Universidade Federal de Campina Grande
Centro de Engenharia Elétrica e Informática
Coordenação de Pós-Graduação em Informática

Dissertação de Mestrado

O Impacto de Calotes e Múltiplas Personalidades no BitTorrent

Felipe Barros Pontes

Campina Grande, Paraíba, Brasil

Março - 2008

Universidade Federal de Campina Grande
Centro de Engenharia Elétrica e Informática
Coordenação de Pós-Graduação em Informática

O Impacto de Calotes e Múltiplas Personalidades no BitTorrent

Felipe Barros Pontes

Dissertação submetida à Coordenação do Curso de Pós-Graduação em Informática da Universidade Federal de Campina Grande - Campus I como parte dos requisitos necessários para obtenção do grau de Mestre em Ciência da Computação.

Área de Concentração: Ciência da Computação
Linha de Pesquisa: Redes de Computadores e Sistemas Distribuídos

Francisco Vilar Brasileiro
(Orientador)

Campina Grande, Paraíba, Brasil

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA CENTRAL DA UFCG

P814i

2008 Pontes, Felipe Barros

O impacto de calotes e múltiplas personalidades no bittorrent / Felipe Barros Pontes – Campina Grande: 2008.

56f.: il.

Dissertação (Mestrado em Ciência da Computação) - Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática.

Referências.

Orientador: Dr. Francisco Vilar Brasileiro

1. Ataques no Bittorrent. 2. Sybil. I-Título

CDU 004.491 (043)

Resumo

A geração de identidades e a associação destas às entidades de um enxame (*swarm*) BitTorrent, um dos sistemas de distribuição de conteúdo mais populares do momento, é feita normalmente de forma autônoma. Além disso, o mecanismo de incentivo do BitTorrent usa uma escolha aleatória na descoberta de novos parceiros. Essas duas características tornam o sistema vulnerável a um ataque *sybil* realizado por um caloteiro, no qual uma entidade associa múltiplas identidades a ela mesma na tentativa de enganar os outros nós e incrementar sua utilidade. Neste trabalho, foi avaliado o impacto que um ataque desse tipo pode ter sobre o mecanismo de incentivo do BitTorrent. Definiu-se um modelo matemático e suas estimativas foram utilizadas para ajudar a parametrizar simulações. A análise matemática utiliza duas métricas para avaliar tal impacto. A primeira é o número de identidades que um caloteiro precisa criar para que o seu tempo de download (utilidade) seja menor ou igual ao tempo de download dos nós que cooperam com o sistema. A segunda métrica avalia quão mais rápido um caloteiro faz download de um arquivo, em comparação com um nó que coopera com o sistema, à medida que o número de identidades do caloteiro cresce. Os resultados mostram que o número de identidades necessárias para ter sucesso em um ataque é relativamente pequeno. Além disso, a análise experimental observou a dinâmica das interações entre os nós para estudar o comportamento do ataque em enxames típicos de uma comunidade que compartilha arquivos de livre distribuição. Os resultados mostram que, em geral, o ataque é efetivo na maioria dos cenários estudados. Em outros poucos cenários, depende do momento na vida do enxame e do número de identidades.

Abstract

BitTorrent, one of the most popular content distribution protocols nowadays, has an identification generation scheme that is completely autonomous. Furthermore, BitTorrent uses a random mechanism to discover new peers. This leaves the system vulnerable to a sybil attack, by which an entity associates multiple identifications to itself in an attempt to fool the other peers that execute the agreed protocol and increase its utility. In this work we evaluate the impact of such an attack. We present an analytical model and validate it through simulations. Our initial analysis uses two metrics. The first one is the number of different identifications that an attacker must have in order to experience download times (utility) equal or smaller to that experienced by the other peers that collaborate resources to the system. The other metric assesses how faster an attacker downloads a file, compared to collaborators, as we increase the number of identifications that the attacker has. Our results show that the number of different identities required is relatively small. In addition, our analysis observed the dynamic of the interactions among the peers aiming at studying the behavior of attacks in typical swarms of a community that shares files for free distribution. Our results show that, in general, the attack is effective when considering the majority of the studied scenarios. In other few scenarios, it depends on the moment of the swarm's life and on the number of identities.

Agradecimentos

Agradeço primeiramente a quem, confiando na minha capacidade, me deu muito apoio nessa empreitada: minha família. Toda ela foi importante para que eu tivesse a serenidade necessária para realizar esse trabalho. Agradeço em especial a meus pais, Ismá e Gilvanize, e aos meus irmãos, Monize e Danilo. Sem vocês eu não estaria aqui.

Outra pessoa que foi e sempre será importante para mim e que me ajudou me dando forças e coragem foi minha namorada Monike. Seus conselhos, seus carinhos, seu afeto, sua compreensão e seu companheirismo foram de suma importância para que eu conseguisse atingir meu objetivo.

Tenho muitos amigos e seria inviável agradecer a todos eles individualmente. Por isso, para representá-los escolhi os que estão mais perto de mim. Agradeço muito a Mario e Leandro por terem compartilhado a maior parte de seus tempos nesses dois anos comigo. Descobrimos que a convivência não é uma tarefa fácil, mas tenho certeza que sempre seremos amigos. Marcinho (plim-plim), Edu, Guto, Willy, Marcus e Manfredi, apesar de morando em outras cidades, também foram importantes para que eu pudesse desenvolver meu trabalho.

Também fiz novos amigos aqui em Campina Grande. Apesar de nos separarmos agora, espero reencontrar Saulo, Celso, Rostand, Sidney (Zé colmeia) e Degas em outras ocasiões.

Agradeço também à colônia de alagoanos da UFCG, principalmente os do laboratório de sistemas embarcados (embedded).

Encerro meus agradecimentos agradecendo a todos quem fazem parte da família LSD. Encontrei aqui um ambiente de trabalho incrível e que sentirei falta pelo resto da vida. Todos foram importantes de alguma maneira. Walfredo, apesar do pouco tempo de convívio, foi quem abriu as portas para mim no LSD. Fubica, pela descontração, pelos puxões de orelha necessários e pela ótima orientação dada. Nazareno, pelas dicas e conselhos de um quase orientador. Jaindson por sempre estar disposto a ajudar. E Flávio (barata) pela sua descontração diária.

Este trabalho foi desenvolvido em colaboração com a HP Brasil P&D.

A todos vocês, meu muito obrigado.

Conteúdo

Lista de Acrônimos

GPS - *General Peer-to-Peer Simulator*

HTTP - *HyperText Transfer Protocol*

IP - *Internet Protocol*

P2P - *Peer-to-Peer*

TCP - *Transmission Control Protocol*

TOR - *The Onion Router*

NAT - *Network Address Translation*

MD5 - *Message-Digest algorithm 5*

HTML - *HyperText Markup Language*

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Com o crescimento e aperfeiçoamento da Internet, observou-se a possibilidade de se compartilhar recursos entre usuários da rede com interesses afins. Aproveitando essa oportunidade, vários sistemas foram desenvolvidos com base na arquitetura entre-pares ou *peer-to-peer* (*p2p*). Nela, usuários compartilham seus recursos diretamente com seus pares sem a necessidade de entidades centralizadas. Essas entidades são utilizadas somente para auxiliar em atividades secundárias, como a descoberta dos nós. Entretanto, tais sistemas devem possuir mecanismos de incentivo à colaboração, como forma de evitar que uma porção significativa dos usuários sejam "caronas" (*free-riders*) ou "caloteiros"[1], e utilizem os recursos providos no sistema sem colaborar com a comunidade [1].

Em sistemas *p2p* a associação de identidades a entidades do sistema é normalmente feita de forma autônoma pela própria entidade. Dessa forma, se por um lado elimina-se a necessidade de um componente central que assegure identificações únicas para entidades, por outro lado a geração de novas identidades para uma entidade pode ser feita com um custo muito baixo, o que torna o sistema vulnerável a um ataque *sybil*[2] [9]. Em um ataque *sybil* uma entidade maliciosa cria identidades suficientes para constituir uma grande fração da comunidade e enganar o sistema. De modo geral, sistemas que possuem um controle fraco do esquema de criação e associação de identidades são suscetíveis a ataques *sybil*.

[1]Neste trabalho será dada preferência ao termo caloteiro, uma vez que o comportamento que será avaliado é executado por um usuário que tem intenção explícita de burlar o protocolo; esse não é necessariamente o comportamento de um "carona".

[2]Esse termo é uma referência à personagem do livro homônimo, o qual relata o estudo de um distúrbio psicológico de uma paciente que apresentava múltiplas personalidades [25].

Ataques *sybil* podem se materializar de diferentes formas. Neste trabalho, aborda-se o ataque em sistemas *p2p* de compartilhamento de recursos, nos quais um atacante tenta contornar os mecanismos de incentivo à colaboração e conseguir tanta utilidade quanto possível, sem contribuir com quaisquer recursos para o sistema, ou seja, dando um "calote" nos seus pares. Em particular, alguns sistemas realizam sorteios para decidir entre entidades que estejam em igualdade de condições quem terá preferência na utilização de recursos. Isso pode potencializar os efeitos do ataque porque, em geral, quanto mais identidades um atacante possuir maior será a probabilidade dele ser escolhido.

O BitTorrent [8] é um dos sistemas de distribuição de conteúdo mais populares do momento. De acordo com a BigChampagne[3] existiam aproximadamente 10 milhões de usuários simultâneos em agosto de 2005. Além disso, de acordo com um estudo feito pela CacheLogic[4] o tráfego BitTorrent em junho de 2004 representava 53% de todo o tráfego *p2p* na Internet.

A geração de identidades no BitTorrent é feita normalmente de forma autônoma pelos próprios nós. Além disso, o mecanismo de incentivo do BitTorrent usa uma escolha aleatória na descoberta de novos parceiros, como será detalhada no Capítulo 3. Essas duas características tornam o sistema vulnerável a um ataque *sybil* realizado por um caloteiro. Uma variação deste tipo de ataque nesse sistema já foi realizada eficientemente por um canal de televisão a cabo. Nele, vários nós foram utilizados para distribuir conteúdo corrompido dos episódios de uma série transmitida pelo canal [10].

O objetivo deste trabalho é avaliar o impacto de ataques *sybil* no BitTorrent, nos quais o atacante está interessado em burlar o sistema para conseguir uma melhor utilidade. Cenários nos quais o atacante está interessado em impedir o compartilhamento dos arquivos não fazem parte do escopo deste trabalho. Espera-se uma avaliação mais concreta do impacto desses ataques nesses sistemas, dando subsídios para o desenvolvimento de alternativas que visem minimizar seus efeitos. Para isso, ataques *sybil* foram estudados pontualmente e dinamicamente no BitTorrent. Definiu-se um modelo matemático que pode ser usado para analisar pontualmente o impacto que um ataque desse tipo pode ter sobre o mecanismo de incentivo do BitTorrent. Em particular, pretende-se avaliar qual é o número mínimo de identidades

[3] http://www.bigchampagne.com

[4] http://www.cachelogic.com

que precisam ser geradas por um nó caloteiro de forma que o tempo de *download* dele seja menor ou igual ao tempo de *download* dos outros nós que cooperam com o sistema. Visto que um caloteiro não emprega seus recursos para cooperar com o sistema, considera-se que isso já é suficiente para que o ataque tenha sido eficaz. Simulações foram realizadas para avaliar dinamicamente o impacto do ataque e verificar quão próximo o modelo analítico está da realidade. Além disso, foi feita uma classificação dos enxames[5] mais comuns encontrados em comunidades de compartilhamento de arquivos e quais são os melhores momentos para atacá-los.

Os resultados demonstram a vulnerabilidade do BitTorrent a ataques *sybil*. Na maioria dos cenários analisados poucas identidades são necessárias para o atacante ter um tempo de *download* inferior ao tempo de um colaborador. Entretanto, cenários nos quais o atacante não obtém sucesso mesmo aumentando a quantidade de identidades geradas foram identificados.

O restante do trabalho está organizado da seguinte forma. O Capítulo 2 define e exemplifica ataques *sybil*, apresentando as principais dificuldades encontradas para proteger os sistemas desse tipo de ataque. No Capítulo 3, apresenta-se o funcionamento básico do BitTorrent, abordando seu mecanismo de incentivo à colaboração e como um atacante poderia explorar características do mecanismo para não precisar contribuir com o sistema, além de discutir os principais trabalhos relacionados a esta pesquisa. No Capítulo 4 é apresentado um modelo matemático para representar o ataque de um nó caloteiro com múltiplas identidades sobre um enxame do BitTorrent e possibilitar a estimativa de quantas identidades são necessárias para ele conseguir uma utilidade maior que aquela de um colaborador. Diferentemente, o Capítulo 5 apresenta resultados da avaliação da dinâmica do ataque durante um intervalo de tempo. Finalmente, o Capítulo 6 conclui o trabalho apresentando as considerações finais e uma indicação dos trabalhos futuros a serem realizados.

[5]Conjunto de nós participando da distribuição de um arquivo.

Capítulo 2

Ataques *Sybil*

Este capítulo visa definir e exemplificar ataques *sybil*, bem como descrever as dificuldades encontradas para defender os sistemas *p2p* desses ataques. Na Seção 2.1 é apresentada a definição de ataques por múltiplas identidades. Na Seção 2.2 exemplos de sistemas suscetíveis a este tipo de ataque são descritos. Finalizando o capítulo, a Seção 2.3 descreve como um ataque *sybil* pode ser minimizado ou até evitado.

2.1 Definindo Ataques *Sybil*

Em sistemas *p2p*, identidades podem ser utilizadas como uma abstração para facilitar a identificação de uma entidade. Muitos sistemas assumem que cada entidade participante possui apenas uma identidade. Porém, isso nem sempre acontece. Quando o custo de criar uma identidade é baixo e não existe uma infra-estrutura de segurança que assegure a unicidade da relação entre entidades e identidades, usuários mal intencionados podem executar ataques *sybil*. Nesse tipo de ataque, um usuário cria múltiplas identidades para burlar o funcionamento dos sistemas.

Ataques *sybil* podem estar presentes em diferentes tipos de sistemas. A próxima seção descreve alguns exemplos de cenários onde o ataque pode ser efetivo.

2.2 Cenários Propícios a Ataques *Sybil*

O exemplo mais conhecido do ataque *sybil* é a distribuição indiscriminada de mensagens eletrônicas (*spam*), um dos principais problemas da comunicação eletrônica atualmente. Nessa modalidade do ataque, um atacante cria várias contas de correio eletrônico, na maioria das vezes inativas, para enviar mensagens indesejadas, dificultando a ação de filtros que usem o endereço do remetente como parâmetro de filtragem. Atacantes freqüentemente utilizam programas que facilitam ou automatizam a obtenção de endereços e o envio a um grande número de destinatários. Existem diversos métodos para um atacante obter uma lista de endereços. Um dos procedimentos mais comuns é utilizar programas de interpretação de textos que executam varreduras em ambientes com um número potencialmente grande de endereços disponíveis, como páginas da Internet. A principal motivação para a prática do *spamming* é o baixo custo associado à obtenção de uma identidade (endereço eletrônico) [30].

Um dos cenários mais propícios à realização de ataques por múltiplas identidades é o de redes de sensores. Newsome *et al.* [19] descreveram as possíveis maneiras de um atacante atacar uma rede de sensores. Sensores são cada vez mais utilizados por proporcionarem a realização de monitoramento contínuo e em tempo real de sistemas, sendo utilizados até em sistemas que oferecem risco à vida, como no acompanhamento da resistência estrutural de pontes, viadutos e equipamentos industriais [29]. Geralmente estão distribuídos em uma determinada região e precisam se comunicar, senão entre eles, com algum ponto de coleta que deve utilizar as informações para tomar alguma decisão. Como os sensores possuem restrições de consumo de energia e de processamento, a utilização de infra-estruturas de segurança quase nunca é considerada. Assim, a ausência de mecanismos de segurança nessas redes em conjunto com a necessidade de difundir informações em um ambiente de comunicação aberto, faz com que um atacante *sybil* se passando por vários nós acabe com a divisão eqüitativa dos tempos de acesso ao meio e consiga um maior tempo de utilização do canal para facilmente realizar um ataque nesse tipo de sistema, o que pode ocasionar desde pequenos inconvenientes até graves acidentes.

Em redes de roteamento um ataque também pode ser prejudicial. Um atacante pode criar várias identidades se fazendo passar por vários nós legítimos para criar falsos caminhos de roteamento e evitar que novas rotas sejam criadas para um nó ser alcançado. Isso pode evitar

que nós participem do roteamento e consigam obter crédito por ter feito o roteamento em sistemas guiados por mecanismos de reputação ou até evitar que pacotes sejam entregues a um determinado nó.

Sistemas de armazenamento distribuído podem replicar dados em diferentes nós para evitar suas perdas ou fragmentá-los para preservar o sistema contra a perda de privacidade. Quando um nó seleciona um conjunto de outros nós para realizar a replicação ou fragmentação dos dados, um nó malicioso pode ser escolhido e prejudicar o armazenamento. Caso um atacante gere identidades suficientes ele pode até eliminar a informação que estaria replicada ou evitar que ela seja reconstituída a partir dos fragmentos que estariam armazenados nos nós que ele controle.

Em um sistema de votação *online*, no qual o custo de geração de identidades seja baixo, um usuário malicioso com mais de uma identificação pode votar mais de uma vez e comprometer o resultado final da votação. O sistema de classificação de páginas *Web* do sistema de busca da *google* classifica as páginas com base nas quantidades de *hyperlinks* (ou votos) que ela possui de outras páginas [20]. Nesse caso, um atacante pode criar várias páginas e utilizá-las para votar em uma página específica, melhorando a colocação dela nos resultados das pesquisas realizadas.

Por fim, ataques *sybil* podem ser realizados em sistemas de compartilhamento de recursos para contornar mecanismos de incentivo à colaboração e utilizar os recursos sem contribuir com o sistema ou até evitar que eles sejam compartilhados. A rede de favores (ou NoF, do inglês *Network of Favors*) é um esquema autônomo de reputação para ajudar nós com recursos ociosos a determinar para qual nó requisitante doar recursos. A idéia central da NoF é que os usuários que são os maiores cooperadores possuem as mais altas pontuações e devem ter maior prioridade para consumir recursos da comunidade [3]. Dessa maneira, usuários caronas são identificados e marginalizados eficientemente. Entretanto, quando dois ou mais usuários possuem mesmas pontuações um sorteio deve ser realizado para decidir quem irá receber o recurso. Um ataque *sybil* pode ser realizado na NoF com o intuito de aumentar a probabilidade de, em igualdade de condições com outros nós requisitantes, um atacante ser escolhido para receber um recurso. Já o BitTorrent, sistema-alvo deste estudo, proporciona o compartilhamento eficiente de arquivos distribuindo o custo de *upload* entre os nós interessados no arquivo. O funcionamento do BitTorrent e como um atacante pode gerar

múltiplas identidades e fazê-las compartilhar o mesmo recurso físico (rede e processamento) e o mesmo documento sendo baixado para conseguir uma banda de *download* maior que a de um colaborador, mesmo sem fazer *upload*, são descritos no Capítulo 3.

Acredita-se que, de maneira geral, sistemas que utilizam sorteios e possuem baixo custo na associação de identidades são vulneráveis a ataques *sybil*. Resultados semelhantes aos encontrados neste trabalho foram concluídos em um estudo sobre ataques *sybil* na rede de favores realizado em parceria com Miranda Mowbray (pesquisadora do *HP Labs*). Mais detalhes sobre esse estudo são descritos no Apêndice A.

2.3 Evitando Ataques

Dada a diversidade de cenários que um ataque pode ser realizado, atualmente, não existe uma solução que seja universalmente aplicável [16]. Douceur [9] mostrou que, na prática, é impossível evitar ataques *sybil* sem que haja a presença de uma entidade central que certifique confiavelmente as identidades geradas no sistema. A utilização de uma entidade desse tipo pode gerar custos proibitivos ou criar gargalos no desempenho dos sistemas. Além disso, essas entidades devem garantir que identidades perdidas ou roubadas sejam descobertas e anuladas. Isso também acrescenta complexidade ao desenvolvimento do sistema. Entretanto, é uma abordagem aconselhada para sistemas com requisitos estritos de segurança.

Como nem todo sistema pode assumir os custos da implementação de uma infra-estrutura de segurança, outros mecanismos são utilizados para minimizar os efeitos de um ataque *sybil*. Teste de recursos é a alternativa mais utilizada para reduzir a eficiência de um ataque [16]. Essa abordagem baseia-se no princípio de que uma entidade maliciosa não conseguirá desempenhar o trabalho que o número de entidades pelas quais ela está se fazendo passar conseguiria. Esses testes incluem checagem por capacidade computacional (fatoração de números grandes, por exemplo), capacidade de armazenamento, largura de banda etc. Aspnes *et al.* [5] apresentam uma solução baseada em teste de recursos para evitar que um atacante *sybil* consiga enganar um mecanismo de consenso na presença de falhas bizantinas [14]. Tal solução requer que os nós validem suas identidades através de um processo de três rodadas. Na primeira, cada nó envia um pedaço de um desafio para todos os outros. Na segunda, cada nó determina o desafio a partir dos pedaços recebidos, computa tantas

soluções quanto possível e envia o desafio juntamente com as soluções de volta para todos os nós no sistema. Na última rodada, cada nó verifica as soluções recebidas e consegue determinar o número correto de identidades que um nó tem. Entretanto, Douceur também mostrou a ineficácia de testes de recursos. Sistemas distribuídos de larga escala podem ser bastante heterogêneos, o que ocasiona disparidades entre os recursos no sistema e dificuldade em comparar os recursos dos nós.

Yu *et al.* [32] apresentaram um protocolo para limitar os efeitos de um ataque *sybil* baseado em redes sociais. Nesse protocolo, uma ligação entre máquinas indica um relacionamento confiável estabelecido entre humanos. Dessa maneira, usuários maliciosos podem criar quantas identidades desejarem, mas a criação de relacionamentos confiáveis é dificultada.

Enquanto que a maioria das soluções tenta evitar ou dificultar o ataque, em outro estudo realizado por Margolin e Levine [18] o objetivo é detectá-lo. Nele, é proposto um mecanismo econômico de detecção de ataques *sybil*, no qual um atacante recebe uma gratificação para se revelar. Porém, estratégias baseadas em economia tendem a inserir complexidade na implementação dos sistemas. Deve existir uma entidade para confirmar o saldo das entidades, caso contrário uma entidade pode anunciar um saldo falso e enganar o sistema, por exemplo.

Outra idéia é a de evitar que um mesmo IP estabeleça mais de uma conexão com diferentes identidades. Isso não se configura como sendo uma boa abordagem porque dois nós podem estar em uma mesma rede e estarem utilizando um esquema de tradução de endereços baseado em NAT ou um mesmo servidor *proxy*. Assim, apenas um nó poderá utilizar o sistema em questão.

Outra maneira de minimizar os efeitos de um ataque *sybil* é banir os nós que não estão cooperando com o sistema. Porém, nem sempre é possível identificar quando um nó é um carona ou quando ele está com problemas de conectividade.

Esses problemas encontrados para evitar um ataque *sybil* motivaram este estudo no sistema de compartilhamento de arquivos BitTorrent. Uma descrição do funcionamento do sistema e como um caloteiro pode obter vantagens realizando um ataque desse tipo ao BitTorrent são apresentados no Capítulo 3.

Capítulo 3

O BitTorrent

Neste capítulo são apresentados os principais conceitos relacionados ao BitTorrent, é indicado como um atacante com múltiplas identidades pode atacar um conjunto de nós compartilhando um arquivo (enxame) e são descritos trabalhos relacionados ao desta dissertação. Na Seção 3.1 é abordado como o conteúdo é distribuído, como funciona o mecanismo que incentiva os nós a colaborarem com o sistema e como é o processo de identificação dos nós. A Seção 3.2 conclui o capítulo apresentando os principais trabalhos relacionados a este.

3.1 O Protocolo do BitTorrent

Quando um arquivo é disponibilizado através do protocolo HTTP, todo o custo de distribuição desse arquivo é assumido pela máquina hospedeira. O protocolo BitTorrent permite que grandes volumes de conteúdo digital possam ser distribuídos de forma eficiente usando um sistema *p2p*. Quando vários nós estão baixando o mesmo arquivo ao mesmo tempo, eles enviam partes do arquivo uns para os outros. Isso faz com que o custo da distribuição do arquivo seja compartilhado entre os nós envolvidos na sua distribuição.

Para manter a informação de quais pedaços do arquivo os nós possuem, um arquivo é dividido em partes de $256KB$. Isso possibilita a melhor difusão do arquivo porque faz com que nós mantenham conexões com nós que possuem pelo menos uma parte que lhes seja de interesse. Entretanto, a unidade mínima de transferência no BitTorrent é uma subparte. Cada parte possui tipicamente 16 subpartes. Isso facilita a implementação da técnica de *pipelining*, na qual várias requisições são mantidas por vez para evitar o atraso entre partes, o que pode

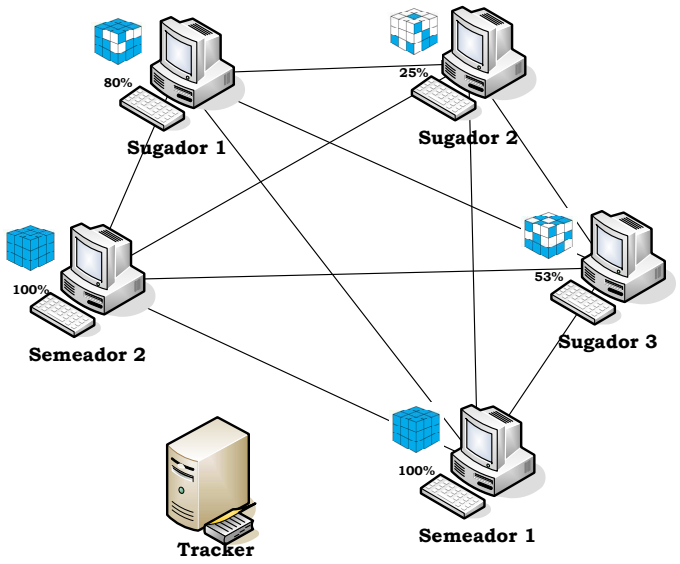


Figura 3.1: Conexões entre os nós.

reduzir as taxas de transferência. Tipicamente 5 requisições são mantidas ao mesmo tempo.

No BitTorrent, um nó que possui uma cópia completa do arquivo e deseja compartilhar partes dele é denominado de semeador (*seeder*). Já um nó que ainda não possui uma cópia completa do arquivo e deseja fazer o *download* dele é denominado de sugador (*leecher*). Importante notar que mesmo sem possuir a cópia completa do arquivo um sugador também pode fazer *upload* de partes do arquivo para outros nós. O conjunto desses nós participando da distribuição do conteúdo de um arquivo é denominado de enxame (*swarm*) e está ilustrado na Figura 3.1. Na figura também se pode notar a presença de um nó chamado *tracker*. *Trackers* são nós específicos que implementam um serviço de *rendez-vous*, permitindo que um nó que se junte ao enxame possa encontrar outros nós naquele enxame[1].

3.1.1 Distribuição do Conteúdo

Um arquivo de metadados (identificado pela terminação *.torrent*) com informações necessárias à distribuição do arquivo deve ser criado e disponibilizado para acesso público, geralmente em sítios *Web*. Os principais componentes do arquivo de metadados são:

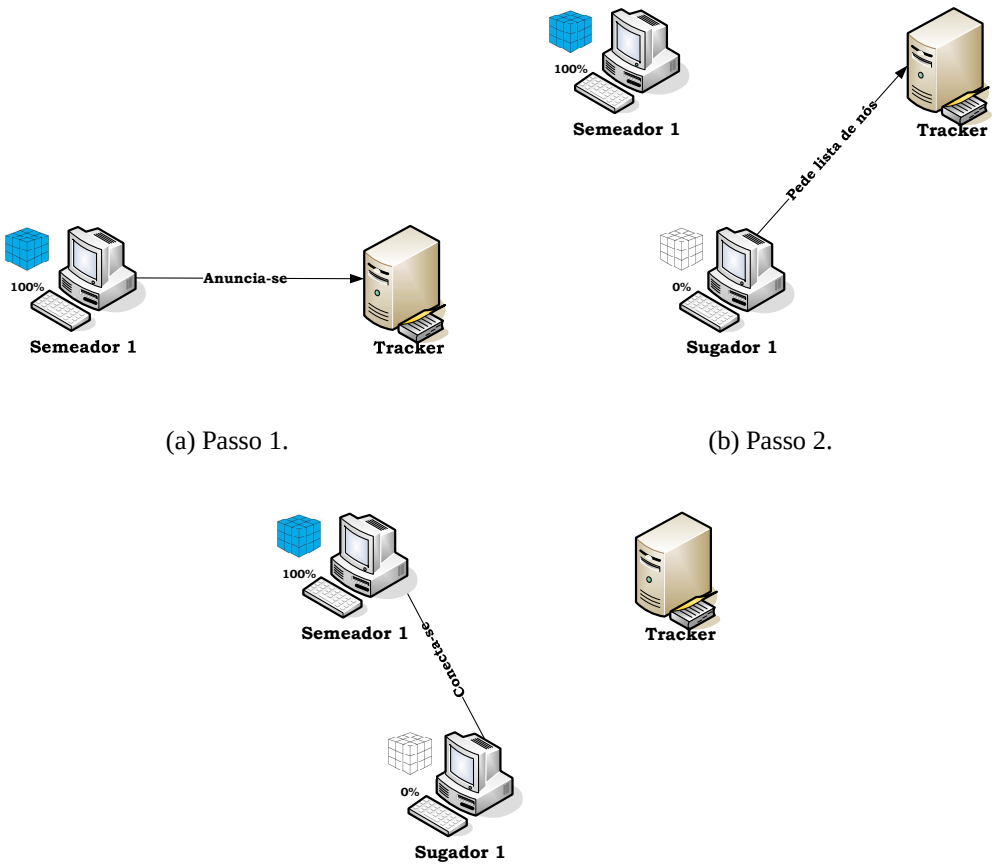
- nome, tamanho (em *bytes*) e uma soma MD5 do arquivo a ser compartilhado;

[1]Para tornar o texto mais fluido, sem perda de generalidade, será assumido a partir desse ponto que existe um único *tracker* por enxame.

- o endereço do *tracker*;
- a data de criação do arquivo de metadados;
- comentários textuais do criador do arquivo de metadados;
- nome e versão do programa utilizado para criar o arquivo de metadados;
- *string* consistindo da concatenação dos valores *hash* de cada parte do arquivo;
- tamanho (em *bytes*), geralmente em potência de 2, de cada parte do arquivo - tipicamente escolhido com base no tamanho do arquivo e restringido pelo fato de que partes muito grandes podem causar ineficiência nas taxas de *download* e muito pequenas podem resultar em grandes arquivos *.torrent*. Cada parte tem mesmo tamanho, exceto a última que pode ter tamanho menor que as outras.

Os passos para um arquivo ser distribuído estão ilustrados na Figura 3.2. No primeiro passo, um semeador após divulgar o arquivo de metadados se anuncia ao tracker para que sugadores possam se conectar diretamente a ele e começar a baixar partes do arquivo. No segundo passo, um sugador deverá obter o arquivo *.torrent* e se conectar ao *tracker* para obter uma lista de nós (geralmente contendo 50 nós) que possuem partes daquele arquivo. A lista é composta por identidades dos nós seguidas de IP e porta que serão usados para o estabelecimento da conexão e os nós devem contactar o *tracker* de tempos em tempos tentando descobrir outros nós. No terceiro passo, o nó sugador poderá se conectar diretamente aos nós da lista obtida do *tracker* utilizando o IP e a porta retornados na lista e iniciar o compartilhamento. Dessa maneira, toda a logística necessária para o compartilhamento dos arquivos é realizada a partir de interações entre os nós. Informações sobre *upload* e *download* são enviadas para o *tracker* mas são apenas para a coleta de estatísticas. As responsabilidades do *tracker* são estritamente limitadas a ajudar os nós a se conhecerem [8]. Desse modo, apesar do *tracker* ser uma entidade centralizada na arquitetura do BitTorrent, ele fica na periferia da rede *p2p* e não participa efetivamente do compartilhamento de arquivos.

No momento da conexão, os nós trocam mapas contendo informações sobre a disponibilidade das partes dos arquivos. Com isso, um nó sabe quais partes os nós com quem está se conectando possuem. Caso as partes de pelo menos um nó interessem ao outro, a conexão



(a) Passo 1.

(b) Passo 2.

(c) Passo 3.

Figura 3.2: Passos para o início da troca de arquivos.

é mantida e o nó é dito estar interessado no outro. Além disso, quando um nó termina de baixar uma parte ele informa a todos os nós com quem está conectado que a partir daquele momento possui aquela parte. Isso é importante para a implementação da política de escolha de partes predominantemente utilizada no BitTorrent. Quando um nó recebe uma parte ele usa informações dos mapas de partes de seus vizinhos para decidir qual a próxima parte a ser solicitada. Tal política é chamada de "parte mais rara primeiro" (*rarest-first*) e consiste em escolher a parte mais rara entre seus vizinhos para aumentar a probabilidade dele passar a ter uma parte que interesse a outros nós. De acordo com Legout *et al.* o emprego desta técnica evita problemas quando uma ou mais partes tornam-se pouco comuns em um enxame e quando um nó precisa de apenas uma parte para terminar de baixar o arquivo [15]. Uma

exceção para a utilização da política de escolha de parte a ser requisitada acontece quando um nó está iniciando o *download* e ainda não possui nenhuma parte completa. Nesse caso, ele escolhe aleatoriamente uma parte para completá-la o mais rápido possível e poder fazer *upload* para outros nós.

Nós sugadores, ao concluírem o *download* do arquivo, podem permanecer no enxame por algum tempo, atuando como novos nós semeadores. Entretanto, não há incentivos a esse tipo de comportamento. Um semeador que permanece no sistema faz isso por motivos meramente altruísticos. Por outro lado, o BitTorrent possui um mecanismo para incentivar sugadores a colaborarem com o sistema.

3.1.2 Mecanismo de Incentivo

Apesar do protocolo BitTorrent não evitar que nós caronas possam utilizar o sistema, e em alguns casos isso faz com que eles consigam uma boa taxa de *download* (ver Capítulo 5), existe um mecanismo para incentivar um nó a colaborar com o enxame.

Cada nó tem um número máximo de conexões permitidas (tipicamente 55), das quais uma parte é usada para que o nó tente se conectar com seus pares, enquanto que outra parte é usada para permitir que outros nós se conectem a ele. Se por um lado um nó pode, potencialmente, receber dados (i.e. fazer *download*) de todas as suas conexões ativas, por outro lado ele utiliza apenas um pequeno número dessas conexões, chamadas de não sufocadas (*unchoked*), para fazer *upload* para os seus pares (tipicamente 5). Nesse contexto, o mecanismo de incentivo é implementado através da forma como um sugador escolhe entre as suas conexões ativas, aquelas que não serão sufocadas, ou seja, aquelas nas quais ele fará *upload*. Essa escolha é baseada em uma política *tit-for-tat*, baseada em retaliações, na qual o nó prioriza a realização de *upload* para os nós que lhe ofereceram, em um momento próximo, uma melhor taxa de *download*. Dessa forma, os nós com maiores taxas de *upload* (maiores contribuintes) irão ter maiores taxas de *download*. Portanto, colaborar com o sistema é normalmente a estratégia mais interessante para um nó.

Como um sugador não possui uma visão completa do sistema (enxames podem ser formados por milhares de nós), o mecanismo de incentivo requer que o nó tente, de tempos em tempos (tipicamente a cada 30 segundos), descobrir se existem conexões que poderiam lhe ofertar melhores taxas de *download* do que aquelas que ele não está sufocando durante

um certo intervalo de tempo. Para tal, o sugador sufoca uma parcela das conexões não sufocadas que estão lhe dedicando as menores taxas de *upload* e escolhe aleatoriamente o mesmo número de conexões sufocadas (*choked*) que passarão a não ser mais sufocadas. Esse mecanismo é chamado de "dessufocamento otimista" (*optimistic unchoking*) e, tipicamente, uma única conexão por vez é envolvida nessa operação.

No caso dos nós semeadores, o mecanismo de "dessufocamento otimista" é um pouco diferente. Um nó conectado a um semeador não faz *upload* de conteúdo para este. Desse modo, ao invés de considerar a taxa de *upload* recebida em suas conexões, um nó semeador considera a taxa de *download* que os outros nós estão conseguindo dele, priorizando aqueles com maiores taxas. Essa estratégia faz com que semeadores difundam o conteúdo de forma mais eficiente.

3.1.3 Identificação

Antes de se juntar a um enxame, cada nó é responsável por gerar aleatoriamente sua própria identidade, uma *string* de 20 *bytes*. O esquema de geração de identidades independente e de baixo custo, combinado com a característica aleatória do algoritmo de dessufocamento otimista provê um cenário ideal para um nó caloteiro realizar um ataque de múltiplas identidades em enxames BitTorrent. Um caloteiro pode inundar a lista de nós distribuída pelo *tracker* com suas identidades, aumentando a probabilidade de ser contactado por outros nós que consultem aquele *tracker*. Isso também aumenta a probabilidade de uma das identidades do nó caloteiro ser escolhida aleatoriamente através do algoritmo de "dessufocamento otimista". Além disso, nós não podem ser diferenciados através de seus endereços IP, pois eles podem utilizar o mesmo IP quando estão em uma NAT ou quando utilizam o mesmo servidor *proxy*.

3.2 Avaliações de Ataques no BitTorrent

Nesta seção, este trabalho é situado em relação a outros trabalhos que estudaram ataques *sybil* no BitTorrent. No contexto de modelagem matemática do ataque no BitTorrent, até onde se sabe, este trabalho apresenta a primeira contribuição. Na avaliação do impacto, existe a abordagem de realizar o ataque com o objetivo de inviabilizar o compartilhamento

de arquivos e a de burlar o sistema para conseguir vantagens como a de ter um tempo de *download* menor que o de um colaborador.

3.2.1 Avaliações Matemáticas

Alguns trabalhos foram realizados com o intuito de modelar matematicamente sistemas BitTorrent [4], [11] e [22]. No entanto, diferentemente deste trabalho (e até onde se sabe), nenhum deles visou representar um ataque *sybil*. Dessa maneira, o modelo apresentado no Capítulo 4 representa uma das contribuições deste trabalho.

De maneira geral, foram propostos modelos para investigar de que maneira a participação de nós semeadores e sugadores no sistema interfere no desempenho do mesmo. Guo *et al.* [11], com base em um modelo de fluido [22], propuseram alterações para melhorar o desempenho de sistemas BitTorrent através da participação simultânea dos nós em múltiplos enxames. Já no trabalho de Andrade *et al.* [4], também com base no modelo de fluido, foi analisado como a inserção de uma entidade servidora pode ser útil para melhorar a qualidade de serviço na distribuição do conteúdo em sistemas BitTorrent.

3.2.2 Avaliações da Dinâmica do Ataque

As avaliações da dinâmica de ataques *sybil* no BitTorrent podem ser divididas em duas partes. Em uma delas o atacante está interessado em inviabilizar o compartilhamento dos arquivos, ou seja, realizar um ataque de negação de serviço. Na outra, como neste trabalho, a intenção é a de reduzir o tempo de *download* do atacante.

Recentemente, Konrath *et al.* [13] analisaram duas estratégias para tentar destruir um enxame BitTorrent através de um ataque *sybil*. A primeira delas consiste em utilizar um grande número de identidades para burlar o protocolo de escolha de parte a ser requisitada. Mentindo ao anunciar quais partes possui, um atacante pode artificialmente tornar uma parte mais rara, fazendo com que as partes que são realmente raras tendam a desaparecer do enxame e impedindo que os outros nós completem o *download* com sucesso. Com isso, o atacante aumenta o nível de replicação de uma parte e acaba induzindo os outros nós a fazer *download* de outras partes primeiro. Assim, as partes mentidas pelo atacante poderiam desaparecer do sistema. A segunda estratégia utilizada no trabalho consiste em inundar o enxame

com personalidades de caloteiros para que os nós se conectem predominantemente com eles e não consigam baixar partes do arquivo. Embora sejam necessárias várias identidades para que isso seja possível, os autores argumentam que como o atacante não faz *download* do arquivo o tráfego gerado por ele está restrito apenas ao plano de controle.

De acordo com o protocolo BitTorrent, na mensagem de anúncio ao *tracker* um nó pode informar um IP e porta a serem utilizados. Com base nisso, Sia descreveu em seu trabalho [26] uma maneira de inundar as conexões de um nó de modo que ele não consiga continuar participando do enxame. A idéia daquele trabalho é realizar um ataque de negação de serviço distribuído onde os atacantes seriam os outros nós participantes do enxame. Para conseguir isso, um atacante poderia fazer vários anúncios com identidades diferentes, mas com mesmo IP e porta da vítima. Assim, o *tracker* iria difundir essa informação e os outros nós do enxame iriam tentar se conectar com a vítima. Este caso pode ser interpretado como um ataque *sybil* às avessas, pois a vítima é quem possui múltiplas identidades. Importante notar que se um *sybil* gerar identidades de maneira indiscriminada ele pode acabar ficando sobrecarregado e não conseguir continuar baixando o arquivo.

Diferente dos estudos supracitados, este trabalho investiga o impacto de um ataque *sybil* em sistemas BitTorrent quando o atacante não está interessado em inviabilizar o compartilhamento de arquivos, mas sim em baixar o arquivo em um tempo menor que aquele de um nó colaborador.

O trabalho mais semelhante a este é o de Sun [28]. Nele, foram realizados experimentos em redes BitTorrent com o intuito de analisar se um atacante *sybil* que colabora com o sistema consegue diminuir seu tempo de *download*. Para isso, Sun avaliou o ataque de acordo com a estratégia de divisão das partes do arquivo entre as identidades do atacante. A cada identidade gerada um novo processo era criado, de modo a permitir que os processos pudessem separadamente baixar conjuntos diferentes de partes. Os resultados mostram que em determinados cenários um atacante consegue um tempo de *download* menor que aquele de um sugador. Entretanto, como os processos, se executados em uma mesma máquina, compartilham a mesma largura de banda, quanto mais processos forem executados menor será a largura de banda de *upload* disponível para cada um deles. Com isso, eles quase nunca estarão entre os nós que fazem mais *upload* e quase sempre serão sufocados. Logo, realizar um ataque *sybil* quando o atacante deseja colaborar com o sistema não é a estratégia

mais adequada para um nó diminuir seu tempo de *download*. Diferentemente daquele estudo, este trabalho apresenta uma avaliação do ataque quando um nó não deseja contribuir com o sistema.

Sirivianos *et al.* propuseram mecanismos para aumentar o número de conexões de um nó carona e aumentar a probabilidade dele ser escolhido pelos outros nós para receber subpartes do arquivo, melhorando sua taxa média de *download* [27]. Segundo os autores, um nó carona conseguiria isso eliminando a restrição no número máximo de conexões e requisitando mais freqüentemente a lista de nós ao *tracker*. Os autores descrevem a realização de experimentos que comprovam suas suspeitas. Entretanto, a metodologia utilizada pode ter influenciado os resultados. Nós caronas e sugadores entram no enxame ao mesmo tempo. Assim, fica difícil afirmar se um nó carona teve uma taxa de *download* maior do que a do sugador por conta do aumento no número de conexões ou da influência disso nas conexões estabelecidas pelo sugador. Além disso, a abordagem daquele trabalho evita que um nó carona estabeleça mais de uma conexão com um mesmo nó. Com um ataque *sybil*, duas identidades distintas podem fazer *download* de um mesmo nó. Isso pode aumentar a largura de banda conseguida pelo atacante porque um mesmo nó pode destinar mais de uma conexão de *upload* para o atacante.

Como pôde ser visto neste capítulo, ataques *sybil* representam uma ameaça ao BitTorrent. Entretanto, nos trabalhos pesquisados, o ataque foi analisado sem ter uma estimativa de quantas identidades utilizar ou em que momento do ciclo de vida de um enxame o ataque deveria ser executado para um atacante obter melhores resultados. Neste trabalho, o número de identidades é estimado com base no modelo matemático descrito no Capítulo 4. Já os momentos na vida de um enxame mais propícios ao ataque são investigados no Capítulo 5 através de simulações parametrizadas com ajuda do modelo.

Capítulo 4

Avaliando Pontualmente Ataques *Sybil* no BitTorrent

Neste capítulo é apresentado um modelo matemático que indica quantas identidades são necessárias para, de acordo com o estado do enxame fornecido, um caloteiro obter uma taxa de *download* maior que aquela de um sugador. Na Seção 4.1 são descritas as equações que compõem o modelo e como elas foram produzidas. Na Seção 4.2 são apresentados resultados da análise do modelo com base na viabilidade do ataque e no benefício que um caloteiro pode conseguir. Tais resultados dão continuidade ao estudo realizado em outro trabalho [21].

4.1 Modelo Analítico

O modelo apresentado abaixo considera um instante de tempo particular na vida de um enxame quando existem L nós sugadores, S nós semeadores e um nó caloteiro que realiza um ataque *sybil* com Σ personalidades (cada uma representada por uma identidade diferente) fazendo parte do enxame. O modelo permite calcular a taxa média de *download* agregada de um nó, o que representa sua utilidade. Esta por sua vez é dada pela soma da taxa de *upload* que cada um dos nós com os quais ele está conectado lhe dedica. Nossa modelagem do ataque *sybil* parte do princípio de que um ataque é vantajoso quando a taxa de *download* agregada de um atacante é maior ou igual que aquela de um sugador médio. Caso isso aconteça, é mais vantajoso não contribuir com o sistema.

A taxa de *upload* que um nó n_i dedica a outro nó n_j com o qual está conectado, em

um determinado instante de tempo, depende da taxa máxima de *upload* de n_i, do papel que n_i está desempenhando (sugador, semeador ou caloteiro), do número de outros nós com os quais n_i está conectado, dos papéis que esses outros nós estão desempenhando, e, dependendo do papel de n_i, da taxa de *upload* ou de *download* de seus pares. Sejam c_a, c_u e c_o, respectivamente, o número de conexões ativas que o nó tem, o número de conexões não sufocadas e o número de conexões sujeitas ao mecanismo de *optimistic unchoking*. No que segue, nós fazemos as seguintes hipóteses simplificadoras:

i os semeadores, os sugadores e o caloteiro têm sempre o mesmo valor para c_a, c_u e c_o;

ii $c_a \geq c_u$;

iii todos os nós semeadores têm os mesmos recursos de processamento e de rede e a taxa máxima de *upload* de um semeador (*s*) é U_s^{max};

iv todos os nós sugadores e o nó caloteiro têm os mesmos recursos de processamento e de rede e a taxa máxima de *upload* de um sugador (*l*) é U_l^{max}.

O número de semeadores, sugadores e caloteiros conectados a um nó define uma *configuração* para o nó. Dada uma configuração qualquer $\mathcal{C}_n$, sejam $\mathcal{C}_n^l$, $\mathcal{C}_n^s$ e $\mathcal{C}_n^\sigma$, respectivamente, o número de sugadores, semeadores e personalidades do caloteiro conectados ao nó n na configuração $\mathcal{C}_n$. De maneira geral, $0 \leq \mathcal{C}_n^l \leq c_a$, $0 \leq \mathcal{C}_n^s \leq c_a$, $0 \leq \mathcal{C}_n^\sigma \leq c_a$ e $\mathcal{C}_n^l + \mathcal{C}_n^s + \mathcal{C}_n^\sigma = c_a$. É assumido que uma personalidade do nó caloteiro consegue identificar as outras personalidades do nó caloteiro e, dessa forma, nós caloteiros não se conectam uns com os outros, ou seja, quando n é uma personalidade do caloteiro, $\mathcal{C}_n^\sigma = 0$. De forma análoga, os semeadores também conseguem identificar outros semeadores (eles têm todas as peças do arquivo e essa informação é trocada quando dois nós tentam se conectar); assim, quando n é um semeador, $\mathcal{C}_n^s = 0$.

A taxa de *upload* que um nó sugador n_l dedica a cada um dos nós com os quais está conectado vai depender da configuração de n_l, ou seja, do número de semeadores, sugadores e personalidades do nó caloteiro com os quais esteja conectado. Por exemplo, se n_l está conectado a um outro nó sugador $n_{l'}$ e todos os outros pares de n_l são semeadores (como ilustrado na Figura 4.1), então toda a banda de *upload* de n_l é dedicada a $n_{l'}$. Em geral, como

será apresentado a seguir, há uma taxa média de *upload* particular a ser dedicada a cada nó conectado a n_l para cada uma das *configurações* de papéis possíveis para as conexões de n_l.

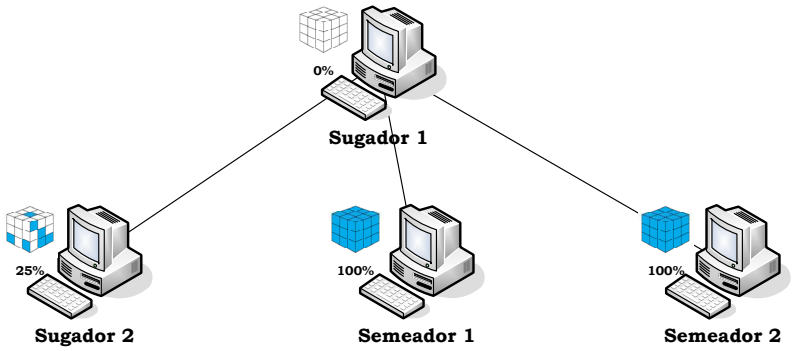


Figura 4.1: Exemplo de configuração de conexões para um nó.

Primeiro é definido como se calcula a fatia da banda de *upload* $U_{l \leftarrow l, \mathcal{C}_{n_l}}$ que um nó sugador n_l dedica ao conjunto dos outros nós sugadores em uma configuração $\mathcal{C}_{n_l}$. Obviamente, se $\mathcal{C}^l_{n_l} = 0$, trivialmente $U_{l \leftarrow l, \mathcal{C}_{n_l}} = 0$. A seguir consideramos as configurações nas quais n_l está conectado a pelo menos um nó sugador, ou seja, $1 \leq \mathcal{C}^l_{n_l} \leq c_a$. Dado que os nós sugadores são idênticos, qualquer nó sugador conectado a n_l vai receber deste a mesma taxa média de *download*. Além disso, essas taxas são recebidas a partir das conexões não sufocadas de n_l. Essas, por sua vez são disputadas apenas por sugadores e personalidades do caloteiro. Quando o número de sugadores da configuração é maior ou igual ao número de conexões não sufocadas de n_l ($\mathcal{C}^l_{n_l} \geq c_u$), os $\mathcal{C}^l_{n_l}$ nós sugadores dividem equanimemente todas essas conexões. Porém, devido ao mecanismo de *optimistic unchoking*, algumas dessas conexões (c_o) são divididas também com as personalidades do caloteiro. Desse modo, a fatia de banda de *upload* que n_l dedica a todos os nós sugadores na configuração $\mathcal{C}_{n_l}$ é dada pela equação 4.1.

$$U_{l \leftarrow l, \mathcal{C}_{n_l}} = \left(c_u - c_o + \frac{\mathcal{C}^l_{n_l}}{\mathcal{C}^l_{n_l} + \mathcal{C}^\sigma_{n_l}} \cdot c_o \right) \cdot \left(\frac{U^{max}_l}{c_u} \right) \tag{4.1}$$

Quando o número de sugadores é menor que o número de conexões não sufocadas ($\mathcal{C}^l_{n_l} < c_u$), então cada sugador receberá a banda que n_l dedica a cada uma de suas conexões não sufocadas. Nesse caso, a fatia de banda de *upload* que n_l dedica a todos os nós sugadores na configuração $\mathcal{C}_{n_l}$ é dada pela equação 4.2.

$$U_{l\leftarrow l,\mathcal{C}_{n_l}} = \mathcal{C}^l_{n_l} \cdot \frac{U^{max}_l}{\min(c_u, \mathcal{C}^l_{n_l} + \mathcal{C}^\sigma_{n_l})} \tag{4.2}$$

Por outro lado, a fatia da banda de *upload* obtida pelas personalidades do nó caloteiro conectado ao nó sugador n_l em uma configuração $\mathcal{C}_{n_l}$, $U_{\sigma\leftarrow l,\mathcal{C}_{n_l}}$, é dada pela diferença entre a taxa máxima de *upload* de n_l e a fatia da banda de *upload* que é fornecida para os $\mathcal{C}^l_{n_l}$ sugadores conectados a n_l na configuração $\mathcal{C}_{n_l}$, como indicado na equação 4.3.

$$U_{\sigma\leftarrow l,\mathcal{C}_{n_l}} = U^{max}_l - U_{l\leftarrow l,\mathcal{C}_{n_l}} \tag{4.3}$$

Note que a taxa individual de *upload* obtida por um nó sugador e por uma personalidade do caloteiro em duas configurações distintas que tenham o mesmo número de semeadores, sugadores e personalidades do caloteiro é a mesma. Dessa forma, sejam $\mathcal{C}_{n_l}[1], \mathcal{C}_{n_l}[2], ..., \mathcal{C}_{n_l}[k]$ as k configurações possíveis e distintas para um nó sugador n_l do enxame, tal que para quaisquer duas configurações $\mathcal{C}_{n_l}[i]$, $\mathcal{C}_{n_l}[j]$, $i \neq j$, $\mathcal{C}^l_{n_l}[i] \neq \mathcal{C}^l_{n_l}[j] \vee \mathcal{C}^s_{n_l}[i] \neq \mathcal{C}^s_{n_l}[j] \vee \mathcal{C}^\sigma_{n_l}[i] \neq \mathcal{C}^\sigma_{n_l}[j]$. Se $p(\mathcal{C}_{n_l}[i])$ é a probabilidade da configuração $\mathcal{C}_{n_l}[i]$ acontecer, então, as fatias médias de banda de *upload* que um nó n_l dedica ao conjunto de sugadores e ao conjunto de personalidades do caloteiro, são dadas, respectivamente, pelas equações 4.4 e 4.5.

$$U_{l\leftarrow l} = \sum_{i=1}^{k} p(\mathcal{C}_{n_l}[i]) \cdot U_{l\leftarrow l,\mathcal{C}_{n_l}[i]} \tag{4.4}$$

$$U_{\sigma\leftarrow l} = \sum_{i=1}^{k} p(\mathcal{C}_{n_l}[i]) \cdot U_{\sigma\leftarrow l,\mathcal{C}_{n_l}[i]} \tag{4.5}$$

Como definido anteriormente, $p(\mathcal{C}_{n_l}[i])$ é a probabilidade da configuração $\mathcal{C}_{n_l}[i]$ ocorrer. As conexões que um nó estabelece dependem das identidades que ele obtém do *tracker*. Este, por sua vez, repassa identidades fazendo uma escolha aleatória entre as identidades que ele conhece. Dessa forma, para as configurações $\mathcal{C}_{n_l}[i]$ que podem ocorrer para um nó sugador n_l, $p(\mathcal{C}_{n_l}[i])$ é dada por:

$$p(\mathcal{C}_{n_l}[i]) = \frac{\binom{L-1}{\mathcal{C}^l_{n_l}[i]} \cdot \binom{S}{\mathcal{C}^s_{n_l}[i]} \cdot \binom{\Sigma}{\mathcal{C}^\sigma_{n_l}[i]}}{\binom{L-1+S+\Sigma}{c_a}} \tag{4.6}$$

Outra hipótese simplificadora que é feita neste modelo é a de que a probabilidade de um sugador não ter partes que interessem a outros nós é desconsiderada. Se isso acontece na prática, não haverá o compartilhamento naquela conexão.

Agora define-se como se calcula a fatia de *upload* $U_{l \leftarrow s, \mathcal{C}_{n_s}}$ que um nó semeador n_s dedica a um nó sugador em uma configuração $\mathcal{C}_{n_s}$. O mecanismo de incentivo implementado pelo semeador leva em consideração as taxas de *download* dos nós, priorizando aqueles com taxas mais elevadas. Para cada personalidade utilizada por um caloteiro, novas conexões serão criadas, o que reduz a taxa de *download* em cada uma delas. Dessa maneira, as personalidades do caloteiro serão preteridas pelos semeadores sempre que houver sugadores suficientes para ocupar todas as conexões não sufocadas do semeador. Nesse caso, uma personalidade do caloteiro só recebe *upload* de um semeador quando ela é escolhida pelo algoritmo de *optimistic unchoking*. Desse modo, a fatia de *upload* recebida de um nó semeador n_s pelo conjunto de nós sugadores na configuração $\mathcal{C}_{n_s}$ é dada pela equação 4.7.

$$U_{l \leftarrow s, \mathcal{C}_{n_s}} = \left(c_u - c_o + \frac{\mathcal{C}^l_{n_s}}{\mathcal{C}^l_{n_s} + \mathcal{C}^\sigma_{n_s}} \cdot c_o \right) \cdot \left(\frac{U^{max}_s}{c_u} \right) \tag{4.7}$$

Por outro lado, a fatia de *upload* obtida pelo conjunto de personalidades do nó caloteiro conectado ao nó semeador n_s em uma configuração $\mathcal{C}_{n_s}$, $U_{\sigma \leftarrow s, \mathcal{C}_{n_s}}$, é dada pela diferença entre a taxa máxima de *upload* de n_s e a fatia de *upload* que é fornecida para os $\mathcal{C}^l_{n_s}$ sugadores conectados a n_s na configuração $\mathcal{C}_{n_s}$, como explicitado abaixo:

$$U_{\sigma \leftarrow s, \mathcal{C}_{n_s}} = U^{max}_s - U_{l \leftarrow s, \mathcal{C}_{n_s}} \tag{4.8}$$

Novamente, a fatia de *upload* obtida de um nó semeador n_s pelo conjunto de nós sugadores e pelo conjunto de personalidades do caloteiro em duas configurações distintas que tenham o mesmo número de sugadores e personalidades do caloteiro é a mesma. Dessa forma, sejam $\mathcal{C}_{n_s}[1], \mathcal{C}_{n_s}[2], ..., \mathcal{C}_{n_s}[k']$ as k' configurações possíveis e distintas para um nó semeador n_s do enxame, tal que para quaisquer duas configurações $\mathcal{C}_{n_s}[i]$, $\mathcal{C}_{n_s}[j]$, $i \neq j$, $\mathcal{C}^l_{n_s}[i] \neq \mathcal{C}^l_{n_s}[j] \vee \mathcal{C}^\sigma_{n_s}[i] \neq \mathcal{C}^\sigma_{n_s}[j] \wedge \mathcal{C}^s_{n_s}[i] = \mathcal{C}^s_{n_s}[j] = 0$. Se $p(\mathcal{C}_{n_s}[i])$ é a probabilidade da configuração $\mathcal{C}_{n_s}[i]$ acontecer, então, as fatias médias da banda de *upload* que o conjunto de nós sugadores e o conjunto de personalidades do nó caloteiro conseguem de n_s, são dadas, respectivamente, pelas equações 4.9 e 4.10.

$$U_{l \leftarrow s} = \sum_{i=1}^{k'} p(\mathcal{C}_{n_s}[i]) \cdot U_{l \leftarrow s, \mathcal{C}_{n_s}[i]} \tag{4.9}$$

$$U_{\sigma \leftarrow s} = \sum_{i=1}^{k'} p(\mathcal{C}_{n_s}[i]) \cdot U_{\sigma \leftarrow s, \mathcal{C}_{n_s}[i]} \tag{4.10}$$

Por outro lado, a probabilidade de uma configuração $\mathcal{C}_{n_s}[i]$ ocorrer é dada por:

$$p(\mathcal{C}_{n_s}[i]) = \frac{\binom{L}{\mathcal{C}^l_{n_s}[i]} \cdot \binom{\Sigma}{\mathcal{C}^\sigma_{n_s}[i]}}{\binom{L+\Sigma}{c_a}} \tag{4.11}$$

Finalmente, para calcular D_σ, a taxa de *download* agregada média do caloteiro, basta somar a taxa de *download* agregada que ele consegue em todas as configurações possíveis e distintas recuperadas por ele a partir de solicitações ao *tracker* (k''), ponderando essas taxas pela probabilidade da configuração ocorrer. Ou seja, D_σ é dada pela equação 4.12.

$$D_\sigma = \sum_{i=1}^{k''} \frac{\binom{L}{\mathcal{C}^l_{n_\sigma}[i]} \cdot \binom{S}{\mathcal{C}^s_{n_\sigma}[i]}}{\binom{L+S}{c_a}} \cdot \left(U_{\sigma \leftarrow l} \cdot \mathcal{C}^l_{n_\sigma}[i] + U_{\sigma \leftarrow s} \cdot \mathcal{C}^s_{n_\sigma}[i] \right) \tag{4.12}$$

A taxa de *download* agregada média obtida por um nó sugador qualquer é dada pela diferença entre a banda total de *upload* disponível no sistema e a taxa de *download* agregada média do caloteiro, como mostra a equação 4.13.

$$D_l = \frac{U_l^{max} * L + U_s^{max} * S - D_\sigma}{L} \tag{4.13}$$

Porém, um nó não consegue utilizar uma banda maior que sua capacidade real. Desta maneira, a largura de banda nominal de *download* de um nó é o mínimo entre D_l ou D_σ e a capacidade real de largura de banda para *download* de sugadores e caloteiros. Logo, a banda calculada aqui para um nó caloteiro estará limitada pela sua largura de banda.

4.2 Análise

Nesta seção apresenta-se uma análise de resultados provenientes do modelo introduzido na Seção 4.1. Aqui, analisa-se o impacto de um ataque *sybil* em sistemas BitTorrent, investigando a sua viabilidade e o benefício que um caloteiro pode conseguir através desse ataque.

A viabilidade é expressa em termos do número de identidades relativo ao tamanho do enxame necessário para que um caloteiro consiga um tempo de *download* igual ou menor que o de um sugador. Já o benefício é uma medida do quão melhor o tempo de *download* de um caloteiro pode ser em relação ao de um sugador.

Esta análise assume sempre os mesmos valores para c_a, c_u e c_o, quais sejam: $c_a = 55$, $c_u = 5$ e $c_o = 1$. Esses são valores comumente utilizados por implementações de clientes BitTorrent. Também é assumido para todos os resultados apresentados que $U_l^{max} = U_s^{max} = 17KB/s$ e a largura de banda de *download* dos nós é de $75KB/s$. Esses valores são derivados de observações de enxames feitas por Bellissimo *et al.* [6]. Outra consideração feita é que o tamanho do arquivo não influencia em como as interações acontecem. Em determinadas situações, porém, esse pode ser um fator importante para o sucesso do ataque. No Capítulo 5 fatos que justificam essa afirmação são apresentados.

Um ataque é dito bem sucedido se a banda de *download* obtida por um caloteiro for maior ou igual àquela obtida por um sugador médio ($D_\sigma \geq D_l$). A Figura 4.2 mostra o comportamento da quantidade de identidades necessárias para que um ataque tenha sucesso em função da relação entre o número de sugadores e de semeadores presentes em um enxame com 1.000 nós. Cada valor no eixo das ordenadas do gráfico representa o número de identidades calculado pelo modelo para que o ataque seja bem sucedido. Inicialmente, o número de identidades necessárias cresce rapidamente, enquanto a quantidade de semeadores diminui, por conta do aumento no número de sugadores e conseqüente aumento da concorrência pelos recursos. Entretanto, com menos semeadores no sistema a banda que um sugador médio consegue diminui. Assim, o caloteiro consegue ganhar uma banda maior que a de um sugador de maneira mais fácil. Por isso, à medida que o número de semeadores diminui, o caloteiro precisa gerar cada vez menos identidades de maneira que a banda do sugador decai até praticamente não se modificar. Quando isso acontece, a quantidade de identidades se estabiliza e a parcela de contribuição de sugadores se aproximará da parcela de contribuição de semeadores, fazendo com que, do ponto de vista das personalidades dos caloteiros, praticamente não se consiga fazer uma distinção entre eles (veja as equações 4.1 e 4.7). De qualquer maneira, o número de identidades necessárias para que a utilidade do caloteiro seja igual ou maior que a de nós colaboradores em um enxame contendo 1.000 nós é relativamente pequeno. Aproximadamente 150 identidades no máximo.

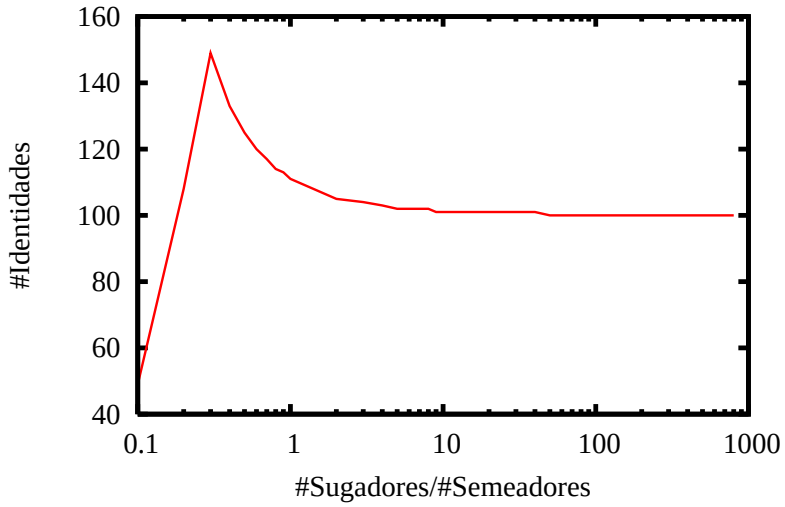


Figura 4.2: Comportamento do número de identidades necessárias em função da relação entre o número de sugadores e semeadores.

Agora, avalia-se o impacto do tamanho do enxame no número de identidades necessárias para que o ataque seja bem sucedido. A Figura 4.3 mostra o número de identidades requeridas quando aumenta-se a população de um enxame para diferentes relações entre o número de sugadores e de semeadores (L/S). O gráfico mostra que a quantidade de identidades aumenta linearmente com o tamanho do enxame, ou seja, quanto menor o enxame, mais barato é para um caloteiro atacá-lo. Isso pode ser explicado pelo fato de que aumentando-se a quantidade de nós em um enxame aumenta-se também a concorrência pelos recursos. No entanto, quanto maior a quantidade de semeadores em relação ao número de sugadores, pior para o caloteiro. Da mesma maneira da situação apresentada na Figura 4.2, quando o número de semeadores é grande, um sugador médio obtém mais facilmente uma banda de *download* alta. Assim, um caloteiro precisa gerar mais identidades para que sua banda seja no mínimo igual à dos sugadores. Mesmo assim, o número de identidades necessárias é relativamente pequeno com relação ao tamanho da população de um enxame (por volta de 15% do número de nós, no pior caso).

A Figura 4.4 apresenta quão melhor pode ser a utilidade do caloteiro em relação àquela de um sugador em um enxame com 1.000 nós para diferentes quantidades de semeadores e sugadores. Pode-se observar que o benefício de um caloteiro aumenta bastante rápido à medida que se aumenta o número de suas identidades até quase se estabilizar em um deter-

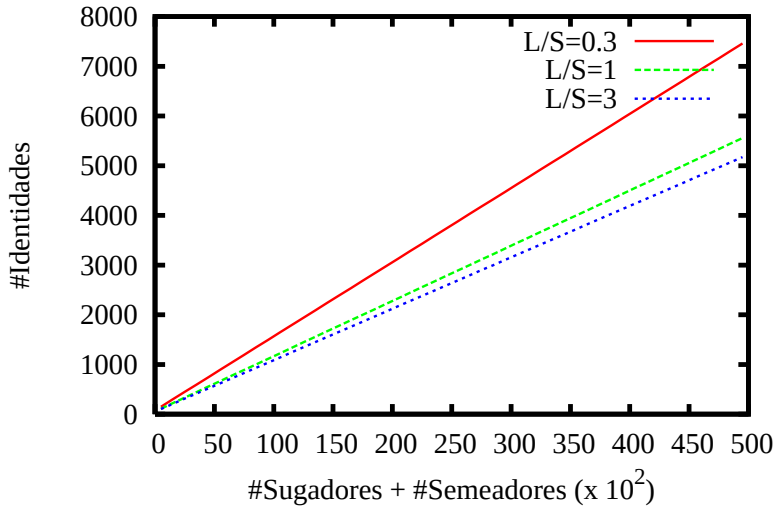


Figura 4.3: Comportamento do número de identidades necessárias em função da população de um enxame.

minado momento. Isso acontece porque a partir daí a banda conseguida por um caloteiro fica limitada pela sua largura de banda. A mesma explicação para os casos anteriores também se aplica para a análise do benefício. Aumentar o número de identidades faz com que a banda do caloteiro cresça mais lentamente em cenários com uma maior quantidade de semeadores.

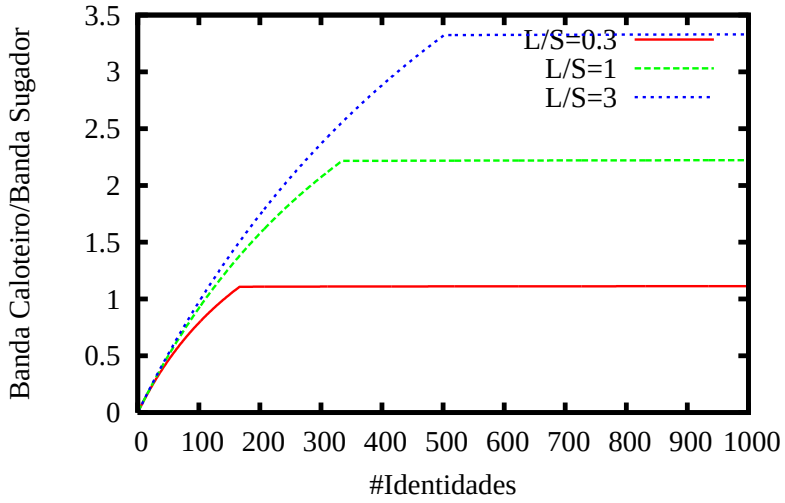


Figura 4.4: Benefício do ataque em função do número de identidades.

O modelo apresentado neste capítulo além de considerar algumas simplificações deixa de avaliar as implicações provenientes da interação entre os nós. Resultados mais realistas com

base em simulações de enxames típicos de uma comunidade formada pelos participantes de um fórum são descritas no Capítulo 5.

Capítulo 5

Simulando Calotes com Múltiplas Personalidades no BitTorrent

Diferentemente do Capítulo 4, onde os enxames são analisados em instantes particulares de tempo, apresenta-se aqui os resultados de simulações utilizadas para avaliar o impacto do ataque no BitTorrent durante um intervalo de tempo. A Seção 5.1 apresenta o simulador utilizado nas simulações. A Seção 5.2 descreve como foram encontrados e quais foram os parâmetros utilizados. Os cenários utilizados são descritos na Seção 5.3. Por fim, os resultados da avaliação e uma análise desses resultados são apresentados na Seção 5.4.

5.1 *General Peer-to-Peer Simulator* (GPS)

O *General Peer-to-Peer Simulator* (GPS) [31] é um *framework* de simulação dirigido a eventos - isso significa que o tempo de simulação avança de acordo com o processamento dos eventos de simulação e não de acordo com incrementos fixos de tempo - cujo propósito é permitir a simulação e modelagem de protocolos e aplicações *p2p* de forma eficiente e precisa. Entretanto, falhas foram encontradas na implementação do protocolo do BitTorrent disponibilizada juntamente com o *framework*. Pôde-se perceber isso mais nitidamente com a implementação de testes de unidade e de integração. O teste de unidade se preocupa em testar individualmente o funcionamento da menor unidade de um *software*. No caso do simulador, a menor unidade é uma classe Java. Já o teste de integração testa a interação entre as unidades do sistema. Após a realização de modificações pôde-se ter mais confiança nos

resultados apresentados pelas simulações. Além disso, foram implementadas modificações que permitiram simular ataques *sybil* no BitTorrent.

O GPS permite a simulação de enxames do protocolo BitTorrent com as características descritas no Capítulo 3. Para isso, utiliza alguns arquivos de configuração que possuem os parâmetros a serem utilizados nas simulações. Os mais importantes são:

- quantidade de nós;
- quantidade de caloteiros;
- largura de banda;
- tamanho do arquivo;
- tempos de entrada dos nós no enxame;
- taxas de saída de sugadores e de semeadores.

Para a análise dos resultados, o GPS fornece as seguintes métricas:

- quantidade de sugadores e de semeadores em um dado momento;
- taxa média de *download* de um nó quando ele termina de baixar o arquivo;
- tempo de *download* de um nó;
- taxa média de *download* para sugadores e caloteiros em um dado momento;
- taxa média de *download* para um sugador específico;
- quantidade média de conexões de cada nó.

A métrica que identifica as quantidades de sugadores e de semeadores foi importante para a identificação de pontos interessantes para o ataque. Já a métrica que permite avaliar se o ataque foi eficiente é a que representa as taxas médias de *download* dos sugadores e dos caloteiros. Caso a taxa média de *download* de um caloteiro seja maior ou igual àquela de um sugador, assume-se que o ataque foi eficaz, já que o atacante não colabora com o sistema.

Embora o simulador implemente o protocolo BitTorrent de maneira bem próxima da real, os nós podem se comunicar sem falhas. Apesar de ser possível um nó desistir do *download*,

ele avisa a todos os nós com quem está conectado que está desistindo do *download*. Isso é necessário para evitar referências inválidas nos nós e inviabilizar o restante da simulação. Acredita-se que isso não reduza a importância dos resultados encontrados.

5.2 *Trace*

Para que os resultados das simulações se aproximem da realidade, os parâmetros a serem utilizados nas simulações devem ser os mais realistas possíveis. Porém, escolher aleatoriamente valores para os parâmetros pode produzir situações improváveis de acontecer na prática. Assim, optou-se por parametrizar as simulações com base em informações reais retiradas de um *trace*, coletado por Bellissimo *et al.* [6], de uma comunidade que compartilha arquivos de mídia de livre distribuição produzidos pelos participantes do fórum online *Alluvion* (`http://alluvion.org/`). Tal *trace* foi disponibilizado[1] após a utilização de um *crawler* para coletar páginas HTML geradas pelo *tracker* da comunidade com informações que os nós enviam periodicamente sobre seus estados atuais. O *crawler* foi utilizado ao longo de aproximadamente 50 dias entre outubro e dezembro de 2003 e baixava, a cada hora, todas as páginas com informações que não possibilitam identificar os nós ou o conteúdo dos enxames, preservando a privacidade dos usuários.

Neste trabalho, considera-se que um nó pode sair do enxame. Isto pode acontecer na prática porque o nó pode estar com problemas de conectividade, não estar conseguindo uma boa taxa de *download*, caso seja um sugador, ou simplesmente por ter feito o *download* de todo o arquivo e não querer mais continuar no enxame, já que no BitTorrent não existe incentivo para um semeador permanecer no enxame. Considerando isso, os principais parâmetros utilizados nas simulações por enxame são a taxa de saída dos sugadores (θ), a taxa de saída dos semeadores (γ), o tamanho do arquivo sendo compartilhado no enxame e as larguras de banda dos nós.

Este trabalho utilizou a mesma metodologia e o mesmo ferramental utilizados por Andrade [2]. Nele, γ e θ são estimados como sendo os inversos dos tempos médios de permanência de semeadores e de sugadores nos enxames, respectivamente. Os tamanhos de arquivos foram obtidos diretamente da descrição dos enxames. As larguras de banda dos nós

[1] Disponível em http://traces.cs.umass.edu/.

foram estimadas de acordo com a quantidade de *bytes* baixados e enviados e informados ao *tracker* a cada interação. Por outro lado, a taxa com que os sugadores entram no sistema e um parâmetro de atenuação da popularidade do enxame, embora não utilizados diretamente nas simulações, são estimados com base em regressões lineares e foram importantes para o agrupamento dos enxames. Entretanto, alguns enxames não puderam ser utilizados por possuírem dados que invalidavam as regressões ou produziam estimativas irreais. Por conta disso, houve a necessidade de tratar e filtrar esses dados, executando as etapas a seguir:

- apenas os enxames que possuíam pelo menos dois nós com tempos distintos de entrada no enxame foram selecionados;
- foram selecionados os enxames que têm pelo menos um semeador e quatro sugadores;
- os enxames que têm menos de um dia de duração foram eliminados;
- quando foram encontrados nós com mesmos tempos de entrada no enxame, tais tempos foram alterados de modo a ficarem igualmente distribuídos entre o tempo inicial e o final do enxame.

Além da filtragem supracitada, só interessavam enxames que se iniciaram e terminaram dentro do período de observação. Mas se todos fossem considerados, a tendência é que enxames com tempos de duração menores fossem escolhidos, o que poderia influenciar nos resultados das simulações. Isso acontece porque supondo que dois enxames se iniciem no mesmo instante, a probabilidade do enxame maior estar totalmente contido no período de observação do *trace* é menor. Uma alternativa para minimizar isso é considerar um tempo qualquer (t) e escolher os enxames que nasceram antes de t, que terminaram entre o tempo inicial e o final de observação do *trace* e que possuem tempo de duração no máximo igual a t, como proposto em [24]. Assim, o número de enxames foi reduzido de 1.528 para 323.

Apesar da redução considerável no número de enxames, ainda ficaria inviável executar simulações com todos eles porque iria dificultar sobremaneira a análise dos resultados. Assim, era necessária a escolha de alguns enxames que retratassem bem toda a amostra. Para isso, os enxames similares foram agrupados e depois escolhidos os que representassem bem cada grupo.

5.2.1 Agrupamento dos Enxames

A técnica de Agrupamento Hierárquico Aglomerativo (do inglês *Agglomerative Hierarchical Clustering*) [12] foi utilizada para formar os grupos de enxames. Nessa técnica, os elementos vão sendo agrupados de modo a formarem uma árvore. No topo da árvore estão os elementos em grupos unitários e a cada passo do agrupamento os dois grupos que são mais similares são combinados em um novo grupo. O processo é repetido até que todos os elementos finalmente são combinados em um único grupo na raiz da árvore. Neste trabalho, a similaridade foi calculada com base na distância euclidiana entre os enxames. Quanto maior é o valor da distância, menos similares os objetos são.

Naturalmente, os elementos em um mesmo grupo são mais similares que os de outros grupos. Dessa maneira, à medida que os grupos vão sendo combinados, a heterogeneidade entre os elementos vai aumentando. Como não se deseja que todos os elementos sejam combinados em apenas um grupo, pode-se observar quão heterogêneos eles estão e quando um passo juntou elementos muito diferentes. A partir desse ponto não é interessante combinar mais os elementos. A árvore é então "cortada" e os grupos recuperados. Se for monitorada a heterogeneidade à medida que o número de grupos diminui, pode-se observar quando dois grupos muito diferentes foram unidos. A Figura 5.1 retrata a heterogeneidade derivada dos agrupamentos realizados com os enxames. Nela, pode-se observar que a heterogeneidade aumenta suavemente à medida que os grupos vão sendo unidos até um ponto no qual ela passa a crescer mais fortemente. Isso indica a quantidade de grupos que devem ser escolhidos para que os elementos não estejam tão heterogêneos.

No caso deste trabalho, 14 grupos foram recuperados. Destes, 6 grupos foram formados por apenas um enxame e 1 grupo foi formado por dois enxames. Assim, eles foram desconsiderados por não representarem bem a amostra. Outro grupo desconsiderado possuía como enxame mais representativo um bem semelhante aquele de outro grupo. Por fim, outro grupo também foi desconsiderado porque os nós do enxame mais representativo não permaneceram por tempo suficiente para produzir informações úteis nas simulações. Assim, restaram apenas 5 grupos que são detalhados na Subseção 5.2.2.

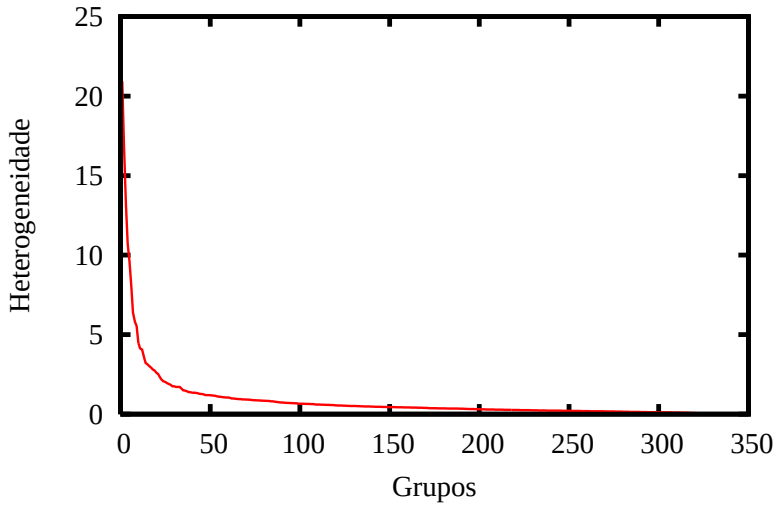


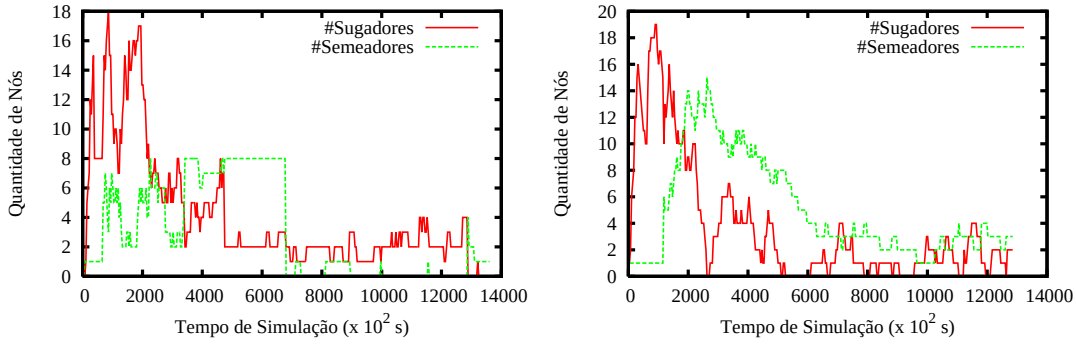
Figura 5.1: Heterogeneidade a cada passo do processo de agrupamento.

5.2.2 Caracterização dos Enxames

Após o agrupamento era necessária a escolha de um enxame que representasse bem cada grupo. Uma maneira de escolher tal enxame é verificar qual o mais similar aos valores médios dos parâmetros do grupo. Dessa maneira, os enxames que tiveram a menor distância euclidiana do vetor que representa o ponto médio de todas as observações no grupo foram escolhidos como os mais representativos daquele grupo.

Alguns parâmetros tiveram de ser ajustados, pois a estimativa deles não possibilitou a execução de simulações com os mesmos. Por exemplo: a taxa de saída dos semeadores era maior em alguns enxames do que o necessário para que o semeador inicial permanecesse no enxame até que um sugador qualquer terminasse de baixar o arquivo. Nesses casos, as execuções não trariam informações úteis à análise porque as taxas médias de *download* não poderiam ser calculadas e a taxa de saída dos semeadores teve de ser ajustada. Contudo, mesmo com essas alterações os resultados das simulações ficaram semelhantes aos resultados do *trace* em termos do comportamento das quantidades de nós durante o tempo. A Figura 5.2 apresenta dois gráficos da quantidade de nós em função do tempo de simulação obtidos a partir de informações do *trace* e de simulações.

As informações mais importantes dos enxames mais representativos, bem como a quantidade de enxames em cada grupo, podem ser visualizados na Tabela 5.1. Nela, estão presentes as taxas de saída para sugadores e semeadores, os tamanhos dos arquivos e as larguras de



(a) A partir de informações do *trace*.

(b) A partir de informações de simulações.

Figura 5.2: Comportamento típico do número de nós em função do tempo em um enxame.

banda de *download* e *upload* para cada enxame.

Com relação às taxas de saída dos nós pôde-se analisar enxames com variadas taxas de saída de semeadores. No enxame do grupo 3 os semeadores saem do sistema com maior freqüência do que nos outros grupos, por exemplo. Já no enxame do grupo 1 os semeadores passam mais tempo contribuindo com o sistema. Com relação à taxa de saída dos sugadores, o enxame que teve o maior índice de desistência de *downloads* foi o do grupo 5. Isso pode ter acontecido por conta da pequena banda de *download* dos nós. Já os sugadores do enxame do grupo 1 desistem com a menor freqüência.

Em termos de popularidade, o enxame do grupo 1 se mostrou como sendo o mais popular. Isso já era esperado por conta de seu tamanho ser o mesmo da capacidade da mídia de gravação de dados mais utilizada na época da coleta do *trace* (a do *compact disc*). Por outro lado, o enxame do grupo 5 é o menos popular. Em uma comunidade que compartilha arquivos de livre distribuição a tendência é que os usuários passem mais tempo no enxame por não estarem infringindo a lei por compartilhar arquivos proprietários sem autorização. Logo, a quantidade de enxames no *trace* com altas taxas de saída de nós tende a ser reduzida. No *trace* em questão, aproximadamente 7% dos enxames apresentaram taxas de saída de sugadores com a mesma ordem de grandeza do enxame do grupo 5. Todos os outros enxames têm taxas de saída de sugadores com ordens de grandeza inferiores a dele.

Com base nas observações acima e na metodologia de agrupamento utilizada, acredita-se que os enxames escolhidos foram representativos o suficiente para a análise do comporta-

Grupo	**Enxames**	γ **(s)**	θ **(s)**	**Tamanho (MB)**	***upload* (KB/s)**	***download* (KB/s)**
1	169	$7,5.10^{-6}$	$1,67.10^{-5}$	$700,68$	$10,32$	$16,18$
2	32	$1,2.10^{-5}$	$5,9.10^{-5}$	$143,32$	$5,43$	$7,86$
3	64	$4,5.10^{-4}$	$3,24.10^{-5}$	$380,89$	$252,49$	$304,31$
4	34	1.10^{-4}	$2,3.10^{-5}$	1024	$278,61$	$1261,4$
5	6	1.10^{-5}	$2,1.10^{-4}$	$0,49$	$0,14$	$0,099$

Tabela 5.1: Características dos enxames.

mento dos enxames mais comuns encontrados nas comunidades e possibilita a criação de variados cenários para a execução das simulações.

5.3 Cenários

Tipicamente um enxame se inicia com um semeador inicial e os sugadores vão entrando à medida que o enxame vai se popularizando. Após um certo tempo, que depende de cada enxame, os sugadores vão terminando o *download* e se transformando em semeadores. Entretanto, diferentemente de outros sistemas, no BitTorrent existe um tempo de vida para cada enxame. A partir de um momento, os semeadores começam a sair do sistema e com a queda de popularidade do enxame, novos sugadores não entram mais no sistema. Com isso, em algum momento o enxame deixará de existir. Tal comportamento está ilustrado na Figura 5.2.

Caracterizados os enxames mais representativos, pretende-se identificar em quais momentos do tempo de vida deles é mais vantajoso realizar o ataque. Para isso, 3 tempos de ataque considerados interessantes para observá-los foram escolhidos. O tempo de início do enxame (t_0), o tempo em que o enxame atinge o número máximo de sugadores (t_1) e o tempo em que o número de semeadores no enxame ultrapassa predominantemente o número de sugadores (t_2).

O tempo t_0 corresponde ao tempo inicial do enxame e foi escolhido para observar o comportamento do sistema quando os nós ainda estão se conhecendo. O tempo t_1 foi escolhido por conta de ser o momento de maior contenção de recursos (a concorrência pelos recursos está maior) na vida do enxame. O tempo t_2 foi escolhido por ser o momento na vida de um enxame que não existe contenção. Sempre existe largura de banda disponível. Espera-se que o caloteiro consiga um bom resultado realizando ataques neste tempo.

Grupo	Tempo	Instante (horas)	Semeadores	Sugadores	Caloteiros
1	t_0	0	1	1	6
	t_1	$24,58$	1	19	7
	t_2	52	10	9	5
2	t_0	0	1	1	6
	t_1	$3,33$	1	12	8
	t_2	$27,8$	7	6	3
3	t_0	0	1	1	5
	t_1	$0,5$	1	19	7
	t_2	$0,66$	13	6	2
4	t_0	0	1	1	10
	t_1	$1,14$	1	27	7
	t_2	$1,55$	17	15	13
5	t_0	0	1	1	2
	t_1	$0,55$	1	4	2
	t_2	$10,27$	3	2	2

Tabela 5.2: Caracterização dos cenários de simulação.

Os tempos e as quantidades de sugadores, semeadores e caloteiros estão descritas na tabela 5.2. As quantidades de caloteiros utilizados em cada cenário foram estimados com base no modelo matemático descrito no Capítulo 4. Porém, a quantidade de caloteiros para o tempo t_0 não pôde ser estimada com ajuda do modelo matemático porque a quantidade de sugadores deve ser maior que o número de conexões. Como só existe um sugador e um semeador no tempo t_0, o número de identidades foi calculada como sendo a média aritmética das quantidades de caloteiros nos tempos t_1 e t_2.

A presença de um caloteiro pode interferir na taxa de *download* médio que um sugador pode obter. Assim, se tal taxa fosse observada na presença do ataque não se teria como garantir que a presença do caloteiro não teve influência nela. Para os resultados serem mais justos, as taxas de *download* de caloteiros e sugadores foram observadas em rodadas separadas. Em uma rodada, executou-se a simulação sem caloteiros e observou-se a taxa média

de *download* de um sugador. Em outra, substituiu-se o sugador por um caloteiro com várias identidades. Por fim, um carona substituiu o sugador para o mecanismo de incentivo do BitTorrent ser analisado em tais cenários.

5.4 Resultados

Nesta seção são apresentados os resultados de simulações realizadas nos cenários descritos na Seção 5.3. Assume-se que um ataque é eficaz se a taxa média de *download* do caloteiro for maior ou igual àquela conseguida por um sugador, visto que um caloteiro não colabora com o sistema. Além disso, optou-se por fazer simulações utilizando a topologia estrela, na qual existe um nó central que interliga todos os outros nós, e com os nós possuindo mesmas larguras de banda de *download* e de *upload* para simplificar a análise dos resultados. Os resultados descritos aqui assumem um nível de confiança de 95% e erros de no máximo $\pm 5\%$.

Os resultados das simulações estão resumidos na Tabela 5.3. Nela, estão presentes o número de identidades estimado pelo modelo matemático (vide Capítulo 4) para cada cenário e a taxa média de *download* em KB/s com seus respectivos intervalos de confiança para caloteiros, sugadores e caronas[2]. Pôde-se observar que nos cenários analisados à medida que o enxame evolui a taxa média de *download* dos nós aumenta. Esse era um resultado esperado, uma vez que à medida que o tempo passa os nós vão interagindo, baixando partes do arquivo e se transformando em semeadores. Logo, a banda de *upload* deles tende a ser melhor utilizada.

Os resultados indicam que as estimativas produzidas pelo modelo matemático estão próximas da realidade na maioria dos casos. Apenas nos cenários dos tempos t_1 e t_2[3] do grupo 4 e t_1 dos grupos 1 e 3 o ataque não foi vantajoso para o caloteiro. Analisando os arquivos produzidos pelo simulador, constatou-se que nesses cenários a quantidade de nós que estão no enxame no momento do ataque é maior que em outros cenários e suficiente

[2]Como as simulações foram executadas com e sem caloteiros, quantidades diferentes de rodadas foram necessárias para garantir os 95% de nível de confiança. Assim, optou-se por utilizar o teste visual de intersecção dos intervalos de confiança para analisar os resultados [23].

[3]Neste cenário, um teste de hipótese foi realizado para confirmar que a taxa média do sugador é realmente maior que a do caloteiro. Tal teste pode ser conferido no Apêndice **??**.

Grupo	Ids	Caloteiros (KB/s)	Sugadores (KB/s)	Caronas (KB/s)
1	6	$7,74$	$6,24$	$5,84$
	7	$10,24$	$13,73$	$8,85$
	5	$15,69$	$15,76$	$15,58$
2	6	$4,88$	$4,85$	$4,74$
	8	$7,04$	$7,17$	$6,17$
	3	$7,67$	$7,27$	$7,48$
3	5	$234,91$	$214,36$	$149,82$
	7	$266,43$	$291,14$	$254,83$
	2	$298,28$	$299,72$	$295,95$
4	10	$262,44$	$238,58$	$171,16$
	7	$433,45$	$536,57$	$392,66$
	13	$774,94$	$813,81$	$733,14$
5	2	$0,067$	$0,060$	$0,055$
	2	$0,070$	$0,068$	$0,062$
	2	$0,098$	$0,098$	$0,098$

Tabela 5.3: Taxas médias de *download* para caloteiros, sugadores e caronas. Resultados com 95% de nível de confiança e erro máximo de $\pm 5\%$.

para fazer com que a maioria das conexões dos nós destinadas a *upload* estejam ocupadas, indicando um aumento na contenção de recursos. Assim, em cenários com maior contenção, os caloteiros têm dificuldade em conseguir fazer o *download* de partes do arquivo e isso evita que eles obtenham uma taxa média de *download* superior à de sugadores. Entretanto, nos cenários do grupo 4 basta aumentar o número de identidades para que o ataque passe a ser eficaz, como mostrado na Tabela 5.4 - as quantidades de identidades utilizadas nos novos cenários foram 10 vezes maiores que as estimadas pelo modelo analítico. O aumento do número de identidades faz com que o caloteiro passe a ficar mais conhecido e consiga uma maior taxa média de *download*, principalmente proveniente de conexões utilizadas pelo algoritmo de dessufocamento otimista.

Aumentar a quantidade de identidades nem sempre faz com que a taxa média de *down-*

Tempo	**Ids**	**Caloteiros (KB/s)**	**Sugadores (KB/s)**	**Caronas (KB/s)**
t_1	70	$554,04$	$536,57$	$392,66$
t_2	130	$951,94$	$813,81$	$733,14$

Tabela 5.4: Taxas médias de *download* para o número de identidades 10 vezes maior no cenário de tempos t_1 e t_2 do enxame do grupo 4.

load do caloteiro seja maior ou igual àquela do sugador. Em geral, essa estratégia ocasiona o aumento da taxa média de *download* do caloteiro até um ponto em que praticamente se estabiliza. Um exemplo disso acontece para o tempo t_1 do grupo 3 e está apresentado na Tabela 5.5. Nesse cenário, o incremento do número de identidades aumentou a taxa média de *download* do caloteiro, mas não foi suficiente para que ela ultrapassasse a de um sugador. Nele, o caloteiro não obtém sucesso por conta da configuração do enxame no momento do ataque. O tempo de ataque nesse cenário é bem próximo do tempo em que boa parte dos sugadores se transformam em semeadores. Além disso, a taxa de saída dos semeadores é alta enquanto que a taxa de saída dos sugadores é baixa. Assim, o caloteiro entra em um momento em que os nós que possuem as maiores quantidades de partes do arquivo estão prestes a sair do sistema. Pode-se observar esse comportamento na Figura 5.3.

Ids	**Taxa (KB/s)**	**Ids**	**Taxa (KB/s)**
30	$284,53$	200	$281,66$
60	$281,80$	300	$281,97$
90	$284,82$	400	$282,01$
120	$279,04$	500	$282,01$
150	$282,06$	600	$280,38$

Tabela 5.5: Variação das taxas médias de *download* com o aumento do número de identidades para o cenário de tempo t_1 do grupo 3.

Em casos extremos, como no tempo t_1 do grupo 1, não é bom para o caloteiro aumentar o número de identidades do ataque porque isso ocasiona a saída do semeador inicial antes que um sugador qualquer consiga terminar de baixar o arquivo. Isso também faz com que o atacante não consiga terminar de baixar o arquivo e o ataque não traga vantagem para o

caloteiro. Outro exemplo no qual não adianta aumentar o número de identidades do caloteiro é no cenário do tempo t_1 do grupo 5. Nele, quando um caloteiro possui 150 identidades sua taxa média de *download* cai para 0.063, por exemplo. Isso se deve ao fato que como a largura de banda de *download* dos nós é pequena, o tempo de *download* de cada subparte aumenta e os nós demoram a obter a primeira parte do arquivo. Assim, quando o caloteiro entra no sistema ele recebe *upload* apenas do semeador e a partir de conexões destinadas ao dessufocamento otimista. Na maioria das vezes que o algoritmo é executado o caloteiro é escolhido para receber *upload*. Entretanto, uma outra conexão com outra identidade do caloteiro é sufocada antes de receber a primeira subparte para que o novo escolhido seja dessufocado. Assim, quanto menos identidades um caloteiro possuir menor será a probabilidade dele ser sufocado antes que uma subparte seja recebida e sua taxa média de *download* tende a aumentar.

A divergência para alguns cenários entre a quantidade de identidades estimada pelo modelo e o necessário na prática para que o caloteiro tenha uma taxa de *download* maior ou igual àquela de um sugador derivam do fato que o modelo estima o número de identidades com base em todas as configurações possíveis de um cenário. Assim, ele considera que em algumas configurações uma parte das conexões destinadas a *upload* são compartilhadas entre sugadores e caloteiros. Isso nem sempre acontece em determinados momentos da vida de um enxame, nos quais as conexões de *upload* dos nós podem estar ocupadas pelos sugadores.

Comparando as taxas entre caloteiros e caronas, observa-se que na maioria dos cenários é melhor ser um caloteiro que um carona. Já entre sugadores e caronas, também na maioria dos cenários é melhor contribuir com o sistema. Entretanto, em determinados cenários[4], os quais não apresentam contenção de recursos, não foi possível afirmar que existe diferença entre as taxas de caloteiros e caronas e entre as taxas de sugadores e caronas. Portanto, em cenários onde a contenção é pequena até mesmo um carona consegue uma taxa média de *download* semelhante à taxa de *download* de caloteiros e de sugadores, ou seja, parece não haver diferença nisso. Esse resultado confirma as conclusões do trabalho de Locher *et al.* [17]. Naquele trabalho, os autores concluem que um carona consegue ter uma boa taxa de *download*. Ainda que eles não tenham explicitado isso, acredita-se que os resultados

[4]Tempo t_2 do grupo 1, tempos t_0 e t_2 do grupo 2 e no tempo t_2 do grupo 5.

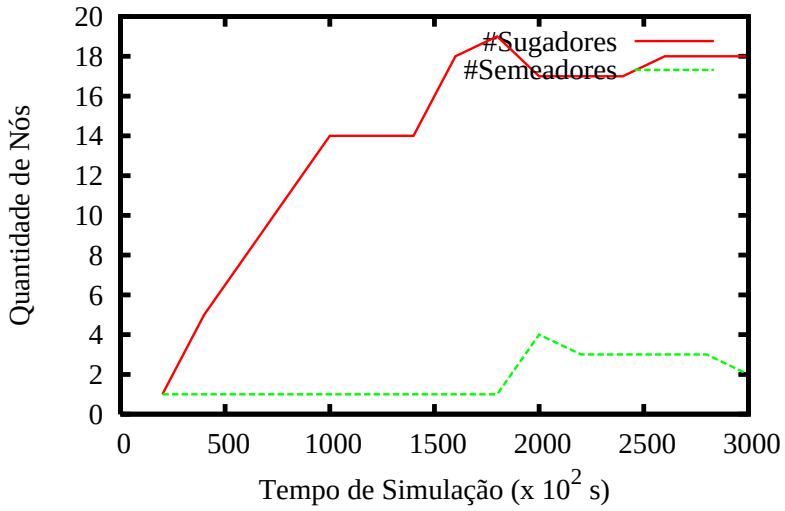


Figura 5.3: Quantidades de nós no tempo t_1 do enxame do grupo 3.

conseguidos naquele trabalho derivaram do fato de que não há contenção em quase todos os enxames analisados.

Em termos gerais, pode-se concluir dos resultados apresentados neste capítulo que um caloteiro pode conseguir uma taxa média de *download* semelhante ou até maior que a de um sugador em vários cenários. Logo, um caloteiro que vise burlar o sistema através de um ataque *sybil* pode desestimular os outros nós a colaborar, o que poderia provocar uma degradação no desempenho de redes BitTorrent. Entretanto, aumentar o número de identidades sem analisar o estado atual do enxame pode fazer com que o ataque falhe.

Capítulo 6

Conclusões

Neste trabalho, foi realizado um estudo do impacto que um ataque *sybil* pode causar em sistemas BitTorrent. Avaliou-se tal impacto analiticamente, por meio de um modelo matemático que apresentou uma noção dos efeitos do ataque em um determinado momento do tempo de vida de um enxame, e experimentalmente, através de simulações para avaliar a dinâmica das interações entre os nós ao longo do tempo e confrontar os resultados com os do modelo analítico.

Mesmo com números relativamente pequenos da quantidade de identidades estimados através do modelo, os resultados das simulações na maioria dos cenários mostram que o ataque pode degradar o desempenho de redes BitTorrent porque mesmo sem colaborar com o sistema um caloteiro pode burlar o mecanismo de incentivo e obter um tempo de *download* menor que o de um colaborador, desmotivando os nós a colaborarem com o sistema. Entretanto, nem sempre as estimativas do modelo matemático fizeram com que o ataque fosse eficaz.

As características e a configuração do enxame também podem influenciar na eficácia do ataque. Quando não existe contenção a quantidade de identidades estimada no modelo faz com que o ataque seja eficaz. Por outro lado, quando existe contenção o atacante não obtém sucesso em alguns cenários. Logo, é melhor ser um caloteiro quando o enxame não é tão popular, o que acontece antes ou depois do enxame atingir o número máximo de sugadores. Para isso, um atacante pode monitorar o enxame e com base em trabalhos como o de Guo *et al.* [11] ter uma estimativa de sua popularidade para decidir o momento certo de atacá-lo. Ainda assim, se um caloteiro quiser atacar um enxame quando ele estiver mais concorrido

ele pode aumentar o número de identidades. Entretanto, fazer isso de maneira indefinida nem sempre é uma boa estratégia porque pode ocasionar a saída prematura do semeador inicial em alguns cenários e evitar que um caloteiro obtenha sucesso. Assim, o ataque pode não ser eficaz em determinadas situações.

Este trabalho pode ser estendido de várias maneiras. Alterar o modelo matemático para que ele reflita melhor a realidade, realizar novas simulações utilizando novos cenários, executar experimentos em enxames reais para confrontar os resultados encontrados pelo modelo e pelas simulações e implementar e avaliar mecanismos que visem minimizar os efeitos de um ataque *sybil* no BitTorrent são exemplos de possíveis modos de dar continuidade a este trabalho.

Algumas simplificações foram consideradas no desenvolvimento do modelo matemático do Capítulo 4. Assim, o modelo pode ser alterado para que se aproxime da realidade nos cenários onde o número de identidades não foi suficiente para o caloteiro obter um tempo de *download* menor que o de um sugador.

Novas simulações podem ser realizadas considerando *traces* de outras comunidades, nós com diferentes larguras de banda, cenários com topologias mais próximas das encontradas na prática e falhas nos nós. Ainda assim, espera-se que essas novas simulações não apresentem resultados muito diferentes dos concluídos neste trabalho porque foram encontrados indícios de que o sucesso de um ataque *sybil* parece não depender desses fatores.

Outra maneira de dar continuidade a esse estudo é realizar experimentos em enxames reais utilizando o número de identidades estimado pelo modelo modificado para confrontar os resultados com os das simulações e concluir a avaliação do impacto do ataque de maneira mais satisfatória.

Por fim, mecanismos que minimizem os efeitos de um ataque *sybil* no BitTorrent podem ser propostos para dar continuidade a esse estudo. O objetivo geral seria reduzir a banda de *download* de um atacante, evitando que a estratégia mais interessante seja criar múltiplas identidades para atacar um enxame BitTorrent. Como um caloteiro não contribui com o sistema, ele não tem preferência para utilizar as conexões de *upload* dos outros nós. Assim, a principal fonte de *upload* para atacantes é oriunda das conexões destinadas aos nós escolhidos no sorteio realizado através do algoritmo de *optimistic unchoking*. Logo, a idéia de uma possível solução seria modificar tal algoritmo para que ele passe a também priorizar nós que

lhe ofereçam as melhores taxas de *upload*. Acredita-se que não seria interessante eliminar o algoritmo de *optimistic unchoking* do BitTorrent pela sua importância na difusão do arquivo, já que ele dá oportunidade a nós que possuem poucas (ou até nenhuma) partes do arquivo de obter mais partes e poder compartilhá-las com os outros nós.

Acredita-se que um mecanismo deve atender a alguns requisitos básicos para preservar o bom funcionamento do BitTorrent. O desempenho do sistema não deve ser reduzido consideravelmente com a sua implementação. Isso poderia afastar colaboradores em potencial. O mecanismo deve ainda evitar que haja desperdício de banda. Se um nó tem conexões de *upload* disponíveis e partes do arquivo que interessem a outros nós, então ele deve continuar fazendo *upload*, mesmo que seja para nós que não fizeram *upload* para ele. Por fim, acredita-se que um mecanismo que diminua a banda obtida por um colaborador não é interessante porque isso poderia desmotivar usuários a utilizarem o sistema.

Uma possível implementação de um mecanismo para reduzir os efeitos de um ataque *sybil* no BitTorrent seria direcionar o sorteio realizado pelo algoritmo de *optimistic unchoking* de maneira a também priorizar os maiores colaboradores. Isso poderia ser implementado através da modificação na maneira de realizar o sorteio. Ele deixaria de ser meramente aleatório para ser probabilístico. Com o sorteio probabilístico, nós que doaram mais durante o intervalo de tempo entre as rodadas do algoritmo teriam maior probabilidade de serem sorteados. Para isso, bastaria replicar proporcionalmente os nós na lista do sorteio. Os maiores colaboradores seriam replicados mais vezes. A tendência é que nós maliciosos sejam marginalizados de maneira eficaz porque eles não contribuem com o sistema e seriam escolhidos com uma menor probabilidade.

Referências Bibliográficas

[1] Eytan Adar and Bernardo A. Huberman. Free Riding on Gnutella. *First Monday*, Outubro 2000.

[2] Nazareno Andrade. *Alocação de Recursos Baseada em Padrões de Uso para a Distribuição Colaborativa de Conteúdo*. Proposta de Doutorado em Ciência da Computação, Universidade Federal de Campina Grande, Outubro 2007.

[3] Nazareno Andrade, Francisco Brasileiro, Walfredo Cirne, and Miranda Mowbray. Automatic Grid Assembly by Promoting Collaboration in Peer-to-Peer Grids. *J. Parallel Distrib. Comput.*, 67(8):957–966, 2007.

[4] Nazareno Andrade, Jaindson Santana, Francisco Brasileiro, and Walfredo Cirne. On the Efficiency and Cost of Introducing QoS in BitTorrent. In *Seventh International Workshop on Global and Peer-to-Peer Computing - GP2PC*, 2007.

[5] James Aspnes, Collin Jackson, and Arvind Krishnamurthy. Exposing Computationally-Challenged Byzantine Impostors. Technical Report YALEU/DCS/TR-1332, Yale University Department of Computer Science, julho 2005.

[6] Anthony Bellissimo, Prashant Shenoy, and Brian Neil Levine. Exploring the Use of BitTorrent as the Basis for a Large Trace Repository. Technical Report 04-41, Department of Computer Science, University of Massachusetts, Junho 2004.

[7] Walfredo Cirne, Francisco Brasileiro, Nazareno Andrade, Lauro Costa, Alisson Andrade, Reynaldo Novaes, and Miranda Mowbray. Labs of the World, Unite!!! *Journal of Grid Computing*, 4(3):225–246, 2006.

[8] Bram Cohen. Incentives Build Robustness in BitTorrent. In *Workshop on Economics of Peer-to-Peer Systems*, Berkeley, CA, EUA, Maio 2003.

[9] J. R. Douceur. The Sybil attack. *Lect. Note. Comput. Sci.*, 2429:251–260, 2002.

[10] Elle Cayabyab Gitlin. HBO doesn't want Rome downloaded in a day. Publicado no sítio de notícias tecnológicas da Ars Technica., Outubro 2005. `http://arstechnica.com/news.ars/post/20051007-5401.html`.

[11] Lei Guo, Songqing Chen, Zhen Xiao, Enhua Tan, Xiaoning Ding, and Xiaodong Zhang. Analyzing Torrent Evolution and Performance of BitTorrent-like File Sharing Systems. Technical report, Ohio State University, George Mason University and AT&T Labs-Research, 2005.

[12] Joseph F. Hair, William C. Black, Barry J. Babin, Rolph E. Anderson, and Ronald L. Tatham. *Multivariate Data Analysis*. Pearson, sixth edition, 2006.

[13] Marlon A. Konrath, Marinho P. Barcellos, Juliano F. Silva, Luciano P. Gaspary, and Rafael Dreher. Atacando um Enxame com um Bando de Mentirosos: vulnerabilidades em BitTorrent. In *Anais do 25º Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC'07)*, Belém, Pará, Brasil, 2007. Sociedade Brasileira de Computação.

[14] Leslie Lamport, Robert Shostak, and Marshall Pease. The Byzantine Generals Problem. In *Advances in Ultra-Dependable Distributed Systems, N. Suri, C. J. Walter, and M. M. Hugue (Eds.), IEEE Computer Society Press*. 1995.

[15] Arnaud Legout, G. Urvoy-Keller, and P. Michiardi. Rarest First and Choke Algorithms are Enough. In *IMC '06: Proceedings of the 6th ACM SIGCOMM on Internet measurement*, pages 203–216, Nova York, NY, EUA, 2006. ACM Press.

[16] Brian Neil Levine, Clay Shields, and N. Boris Margolin. A Survey of Solutions to the Sybil Attack. Technical Report 2006-052, University of Massachusetts Amherst, Amherst, MA, Outubro 2006.

[17] Thomas Locher, Patrick Moor, Stefan Schmid, and Roger Wattenhofer. Free Riding in BitTorrent is Cheap. In *5th Workshop on Hot Topics in Networks (HotNets), Irvine, California, EUA*, Novembro 2006.

[18] N. Boris Margolin and Brian Neil Levine. Informant: Detecting Sybils Using Incentives. In *Proc. Financial Cryptography (FC)*, February 2007.

[19] James Newsome, Elaine Shi, Dawn Song, and Adrian Perrig. The Sybil Attack in Sensor Networks: Analysis & Defenses. In *IPSN '04: Proceedings of the third international symposium on Information processing in sensor networks*, pages 259–268, Nova York, NY, EUA, 2004. ACM Press.

[20] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank Citation Ranking: Bringing Order to the Web. In *7th International World Wide Web Conference*, pages 161–172, 1998.

[21] Felipe Pontes, Francisco Brasileiro, and Nazareno Andrade. Sobre Calotes e Múltiplas Personalidades no BitTorrent. In *Anais do III Workshop de Peer-to-Peer*, pages 75–86, Belém, Pará, Brasil, Junho 2007. Sociedade Brasileira de Computação.

[22] Dongyu Qiu and R. Srikant. Modeling and Performance Analysis of BitTorrent-like Peer-to-Peer Networks. In *SIGCOMM '04: Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 367–378, Nova York, NY, EUA, 2004. ACM Press.

[23] Raj Jain. *The Art of Computer Systems Performance Analysis*. Wiley, 1991.

[24] Drew Roselli, Jacob R. Lorch, and Thomas E. Anderson. A Comparison of File System Workloads. In *ATEC'00: Proceedings of the Annual Technical Conference on 2000 USENIX Annual Technical Conference*, pages 41–54, Berkeley, CA, EUA, 2000. USENIX Association.

[25] F. R. Schreiber. *Sybil*. Warner Books, 1973.

[26] Ka Cheung Sia. DDoS Vulnerability Analysis of the Bittorrent Protocol. Technical report, University of California, Junho 2006.

[27] Michael Sirivianos, Jong H. Park, Rex Chen, and Xiaowei Yang. Free-riding in BitTorrent Networks with the Large View Exploit. In *IPTPS*, 2007.

[28] Jin Sun. *Multiple Identities in BitTorrent Networks*. Mestrado em Ciência da Computação, University of California Riverside, Dezembro 2006.

[29] Inovação Tecnológica. Redes de Sensores vão Monitorar Integridade Estrutural de Aviões, Julho 2007. `http://www.inovacaotecnologica.com.br/noticias/noticia.php?artigo=010110070727`.

[30] Wikipedia. Spam, Fevereiro 2008. `http://pt.wikipedia.org/wiki/Spam`.

[31] W. Yang and N. Abu-Ghazaleh. GPS: a General Peer-to-Peer Simulator and its use for Modeling BitTorrent. *Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2005. 13th IEEE International Symposium on*, pages 425–432, 2005.

[32] Haifeng Yu, Michael Kaminsky, Phillip B. Gibbons, and Abraham Flaxman. SybilGuard: Defending Against Sybil Attacks via Social Networks. In *SIGCOMM '06: Proceedings of the 2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 267–278, New York, NY, USA, 2006. ACM Press.

Apêndice A

Ataques *Sybil* na Rede de Favores

Este estudo foi realizado em parceria com Miranda Mowbray do HP Labs.

Neste anexo são apresentados um modelo matemático capaz de indicar quantas identidades são necessárias para um atacante obter uma maior utilidade que aquela de um colaborador na rede de favores (ou NoF, do inglês *Network of Favors*), o mecanismo de incentivo do OurGrid [7], além de uma análise desse modelo. Na Seção A.1 apresenta-se uma introdução sobre ataques *sybil* na NoF. Na Seção A.2 é apresentado o modelo matemático. A Seção A.3 apresenta os resultados analíticos do modelo matemático. Por fim, a Seção A.4 apresenta as conclusões sobre ataques *sybil* na NoF.

A.1 Introdução

A rede de favores é um esquema autônomo de reputação para ajudar nós com recursos ociosos a determinar para qual nó requisitante doar. A idéia central da NoF é que os usuários que são os maiores cooperadores devem ter prioridade mais alta para consumir recursos da comunidade. Este princípio age como um guia para a divisão dos recursos entre os usuários que estão requisitando os mesmos, logo, age como um incentivo para colaboração [3]. Dessa maneira, *free-riders* são identificados e marginalizados eficientemente.

Entretanto, esse mecanismo não evita ataques *sybil* [9], no qual um nó malicioso pode criar identidades suficientes para constituir uma grande fração da comunidade e enganar o

sistema, conseguindo tanta utilidade quanto possível para ele mesmo sem contribuir com o sistema, isto é, *free-riding*. Este tipo de ataque é possível em sistemas de fácil obtenção de identidades. Em sistemas com mecanismos para dificultar a geração de identidades, ataques *sybil* são mais difíceis. Apesar dos nós na rede de favores pertencerem a instituições reais e confiáveis, seu esquema de geração de identidades permite a fácil criação de múltiplas identidades, provendo um cenário propício para um nó malicioso realizar um ataque *sybil*.

Na rede de favores, o compartilhamento de um recurso é encarado como um favor. Então, se um nó **A** doa um recurso para um nó **B** ele estará realizando um favor, de modo que quando ele precisar de recursos o nó **B** irá privilegiá-lo na doação. Para que isso seja possível, um nó armazena os registros dessas doações em uma lista de interações com os outros nós. Dessa maneira, um atacante pode preencher essa lista, fazendo com que a probabilidade dele conseguir um recurso pela multiplicação do seu número de identidades aumente. Quando existirem nós com o mesmo saldo requisitando um recurso, um colaborador decidirá aleatoriamente para quem doar. Portanto, se novatos e atacantes estiverem requisitando um recurso de um colaborador ao mesmo tempo, a probabilidade de um nó malicioso conseguir o recurso é tão grande quanto seu número de identidades.

Neste apêndice serão apresentados um modelo analítico e os resultados de um estudo sobre ataques *sybil* na rede de favores. O modelo analítico permite calcular quantas identidades são necessárias para que a utilidade de um atacante seja maior que a de um colaborador. Por outro lado, os resultados mostram como se comporta o modelo diante de variações em certos parâmetros.

A.2 Modelo Analítico

Nesse modelo, um atacante não está tentando destruir o sistema, mas conseguir tanta utilidade quanto possível. Nesse contexto, sejam C o conjunto de colaboradores, de tamanho c, os quais possuem reputação positiva diante dos outros nós e SY o conjunto de nós *sybil*, de tamanho $sy > 1$, os quais possuem reputação nula diante dos outros nós; os nós possuem probabilidade independente p de estarem consumindo um recurso, consumidores e novatos (nós que ainda não interagiram) que não estão consumindo estão doando um recurso e um nó doa um recurso com utilidade 1, sendo v o custo da doação, $0 < v < 1$. Além disso,

atacantes requisitam recursos continuamente.

A utilidade esperada de atacantes e de colaboradores depende de configurações do sistema, isto é, todas as possíveis combinações de nós consumindo ou doando um recurso, e é influenciada de acordo com a probabilidade da combinação ocorrer.

Para $0 <= c' <= c$ e $0 <= n' <= n$ existem $\binom{c}{c'}.\binom{n}{n'}$ combinações nas quais o número de nós em C e o número de nós em N consumindo um recurso são c' e n' respectivamente. Cada combinação ocorre com probabilidade $p^{(c'+n')}.(1-p)^{(c+n-c'-n')}$.

Numa configuração com c' nós em C e n' nós em N existem $c-c'$ colaboradores doando recursos, os quais possuem um custo conjunto de doação $(c-c').v$. Os c' colaboradores que estão consumindo um recurso juntamente recebem recursos de utilidade 0 se $c'=0$ e $c-c'+(n-n').c'/(c'+n'+sy)$ se $c'>0$. Por outro lado, o atacante consegue recursos de utilidade $(c+n-n').sy/(n'+sy)$ se $c'=0$ e $(n-n').sy/(c'+n'+sy)$ se $c'>0$.

Daí, a utilidade esperada obtida por um colaborador considerando as configurações possíveis de consumo e doação de recursos para o sistema ponderado de acordo com as probabilidades delas ocorrerem é

$$\frac{1}{c}.\left[\sum_{c'=1}^{c}\sum_{n'=0}^{n}\left(c-c'+\frac{(n-n').c'}{c'+n'+sy}\right).\binom{c}{c'}.\binom{n}{n'}.p^{(c'+n')}.(1-p)^{(c+n-c'-n')}-\sum_{c'=0}^{c}v.(c-c').\binom{c}{c'}.p^{c'}.(1-p)^{c-c'}\right].$$

Já a utilidade esperada obtida por um atacante, ou seja, a soma das utilidades conseguidas por todos os nós *sybil* considerando todas as configurações possíveis do sistema é

$$p.\left[\sum_{n'=0}^{n}\frac{(c+n-n').sy}{n'+sy}.\binom{n}{n'}.p^{n'}.(1-p)^{(c+n-n')}+\sum_{c'=1}^{c}\sum_{n'=0}^{n}\frac{(n-n').sy}{c'+n'+sy}.\binom{c}{c'}.\binom{n}{n'}.p^{(c'+n')}.(1-p)^{(c+n-c'-n')}\right].$$

Portanto, é possível calcular quão grande sy precisa ser para fazer com que a segunda expressão seja maior que a primeira, isto é, a utilidade recuperada por um atacante seja maior que a utilidade recuperada por um colaborador.

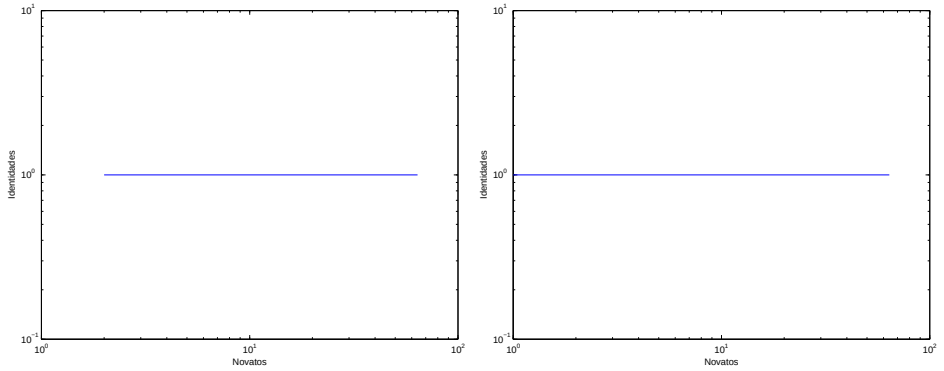
A.3 Análise

Esta subseção descreve resultados do modelo analítico que possibilita o cálculo da quantidade de identidades necessárias para um atacante conseguir uma maior utilidade que um

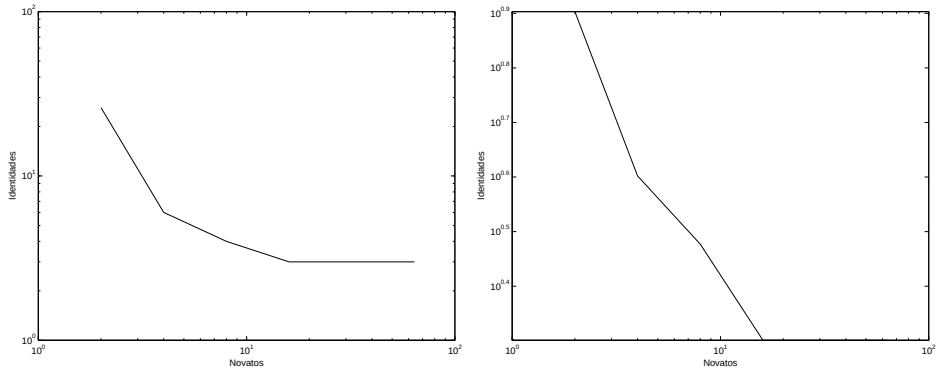
colaborador. Os resultados analisados levam em consideração variações no número de novatos no sistema, na probabilidade de um nó estar consumindo um recurso e no custo de doação para observar como se comporta o número de identidades.

Figuras A.1 e A.2 mostram o número de identidades necessárias para a utilidade de um atacante ser maior que a de um colaborador quando a quantidade de novatos presente no sistema varia, para 10 e 100 colaboradores respectivamente. Observa-se que o número de identidades depende fortemente da quantidade de novatos. Em geral, quando o número de novatos cresce, o número de identidades decresce, ou seja, o número de identidades é inversamente proporcional ao número de novatos. Portanto, nesse modelo, o número de novatos é muito importante para um ataque *sybil* ser realizado. Além disso, como era esperado, o número de identidades aumenta quando o número de colaboradores também aumenta. Esse resultado pode ser interpretado da seguinte maneira. Colaboradores são privilegiados por causa de suas reputações positivas. Isso significa que quando um colaborador requisita um recurso ao mesmo tempo que um novato ou um atacante ele será escolhido. Além disso, um novato recebe recursos de outros novatos e oferece seus recursos a colaboradores e atacantes. Por outro lado, um atacante recebe recursos apenas de novatos enquanto eles permanecem novatos. O atacante ganha utilidade de um colaborador apenas quando todos os colaboradores estão doando recursos. A utilidade conseguida por um colaborador é predominantemente obtida a partir de outros colaboradores. Enquanto que a utilidade ganha por um atacante é predominantemente conseguida de novatos. Isto é representado na figura A.3 e reflete nos cálculos das utilidades.

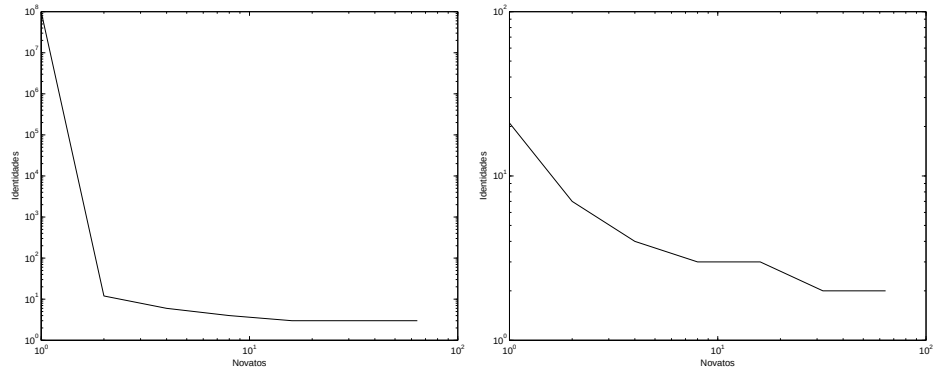
Outras conclusões podem ser extraídas dos gráficos. Quando a probabilidade de consumo de um recurso é pequena ($0,1$), o número de identidades necessárias para um atacante conseguir maior utilidade que um colaborador é pequeno também. Isso acontece devido à pequena quantidade de nós consumindo recursos, ou seja, recursos sobram no sistema. Quando o número de nós consumindo recursos cresce não há sobra de recursos. Daí, um atacante deve gerar mais identidades para que o ataque seja efetivo. Além disso, quando a probabilidade de nós estarem consumindo um recurso permanece invariável e o custo de doação de um recurso aumenta o número de identidades diminui. Intuitivamente, quando o custo de doação de um recurso cresce, a utilidade obtida pelos colaboradores decresce. Logo, a utilidade de um atacante irá exceder a utilidade de um colaborador mais rapidamente. Deste



(a) $p = 0.1$ e $v = 0.1$ (b) $p = 0.1$ e $v = 0.4$

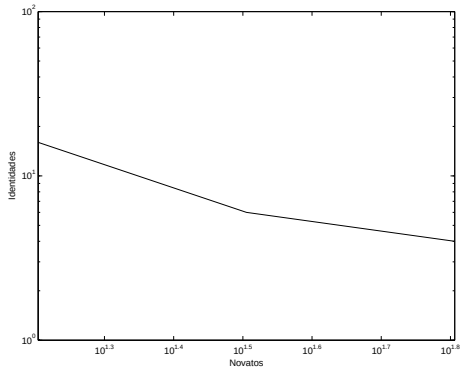


(c) $p = 0.5$ e $v = 0.1$ (d) $p = 0.5$ e $v = 0.4$

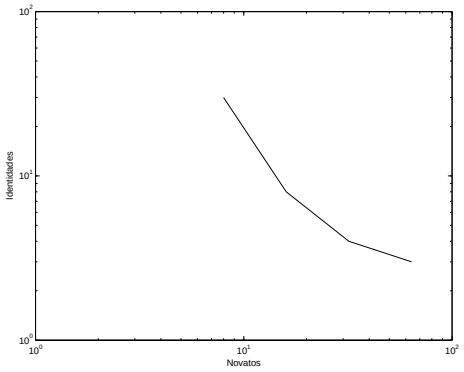


(e) $p = 0.9$ e $v = 0.1$ (f) $p = 0.9$ e $v = 0.4$

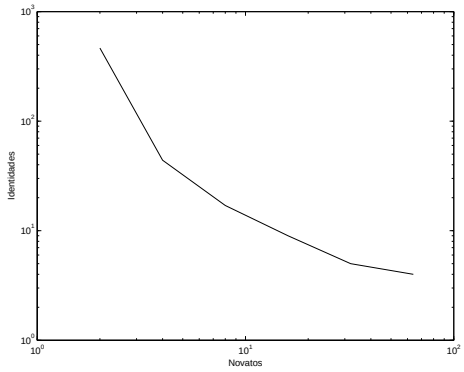
Figura A.1: Variação das identidades quando o número de colaboradores é 10.



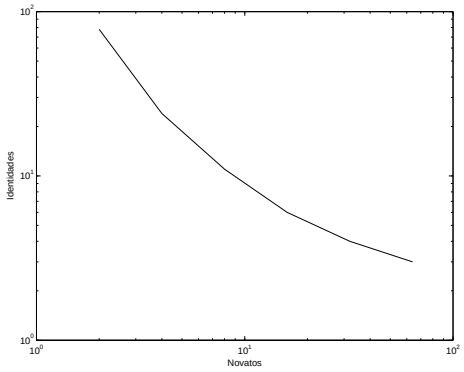
(a) $p = 0.1$ e $v = 0.1$



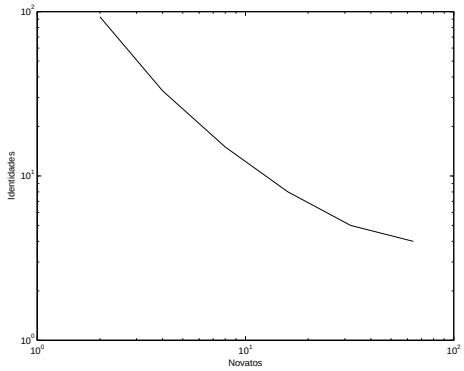
(b) $p = 0.1$ e $v = 0.4$



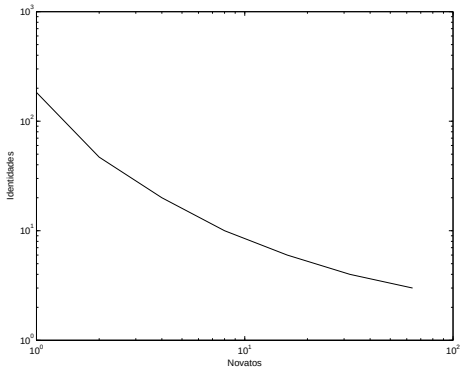
(c) $p = 0.5$ e $v = 0.1$



(d) $p = 0.5$ e $v = 0.4$



(e) $p = 0.9$ e $v = 0.1$



(f) $p = 0.9$ e $v = 0.4$

Figura A.2: Variação das identidades quando o número de colaboradores é 100.

modo, o número de identidades necessárias também se reduz.

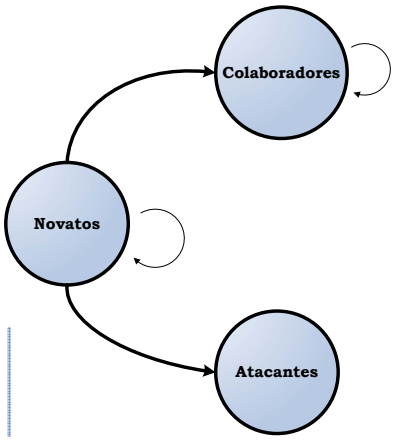


Figura A.3: Relacionamentos entre colaboradores, novatos e atacantes.

Entretanto, existe uma situação na qual o ataque falha. Nesse caso, o número de identidades tende ao infinito e não é possível calcular exatamente o número de identidades necessárias. Quando isso acontece, $c'/(c' + n' + sy) \rightarrow 0$, $sy/(c' + n' + sy) \rightarrow 1$ e $sy/(n' + sy) \rightarrow 1$. Com isso, $v \leq 1 - n.p - (1 + p.c).[(1 - p)^{(c-1)}]$. Quando essa inequação é satisfeita, a utilidade esperada para um colaborador é igual ou maior que a utilidade esperada para um atacante e um ataque *sybil* não é efetivo. Deste modo, a utilidade de um atacante nunca será maior que a de um colaborador, não importa quantas identidades ele consiga gerar. Isso pode ser observado em alguns gráficos das figuras A.1 e A.2. Neste caso, para alguns valores de novatos, não existe um valor para o número de identidades.

A.4 Discussão

Ataques *sybil* podem representar uma ameaça para a rede de favores. Em geral, quanto maior o número de identidades, maior será a utilidade conseguida por um atacante.

Na NoF, um ataque pode ser potencializado através do aumento no número de novatos, pois um atacante ganha tantos recursos quanto maior for a quantidade de novatos, e pelo incremento do custo de doação de um recurso, por causa da redução na utilidade obtida pelos colaboradores.

O número de identidades decresce quando a quantidade de novatos cresce. Daí, é bom

para a realização de um ataque quando o sistema é composto por vários novatos. Evitar um ataque *sybil* limitando o número de novatos é impraticável. No entanto, na comunidade de um grid geralmente existe um momento no qual os nós são na sua maioria colaboradores. Logo, a quantidade de novatos deve permanecer pequena e o ataque pode não ser efetivo.

A redução do custo de doação de um recurso pode evitar os efeitos nocivos de um ataque *sybil* na NoF. Em uma determinada situação, existe uma condição para o custo de doação na qual um ataque *sybil* falha. Entretanto, essa mudança pode não ser boa para incentivar os nós a doarem seus recursos, porque a recompensa pelo favor será reduzida se for considerado que a utilidade perdida por um doador como resultado de uma doação é um múltiplo fixo da utilidade ganha por um consumidor [3]. Além disso, na atual política da NoF, um nó é livre para decidir seu próprio custo de doação.