\frac{dN(t)_p^i}{dt} = \lambda^i \, (1 - P(t)_b^i) \, M \, \delta \tag{4.9}$$

$$P(t)_b = \frac{\sum_{i=1}^{T} N(t)_b^i}{\sum_{i=1}^{T} N(t)_b^i + N(t)_p^i} \tag{4.10}$$

As equações 4.7, 4.8 e 4.9 capturam os mesmos aspectos das equações 4.1, 4.2 e 4.3, respectivamente. A equação 4.10 define $P(t)_b$, a probabilidade de se obter uma cópia de uma versão não poluída que é obtida através da fração de cópias não poluídas de todos os títulos do sistema.

Esse modelo assume que todos os pares têm o mesmo comportamento e sempre compartilham seus objetos, premissas que o simulador também possui. Para o funcionamento correto do modelo, também deve-se ter que $t < \frac{1}{\lambda} M(N(0)_b + N(0)_p)$, ou seja, t não deve ser maior que o tempo necessário para que todos os objetos do sistema tenham sido obtidos por todos os pares.

O modelo criado representa apenas o processo de disseminação pelo mecanismo de *inserção de versões falsas*. Houve um grande esforço para criar um modelo que representasse a introdução através da *corrupção do identificador*. Porém, como discutido no capítulo 3, as versões dos títulos podem conter cópias poluídas e não poluídas quando é utilizado a *corrupção do identificador*. Assim, variáveis como o número de fontes de *download* (F) e a porcentagem de dados corrompidos de uma cópia (H), influenciam a probabilidade $P(t)_b$ de um par obter uma cópia não poluída no tempo t. A influência dessas variáveis impõem grandes dificuldades para a criação do modelo.

A próxima seção apresentará os resultados da avaliação da disseminação de conteúdo poluído, quando os mecanismos de *inserção de versões falsas* e *corrupção do identificador* são utilizados. Dentre esses resultados, está a validação do simulador através do modelo analítico apresentado nesta seção.

4.2 Disseminação de Poluição

Esta seção avalia o processo de disseminação de poluição para os dois mecanismos de introdução de conteúdo poluído. Três aspectos principais serão discutidos. Primeiramente, essa avaliação irá mostrar o impacto da disseminação de conteúdo poluído no sistema, quando os mecanismos de introdução por *inserção de versões falsas* e a *corrupção do identificador* são utilizados. Os resultados para ambos os mecanismos de introdução serão comparados e será apresentada qual deles consegue disseminar objetos poluídos mais rapidamente.

Além disso, outro aspecto importante abordado nesta avaliação é a validação do simulador através do modelo analítico proposto neste capítulo. Os resultados, obtidos com o modelo e por simulação, da avaliação da disseminação de poluição pelo mecanismo de introdução por *inserção de versões falsas*, serão contrastados.

Por fim, será avaliado o impacto da ação voluntária dos usuários apagarem seu conteúdo poluído. Para fazer esta avaliação assume-se que, imediatamente após o *download*, um par recebe um incentivo para verificar o conteúdo obtido e apagá-lo caso esteja poluído. A reação a esse incentivo recebido é modelado através do parâmetro δ, introduzido e discutido no capítulo 3. O modelo de δ utilizado nesta avaliação será o *fixo* (equação 3.1), que atribui uma probabilidade fixa para o par não malicioso apagar o conteúdo poluído obtido, imediatamente após o *download*. É importante ressaltar que nas avaliações deste capítulo, apagar o conteúdo obtido a única forma de combater a disseminação de conteúdo poluído.

Foram realizados uma série de experimentos considerando a configuração apresentada na Tabela 4.2. Todos os resultados, obtidos por simulação, são médias de 5 execuções e, com um nível de confiança de 95% a largura do intervalo de confiança foi no máximo 2% da média. Os resultados analíticos foram obtidos através do método Runge-Kutta de quarta Ordem [15], um método iterativo para solução de equações diferenciais ordinárias. O método foi executado com um passo de $0,005$ dias, apresentando um erro da ordem de $0,625 \times 10^{-9}$.

A Figura 4.1 inicia a avaliação, mostrando a fração de *downloads* não poluídos ao longo do tempo, quando o mecanismo de *inserção de versões falsas* é utilizado. Nesta avaliação, considera-se diferentes valores de probabilidades do par não malicioso apagar um objeto poluído obtido (δ), imediatamente após o *download*. Os resultados foram obtidos tanto através de simulação quanto pelo modelo analítico.

Primeiramente, pode-se observar que os resultados obtidos com o modelo analítico e por simulação são muito próximos. Na Figura 4.1, os resultados obtidos com o modelo analítico estão representados por uma linha e os resultados de simulação por pontos. Para qualquer

PARÂMETRO	VALOR	
# títulos únicos (T)	100	
# versões únicas (V)	400	
α (parâmetro da Zipf)	$0,8$	
# múltiplas fontes (F)	10	
% poluída dos dados (H)	100%	
PARÂMETRO	PARES NÃO MALICIOSOS	POLUIDORES ATIVOS
# pares	$N_b = 1000$	$N_p = 250$
# objetos compartilhados por par (início)	$O_b = 50$	$O_p = 100$
taxa de *download* de cada par ($\lambda_{\text{download}}$)	4 objetos/dia	-
taxa de entrada de cada par (λ_{entrada})	2 vezes/dia	-
taca de saída de cada par ($\lambda_{\text{saída}}$)	2 vezes/dia	-

Tabela 4.2: Valores dos Parâmetros da Simulação

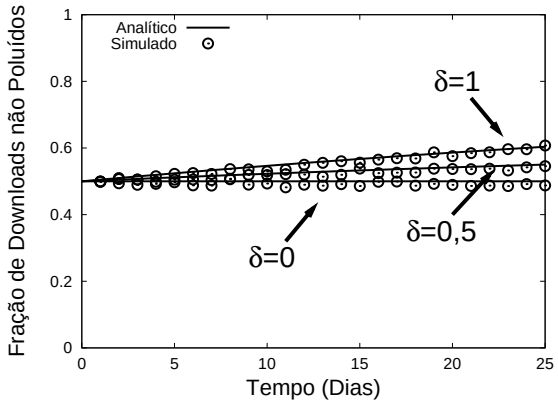


Figura 4.1: Disseminação de poluição pela Inserção de Versões Falsas ao longo do tempo para três valores de probabilidade de apagar o conteúdo poluído obtido.

instante, a diferença entre as curvas obtidas foram menores do que 3%. Isso mostra que, para a *inserção de versões falsas* o simulador foi validado, pois obtém resultados muito próximos aos do modelo.

Além disso, pode-se observar que a fração de *downloads* não poluídos no início da simulação é $0,5$. Esse valor é derivado do fato que a porcentagem de cópias não poluídas **ativas** é 50%. O total de cópias poluídas é conseguido multiplicando o número de cópias poluídas por par (O_p) com o número total de pares (N_p). Já o número de cópias não poluídas é conseguido multiplicando o número de cópias não poluídas (O_b) por par com o número total de **pares não maliciosos ativos** ($N_b\ \lambda_{\text{saída}}/(\lambda_{\text{saída}} + \lambda_{\text{entrada}})$). Utilizando os valores apresentados da tabela 4.2 chega-se na porcentagem de 50% de cópias não poluídas ativas no início da simulação.

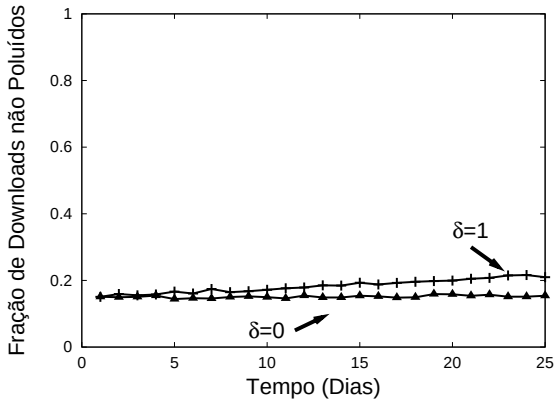


Figura 4.2: Disseminação de poluição pela Corrupção do Identificador ao longo do tempo para dois valores de probabilidade de apagar o conteúdo poluído obtido. ($H = 100\%$ e $F = 10$)

Pode-se observar também que quando nenhum par apaga seus objetos poluídos ($\delta = 0$), a fração de *downloads* não poluídos ao longo do tempo se mantém constante. Isso acontece pois novos objetos não são inseridos ou retirados do sistema, e assim a porcentagem de cópias não poluídas se mantém contante ao longo do tempo.

Além desses resultados, pode-se observar que se os pares não maliciosos apagam seus objetos poluídos ($\delta \neq 0$), a fração de *downloads* não poluídos aumenta com o passar do tempo. Porém, o crescimento observado é muito pequeno. Se a probabilidade do par não malicioso apagar o objeto poluído obtido é 0,5, então a fração de *downloads* não poluídos aumenta de $0,5$ para $0,55$, no dia 25. Mesmo se todos os pares não maliciosos apagarem seus objetos poluídos após o *download* ($\delta = 1$) a fração de objetos não poluídos cresce só de $0,5$ para $0,6$, no dia 25.

A partir desse resultado, observa-se que a disseminação de poluição não pode ser efetivamente contida apenas através da ação voluntária dos pares não maliciosos apagarem seus objetos poluídos. Com essa observação, conclui-se que os poluidores ativos também causam um grande impacto na disseminação de conteúdo poluído. Portanto, incentivar os poluidores passivos a apagar suas cópias poluídas tem pouco efeito se os poluidores ativos não forem isolados.

A figura 4.2 apresenta a avaliação para o mecanismo de *corrupção do identificador*, considerando diferentes probabilidades do par não malicioso apagar o objeto poluído obtido (δ). Nesse experimento, é considerado que o número de fontes de *download* (F) é no máximo 10 e porcentagem dos dados poluídos de um objeto introduzido por um poluidor ativo é 100% (H).

Primeiramente, observa-se que quando os pares não maliciosos não apagam seus objetos poluídos ($\delta = 0$), a fração de *downloads* não poluídos é $0,15$ no início da simulação e se mantém constante durante toda a simulação. Isso acontece pois não existem objetos sendo inseridos ou

Número de Fontes	**Fração de *Downloads* não Poluídos no 25^o dia**
1	0,5
2	0,33
5	0,19
10	0,15
100	0,14

Tabela 4.3: Disseminação de poluição pela corrupção do identificador, variando o parâmetro F ($H = 100\%$ e $\delta = 0$)

retirados do sistema, assim como acontece com a *inserção de versões falsas*.

Além disso, como na *inserção de versões falsas*, a ação voluntária dos usuarios apagarem suas cópias poluídas tem pouco impacto. Mesmo se todos os pares não maliciosos apagarem o conteúdo poluído obtido, a fração de *downloads* poluídos aumenta de $0,15$ para $0,2$, no vigésimo quinto dia. Esse resultado reforça a conclusão de que os poluidores ativos precisam ser punidos e isolados para uma redução significativa da poluição.

Pode-se observar também que a fração de *downloads* não poluídos é muito menor que na *inserção de versões falsas*. Isso mostra que o mecanismo de *corrupção de identificador* consegue disseminar conteúdo poluído mais rapidamente que o mecanismo por *inserção de versões falsas*. Essa grande velocidade de disseminação ocorre pois as versões de títulos na *corrupção do identificador* podem possuir cópias poluídas e não poluídas. Dessa forma, qualquer *pedaço de dados* poluído recebido no processo de *download* torna a cópia obtida poluída. Como o número de fontes de *download* (F) é 10, é alta a probabilidade de um par não malicioso interagir com um par poluidor e receber dados poluídos.

Essa afirmação pode ser comprovada a partir da Tabela 4.3. A tabela mostra como o número de fontes afeta a fração de *downloads* não poluídos na *corrupção do identificador*. Esses experimentos consideraram que a porcentagem dos dados poluídos de um objeto introduzido por um poluidor ativo é 100% (H) e os pares nunca apagam o conteúdo poluído obtido ($\delta = 0$). Como mostrado nos resultados anteriores, sempre que os pares não apagam os seus objetos poluídos, a fração de *downloads* não poluídos se mantém constante. Portanto, os dados apresentados na tabela se referem ao vigésimo quinto dia de simulação.

Como esperado, quanto maior o número de fontes, maior é a probabilidade de um par não malicioso receber um *pedaço de dado* de uma fonte poluidora. Pode-se observar também que a diferença da fração de *downloads* não poluídos para as curvas de 10 e 5 fontes é muito pequena e a diferença entre 10 ou de 100 fontes é quase desprezível.

Porcentagem poluída dos dados (H)	**Fração de *Downloads* não Poluídos no 25^o dia**
10%	0,75
20%	0,58
30%	0,45
40%	0,34
50%	0,28
60%	0,22
70%	0,18
80%	0,16
90%	0,15
100%	0,15

Tabela 4.4: Disseminação de poluição pela corrupção do identificador, variando o parâmetro H ($F = 10$ e $\delta = 0$)

Finalmente, a Tabela 4.4 mostra variação da porcentagem de dados do objeto que pode ser poluído, para o mecanismo de *corrupção do identificador*. Nessa avaliação, foi assumido que nenhum par não malicioso apaga seus objetos poluídos ($\delta = 0$) e que o número de fontes de *download* simultâneos é 10 ($F = 10$). Assim como na avaliação anterior, os dados apresentados se referem ao vigésimo quinto dia de simulação.

Os dados mostram que, se a porcentagem de dados poluídos de um objeto é maior que 80% ($H > 80\%$), o impacto no sistema é semelhante a quando $H = 100\%$. Além disso, valores de $H = 70\%$ e 60% ainda têm impacto no sistema, fazendo com que a *corrupção do identificador* dissemine poluição muito rapidamente. É interessantes notar que, quando a porcentagem de dados poluídos é menor que 30%, o mecanismo de *inserção de versões falsas* dissemina poluição mais rapidamente que a *corrupção do identificador*. Esse fato ocorre pois se a porcentagem de dados poluídos é baixa, a probabilidade de um poluidor enviar um pedaço de dado não poluído aumenta. Lembrando que como foi discutido no capítulo 3, no sistema Kazaa é possível poluir 88% e 99,5% dos dados de típicos objetos de áudio (5 MB) e vídeo (700 MB), respectivamente.

Este capítulo apresentou um modelo analítico construído para representar a disseminação de poluição pelo mecanismo de introdução de versões falsas. Esse modelo foi utilizado para validar o simulador apresentado no capítulo 3. Além disso, foram apresentados os resultados da avaliação da disseminação de poluição pelos mecanismos de *inserção de versões falsas* e *corrupção do identificador*. Foi concluído que o mecanismo de *corrupção do identificador* dissemina poluição mais rapidamente que a *inserção de versões falsas*. Além disso, foi verificado que a disseminação de poluição não pode ser efetivamente contida apenas através da ação

voluntária dos usuários apagarem seus objetos poluídos, e portanto se faz necessário outras estratégias mais eficazes. O próximo capítulo apresentará estratégias de combate para redução da disseminação de conteúdo poluído.

5 *Estratégias de Combate a Poluição*

O capítulo 4 verificou que a disseminação de poluição não pode ser efetivamente contida apenas através da ação voluntária dos usuários apagarem seus objetos poluídos. Isso ocorre pois os poluidores ativos têm um grande impacto na disseminação de poluição e portanto, esse problema deve ser atacado.

Este capítulo apresenta as estratégias de combate à poluição avaliadas nesta dissertação. A seção 5.1 apresenta uma estratégia para reduzir a poluição baseada na ação de moderadores. A seção 5.2 apresenta o sistema de reputação *Scrubber*, proposto nesta dissertação, no qual os pares atribuem reputação uns para os outros como fontes de conteúdo poluído. Finalmente, a Sessão 5.3 apresentada o sistema *Híbrido*, também proposto nesta dissertação, e que combina funcionalidades dos sistemas de reputação de pares (ex.: *Scrubber*) e de classificação de objetos (ex.: *Credence*).

5.1 Censura por Moderador

Supondo que seja possível controlar as pesquisas por determinados objetos em um sistema P2P, uma estratégia simples e eficaz pode ser aplicada para evitar a disseminação de conteúdo poluído. Ao fim de cada *download*, o par analisa o objeto recebido, verificando se ele está poluído ou não. Através de uma interface no cliente P2P, o usuário relata que o objeto *o* recebido está poluído. O relato é enviado para um moderador. O moderador, por sua vez, realiza *download* do objeto reportado, verifica sua autenticidade e, caso seja poluído, censura o objeto. A censura torna o objeto indisponível para todos os pares do sistema.

A implantação da censura por moderadores é possível na maioria dos sistemas P2P atuais. Por exemplo, a busca do objeto no sistema BitTorrent não faz parte do protocolo. Existem sítios na *Internet* que disponibilizam e indexam os arquivos *.torrent*, responsáveis por armazenar os dados necessários para a realização do *download*, como visto no capítulo 2. Dessa forma, sempre que um objeto for considerado poluído pelos usuários, o moderador do sítio tem o poder

de retirar o arquivo *.torrent* e assim tornar objeto não disponível.

A implantação de um sistema de moderadores é possível mesmo em sistemas P2P nos quais a busca faz parte do sistema e é realizada de forma distribuída. Nesse caso, o moderador não teria o poder de censurar o objeto. Porém, ele poderia emitir avisos aos usuários sobre a autenticidade de um determinado objeto.

É interessante ressaltar que o moderador é um ponto de contenção, e em sistemas P2P muito populares, ele pode acabar apresentando problemas de escalabilidade , comprometendo a eficácia da estratégia. Além disso, o moderador precisa ser uma entidade confiável pelos pares, pois um moderador malicioso poderia facilmente destruir o sistema.

5.2 O Sistema de Reputação Scrubber

O *Scrubber* é um sistema de reputação projetado para identificar e isolar pares maliciosos que ativamente disseminam conteúdo poluído nos sistemas P2P de compartilhamento de arquivos. Além disso, *Scrubber* permite a reabilitação dos *poluidores passivos* atribuindo incentivo para que eles apaguem o conteúdo poluído que receberam e mantiveram em suas pastas de compartilhamento. O *Scrubber* é descentralizado e distribuído, facilitando assim sua implantação em sistemas P2P reais.

O projeto do *Scrubber* é uma adaptação e extensão de em um sistema de reputação proposto por Rocha *et alii* para combater comportamento egoísta em redes sobrepostas de roteamento [70, 71]. Os principais conceitos da proposta original foram modificados e estendidos para capturar as peculiaridades da disseminação de poluição em sistemas P2P reais de compartilhamento de arquivos, como por exemplo a baixa freqüência de interações entre dois pares do sistema.

No *Scrubber*, os pares atribuem reputação uns para os outros como fontes de conteúdo poluído. Reputações são construídas a partir de dois componentes principais: a *Experiência Individual* e o *Testemunho*. A *Experiência Individual* de um par i a respeito do par j é a confiança que i tem em j, baseada nos objetos obtidos dele anteriormente. Depois que um objeto é obtido, o par i atualiza sua *Experiência Individual* com todas as fontes que enviaram o objeto. A *Experiência Individual* de i com o par j, $I_{i(j)}$, é atualizada da seguinte forma:

$$I_{i(j)} = \begin{cases} max(0, I_{i(j)} - \alpha_d n^2) & \text{se o objeto é poluído} \\ \\ min(1, I_{i(j)} + \alpha_i) & \text{caso contrário} \end{cases} \tag{5.1}$$

onde n é o número de vezes consecutivas que i recebeu objetos poluídos de j. Além disso, α_d e α_i representam a penalidade e recompensa dados para o par j para cada objeto poluído e não poluído enviado, respectivamente. A *Experiência Individual* de qualquer par i em relação a um par j desconhecido é inicialmente fixado em $R_{\text{início}}$. Note que a *Experiência Individual* diminui mais rápido do que aumenta.

Duas ações são tomadas visando identificar rapidamente e penalizar severamente os poluidores que ativamente enviam conteúdo poluído. Primeiro, o *Scrubber* multiplica o fator α_d pelo quadrado do número de objetos poluídos recebidos consecutivamente de um determinado par j. Logo, as penalidades para o envio de objetos poluídos aumentam quadraticamente com ações maliciosas repetidas. Este mecanismo visa evitar ataques de *traidor*, nos quais os pares maliciosos adquirem uma alta reputação, e então começam a enviar conteúdo poluído. Além disso, dado que tipicamente ocorrem poucas interações entre dois pares em sistemas P2P de compartilhamento de arquivos, foram utilizados diferentes fatores de penalidade e recompensa, e proposto $\alpha_d > \alpha_i$.

É interessante observar que se os poluidores ativos utilizarem a *corrupção do identificador*, o par i pode obter um objeto poluído, porém recebendo pedaços não poluídos do par j. Como é difícil para os usuários verificarem cada pedaço recebido, assume-se nesta dissertação que se i recebeu um objeto poluído, então ele pune todos os pares que enviaram conteúdo, mesmo se j enviou um segmento de dados não poluído.

O *Testemunho* captura a opinião da comunidade (ou seja, da rede) sobre um par j. A opinião da comunidade sobre outros pares do sistema é conseguida através da *pesquisa por testemunho*. Periodicamente, cada par i envia uma *pesquisa por testemunho* para um número de pares selecionados aleatoriamente. Esta dissertação considera a seleção de *um* único par, k. O par k retorna para i todos os seus valores de *Experiência Individual*. Esta informação é utilizada por i, antes do *download* do objeto, para atualizar o *Testemunho* da comunidade sobre j, $T_i(j)$, da seguinte maneira:

$$T_{i(j)} = \frac{\sum_{k \in N_{i(j)}} I_{k(j)} R_{i(k)}}{\sum_{k \in N_{i(j)}} R_{i(k)}} \tag{5.2}$$

onde $N_{i(j)}$ é a lista dos pares que enviaram suas experiências sobre j para i no passado e $R_{i(j)}$, definido abaixo, é a reputação atual do par j atribuída por i. Note que as experiências individuais obtidas na comunidade são ponderadas pelas reputações locais de suas fontes para evitar difamação. Se nenhum testemunho sobre j foi coletado o valor inicial do testemuho $T_{i(j)}$ é $R_{\text{início}}$.

Antes e depois que um objeto é obtido, o par i computa a reputação local de todos os pares conhecidos j, $R_{i(j)}$, da seguinte maneira:

$$R_{i(j)} = \beta T_{i(j)} + (1-\beta) I_{i(j)} \tag{5.3}$$

onde β $(0 \leq \beta \leq 1)$ controla os pesos dados para a *Experiência Individual* e o *Testemunho*. O parâmetro $R_{min(i)}$ $(0 \leq R_{min(i)} \leq R_{\text{início}})$ é a reputação mínima que um par deve ter para ser considerado confiável por i. Um par i não envia nem obtém objetos de pares que ele não considera confiáveis. Note que, recusando enviar objetos a esses pares, *Scrubber* dá incentivo para que os *poluidores passivos* apaguem seus objetos poluídos, já que essa é a única forma dos poluidores passivos aumentarem a sua reputação no sistema e ter suas futuras requisições servidas. A opinião da comunidade ($\beta > 0$) ajuda aos pares identificarem poluidores em potencial, assim como promover a *reabilitação* de pares que apagaram o conteúdo poluído de suas pastas de compartilhamento. A *reabilitação* dos poluidores passivos também pode ser facilitada se os pares utilizarem diferentes valores de $R_{min(i)}$, visto que um par considerado não confiável por i pode readquirir sua confiança aumentando sua reputação com outros pares k que possuam valores $R_{min(k)}$ mais baixos, tendo assim a sua reputação aumentada em i através dos testemunhos dos outros pares.

As principais diferenças desta proposta para a proposta original de Bruno *et alii* são três. Primeiramente, foi preciso estender o sistema original para conseguir aplicar punições mais rápidas, já que o número de interações em sistemas P2P de compartilhamento de arquivos é pequeno. Portanto, foram propostos diferentes fatores de punição (α_d) e recompensa (α_i), além de assumir que $\alpha_d > \alpha_i$. Uma outra modificação foi a criação da *pesquisa por testemunho*. O tra-

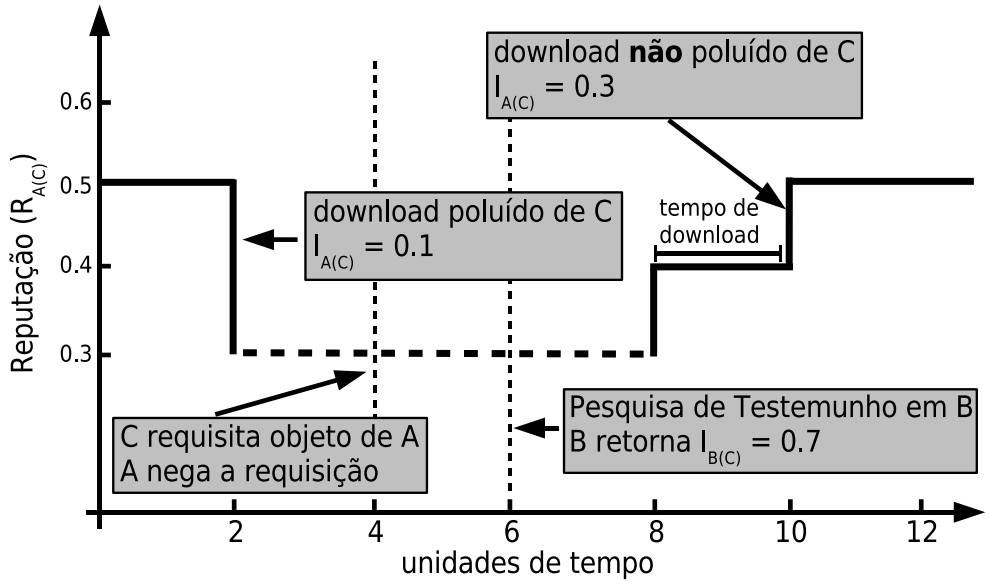


Figura 5.1: Funcionamento do Scrubber em um Sistema com Três Pares (A, B e C).

balho de Rocha *et alii* assume que os pares têm, em todo instante, informação dos Testemunhos sobre todos os outros pares. Logo, visando facilitar a implantação do *Scrubber* em sistemas P2P de compartilhamento de arquivos, foi proposta a *pesquisa por testemunho* como forma de coletar a opinião da comunidade em sistemas com uma grande quantidade de usuários.

Finalmente, a proposta original foi modificada para tornar o *Scrubber* robusto contra ataques de conluio. Uma avaliação do *Scrubber*, apresentada no capítulo 6, mostrou que o sistema é pouco eficaz sob os ataques de conluio de difamação se todos os pares conhecidos por *i* puderem ser escolhidos para retornar suas *Experiências Individual* em uma *pesquisa por testemunho*. Para tornar o *Scrubber* resistente contra ataques de conluio a seleção dos pares é realizada considerando somente os pares conhecidos e que possuem *Experiência Individual* e *Testemunho* maiores que $R_{min(i)}$. Utilizando $T_{i(j)}$ e $I_{i(j)}$ separadamente (ao invés da métrica $R_{i(j)}$, que é agregada) e considerando somente pares conhecidos, as chances do *Scrubber* contactar um par malicioso, que irá retornar reputações difamatórias, é menor. Hélio *et alii* [2] propuseram, paralelamente a esta dissertação, uma modificação similar a essa apresentada para solucionar o problema de ataques conluio no sistema original de Bruno *et alii*.

Para o funcionamento do *Scrubber*, é necessário que o par *i* armazene localmente todos os valores computados da *Experiência Individual* e da Reputação, assim como os valores coletados dos *Testemunhos*.

A operação do *Scrubber* será ilustrada em um sistema hipotético com três pares, *A*, *B* e *C*. A Figura 5.1 mostra como a reputação de *C* atribuída pelo par *A*, $R_{A(C)}$, evolui com o

decorrer do tempo. Os parâmetros do sistema são $R_{min(A)} = 0,35$, $R_{\text{início}} = 0,5$, $\alpha_i = 0,2$, $\alpha_d = 0,4$, e $\beta = 0,5$. Inicialmente, assume-se que A não conhece C, logo $I_{A(C)} = T_{A(C)} = R_{\text{início}}$, e $R_{A(B)} = 0,7$.

No instante 2, A recebe um objeto poluído de C, diminui $I_{A(C)}$ para 0.1 e $R_{A(C)}$ para $0,3$. Como $R_{A(C)} < R_{min(A)}$, A considera C não confiável e recusa enviar um objeto, duas unidades de tempo depois. Motivado por esta negação de serviço, C apaga seus objetos poluídos. Durante as próximas duas unidades de tempo, B recebe um objeto não poluído de C e aumenta $I_{B(C)}$ para $0,7$ (não mostrado na figura). No instante 6, A envia uma *pesquisa por testemunho* para B e obtém a sua *experiência individual* sobre C, $I_{B(C)}$. No instante 8, antes de obter um novo objeto, A utiliza $I_{B(C)}$ para atualizar $T_{A(C)}$ para $0,7$ e $R_{A(C)}$ para $0,4$. A considera C confiável novamente e obtém o objeto dele. Depois que o processo termina (instante 10), A aumenta $I_{A(C)}$ para $0,3$ e $R_{A(C)}$ para $0,5$.

5.3 O Sistema de Reputação *Híbrido*

O sistema Híbrido estende o *Scrubber* para permitir não somente a atribuição de reputação aos pares, mas também a classificação de objetos. O sistema de reputação *Híbrido* foi concebido para descobrir os objetos poluídos e isolar os nós maliciosos que ativamente enviam poluição, assim como isolar os objetos poluídos presentes no sistema. Esse sistema é distribuído e descentralizado, como o *Scrubber*.

O sistema *Híbrido* possui dois componentes principais: a *Classificação de Objetos* e a *Reputação de Pares*. As próximas seções descrevem estes componentes.

5.3.1 Classificação de Objetos

Assim como outros sistema de classificação de objetos, tais como o *Credence* e o *XREP*, o sistema de Reputação *Híbrido* permite que os usuários verifiquem autenticidade dos objetos no sistema. Para fazer tal verificação, antes de obter um objeto, o par i envia para o sistema uma *pesquisa por votos*, buscando por votos de outros pares sobre o objeto desejado. O voto do par j sobre o objeto o, denotado por $V_j(o)$, pode ter os valores -1, caso o objeto seja considerado poluído por i, ou $+1$, caso o objeto não seja poluído.

Os votos coletados por i, para o objeto o, são então agregados na classificação do objeto, $R_{i(o)}$, como segue:

$$R_{i(o)} = \frac{\sum_{j \in N_{i(o)}} V_{j(o)} R_{i(j)}}{\sum_{j \in N_{i(o)}} R_{i(j)}} \tag{5.4}$$

onde $R_{i(j)}$ é a reputação atual do par j atribuída pelo par i, e $N_{i(o)}$ é o conjunto de pares que reponderam à *pesquisa por votos* promovida por i. Para evitar a difamação do objeto, os votos coletados são ponderados pelas reputações locais, além de serem descartados votos de pares desconhecidos e que possuem a reputação menor do que $R_{min(i)}$. A classificação do objeto $R_{i(o)}$ varia entre $[-1..1]$ e é usada para decidir se o objeto deve ser obtido. Se o objeto for considerado não poluído, então as fontes com valores de reputação, $R_{i(j)}$, maiores que o limiar de confiança, $R_{min(i)}$, são selecionadas.

5.3.2 Reputação de Pares

A coordenação da reputação de pares no sistema *Híbrido* é uma adaptação e extensão do *Scrubber*. O *Híbrido* possui os mesmo componentes principais do *Scrubber*, a saber, a *Experiência Individual* e os *Testemunhos*. Os *Testemunhos* de um par i sobre o par j, $T_{i(j)}$, bem como a *Reputação* de j em i, $R_{i(j)}$, são definidos como no *Scrubber*, segundo as equações 5.2 e 5.2, respectivamente. As *pesquisas por testemunho* também são realizadas da mesma forma que no *Scrubber*.

A *Experiência Individual* do par i a respeito do par j, $I_{i(j)}$, por outro lado, precisa ser estendida para incluir, além da experiência baseada nos objetos obtidos (equação 5.2), os votos retornados pela *pesquisa por votos*. Em outra palavras, a *Experiência Individual* agora precisa ser adaptada para refletir o fato de que um par j pode agir de forma maliciosa, enviando conteúdo poluído e emitindo votos diferentes de i sobre a autenticidade de um objeto. As penalidades atribuídas por i em j, em cada caso, p_i^{poluidor} e $p_i^{\text{mentiroso}}$ respectivamente, são definidas como:

$$p_i^{\text{poluidor}} = max(0,\, I_{i(j)} - \alpha_d^{\text{poluidor}} n^2) \tag{5.5}$$

$$p_i^{\text{mentiroso}} = max(0,\, I_{i(j)} - \alpha_d^{\text{mentiroso}} m^2) \tag{5.6}$$

onde m é o número de votos consecutivos enviados por j que foram inconsistentes com os

dados por *i*, e *n* foi definido na seção 5.1. Depois de realizar *download*, o par *i* vota no objeto *o* recebido, e atualiza $I_{i(j)}$ para todos os pares j que retornaram votos ou que enviaram o objeto *o* para *i*, da seguinte maneira:

$$I_{i(j)} = \begin{cases} p_i^{\text{mentiroso}} & \text{se } V_{j(o)} \neq V_{i(o)} \\ p_i^{\text{poluidor}} & \text{se j enviou um objeto poluído} \\ max(p_i^{\text{mentiroso}}, p_i^{\text{poluidor}}) & \text{se j enviou um objeto poluído } \textbf{e} \\ & V_{j(o)} \neq V_{i(o)} \\ min(1,\, I_{i(j)} + \alpha_i) & \text{caso contrário} \end{cases} \tag{5.7}$$

Depois de atualizar $I_{i(j)}$, o par *i* armazena seu voto localmente, para compartilhar com os outros pares do sistema. Como no *Scrubber*, *Experiência Individual* e o *Testemunho* também são armazenados localmente para serem utilizados na computação das reputações de par e na classificação dos objetos.

Será ilustrada agora a operação do sistema *Híbrido* em um sistema hipotético com quatro pares *A, B, C* e *D*. Considera-se que *A* e *B* são pares não maliciosos e *C* e *D* poluidores ativos. O poluidor ativo *C* compartilha os objetos o_1 e o_2, enquanto o poluidor ativo *D* compartilha o objeto poluído o_3.

Será mostrado o funcionamento do *Híbrido* na visão do par *A*. Os parâmetros do sistema são $R_{min(A)} = 0,35$, $R_{\text{início}} = 0,5$, $\alpha_d^{\text{poluidor}} = \alpha_d^{\text{mentiroso}} = 0,4$, $\alpha_i = 0,2$, e $\beta = 0,5$. Inicialmente, foi considerado que *A* não conhece *B, C* ou *D*.

instante 1: *A* deseja obter o objeto o_1 e realiza uma *pesquisa por votos*. *A* recebe um *voto* de *B* acusando que o objeto está poluído, porém desconsidera o *voto* pois *A* não conhece *B*. *A* então obtém o objeto o_1, poluído, do par *C* e diminui $I_{A(C)}$ para $0,1$ e $R_{A(C)}$ para $0,3$. *A* aumenta $I_{A(B)}$ para 0,7 e $R_{A(B)}$ para $0,6$, pois o voto de *B* foi idêntico ao de *A*.

instante 2: *A* deseja obter um objeto o_2 e realiza uma pesquisa por votos. Desta vez nenhum par retorna a pesquisa. *A* decide obter o objeto mesmo sem indicativo de sua autenticidade. Porém, *C* é a única fonte que possui o objeto e como a reputação de *C* está abaixo do limiar $R_{A(C)}$, então *A* decide não realizar o *download*.

instante 3: *A* deseja obter um objeto o_3 e realiza uma pesquisa por votos. *A* recebe um voto de *B* acusando que o objeto é poluído. *A* calcula $R_{i(o)} = 0$ e decide **não** obter o objeto.

Este capítulo apresentou estratégias para combater a poluição em sistemas P2P de compartilhamento de arquivos. Foi apresentado o sistema de reputação *Scrubber*, que atribuí reputação para os pares como fontes de objetos poluídos, e o sistema *Híbrido*, que agrega as funcionalidades da reputação de pares e classificação de objetos. Além disso, foi apresentada uma estratégia baseada na censura por moderadores. O próximo capítulo irá apresentar uma avaliação dos sistemas apresentados neste capítulo.

6 *Avaliação das Estratégias de Combate à Poluição*

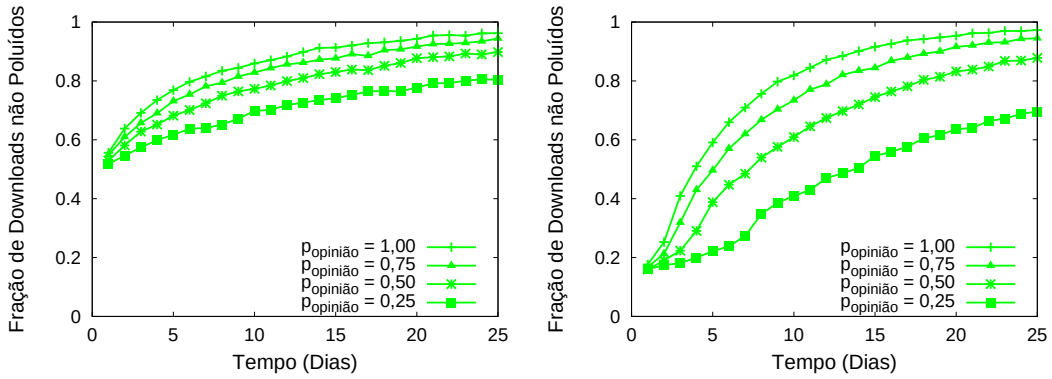
Este capítulo avalia a eficácia das estratégias de combate à disseminação de conteúdo poluído. A seção 6.1 apresenta os resultados para a estratégia baseada na censura por moderadores. A seção 6.2 apresenta os resultados para as estratégias baseadas em reputação, apresentando uma comparação entre os sistemas de reputação implementados, a saber, os sistemas *Scrubber*, *Híbrido* e *Credence*.

Para avaliar os sistemas, duas métricas, a *eficiência* e a *convergência*, foram consideradas. A eficiência está relacionada com a capacidade do sistema reduzir a disseminação de poluição, e é avaliada a partir da fração de *downloads* de cópias não poluídas em um dado instante. A convergência está relacionada com o tempo necessário para que o sistema atinja a sua eficiência máxima.

Os parâmetros do simulador são os mesmos apresentados na tabela 4.2, apresentada no capítulo 4. Para todos os experimentos apresentados neste capítulo, os resultados são médias de 5 execuções possuindo um intervalo de confiança de no máximo 5% da média com um nível de confiança de 95%.

6.1 Censura por Moderadores

Esta seção avalia como a estratégia baseada na censura por moderadores pode reduzir a disseminação de poluição. Assume-se que existe um moderador que tem poder de censurar os objetos poluídos do sistema. Nessa avaliação, o parâmetro opinião do usuário ($p_{\text{opinião}}$) significa a probabilidade do par reportar para o moderador um objeto poluído obtido, imediatamente após a realização do *download*. O único parâmetro específico dessa estratégia é o intervalo de tempo, T_{mod}, que o moderador necessita para verificar a autenticidade de um objeto reportado e excluí-lo do sistema. O intervalo de tempo, T_{mod}, é contado a partir do instante que o par reporta o objeto poluído ao moderador. Além disso, assume-se que os pares não maliciosos não cometem



(a) Inserção de Versões Falsas (b) Corrupção do Identificador

Figura 6.1: Censura por Moderador: Probabilidade do usuário relatar objetos poluídos obtidos ao moderador. ($T_{\text{mod}} = 12$ horas)

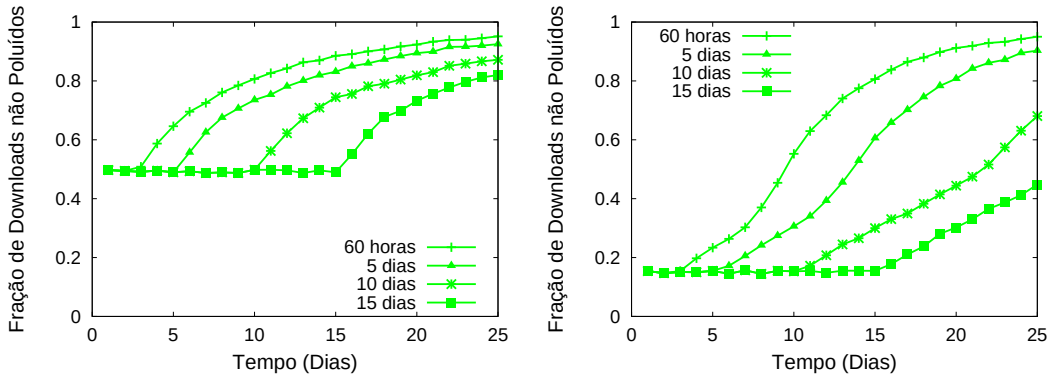
erros em suas opiniões ($p_{erro} = 0$).

A figura 6.1 mostra como a estratégia se comporta variando a probabilidade dos pares reportarem o objeto poluído obtido ($p_{\text{opinião}}$), para os dois mecanismos de introdução de poluição. O parâmetro T_{mod} é fixado em 12 horas. A figura 6.1-a mostra o resultado quando os pares poluidores introduzem objetos poluídos pelo mecanismo de *inserção de versões falsas*. A fração de *downloads* não poluídos é $0,5$ no início da simulação, como discutido no capítulo 4.

Observa-se que a estratégia é muito eficiente para reduzir a disseminação de poluição. Se todos os usuários cooperarem ($p_{\text{opinião}} = 1$) a fração de *downloads* não poluídos chega a $0,96$ ao fim do dia 25. Mesmo se somente 25% dos pares cooperarem com o sistema, uma porcentagem de 80% de *downloads* não poluídos é atingido ao dia 25.

A figura 6.1-b apresenta o resultado quando a poluição é introduzida pelo mecanismo de *corrupção do identificador*. Intuitivamente, como as versões podem possuir cópias poluídas e não poluídas, todas as versões poderiam ser consideradas poluídas pelo moderador, destruindo assim o sistema. Porém, os poluidores inserem cópias poluídas nas versões mais populares do sistema. Logo, as versões que possuem cópias poluídas e não poluídas são censuradas pelo moderador. Dessa forma, as versões pouco populares que não possuem cópias poluídas são selecionadas para a realização do *download*.

Observa-se que se todos os pares cooperarem ($p_{\text{opinião}} = 1$), a fração de *downloads* não poluídos chega a $0,96$ no dia 25, como ocorre para o mecanismo de *inserção de versões falsas*. Como a fração de *downloads* inicialmente é $0,18$, logo o crescimento é mais rápido que na *inserção de versões falsas*. Esse fato ocorre pois como a *corrupção do identificador* dissemina objetos poluídos mais rapidamente, o moderador receberá relatos sobre objetos poluídos mais



(a) Inserção de Versões Falsas (b) Corrupção do Identificador

Figura 6.2: Tempo necessário para o moderador censurar objetos poluídos. ($p_{\text{opinião}} = 1$)

frequentemente. Dessa forma, ele consegue identificar e censurar de forma eficaz as versões que possuem muitas cópias poluídas.

Porém, se os pares não relatarem com frequência os objetos poluídos obtidos, o impacto no sistema é maior. Se somente 25% dos pares cooperarem ($p_{\text{opinião}} = 0,25$), a fração de *downloads* não poluídos aumenta para 70%. Isto ocorre pois o moderador leva mais tempo para censurar as versões mais populares, que possuem muitas cópias poluídas disseminadas. Portanto, até o dia 8, o aumento da fração de *downloads* não poluídos é lento. A partir deste ponto, quando as versões mais populares são censuradas, a fração *downloads* não poluídos cresce rapidamente.

Agora será avaliado, através da fração diária de *downloads* não poluídos, o impacto do intervalo de tempo, T_{mod}, que o moderador necessita para censurar uma versão poluída reportada. A figura 6.2 apresenta esses resultados para a *inserção de versões falsas* e a *corrupção do identificador*. Esses experimentos consideram que 100% dos clientes que recebem conteúdo poluído retornam sua opinião para o moderador ($p_{\text{opinião}} = 1$).

A figura 6.2-a mostra os resultados para a *inserção de versões falsas*. Se o moderador gasta dois dias e meio para censurar um objeto poluído, a fração de *downloads* não poluídos aumenta para 95% no dia 25, uma eficiência muito semelhante aos resultados quando o intervalo de tempo, T_{mod}, é 12 horas. Mesmo se o moderador necessitar de 15 dias para censurar um objeto, uma eficiência de 82% é alcançada no dia 25 de simulação. Isso mostra, que mesmo se o moderador necessitar de um longo tempo para censurar um objeto, a fração de *downloads* não poluídos aumenta rapidamente para a *inserção de versões falsas*.

A figura 6.2-b apresenta os resultados para a *corrupção do identificador*. Pode-se observar que o intervalo de tempo, T_{mod}, tem impacto maior para esse mecanismo de introdução. Se o moderador necessitar de 15 dias para censurar um objeto, a fração de *downloads* não poluídos

chega a somente 0,44 no final do vigésimo quinto dia. Isso ocorre pois assim como na figura 6.1-b, o moderador leva muito tempo para censurar as versões populares, que possuem muitas cópias poluídas disseminadas.

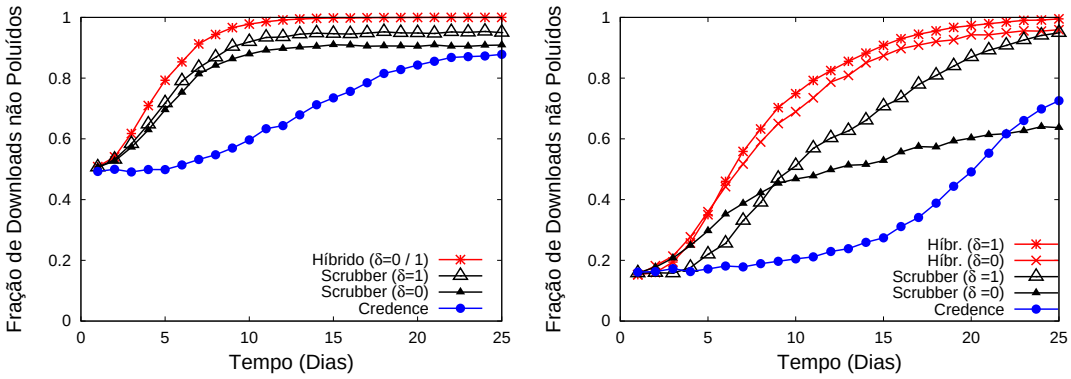
6.2 Sistemas de Reputação

Esta seção apresenta os resultados da avaliação das estratégias baseadas em reputação. Os sistemas cobertos nessa análise são o *Credence*, apresentado no capítulo 2, e os sistemas *Scrubber* e o *Híbrido*, propostos nesta dissertação e apresentados no capítulo 5.

Como o capítulo 5 mostrou, os sistemas *Scrubber* e *Híbrido* punem os pares que compartilham objetos poluídos. Todas as requisições de *download* do par j para i são negadas, se i considerar j não confiável ($R_{i(j)} < R_{min(i)}$). Essa punição pode ser interpretada como um incentivo para que j apague os seus objetos poluídos, para assim aumentar a sua reputação no sistema, e deixar de ter suas requisições negadas. Portanto, assume-se que a probabilidade de j reagir a esse incentivo é capturada pelo parâmetro δ, apresentado no capítulo 3 (equação 3.1). Foram executados experimentos com os três modelos de reação ao incentivo, δ, a saber, *fixo*, *aumento linear* e *aumento quadrático*. Nesses experimentos, a reação ao incentivo consiste em apagar todos os objetos poluídos compartilhados imediatamente após o recebimento da negação da requisição de *download*. É importante ressaltar que, na avaliação apresentada neste capítulo os usuários não apagam voluntariamente seus objetos, como foi avaliado no capítulo 4. O conteúdo poluído só é apagado pelos pares não maliciosos mediante às punições impostas pelo *Scrubber* e *Híbrido*.

O parâmetro $p_{\text{opinião}}$, para os três sistemas de reputação, significa a probabilidade dos pares opinarem sobre a autenticidade do objeto obtido (poluído ou não). Essa opinião é interpretada de forma diferente para cada um dos sistemas analisados. Para o *Credence*, a opinião significa o atribuir o voto sobre autenticidade do objeto obtido e a classificar os pares retornaram votos sobre o objeto. Para o *Scrubber*, a opinião significa atualizar a Experiência Individual de todas as fontes que enviaram o objeto. Finalmente, para o *Híbrido*, significa tanto atribuir o voto sobre autenticidade do objeto quanto a atualizar a Experiência Individual das fontes e dos pares que retornaram votos sobre o objeto obtido.

Além disso, para os experimentos com o sistema *Scrubber* e *Híbrido*, $R_{\text{início}}$ foi fixado em $0,5$ e os valores dos limiares de reputação, $R_{min(i)}$, assume-se sendo distribuídos uniformemente entre $[0,1,\ 0,4]$. Para o sistema *Híbrido* em particular, é assumido, por questões de simplificação, fatores de penalidades iguais tanto para pares que distribuem conteúdo poluído quanto



(a) Inserção de Versões Falsas (b) Corrupção do Identificador

Figura 6.3: Eficácia dos Sistemas de Reputação contra a Disseminação de Poluição ($p_{\text{opinião}} = 1$, $\beta = 1$).

para os que votam em discordância em objetos do sistema. Em outras palavras, assume-se que $\alpha_d = \alpha_d^{\text{mentiroso}} = \alpha_d^{\text{poluidor}}$. Para uma comparação justa, é assumido que a *tabela de correlações* do *Credence* é atualizada antes de cada *download*. Além disso, assume-se que $\alpha_d = 0,4$, $\alpha_i = 0,2$ (para o *Scrubber* e *Híbrido*) e o intervalo T_{busca} entre duas execuções do *protocolo de fofoca* (*Credence*) e a *pesquisa por testemunho* (*Scrubber* e *Híbrido*) é uma hora.

As próximas seções apresentam os resultados dos experimentos. A seção 6.2.1 apresenta resultados para variadas configurações de sistema e comportamento dos pares. A seção 6.2.2 apresenta uma avaliação de sensibilidade dos parâmetros dos sistemas propostos. Finalmente, a seção 6.2.3 apresenta a análise realizada quando os pares poluidores utilizam de ataques de conluio, *whitewashing* e *Sybil* para tornar a disseminação de conteúdo poluído mais efetiva. Todos os experimentos foram executados para os dois mecanismos de introdução de poluição.

6.2.1 Eficácia dos Sistemas de Reputação

Esta seção apresenta uma avaliação da eficácia dos três sistemas de reputação avaliados. A avaliação apresentada considera os dois mecanismos de introdução de poluição, bem como diferentes níveis de cooperação da comunidade, capturados pelos parâmetros δ, $p_{\text{opinião}}$ e p_{erro}.

A figura 6.3 apresenta a variação da fração de *downloads* de objetos não poluídos para os dois mecanismos de introdução de poluição, considerando que todos os pares sempre cooperam com o sistema ($p_{\text{opinião}} = 1$). Para o *Scrubber* e o *Híbrido* assume-se ainda que a probabilidade dos pares reagirem às punições (δ) é fixa, e considera-se os dois cenários extremos: (a) quando todos os pares reagem às punições ($\delta = 1$) e (b) quando nenhum par reage às punições ($\delta = 0$).

A figura 6.3-a apresenta os resultados para a *inserção de versões falsas*. Severamente iso-

lando os poluidores ativos no início da simulação, o *Scrubber* converge rapidamente para uma eficiência máxima de 94%, se todos os pares reagirem às punições ($\delta = 1$). Uma eficiência de 100% não é alcançada devido aos objetos poluídos armazenados, mas não enviados com frequência, pelos *poluidores passivos*. Relembrando que os *poluidores passivos* são *pares não maliciosos* que realizaram *download* de objetos poluídos, e os compartilham por descuido. Esses *poluidores passivos* enviam uma quantidade de objetos não poluídos maior que poluídos e portanto conseguem uma reputação agregada na rede alta o suficiente para escapar da punição. Se nenhum par apaga os seus objetos poluídos ($\delta = 0$), o *Scrubber* alcança uma fração de *downloads* não poluídos de 0,89. Apesar de uma eficiência de 100% não ser alcançada, o *Scrubber* converge para sua eficiência máxima rapidamente, em 12 dias para ambos cenários ($\delta = 0$ e $\delta = 1$).

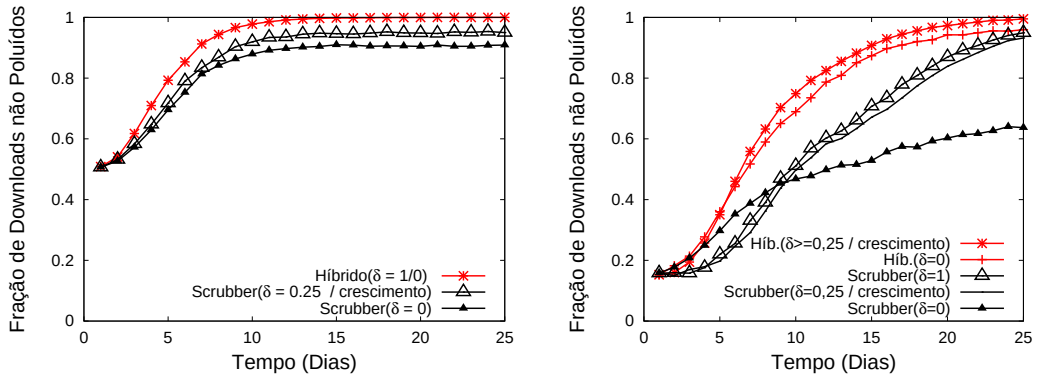
O *Credence*, por outro lado, tem uma eficácia pior e leva 25 dias para alcançar uma eficiência de 0,87. Porém, por estar classificando objetos, o *Credence* eventualmente consegue isolar todas as fontes de conteúdo poluído.

Finalmente, combinando reputação de par e a classificação de objetos, o sistema *Híbrido* consegue isolar os poluidores ativos e os objetos poluídos rapidamente, alcançando uma eficiência de 100% no dia 14. Além disso, o mecanismo sofre pouquíssimas alterações quando o parâmetro δ é variado, fato que será explicado ao longo do texto.

A figura 6.3-b mostra os resultados para a *corrupção do identificador*. Como o capítulo 4 discutiu, esse mecanismo dissemina objetos poluídos em uma velocidade maior que a *inserção de versões falsas*. Portanto, o número de poluidores passivos é maior quando a *corrupção do identificador* é utilizada. As curvas para o *Scrubber* ilustram esse fato. Se todos os pares reagirem às punições ($\delta = 1$), então, como existem muitos poluidores passivos, as punições aplicadas nesses pares é mais frequente. Portanto, muitos pares apagam seus objetos poluídos e o *Scrubber* consegue uma eficiência de 94% ao fim do dia 25, mesma eficiência alcançada quando os poluidores utilizam a *inserção de versões falsas*.

Porém, se os pares nunca reagem às punições ($\delta = 0$), o *Scrubber* sofre muito com o número de poluidores passivos disseminando poluição, alcançando ao fim do dia 25 de simulação uma eficiência de 64%. Isso mostra que o *Scrubber* depende mais do seu mecanismo de punição, e consequentemente da reabilitação.

O *Credence* não possui um mecanismo de punição para incentivar os poluidores passivos a apagar as suas cópias. Porém, classificando objetos, o *Credence* consegue isolar o conteúdo poluído disseminado tanto pelos poluidores ativos quanto pelos poluidores passivos. Dessa forma, o *Credence* possui um crescimento lento, porém estável, alcançando uma eficiência de



(a) Inserção de Versões Falsas (b) Corrupção do Identificador

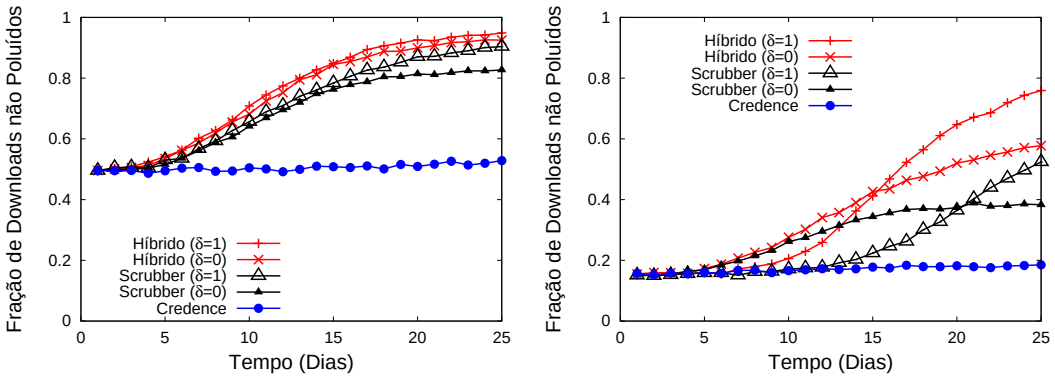
Figura 6.4: Impacto da Probabilidade dos Usuários Reagirem aos Incentivos ($p_{\text{opinião}} = 1$, $\beta = 1$).

74% no fim do dia 25 e ultrapassando o *Scrubber*, para $\delta = 0$, no dia 22.

O sistema *Híbrido*, combinando as reputações de par e a classificação de objetos, consegue uma eficiência de 100% ao fim do dia 25 quando todos os pares reagem às punições ($\delta = 1$). Mesmo se nenhum par reagir às punições ($\delta = 0$), uma eficiência de 96% é alcançada ao fim do dia 25. Isso mostra que o sistema *Híbrido* depende bem menos do mecanismo de punição que o *Scrubber*. Isso acontece pois, assim como no *Credence*, a classificação de objetos do *Híbrido* isola o conteúdo poluído disseminado pelos poluidores passivos. Dessa forma, no sistema *Híbrido*, a necessidade de que os poluidores passivos apaguem suas cópias poluídas é menor.

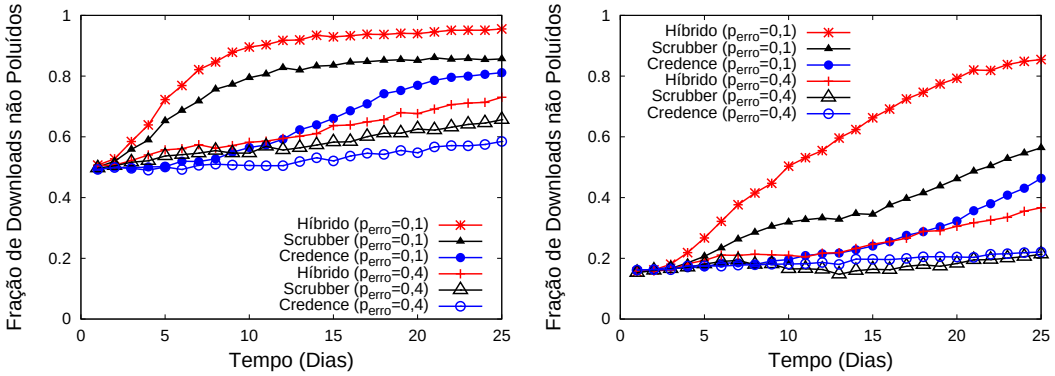
A figura 6.4 avalia a variação da probabilidade dos pares reagirem às punições apagando todos os seus objetos poluídos (δ), para a *inserção de versões falsas* e para a *corrupção do identificador*. O rótulo indicado como *crescimento* na figura se refere aos modelos de reação ao incentivo *crescimento linear* e *quadrático*. As curvas para valores distintos de δ representadas juntas são resultados que apresentaram uma diferença muito pequena entre si. O *Scrubber* é penalizado se nenhum par apaga seus objetos poluídos ($\delta = 0$). Contudo, se até 25% dos pares reagem aos incentivos apagando seu conteúdo poluído ($\delta = 0,25$), o *Scrubber* ainda mantém uma convergência similar quando $\delta = 1$. Isso mostra que o *Scrubber* é penalizado somente se poucos pares não reagem às punições. Por outro lado, como já discutido, o *Híbrido* sobre poucas alterações com a variação da probabilidade dos pares reagirem às punições.

A figura 6.5 mostra o impacto da cooperação dos usuários quando somente 25% dos pares não maliciosos retornam opiniões para o sistema ($p_{\text{opinião}} = 0,25$). A figura 6.5-a mostra o resultado para a *inserção de versões falsas*. Comparando os resultados com a figura 6.3-a pode-se observar que todos os sistemas são severamente penalizados pela falta de cooperação. Se



(a) Inserção de Versões Falsas, $p_{\text{opinião}} = 0.25$ (b) Corrupção do Identificador, $p_{\text{opinião}} = 0.25$

Figura 6.5: Impacto da Cooperação dos Usuários - $p_{\text{opinião}}$ ($\beta = 1$).



(a) Inserção de Versões Falsas (b) Corrupção do Identificador

Figura 6.6: Impacto da Cooperação dos Usuários - p_{error} ($p_{\text{opinião}} = 1$, $\beta = 1$, $\delta = 1$).

nenhum par reage aos incentivos ($\delta = 0$), o *Scrubber* apresenta uma eficiência de 82%, em comparação aos 89% quando todos pares cooperam ($p_{\text{opinião}} = 1$). Para o sistema *Híbrido*, uma eficiência de 92% é alcançada, em comparação aos 100% de eficiência quando todos os pares cooperam. Contudo, o impacto nesses dois sistemas é muito menor que no *Credence*, que não consegue reduzir a disseminação de poluição nesse cenário.

A figura 6.5-b mostra o resultado para a *corrupção do identificador*. Novamente, todos os sistemas são severamente penalizados. Observa-se que pela falta de cooperação dos pares, o parâmetro δ tem um maior impacto no sistema *Híbrido*. Isso ocorre pois a falta de cooperação leva a uma redução da eficiência do *Híbrido* em classificar objetos e isso faz com que ele fique mais sensível à presença dos poluidores passivos, consequentemente dependendo mais do mecanismo de punição e da reação dos pares à punição (δ), assim como acontece com o *Scrubber*. O *Credence* não consegue reduzir a disseminação de conteúdo poluído, assim como para a *inserção de versões falsas*.

Fração de *Downloads* não Poluídos no 25^o dia									
		Inserção de Versões Falsas - α_i				**Corrupção do Identificador - α_i**			
		0,01	**0,1**	**0,2**	**0,4**	**0,01**	**0,1**	**0,2**	**0,4**
Scrubber α_d	**0.1**	0,54	-	-	-	0,21	-	-	-
	0,2	0,65	0,65	-	-	0,47	0,49	-	-
	0,4	0,93	0,92	0,92	-	0,82	0,90	0,94	-
	0,8	0,94	0,93	0,93	0,93	0,59	0,74	0,82	0,92
Híbrido α_d	**0,1**	0,88	-	-	-	0,79	-	-	-
	0,2	0,91	0,91	-	-	0,82	0,82	-	-
	0,4	1,0	1,0	1,0	-	0,95	0,98	0,99	-
	0,8	1,0	1,0	1,0	1,0	0,86	0,94	0,97	0,99

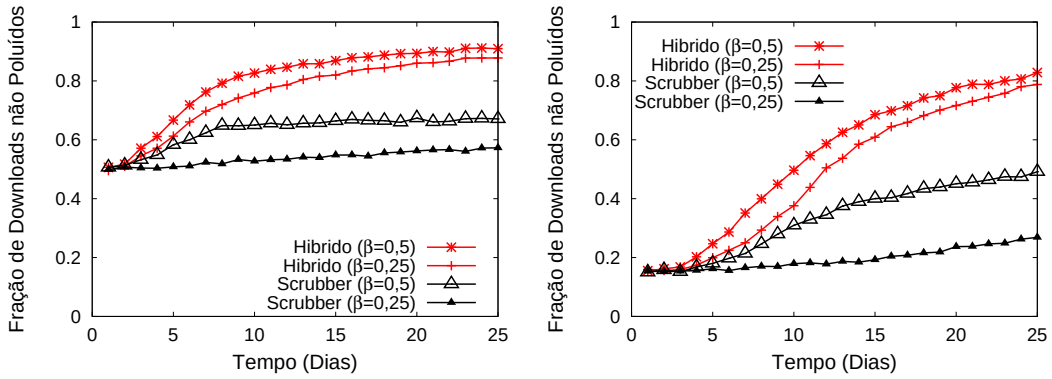
Tabela 6.1: Impacto dos Parâmetros de Penalidade e Recompensa - $\alpha_d > \alpha_i$ ($p_{\text{opinião}} = 1$, $\beta = 1$, $\delta = 1$).

Finalmente, a figura 6.6 apresenta o impacto dos pares retornarem opiniões erradas para o sistema (p_{erro}), ou seja, avaliar a autenticidade dos objetos poluídos como não poluídos, e vice-versa. A figura 6.6-a mostra os resultados para a *inserção de versões falsas* e a figura 6.6-b para a *corrupção do identificador*. Todos os três sistemas são severamente penalizados, mesmo se somente 10% dos pares cometerem erros. Se 40% dos pares errarem suas opiniões, o *Scrubber* e o *Credence* conseguem uma redução muito pequena da disseminação de poluição, tanto para a *inserção de versões falsas* quanto para a *corrupção do identificador*. O sistema *Híbrido*, mesmo sofrendo com imprecisão das opiniões atribuídas para pares e objetos, é mais robusto e consegue reduzir a poluição.

6.2.2 Impacto dos Parâmetros

Esta seção avalia a sensibilidade dos sistemas de reputação aos valores atribuídos aos seus principais parâmetros de configuração. Para o *Scrubber* e o *Híbrido*, avalia-se o impacto dos fatores de recompensa (α_i) e penalidade (α_d), bem como o peso atribuído ao *testemunho* do sistema no cálculo das reputações dos pares (β). Por fim, é avaliado o impacto do intervalo de tempo, T_{busca}, entre execuções consecutivas do *protocolo de fofocas* (*Credence*) e da *pesquisas por testemunho* (*Scrubber* e *Híbrido*).

A tabela 6.1 mostra o impacto dos fatores de penalidade (α_d) e recompensa (α_i) na eficácia dos sistemas de reputação *Scrubber* e *Híbrido*, assumindo $\alpha_d > \alpha_i$. Os valores presentes na tabela correspondem à fração de *downloads* não poluídos no vigésimo quinto dia. Para ambos sistemas, altos valores de α_d implicam em uma punição mais severa e, consequentemente, em uma convergência mais rápida do sistema. Porém, se o valor de α_d for muito alto ($\alpha_d > 0,4$),



(a) Inserção de Versões Falsas (b) Corrupção do Identificador

Figura 6.7: Impacto do β ($p_{\text{opinião}} = 1$, $\delta = 1$).

os poluidores passivos começam a ser excessivamente punidos, reduzindo assim o número de *downloads* realizados e, consequentemente, diminuindo a efetividade do sistema.

A fração de *downloads* não poluídos é bem menos sensível à variação do parâmetro α_i. Para a *inserção de versões falsas* o valor de α_i não altera a eficácia do sistema. Porém, para a *corrupção do identificador* observa-se que quanto maior os valores de α_i, maior é a eficácia do sistema. Isso acontece pois baixos valores de α_i fazem os poluidores passivos serem excessivamente punidos, assim como acontece com altos valores de α_d. Todavia, pode-se observar que o sistema *Híbrido* é menos sensível à variação destes parâmetros.

A figura 6.7 apresenta o impacto de diferentes valores do parâmetro β do *Scrubber* e *Híbrido*, considerando $\delta = 1$ e os dois mecanismos de introdução de poluição. Pode-se observar que quanto maior o valor de β melhor é a eficácia das estratégias de reputação. Isso ocorre pois o testemunho da comunidade ajuda os pares a identificar mais rapidamente os poluidores ativos. Se um peso maior é atribuído à *Experiência Individual*, os poluidores ativos levam um grande tempo para serem identificados, e portanto a eficácia é comprometida. É interessante observar que o *Híbrido*, por também classificar objetos, sofre menos que o *Scrubber* com o fato de não conseguir isolar os poluidores ativos rapidamente.

Porém, menores valores de β tornam a eficácia a *longo prazo* do *Scrubber* mais sensível. A figura 6.8 ilustra o comportamento do *Scrubber* para diferentes valores de δ e β, considerando uma comunidade menor com 200 pares não maliciosos e 50 poluidores ativos ($N_b = 200$ e $N_p = 50$). Os demais parâmetros do sistema são mantidos como na tabela 4.2. Se os pares sempre reagem às punições ($\delta = 1$), a reputação agregada atribuída pela comunidade favorece os poluidores passivos, como discutido na seção 6.2.1. Logo, pequenos valores de β podem aumentar a probabilidade dos poluidores passivos serem punidos e, consequentemente de mais

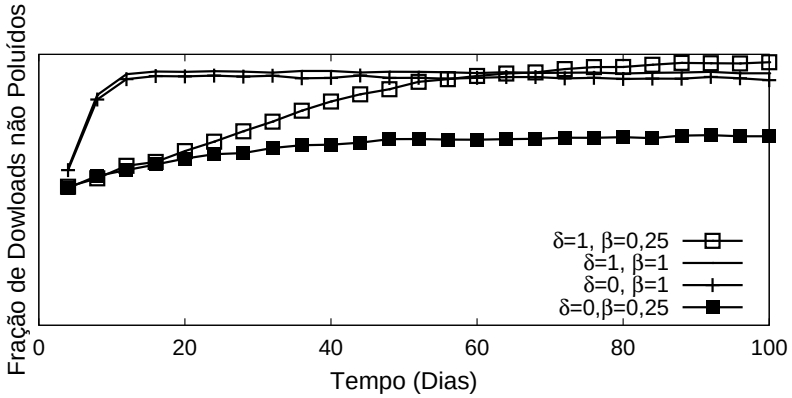


Figura 6.8: Impacto do β para a Inserção de Versões Falsas no *Scrubber* ($p_{\text{opinião}} = 1$, $N_b = 200$, $N_p = 50$).

conteúdo poluído ser apagado. Portanto, apesar de uma convergência mais lenta, valores menores de β levam a uma maior eficiência *a longo prazo* do sistema. Se os pares nunca reagem às punições ($\delta = 0$), o *Scrubber* que demora mais tempo para isolar os poluidores ativos para $\beta = 0,25$, acaba sofrendo muito com objetos poluídos armazenados pelos poluidores passivos. É importante ressaltar que esse efeito não acontece com o sistema *Híbrido* pois a sua classificação de objetos evita que o conteúdo poluído compartilhado pelos poluidores passivos sejam obtidos por outros pares.

Além disso, altos valores altos de β tornam o *Scrubber* e *Híbrido* mais dependentes do seu mecanismo de coleta de opiniões. A tabela 6.2 mostra a eficiência dos três sistemas de reputação quando o intervalo, T_{busca}, entre duas buscas por opinião na comunidade é variado. O *Scrubber* e o *Híbrido* coletam a opinião da comunidade através da *pesquisa por testemunho* e o *Credence* o faz através do seu *protocolo de fofoca*. Note que, o *Credence* é mais robusto a intervalos de tempo, T_{busca}, grandes, uma vez que a opinião da comunidade sobre um par j é utilizada por i somente se j nunca retornou votos para i através da *pesquisa por votos*. Já no *Scrubber* e no *Híbrido*, valores grandes de T_{busca} tem um impacto significativo na eficácia dos sistemas, uma vez que a opinião da comunidade sempre é levada em consideração ($\beta = 1$) nesse cenário. Logo, diferentemente do *Credence*, opiniões desatualizadas e imprecisas sempre são refletidas nos cálculos das reputações.

A tabela 6.3 apresenta os resultados da variação do intervalo de tempo, T_{busca}, quando o *Scrubber* e o *Híbrido* utilizam o valor do parâmetro $\beta = 0,5$. As mesmas conclusões obtidas a partir da tabela 6.2 são válidas. Contudo, é interessante observar que para intervalos de tempo muito grandes ($T_{\text{busca}} = 60$ horas), a eficácia do *Scrubber* e *Híbrido* é superior aos valores apresentados tabela 6.2 para o mesmo intervalo de tempo. A opinião da comunidade, nesse caso

Fração de *Downloads* não Poluídos no 25^o Dia						
T_{busca}	**Inserção de Versões Falsas**			**Corrupção do Identificador**		
horas	***Híbrido***	***Scrubber***	***Credence***	***Híbrido***	***Scrubber***	***Credence***
1	1,0	0,93	0,88	0,99	0,94	0,73
6	0,99	0,90	0,86	0,89	0,65	0,54
12	0,96	0,83	0,85	0,78	0,48	0,49
60	0,71	0,51	0,77	0,26	0,16	0,33

Tabela 6.2: Impacto da Frequência da Coleta da Opinião da Rede - $\beta = 1.0$ ($p_{\text{opinião}} = 1$, $\delta = 1$).

Fração de *Downloads* não Poluídos no 25^o Dia				
T_{busca}	**Inserção de Versões Falsas**		**Corrupção do Identificador**	
horas	***Híbrido***	***Scrubber***	***Híbrido***	***Scrubber***
1	0,91	0,67	0,83	0,49
6	0,92	0,68	0,81	0,45
12	0,90	0,65	0,74	0,41
60	0,73	0,52	0,29	0,20

Tabela 6.3: Impacto da Frequência da Coleta da Opinião da Rede - $\beta = 0.5$ ($p_{\text{opinião}} = 1$, $\delta = 1$).

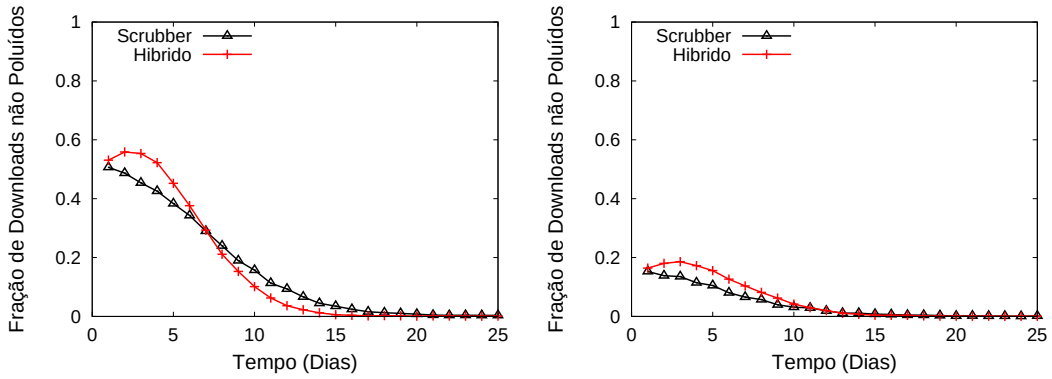
tem um peso menor nos cálculos das reputações. Logo, os valores da Experiência Individual ajudam à minimizar o impacto das opiniões desatualizadas da comunidade.

Todavia, pode-se observar que o sistema *Híbrido* ainda tem uma eficácia superior a outras estratégias, quando o intervalo de tempo, T_{busca}, é menor que 12 horas. Como não é esperado que esses valores de T_{busca} provoquem uma grande sobrecarga (*overhead*) na rede, acredita-se que o sistema *Híbrido* seja a melhor estratégia para cenários práticos.

6.2.3 Sistemas de Reputação sob Ataques

Esta seção apresenta uma avaliação dos três sistemas de reputação em cenários nos quais os pares utilizam ataques de conluio, *whitewashing* e *Sybil* para tornar a disseminação de conteúdo poluídos mais efetiva, a despeito dos mecanismos de reputação. Os modelos de ataque por conluio, *whitewashing* e *Sybil* considerados nesta seção foram discutidos no capítulo 3.

A primeira avaliação trata da robustez dos sistemas ao ataque de conluio. Conforme discutido no capítulo 3, foi considerado o ataque de conluio para difamação de pares. Nesse ataque, os pares maliciosos (poluidores ativos) agem conjuntamente retornando opiniões para os sistemas, de forma a aumentar as suas reputações e diminuir a reputação dos pares não maliciosos. No sistema *Credence*, os pares maliciosos disseminam essas reputações através do *protocolo de*



(a) Inserção de Versões Falsas (b) Corrupção do Identificador

Figura 6.9: Eficácia dos Sistemas Híbrido e Scrubber de Reputação sob o ataque de Conluio **caso os Testemunhos sejam coletados de qualquer par conhecido do sistema** ($p_{\text{opinião}} = 1$, $\beta = 1$, $\delta = 1$).

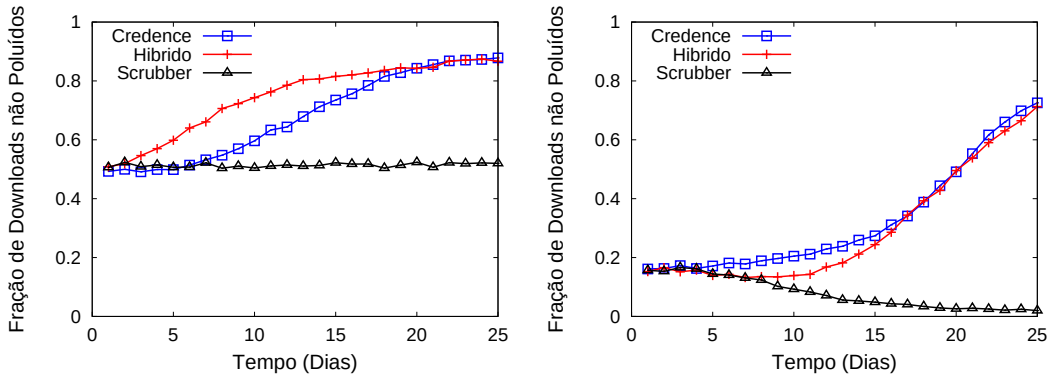
fofoca, e no *Scrubber* e *Híbrido*, elas são disseminadas através da *pesquisa por testemunho*.

Os resultados desta avaliação indicam que nenhum dos três sistemas de reputação é impactado por esse ataque e os resultados obtidos foram os mesmos da figura 6.3. No *Credence*, esse ataque é ineficaz pois um par *i* considera a opinião obtida por um par j, somente se j for um par *conhecido* por *i* e possuir uma *correlação*, θ, maior que $0,5$. Como os poluidores ativos nunca votam em objetos, logo eles nunca são *conhecidos* por nenhum par do sistema, e portanto não conseguem realizar uma difamação.

A robustez tanto do *Scrubber* quanto do *Híbrido* se deve particularmente ao envio da *pesquisa por testemunho* somente para pares conhecidos e que tenham tanto $T_{i(j)}$ quanto $I_{i(j)}$ superiores ao limiar $R_{min(i)}$. Esse mecanismo impede um ataque de conluio pois, com a utilização do parâmetro $\alpha_d = 0,4$, um poluidor ativo j até então desconhecido pelo par *i*, sempre tem $I_{i(j)}$ inferior a $R_{min(i)}$ após enviar um objeto poluído. Além disso, qualquer par *k*, que o obtiver a Experiência Individual de $I_{i(j)}$, terá o valor $T_{k(j)}$ abaixo do limiar $R_{min(i)}$. Logo, os poluidores não conseguem disseminar as suas reputações difamatórias.

As figuras 6.9-a e 6.9-b mostram a eficácia desses dois sistemas, para a *inserção de versões falsas* e a *corrupção do identificador* respectivamente, **caso os Testemunhos fossem coletados de qualquer par conhecido do sistema**. Agindo conjuntamente para aumentar a sua reputação e difamar os pares não maliciosos, os poluidores ativos conseguem inverter o status do sistema, fazendo com que pares não maliciosos sejam identificados como poluidores ativos, e vice-versa. O resultado é desastroso e rapidamente a fração de *downloads* não poluídos chega próximo de zero, para ambos mecanismos de introdução de poluição.

A figura 6.10 mostra o impacto no sistema quando os pares maliciosos utilizam o ataque



(a) Inserção de Versões Falsas (b) Corrupção do Identificador

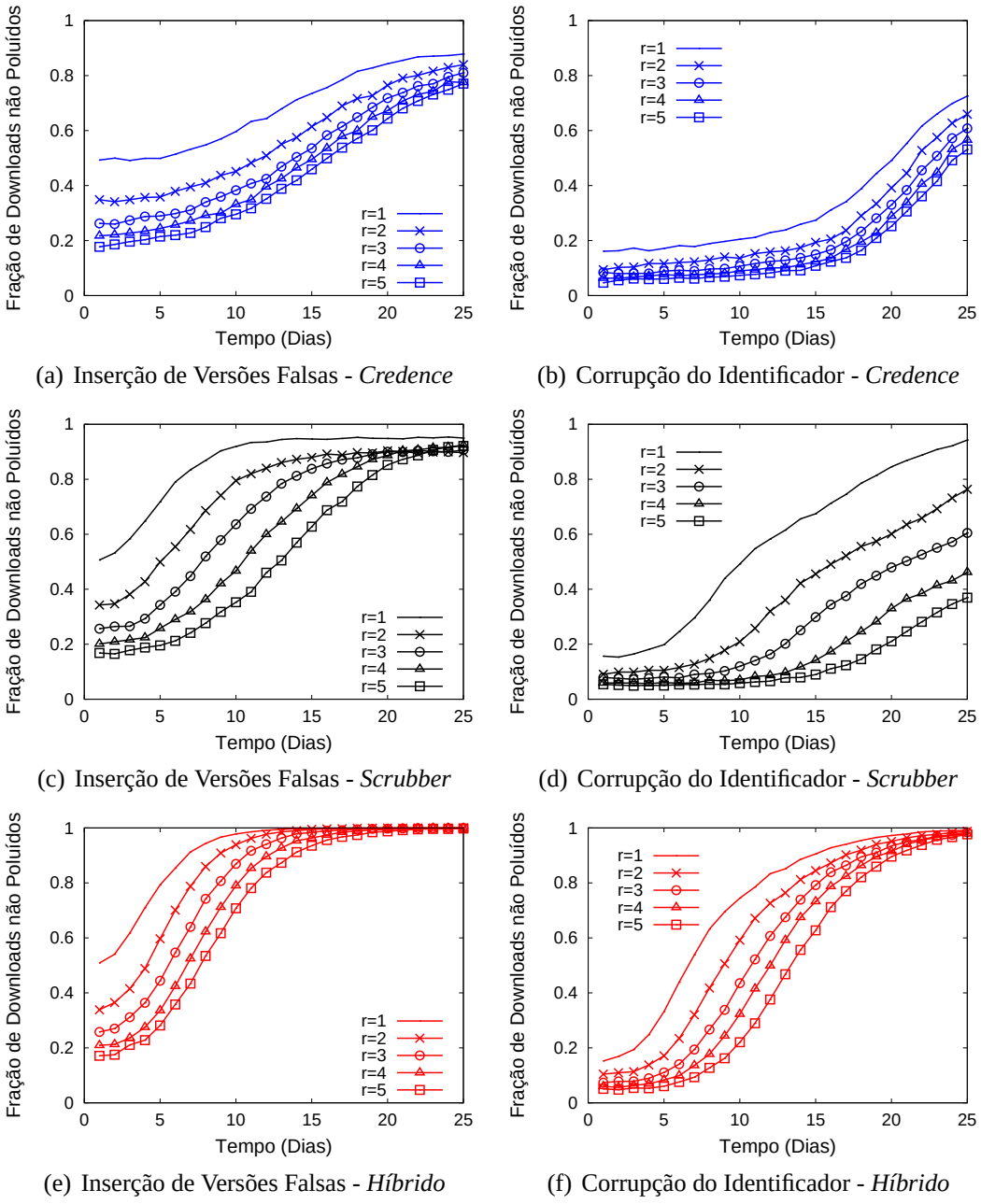
Figura 6.10: Eficácia dos Sistemas de Reputação sob o ataque de Whitewashing ($p_{\text{opinião}} = 1$, $\beta = 1$, $\delta = 1$).

de *whitewashing*, para ambos mecanismos de introdução de poluição. Considera-se que os pares maliciosos mudam sua identidade quando 10% dos pares não maliciosos os identificam como poluidores ($w = 10\%$). Lembrando que no *Scrubber* e *Híbrido* um par é considerado malicioso quando a sua reputação fica inferior ao limiar $R_{min(i)}$. O *Credence* não sofre ataque de *whitewashing* pois ele não atribui reputação aos pares como fontes de objetos poluídos.

O impacto desse ataque no *Scrubber* é significativo. Para a *inserção de versões falsas* o *Scrubber* não consegue reduzir a poluição. Porém, para a *corrupção do identificador* a eficiência do *Scrubber* é reduzida e chega próximo de 0% ao fim do dia 25. Nesse cenário, os poluidores ativos disseminam poluição em uma velocidade muito grande, ao mesmo tempo que trocam constantemente de identidade. Dessa forma, o número de poluidores passivos cresce rapidamente, e os incentivos atribuídos pelo *Scrubber* não são capazes de reabilitar esses pares poluidores. A eficiência do *Híbrido* também é bastante prejudicada, porém ele ainda consegue uma boa redução da poluição trabalhando somente com sua classificação de objetos.

Esta dissertação aborda somente o ataque de *whitewashing* quando os pares trocam a sua identidade. Por esse motivo, o sistema *Híbrido* e o *Credence* têm uma boa eficácia. Além disso, é possível realizar um ataque de *whitewashing de objetos*, no qual os pares maliciosos constantemente renovam as versões poluídas compartilhadas. Os sistemas que classificam a objetos, como *Híbrido* e o *Credence*, podem ser muito prejudicados por esse ataque. A avaliação da eficácia dos sistemas de reputação sob esse ataque é trabalho futuro.

Finalmente, A figura 6.11 apresenta os resultados da avaliação quando os três sistemas de reputação estão sob ataque *Sybil*. Nesse ataque, cada par poluidor se replica r vezes no sistema. Os sistemas *Híbrido* e o *Credence* apresentam uma boa eficácia contra esse tipo de ataque, apresentando *incrementos marginais* (*diminishing returns*) à medida em que o número



(a) Inserção de Versões Falsas - *Credence*

(b) Corrupção do Identificador - *Credence*

(c) Inserção de Versões Falsas - *Scrubber*

(d) Corrupção do Identificador - *Scrubber*

(e) Inserção de Versões Falsas - *Híbrido*

(f) Corrupção do Identificador - *Híbrido*

Figura 6.11: Eficácia dos Sistemas de Reputação sob o ataque *Sybil* ($p_{\text{opinião}} = 1$, $\beta = 1$, $\delta = 1$).

de réplicas (r) aumenta. Maiores valores de r implicam em uma disseminação de poluição mais veloz. Porém, à medida que os pares votam nos objetos, a poluição é rapidamente contida. Isso ocorre pois a quantidade de versões distintas permanece a mesma, apesar do número de poluidores ativos aumentar.

O *Scrubber* também apresenta bons resultados para a *inserção de versões falsas*. Porém, para a *corrupção do identificador*, o *Scrubber* não apresenta resultados tão expressivos. Nesse cenário, o número de pares maliciosos é muito alto, logo os pares não maliciosos levam um

longo tempo para identificá-los. Contudo, mesmo sofrendo com o ataque *Sybil* para a *corrupção do identificador*, o *Scrubber* ainda apresenta uma eficiência melhor que o *Credence* se $r < 5$.

6.2.4 Considerações de Implementação

Esta seção apresenta considerações preliminares sobre a implementação dos sistemas de reputação *Scrubber* e *Híbrido* em sistemas P2P reais de compartilhamento de arquivos.

Os sistemas *Scrubber* e *Híbrido* poderiam ser implementados como extensões de clientes utilizados para conectar nos sistemas P2P. Por exemplo, para esse fim poderiam ser utilizados clientes com *código aberto* [58] (*open source*), como é o caso do *LimeWire* [84], um famoso cliente Gnutella.

A interface gráfica dos clientes deveria ser modificada de forma a: (1) permitir que os usuários possam votar na na autenticidade de um objeto obtido, (2) incluir avisos alertando os usuários, poluidores, sobre punições que eles receberam, isto é, recusa de requisições de *download* por outros usuários e (3) classificação das versões disponíveis para *download* (especificamente para o Híbrido).

A implementação dos componentes dos dois sistemas *Scrubber* ou *Híbrido* é realizada sobre o protocolo do sistema P2P, não necessitando realizar nenhum tipo de alteração no protocolo original. É interessante mencionar que, particularmente para o sistema *Híbrido*, a *pesquisa por votos* poderia ser roteada através do próprio mecanismo de pesquisa de objetos do sistema P2P. Isso significa, por exemplo, utilizar os *super-pares* do sistema Kazaa para pesquisar votos em objetos.

Possíveis alterações nas estratégias também poderiam ser realizadas para otimizar o seu desempenho em sistemas P2P reais. Os componentes do *Scrubber* e *Híbrido* (ex.: *Experiência Individual, Testemunhos e votos*) poderiam ser armazenados em *cache*, evitando assim um alto consumo de memória. Outra modificação seria um mecanismo para apagar *Testemunhos* muito antigos, fazendo com que o sistema utilize somente as opiniões mais recentes, e logo mais precisas, para calcular as reputações. Finalmente, poderia-se alterar os sistemas para atualizar as reputações somente em momentos no quais os clientes estão com baixa carga de processamento. A avaliação do impacto dessas modificações na eficácia das estratégias é deixada para trabalho futuro.

Uma comparação do gasto de memória, processamento e banda de rede pode ser realizada entre os sistemas apresentados. Os requisitos de banda de rede entre os sistemas *Híbrido* e *Credence* é o mesmo, pois ambos possuem a *pesquisa por votos* e buscam por opiniões no

sistema. O sistema *Híbrido* possui uma carga de processamento e memória um pouco maiores que o *Credence*, pois ele armazena, além dos pares votantes, informações sobre as fontes de *download*.

Os requisitos de memória e processamento entre o *Scrubber* e o *Credence* são equivalentes, considerando que o número de opiniões coletadas são iguais ao longo do tempo. Porém, o *Credence* utiliza mais banda de rede que o *Scrubber* já que ele realiza a *pesquisa por votos* antes de cada *download*.

Finalmente, os resultados mostraram que o *Híbrido* possui uma melhor eficácia que o *Scrubber* em todos os cenários avaliados. Porém, existe um compromisso para esta maior eficácia do *Híbrido*. Tipicamente, o número de objetos é muito maior do que o número de pares nos sistemas P2P de compartilhamento de arquivos. Portanto, como o *Híbrido* classifica objetos e reputa pares, ele tem um custo maior que o *Scrubber* para banda de rede, armazenamento e processamento.

7 Conclusões e Trabalhos Futuros

A poluição de conteúdo é um padrão de comportamento malicioso no qual os pares maliciosos inserem conteúdo corrompido, e portanto inútil, nos sistemas P2P de compartilhamento de arquivos. A introdução de conteúdo poluído no sistema reduz a disponibilidade dos objetos não poluídos, diminuindo assim a confiança dos usuários no sistema. Esta dissertação estudou o problema da disseminação de conteúdo poluído nos sistemas P2P de compartilhamento de arquivos, bem como propôs estratégias para combater este padrão de comportamento.

A avaliação do processo de disseminação de conteúdo poluído considerou dois mecanismos de introdução de poluição, a saber, a *inserção de versões falsas* [45] e a *corrupção do identificador*, este último descrito com detalhes nesta dissertação. Foi realizada uma avaliação, via simulação, do processo de disseminação de conteúdo poluído em sistemas P2P, a partir da introdução de poluição pelos dois mecanismos.

Verificou-se que o mecanismo de *corrupção do identificador* dissemina poluição mais rapidamente no sistema e, logo, é mais difícil de combater. Além disso, verificou-se que a disseminação de poluição não pode ser efetivamente contida apenas através da ação voluntária dos usuários apagarem seus objetos poluídos. Logo, faz-se necessário o projeto e a avaliação de estratégias mais eficazes. Estes resultados foram obtidos via simulação e, para a introdução de poluição via *inserção de cópias falsas*, validados por um modelo analítico. Os modelos de simulação e analítico foram propostos nesta dissertação.

A partir desta primeira análise, foram propostas estratégias para combater a disseminação de conteúdo poluído. Foi apresentada uma estratégia para reduzir a poluição baseada na censura por moderadores. Além disso, foram propostos os sistemas de reputação Scrubber e *Híbrido*. No sistema de reputação *Scrubber*, os pares atribuem reputação uns para os outros como fontes de conteúdo poluído. O sistema *Híbrido* estende o *Scrubber*, combinando as funcionalidades de reputação de pares e classificação de objetos.

Os sistemas de combate à poluição propostos foram avaliados considerando os dois mecanismos de introdução de poluição, bem como diversas configurações de parâmetro e compor-

tamento dos usuários, incluindo ataques do tipo conluio, *Sybil* e *whitewashing*. Foi encontrado que o sistema baseado na censura por moderador consegue uma boa redução da poluição, mas a dependência da intervenção de um moderador apresenta desafios quanto à escalabilidade.

Os dois sistemas de reputação foram avaliados, comparando a sua eficácia em relação ao sistema de reputação *Credence* [80–82]. As principais conclusões encontradas foram que o sistema *Híbrido* consegue reduzir a disseminação de conteúdo poluído de forma mais eficaz que os outros sistemas de reputação, mesmo quando os pares maliciosos utilizam ataques de conluio e *Sybil*. Além disso, o sistema *Híbrido* é menos sensível a variações dos seus parâmetros, em comparação com o *Scrubber*. Finalmente, todos os três sistemas dependem da cooperação dos usuários em atribuir opiniões corretas sobre os outros pares ou objetos do sistema. Todavia, mesmo em comunidades onde grande parte dos pares não cooperam ou não são confiáveis, o sistema *Híbrido* ainda consegue reduzir a disseminação de poluição de forma mais eficaz que as outras estratégias.

O trabalho apresentado nesta dissertação pode ser estendido pelo menos em quatro direções. Primeiramente, a avaliação dos sistemas poderia ser estendida para considerar padrões de ataque mais sofisticados. O comportamento dos sistemas poderia ser avaliado sob um ataque de *traidor*, no qual um par malicioso se comporta adequadamente durante certo tempo, para então atacar o sistema. Outro ataque interessante a ser avaliado seria o *whitewashing* de objetos, no qual os pares poluidores constantemente renovam as cópias poluídas compartilhadas, diminuindo assim a eficácia dos sistemas classificação de objetos (ex.: *Credence* e *Híbrido*) em combater a poluição.

Além disso, a avaliação dos protocolos *Scrubber* e *Híbrido* poderia ser estendida para incluir cenários em que os pares coletam informações, através da *pesquisa por testemunho*, para mais de um par ao mesmo tempo, além de considerar diferentes valores as penalidades $\alpha_d^{\text{mentiroso}}$ e $\alpha_d^{\text{poluidor}}$ (*Híbrido*)

Uma outra direção possível para trabalho futuro é a prototipação dos sistemas *Scrubber* e *Híbrido* em um sistema P2P, seguida de uma avaliação em um cenário real. Neste caso, questões práticas de implementação, discutidas na seção 6.2.4, tais como *cache* e o momento de atualização dos *Testemunhos*, devem ser avaliadas.

Extensões dos sistemas *Scrubber* e *Híbrido* para outras aplicações P2P que também podem ser alvo de poluição podem ser consideradas. Como exemplo de tais aplicações pode-se citar as máquinas de buscas descentralizadas [7] e a transmissão de mídia contínua, ao vivo ou sob demanda, em sistemas P2P [22].

Finalmente, a modelagem analítica da disseminação de poluição, particularmente para o mecanismo de *corrupção do identificador*, também pode ser estudada. O modelo apresentado nessa dissertação precisaria ser estendido para capturar o *download* através múltiplas fontes e a porcentagem de dados de um objeto que pode ser poluída, dois fatores que afetam a disseminação pela *corrupção do identificador*.

APÊNDICE A – Mapeamento dos Parâmetros do Simulador para o Modelo

Os parâmetros do modelo e simulador não são os mesmos e um mapeamento de valores precisa ser realizada para conseguir comparar ambas abordagens. Abaixo seguem as equações que configuram os parâmetros do modelo de acordo com os parâmetros do simulador:

$$M = N_b \frac{\lambda_{\text{saída}}}{\lambda_{\text{entrada}} + \lambda_{\text{saída}}} \quad \text{(A.1)}$$

$$N(0)_b = N_b \frac{\lambda_{\text{saída}}}{\lambda_{\text{entrada}} + \lambda_{\text{saída}}} O_b \quad \text{(A.2)}$$

$$N(0)_p = N_p O_p \quad \text{(A.3)}$$

$$\lambda = \lambda_{\text{download}} \quad \text{(A.4)}$$

$$T = T \quad \text{(A.5)}$$

$$P(i) = \text{Zpif}(\alpha, T) \quad \text{(A.6)}$$

A Equação A.1 atribui valor ao número de pares do modelo de acordo com parâmetros do simulador. No modelo, todos os pares fazem *download*. Portanto, somente os pares não maliciosos do simulador devem ser considerados na atribuição do parâmetro M. Outra questão que se deve considerar é o comportamento dinâmico dos pares no simulador, característica que o modelo isso não apresenta. Neste caso, para realizar o mapeamento foi considerada a média de pares ativos no simulador em um dado instante ($\frac{\lambda_{\text{saída}}}{\lambda_{\text{saída}} + \lambda_{\text{entrada}}}$).

O mapeamento do número de cópias não poluídas e poluídas, apresentado nas Equações A.2 e A.3 respectivamente, é realizado multiplicando o número de pares pelos objetos que cada par

possui. Ressaltando que, assim como foi feito na Equação A.1, somente os pares não maliciosos ativos devem ser considerados para o cálculo do número de versões poluídas. Os mapeamentos realizados nas Equação A.4, A.5 e A.6 são atribuições diretas dos parâmetros do simulador.

Referências Bibliográficas

[1] E. Adar and B. A. Huberman. Free riding on gnutella. *First Monday*, setembro 2000.

[2] H. Almeida, T. Macambira, D. Guedes, V. Almeida, and W. Meira. Um sistema de reputação resistente a ataques sybil para redes overlay. In *III Workshop de Peer-to-Peer (WP2P)*, Belém, Brasil, maio 2007.

[3] D. Anderson, N. Asthagiri, D. Brickley, D. Bricklin, A. Brown, L. Cranor, R. Dingledine, R. Dornfest, M. Freedman, M. Hedlund, T. Hong, G. Kan, A. Langley, R. Lethin, J. Miller, N. Minar, D. Molnar, T. O'Reilly, A. Rubin, C. Shirky, W. Tuvell, J. Udell, Marc W., and B. Wiley. *Peer-to-Peer*. O'Reilly, 2001.

[4] N. Andrade, F. Brasileiro, W. Cirne, and M. Mowbray. Discouraging free riding in a peer-to-peer cpu-sharing grid. In *13th IEEE Symposium on High Performance Distributed Computing (HPDC)*, Honolulu, EUA, junho 2004.

[5] S. Androutsellis-Theotokis and D. Spinellis. A survey of peer-to-peer content distribution technologies. *ACM Computing Surveys (CSUR)*, 36(4):335–371, dezembro 2004.

[6] M. Barbosa, M. Costa, J. Almeida, and V. Almeida. Using locality of reference to improve performance of peer-to-peer applications. *ACM SIGSOFT Software Engineering Notes*, 29(1):216–227, janeiro 2004.

[7] M. Bender, S. Michel, P. Triantafillou, G. Weikum, and C. Zimmer. MINERVA: collaborative P2P search. In *31st International Conference on Very Large Data Bases*, Trondheim, Noruega, agosto 2005.

[8] F. Benevenuto, C. Costa, M. Vasconcelos, V. Almeida, J. Almeida, and M. Mowbray. Impact of Peer Incentives on the Dissemination of Polluted Content. In *21st Annual ACM Symposium on Applied Computing (SAC)*, Dijon, França, abril 2006.

[9] F. Benevenuto, J. Ismael Junior, and J. Almeida. Quantitative evaluation of unstructured peer-to-peer architectures. In *IEEE First International WorkShop on Hot Topics in Peer-to-Peer Systems (Hot-P2P'04)*, Volendam, Holanda, outubro 2004.

[10] N. Bisnik and A. Abouzeid. Optimizing random walk search algorithms in p2p networks. *Comput. Networks*, 51(6):1499–1514, 2007.

[11] BitTorrent. http://www.bittorrent.com/, acessado em junho de 2007.

[12] T. Boutell. PNG (Portable Network Graphics) specification. RFC 2083, Internet Engineering Task Force, março 1997.

[13] E. Buchmann and K. Böhm. FairNet – how to counter free riding in peer-to-peer data structures. In *International Conference on Cooperative Information Systems*, Larnaca, Chipre, outubro 2004.

[14] CacheLogic. http://www.cachelogic.com/, acessado em junho de 2007.

[15] Frederico Ferreira Campos. *Algoritmos Numéricos*. LTC, 2001.

[16] N. Christin, A. Weigend, and J. Chuang. Content availability, pollution and poisoning in file sharing peer-to-peer networks. In *6th ACM Conference on Electronic commerce (EC'05)*, Vancouver, Canadá, junho 2005.

[17] C. Costa and J. Almeida. Reputation systems for fighting pollution in peer-to-peer file sharing systems. In *IEEE International Conference on Peer-to-Peer Computing*, Galway, Irlanda, setembro 2007.

[18] C. Costa, V. Soares, J. Almeida, and V. Almeida. Combatendo a disseminação de conteúdo poluído em redes par-a-par para compartilhamento de arquivos. In *25th Simpósio Brasileiro de Redes de Computadores (SBRC)*, Belém, Brasil, maio 2007.

[19] C. Costa, V. Soares, J. Almeida, and V. Almeida. Fighting pollution dissemination in peer-to-peer networks. In *22nd Annual ACM Symposium on Applied Computing (SAC)*, Seoul, Coreia, março 2007.

[20] C. Costa, V. Soares, F. Benevenuto, M. Vasconcelos, V. Almeida, J. Almeida, and M. Mowbray. Disseminação de conteúdo poluído em redes p2p. In *24th Simpósio Brasileiro de Redes de Computadores (SBRC)*, Curitiba, Brasil, maio 2006.

[21] E. Damiani, De Capitani di Vimercati, S. Paraboschi, P. Samarati, and F. Violante. A reputation-based approach for choosing reliable resources in peer-to-peer networks. In *ACM Conference on Computer and Communications Security*, Washington, EUA, novembro 2002.

[22] P. Dhungel, X. Hei, K. W. Ross, and N. Saxena. The pollution attack in p2p live video streaming: Measurement results and defenses. In *Sigcomm P2P-TV Workshop*, Kyoto, Japão, agosto 2007.

[23] J. Douceur. The sybil attack. In *1st International Workshop on Peer-to-Peer Systems*, Cambridge, EUA, março 2002.

[24] D. Dutta, A. Goel, R. Govindan, and H. Zhang. The design of a distributed rating scheme for peer-to-peer systems. In *1st Workshop on Economics of Peer-to-Peer Systems*, Berkeley, EUA, junho 2003.

[25] D. Eastlake and P. Jones. US secure hash algorithm 1 (SHA1). RFC 3174, Internet Engineering Task Force, setembro 2001.

[26] eBay. http://www.ebay.com/, acessado em junho de 2007.

[27] eMule. http://www.emule-project.net/, acessado em junho de 2007.

[28] M. Feldman, K. Lai, I. Stoica, and J. Chuang. Robust incentive techniques for peer-to-peer networks. In *5th ACM conference on Electronic Commerce*, New York, EUA, maio 2004.

[29] M. Feldman, C. Papadimitriou, J. Chuang, and I. Stoica. Free-riding and whitewashing in peer-to-peer systems. In *ACM SIGCOMM workshop on Practice and theory of incentives in networked systems*, Portland, EUA, setembro 2004.

[30] F. Fessant, S. Handurukande, A. Kermarrec, and L. Massoulie. Clustering in peer-to-peer file sharing workloads. In *3rd International Workshop on Peer-to-Peer Systems (IPTPS)*, San Diego, EUA, fevereiro 2004.

[31] N. Garnett. Digital rights management, copyright, and napster. *ACM SIGecom Exchanges*, 2(2):1–5, 2001.

[32] C. Gkantsidis, M. Mihail, and A. Saberi. Random walks in peer-to-peer networks: algorithms and evaluation. *Perform. Eval.*, 63(3):241–263, 2006.

[33] K. Gummadi, R. Dunn, S. Saroiu, S. Gribble, H. Levy, and J. Zahorjan. Measurement, modeling and analysis of a Peer-to-Peer File-Sharing workload. In *19th ACM Symposium of Operating Systems Principles (SOSP)*, Bolton Landing, EUA, outubro 2003.

[34] Gzip. http://www.gzip.org/, acessado em junho de 2007.

[35] Lost in Usenet - References. http://www.faqs.org/usenet/, acessado em junho de 2007.

[36] Jabber. http://www.jabber.org/, acessado em junho de 2007.

[37] A. Kalafut, A. Acharya, and M. Gupta. A study of malware in peer-to-peer networks. In *6th ACM SIGCOMM on Internet Measurement*, Rio de Janeiro, Brasil, outubro 2006.

[38] S. Kamvar, M. Schlosser, and H. Garcia-Molina. Incentives for combatting freeriding on p2p networks. In *9th International Euro-Par Conference*, Klagenfurt, Áustria, agosto 2003.

[39] S.D. Kamvar, M.T. Schlosser, and H. Garcia-Molina. The Eigentrust Algorithm for Reputation Management in P2P Networks. In *International WWW Conference*, Budapest, Hungria, 2003.

[40] Kazaa. http://www.kazaa.com/, acessado em junho de 2007.

[41] B. Kernighan and D. Ritchie. *The C Programming Language*, *Second Edition*. rentice Hall, Inc., 1988.

[42] R. Kumar, D. Yao, A. Bagchi, K.W. Ross, and D. Rubenstein. Fluid modeling of pollution proliferation in P2P networks. In *ACM Sigmetrics*, Saint-Malo, França, junho 2006.

[43] Python Programming Language. http://www.python.org/, acessado em junho de 2007.

[44] U. Lee, M. Choi, J. Cho, M. Sanadidi, and M. Gerla. Understanding pollution dynamics in p2p file sharing. In *5th International Workshop on Peer-to-Peer Systems (IPTPS)*, Santa Barbara, EUA, fevereiro 2006.

[45] J. Liang, R. Kumar, Y. Xi, and K. W. Ross. Pollution in P2P file sharing systems. In *IEEE Infocom*, Miami, EUA, março 2005.

[46] J. Liang, N. Naoumov, and K. Ross. The index poisoning attack in P2P file-sharing systems. In *IEEE Infocom*, Barcelona, Espanha, abril 2006.

[47] J. Liang, N. Naoumov, and K. W. Ross. Efficient blacklisting and pollution-level estimation in P2P file-sharing systems. In *Asian Internet Engineering Conference (AINTEC'05)*, Bangcoc, Tailândia, dezembro 2005.

[48] D. Liben-Nowell, H. Balakrishnan, and D. Karger. Analysis of the evolution of peer-to-peer systems. In *twenty-first annual symposium on Principles of distributed computing*, Monterey, EUA, julho 2002.

[49] K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim. A survey and comparison of peer-to-peer overlay network schemes. *Communications Surveys & Tutorials, IEEE*, 7(2):72–93, Second Quarter 2005.

[50] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker. Search and replication in unstructured peer-to-peer networks. In *16th international conference on Supercomputing*, Nova Iorque, EUA, junho 2002.

[51] R. Mahajan, M. Castro, and A. Rowstron. Controlling the cost of reliability in peer-to-peer overlays. In *2nd International Workshop on Peer-to-Peer Systems*, Berkeley, EUA, fevereiro 2003.

[52] P Maniatis, T. Giuli, M. Roussopoulos, D. Rosenthal, and M. Baker. Impeding attrition attacks on p2p systems. In *11th ACM SIGOPS European Workshop*, setembro 2004.

[53] S. Marti and H. Garcia-Molina. Taxonomy of trust: Categorizing p2p reputation systems. *Computer Networks*, 50(4):472–484, March 2006.

[54] P. Maymounkov and D. Mazieres. Kademlia: A peer-to-peer information system based on the xor metric. In *1st International Workshop on Peer-to-Peer Systems*, Cambridge, EUA, março 2002.

[55] P. Mockapetris. DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION. RFC 1035, Internet Engineering Task Force, novembro 1987.

[56] MP3. http://www.mpeg.org/, acessado em junho de 2007.

[57] Napster. http://www.napster.com/, acessado em junho de 2007.

[58] Open Source Initiative. http://www.opensource.org/, acessado em junho de 2007.

[59] OuriGrid. http://www.ourgrid.org/, acessado em junho de 2007.

[60] Zero Paid. Music industry uses vulnerability in kazaa hash calculations. http://www.zeropaid.com/news/articles/auto/08262003a.php, acessado em junho de 2007.

[61] PDF. http://www.adobe.com/devnet/pdf/pdf_reference.html, acessado em junho de 2007.

[62] PostScript. http://www.postscript.org/, acessado em junho de 2007.

[63] J. Pouwelse, P. Garbacki, D. Epema, and H. Sips. The BitTorrent P2P File-sharing System: Measurements and Analysisng. In *International Workshop on Peer-to-Peer Systems (IPTPS)*, Ithaca, EUA, fevereiro 2005.

[64] RAR. http://www.rarlab.com/, acessado em junho de 2007.

[65] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In *ACM SIGCOMM*, San Diego, EUA, agosto 2001.

[66] V. Reich and D. Rosenthal. Lockss (lots of copies keep stuff safe). In *Preservation 2000*, York, Inglaterra, dezembro 2000.

[67] P. Resnick, K. Kuwabara, R. Zeckhauser, and E. Friedman. Reputation systems. *Communications of the ACM*, 43(12):45–48, dezembro 2000.

[68] Gnutella RFC. http://rfc-gnutella.sourceforge.net/, acessado em junho de 2007.

[69] R. Rivest. The MD5 Message-Digest algorithm. RFC 1321, Internet Engineering Task Force, abril 1992.

[70] B. Rocha, V. Almeida, and D. Guedes. Estratégias para Aumento de Confiabilidade em Redes de Roteamento Sobrepostas com Nós Egoístas. In *24th Simpósio Brasileiro de Redes de Computadores (SBRC)*, Curitiba, Brasil, maio 2006.

[71] B. Rocha, V. Almeida, and D. Guedes. Increasing the Quality of Service in Selfish Overlay Networks. *IEEE Internet Computing*, maio 2006.

[72] S. Saroiu, P. Gummadi, and S. Gribble. A Measurement Study of Peer-to-Peer File Sharing Systems. In *Multimedia Computing and Networking*, San Jose, EUA, janeiro 2002.

[73] Flora Rheta Schreiber. *Sybil*. Regnery, 1973.

[74] J. Silva, M. Barcellos, M. Konrath, L. Gaspary, and R. Antunes. Métodos para contenção de poluição de conteúdo em redes p2p. In *25th Simpósio Brasileiro de Redes de Computadores (SBRC)*, Belém, Brasil, junho 2007.

[75] Skype. http://www.skype.com/, acessado em junho de 2007.

[76] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. Frans Kaashoek, F. Dabek, and H. Balakrishnan. Chord: a scalable peer-to-peer lookup protocol for internet applications. *IEEE/ACM Transactions Network*, 11(1):17–32, fevereiro 2003.

[77] Y. Tang, H. Wang, and W. Dou. Trust based incentive in p2p network. In *E-Commerce Technology for Dynamic E-Business, IEEE International Conference on*, Pequim, China, setembro 2004.

[78] R.W. Thommes and M.J. Coates. Epidemiological Modelling of Peer-to-Peer Viruses and Pollution. In *do IEEE Infocom*, Barcelona, Espanha, abril 2006.

[79] Ogg Vorbis. http://www.vorbis.com/, acessado em junho de 2007.

[80] K. Walsh and E. G. Sirer. Fighting peer-to-peer SPAM and decoys with object reputation. In *ACM SIGCOMM workshop on Economics of Peer-to-Peer Systems*, Philadelphia, EUA, agosto 2005.

[81] K. Walsh and E. G. Sirer. Thwarting P2P pollution using object reputation. TR 2005-1980, Cornell University, fevereiro 2005.

[82] K. Walsh and E. G. Sirer. Experience with a distributed object reputation system for peer-to-peer filesharing. In *USENIX 3rd Symposium on Networked Systems Design & Implementation (NSDI)*, San Jose, EUA, março 2006.

[83] the free encyclopedia Wikipedia. UUHash. http://en.wikipedia.org/wiki/UUHash, acessado em junho de 2007.

[84] Lime Wire. http://www.limewire.com/, acessado em junho de 2007.

[85] Sopcast: Deliver your Media to the World. http://www.sopcast.com/, acessado em junho de 2007.

[86] B. Zhao, L. Huang, J. Stribling, S. Rhea, A. Joseph, and J. Kubiatowicz. Tapestry: A global-scale overlay for rapid service deployment. *IEEE Journal on Selected Areas in Communications*, 22(1), janeiro 2004.

[87] G. Zipf. *Human Behavior and the Principle of Least-Effort*. Addison-Wesley, Cambridge, MA, 1949.

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo