Número 01
Volume 03
Junho / 2010
ISSN 1983-4217

REVISTA BRASILEIRA DE REDES DE COMPUTADORES E SISTEMAS DISTRIBUÍDOS

Lisandro Zambenedetti Granville
Thaís Vasconcelos Batista

Número 1
Volume 3
Junho de 2010
ISSN 1983-4217

Revista Brasileira de Redes de Computadores e Sistemas Distribuídos

Laboratório Nacional de Redes de Computadores

Sociedade Brasileira de Computação

Revista Brasileira de Redes de Computadores e Sistemas Distribuídos /
Laboratório Nacional de Redes de Computadores (LARC), Comissão Especial de Redes de Computadores da Sociedade Brasileira de Computação.
Rio de Janeiro: LARC; SBC, 2010.
v. 3, n. 1 (jan-jun, 2010)
v. : il. cm.
Semestral.
Texto em português ou inglês.
ISSN 1983-4217

1. Redes de computadores 2. Sistemas distribuídos. I. Laboratório Nacional de Redes de Computadores (LARC) II. Sociedade Brasileira de Computação. Comissão Especial de Redes de Computadores. III. Título.

CDD 004.65

Laboratório Nacional de Redes de Computadores (LARC)
Contato: Luciano Paschoal Gaspary
Instituto de Informática - UFRGS
Av. Bento Gonçalves, 9500 - Porto Alegre, RS
Caixa Postal 15064 - CEP 91501-970

Sociedade Brasileira de Computação
Contato: Carlos André Guimarães Ferraz
Av. Bento Gonçalves, 9500 - Porto Alegre, RS
Caixa Postal 15012 - CEP 91501-970

Conteúdo

Carta dos Editores

A Revista Brasileira de Redes de Computadores e Sistemas Distribuídos é um periódico promovido conjuntamente pelo Laboratório Nacional de Redes de Computadores (LARC) e Sociedade Brasileira da Computação (SBC) através de sua Comissão Especial em Redes de Computadores e Sistemas Distribuídos (CE-RESD). Como resultado das ações da comunidade nacional de pesquisa em redes de computadores e sistemas distribuídos, a Revista ter por objetivo se estabelecer como um veículo de divulgação dos avanços científicos e tecnológicos da área, e assim estender o atual alcance do prestigioso e tradicional Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC).

Neste terceiro número, a Revista apresenta versões atualizadas e estendidas de três artigos dentre os artigos ganhadores do prêmio de melhor trabalho da 28ª edição do SBRC, realizado em 2010 em Gramado. O primeiro artigo versa sobre diminuição de consumo de energia em grades oportunistas. O segundo artigo versa sobre comunicação de dados em redes de sensores sem fio. Finalmente, o terceiro artigo versa sobre a conteção de ataques de Sybils

No primeiro artigo, "Usando as Estratégias Sobreaviso e Hibernação para Economizar Energia em Grades Computacionais Oportunistas", de Lesandro Ponciano, Francisco Brasileiro, Jaindson Santana, Marcus Carvalho, Matheus Gaudencio, os autores apresentam uma avaliação do impacto de duas estratégias para diminuir o consumo de energia dos recursos de uma grade oportunista: Sobreaviso (Standby) e Hibernação (Hibernate). Tais estratégias são utilizadas quando as máquinas estão disponíveis para a grade, mas não têm nenhuma tarefa para executar, ou seja, estão inativas. O artigo avalia também, após quando tempo de inatividade as estratégias devem ser utilizadas. Ambas as estratégias de computação verde impactam o tempo de resposta das aplicações executadas, mas reduzem o gasto da infraestrutura com energia.

No segundo artigo, "Comunicação de Dados baseada no Receptor para Redes de Sensores Sem Fio" de autoria de Max do Val Machado, Raquel Mini e Antonio Loureiro, é apresentado o protocolo *Receiver-based Medium Access Control* (Rb-MAC) para ser utilizado em projetos integrados para a comunicação baseada no receptor.Um projeto integrado consiste em uma camada ser capaz de violar a arquitetura da pilha de protocolos para acessar informações disponíveis em outra camada, a fim de explorar vantagens e evitar pontos fracos. Esses modelos são apropriados para as Redes de Sensores Sem Fio por lidarem com restrições de energia e topologia dinâmica. Resultados de simulação mostram que usando o Rb-MAC, o modelo receptor avaliado aumenta a taxa de entrega, reduz o número de transmissões e o consumo de energia quando comparado com um modelo baseado no emissor. Além disso, o Rb-MAC permite um ciclo de trabalho dinâmico e mantém a taxa de entrega para valores reduzidos de ciclo de trabalho.

Por fim, no terceiro artigo, "Dois Pesos, Duas Medidas: Gerenciamento de Identidades Orientado a Desafios Adaptativos para Contenção de Sybils" de autoria de Gustavo Mauch, Flávio Santos, Weverton Cordeiro, Marinho Barcellos, Luciano Gaspary, é apresentado uma abordagem para mitigar o ataque Sybil, que consiste na criação indiscriminada de identidades forjadas por um usuário malicioso. A abordagem apresentada consiste em atribuir ou renovar a concessão de identidades aos usuários solicitantes mediante a resolução de

desafios computacionais, Apesar de suas potencialidades, as soluções baseadas em tal abordagem não distinguem solicitações de usuários legítimos das de atacantes, fazendo com que ambos sejam igualmente penalizados. O artigo propõe o uso de desafios adaptativos como limitante à disseminação de Sybils. Estima-se um grau de confiança da fonte de onde partem as solicitações de identidade em relação às demais. Quanto maior a frequência de solicitação de identidades, menor o grau de confiança e, consequentemente, maior a complexidade do desafio a ser resolvido pelo(s) usuário(s) associado(s) àquela fonte. Resultados obtidos por meio de experimentação mostram a capacidade da solução proposta em diminuir a capacidade dos atacantes de criarem identidades falsas de forma indiscriminada, ao mesmo tempo sendo favorável a usuários legítimos, os quais foram penalizados minimamente.

Lisandro Zambenedetti Granville
Editor
Universidade Federal do Rio Grande do Sul
Instituto de Informática
Av. Bento Gonçalves, 9500
Caixa Postal 15064
91501-970 – Porto Alegre, RS

Thais Vasconcelos Batista
Editor
Universidade Federal do Rio Grande do Norte
Departamento de Informática e
Matemática Aplicada
Av. Sen. Salgado Filho, 3000
59072-970 – Natal, RN

Usando as Estratégias Sobreaviso e Hibernação para Economizar Energia em Grades Computacionais Oportunistas

Lesandro Ponciano, Francisco Brasileiro, Jaindson Santana, Marcus Carvalho, Matheus Gaudencio

Universidade Federal de Campina Grande
Departamento de Sistemas e Computação
Av. Aprígio Veloso, s/n, Bodocongó
58.429-900 - Campina Grande, PB - Brasil
{lesandrop, fubica}@dsc.ufcg.edu.br, {jaindson, marcus, matheusgr}@lsd.ufcg.edu.br

Abstract

Nowadays, opportunistic grids are getting more and more popular. In these systems, idle time of computing resources is used to process third-party workloads. Thus, the energy efficiency of those grids is also an increasing concern. This paper evaluates two green computing strategies to reduce energy consumption in resources of an opportunistic grid, namely: Standby and Hibernate. Both techniques are used when a resource is idle, i.e., available to the grid, but there is no work to be processed. Our evaluation uses a simulation model to assess the cost incurred in terms of increased the response time (makespan) of the jobs executed, as well as the energy savings achieved. We simulate an opportunistic grid that uses both strategies and also a scenario without a green computing strategy. As expected, both techniques increased the makespan of the jobs executed, but improved energy savings when compared to the scenario that does not use a green computing strategy. Moreover, for the scenarios evaluated, the Standby approach resulted in greater savings and in a smaller impact on the application makespan, being a better strategy to be used in such grids.

Keywords: Green computing, energy saving, sleeping strategy, standby, hibernate, opportunistic grids.

Resumo

Grades oportunistas são sistemas computacionais que têm sido amplamente utilizados para execução de aplicações científicas. Nesses sistemas, o tempo ocioso dos recursos computacionais é aproveitado para executar aplicações de terceiros. Nos últimos anos tem aumentado também a preocupação com a eficiência energética dessas grades. Este trabalho avalia o impacto do uso de estratégias para diminuir o consumo de energia dos recursos de uma grade oportunista. Duas estratégias são analisadas: Sobreaviso (Standby) e Hibernação (Hibernate). Elas são utilizadas quando as máquinas estão inativas, i.e., estão disponíveis para a grade, mas não têm nenhuma tarefa para executar. Avaliamos, também, após quando tempo de inatividade (TI) as estratégias devem ser utilizadas. Como esperado, ambas as estratégias de computação verde impactam o tempo de resposta das aplicações executadas, mas reduzem o gasto da infraestrutura com energia. Nos cenários avaliados, a estratégia Sobreaviso resultou em uma economia de energia equivalente à estratégia Hibernação, mas em um menor impacto no tempo de resposta das aplicações. Os resultados obtidos mostram, também, que a estratégia pode ser utilizada tão logo a máquina torne-se inativa, não sendo necessário aguardar um tempo de inatividade. Isso aumenta a economia de energia e não impacta significativamente o tempo de resposta das aplicações.

Palavras-chave: Computação verde, economia de energia, estratégia de economia de energia, sobreaviso, hibernação, grades oportunistas.

1. INTRODUÇÃO

A evolução dos sistemas computacionais tem sido marcada pela busca por mais poder computacional a qualquer custo [9]. No entanto, com o aumento do poder computacional, aumentou-se também o consumo de energia desses sistemas e, por consequência, a emissão de dióxido de carbono (CO_2) no meio ambiente. Os problemas ambientais gerados pelo aumento da emissão de CO_2 e o custo financeiro ocasionado pelo consumo de energia têm impulsionado estudos que visam o desenvolvimento de mecanismos e tecnologias que façam uso mais eficiente da energia. Esses mecanismos e tecnologias são denominados de *computação verde* (*green computing*) [29, 9].

O objetivo de usar a energia de modo mais eficiente tem influenciado diversas áreas da computação, como, por exemplo, o projeto de hardware e a gerência de centros de processamento de dados (*datacenters*) [3] e grades computacionais de serviço [19, 25, 21, 22]. Este trabalho analisa o uso de estratégias para economizar energia em grades computacionais oportunistas. Os recursos pertencentes a este tipo de grade são utilizados de forma oportunista: se o recurso não estiver sendo utilizado pelo dono do recurso, o poder computacional disponível, porém ocioso, pode ser utilizado para executar tarefas de terceiros que tenham sido submetidas à grade.

Quando um computador está disponível para a grade, mas a grade não o utiliza, ele fica em estado `ocioso`. Ao longo do tempo, a permanência dos computadores nesse estado caracteriza ciclos de ociosidade na grade e desperdício de energia. Estudos anteriores mostram que esses ciclos de ociosidade são comuns em diversos tipos de grandes computacionais [13, 17]. Um computador ocioso apresenta menor consumo de energia do que quando está executando alguma aplicação, porém esse consumo de energia é ainda significativo [12].

Duas estratégias para reduzir o consumo de energia nesses ciclos de ociosidade são: Sobreaviso (`Standby`) e Hibernação (`Hibernate`). A estratégia Sobreaviso, também conhecida por Suspensão para a Memória de Acesso Aleatório (RAM, do inglês *Random Access Memory*), consiste em manter a memória RAM ativa e reduzir a atividade do disco rígido e do processador. Já a estratégia Hibernação, também conhecida por Suspensão para o Disco, consiste em salvar o estado da memória RAM no disco rígido, reduzir o uso de energia da RAM, do disco rígido e do processador. Essas estratégias são definidas pelo padrão de configuração avançada e interface de energia (ACPI, do inglês *Advanced Configuration and Power Interface* [8]) - padrão aberto que unifica a configuração dos dispositivos e a interface de gerência de energia pelos sistemas operacionais.

Essas estratégias desativam diversos dispositivos para reduzir o consumo de energia do computador, mas elas permitem que o computador seja reativado eletronicamente através de um comando a algum dispositivo mantido em estado de espera, como: teclado, *modem*, interface de rede local (conhecido como *Wake-on-LAN* - WOL) ou interface de Barramento Serial Universal (USB, do inglês *Universal Serial Bus*), que, ao ser acionado, coloca todo o computador novamente em atividade. O tempo gasto para colocar e retirar um computador do estado Sobreaviso é menor do que do estado Hibernação, uma vez que não é preciso mover os dados entre a memória RAM e o disco rígido. Por outro lado, um computador em Sobreaviso apresenta maior consumo de energia do que em Hibernação, uma vez que a memória RAM permanece energizada.

O uso dessas estratégias em uma grade computacional oportunista, requer uma avaliação tanto da redução do consumo de energia obtida, quanto do custo associado a essa economia em termos do aumento no tempo de resposta das aplicações (também conhecido por *makespan*) submetidas à grade, ocasionado pelo tempo gasto para colocar e retirar os recursos da grade de tais estados. Isso porque, quando uma nova tarefa é submetida à grade, ela precisará esperar o tempo necessário para que o recurso seja colocado de volta em um estado completamente operacional. Além disso, mostra-se necessário avaliar o número de transições realizadas entre o estado ativo e os estados de baixo consumo de energia, uma vez que alguns componentes do computador, como o disco rígido, são fabricados para tolerar um número máximo de transições durante o seu tempo de vida [15]. Nesse sentido, o uso das estratégias Sobreaviso e Hibernação pode reduzir a vida útil dos recursos da grade computacional, caso elas provoquem um aumento excessivo no número de transições [15, 26] levando os componentes a atingirem o limite de transições antes do tempo estimado.

Ao se utilizar as estratégias de economia de energia, como Sobreaviso e Hibernação, é necessário decidir após quando tempo de inatividade do recurso a estratégia deverá ser utilizada. Esse tempo é denominado *tempo de inatividade* (TI). O TI é o tempo máximo em que um recurso deve permanecer ocioso aguardando a chegada de uma nova requisição, antes que seja colocado em um estado de baixo consumo de energia [4, 2]. Se o intervalo entre requisições de recursos é longo, mostra-se vantajoso transitar o recurso para um estado de baixo consumo de energia tão logo ele se torne ocioso, mas se esse intervalo for pequeno é mais vantajoso que o recurso permaneça ocioso aguardando a chegada de uma nova requisição.

Deste modo, usar um TI pequeno possibilita que o recurso seja adormecido tão logo se torne ocioso, o que pode aumentar a economia de energia. No entanto, esse valor de TI pode aumentar o tempo de reposta das aplicações, pois o recurso precisará ser reativado assim que chegar uma nova requisição. Usar um TI grande faz com que o recurso permaneça no estado de ociosidade durante

mais tempo. Esse tempo no estado ocioso pode reduzir a economia de energia, mas permite que o computador responda instantaneamente a uma nova requisição o que pode reduzir o tempo de resposta das aplicações. Geralmente, busca-se definir um TI que equilibre essas duas possibilidades [2, 4, 1]. Neste artigo avaliamos o uso de diferentes valores de TI associados às estratégias Sobreaviso e Hibernação.

O objetivo deste artigo é avaliar como as estratégias Sobreaviso, Hibernação e os valores de TI utilizados impactam na economia de energia, no tempo de resposta e no número de transições realizadas pelos recursos em uma grade computacional oportunista, identificando cenários em que essas estratégias minimizam o consumo de energia e avaliando o impacto no tempo de resposta. As duas principais contribuições deste artigo são:

- Avaliamos o uso das estratégias Sobreaviso e Hibernação em um domínio administrativo de uma grade computacional oportunista. Os resultados mostram que Sobreaviso e Hibernação reduzem o consumo de energia da infraestrutura em mais de 80% em cenários de baixa contenção de recursos. Essas estratégias têm pequeno impacto no tempo de resposta das aplicações, aumentando o mesmo em no máximo 5% com o uso da estratégia Hibernação e em não mais que 1% com o uso da estratégia de Sobreaviso. Na maior parte dos cenários avaliados, a estratégia de Sobreaviso apresentou economia de energia similar a Hibernação e menor impacto no tempo de resposta das aplicações.

- Avaliamos também o uso de diferentes valores de TI associados às estratégias Sobreaviso e Hibernação. Foram avaliados cinco valores de TI utilizados em outros trabalhos disponíveis na literatura: 0 (sem espera), 300 [12, 18, 26], 600 [20, 5], 900 [8] e 1.200 segundos (temporizador mais agressivo). Os resultados obtidos mostram que quanto maior o valor de TI menor é a economia de energia. Além disso, nos cenários onde há grande contenção pelos recursos da grade, observamos que variar o valor de TI não impacta significativamente no tempo de resposta das aplicações. No entanto, valores de TI maiores resultam em menor número de transições entre o estado ativo e os estados de economia de energia.

As demais seções deste artigo estão organizadas da seguinte forma. Na Seção 2 é apresentado o estado da arte do uso de estratégias de economia de energia em sistemas computacionais. Na Seção 3 é apresentado o modelo de simulação de eventos discretos utilizado; em seguida, na Seção 4 , é descrito o projeto dos experimentos. Os resultados da avaliação do uso das estratégias Sobreaviso e Hibernação, e de diferentes valores de TI em uma grade computacional oportunista são apresentados e analisados na Seção 5. Por fim, na Seção 6 são descritas as conclusões e os trabalhos futuros.

2. ESTADO DA ARTE

Nos últimos anos, diversas abordagens têm sido propostas para analisar e reduzir o consumo de energia em sistemas computacionais, através de melhores projetos do hardware e do software desses sistemas, além do ambiente onde os mesmos estão inseridos. Em hardware, tem-se buscado desenvolver computadores mais econômicos em termos de consumo de energia. Já em software, um caminho é o desenvolvimento de softwares mais otimizados a fim de evitar processamento excessivo [3]. Em relação ao ambiente de implantação, toda a infraestrutura de suporte é considerada, incluindo principalmente o sistema de refrigeração das máquinas.

Esta preocupação com o consumo de energia não é nova. As indústrias de dispositivos móveis e *laptops* têm desenvolvido muitas pesquisas nesse sentido. No entanto, a sua motivação é o fato da energia ser algo bastante escasso (uso de baterias recarregáveis com suporte limitado, por exemplo). Além disso, as técnicas utilizam, entre outras coisas, padrões de comportamento do usuário que nem sempre se aplicam a todos os sistemas, o que invalidaria o uso dessas técnicas em outros cenários. Mesmo com a impossibilidade de reuso destas técnicas, tem-se observado uma melhora na relação entre o consumo de energia e utilização (carga de trabalho) dos computadores ao longo dos tempos. No entanto, esta relação ainda pode ser melhorada [3].

Outro aspecto que pode ser explorado visando a diminuição do consumo de energia é a mudança do estado dos dispositivos dependendo da sua carga de uso. Alguns trabalhos já foram realizados com o objetivo de mensurar os gastos e abordar diferentes estratégias. Talebi et al. [28] reporta práticas que podem ser usadas em salas de aula e laboratórios de pesquisa. São destacadas as práticas: *(i)* não utilizar protetor de tela que possui animações gráficas; *(ii)* configurar o monitor para usar o modo *sleep*, no qual ele passa a consumir menos energia (reativação, por exemplo, com a movimentação do *mouse*); *(iii)* configurar o computador para que o disco rígido entre em modo *sleep*, que implica na redução dos movimentos do disco rígido quando o computador está ocioso; *(iv)* configurar o computador para usar a estratégia Sobreaviso, que consiste em manter os dados na memória RAM e reduzir a atividade dos outros componentes; *(v)* configurar o computador para usar a estratégia Hibernação, em que os dados são armazenados no disco rígido e os demais componentes são desligados, de modo que ao ser ligado novamente, os dados sejam recuperados do disco rígido e

o computador retorne ao estado em que estava.

Ao se utilizar estratégias como Sobreaviso e Hibernação que mudam o estado do recurso, mostra-se necessário definir um temporizador para determinar após quando tempo de inatividade (TI) a estratégia deverá ser usada [4]. Temporizadores têm sido usados na gerência de energia em sistemas operacionais para computadores pessoais [8, 12, 20, 5]. Nesses sistemas, o valor de TI varia entre 5 e 15 minutos. O Condor, *middleware* para grade computacional, utiliza um TI de 2 horas [7].

Um projeto da Escola de Educação da Universidade de Indiana [11] visa implantar um mecanismo para colocar computadores *desktops* em modo *sleep* quando não estiverem em uso, utilizando um temporizador de 2 horas e 15 minutos. Em um projeto piloto, obteve-se redução no consumo de energia em $48,3\%$ para um *cluster* de 11 computadores *desktops* e em $30,9\%$ na ala de escritórios. Essa redução equivale a uma economia de até US$ $500.000,00$ por ano para a universidade. Entretanto, o foco do trabalho é a redução do consumo utilizando o estado *sleep*, sem se preocupar com o tempo e o consumo de energia necessários para colocar e retirar os computadores desse estado. Em uma máquina que está sendo explorada de forma oportunista, onde esse tipo de operação de entrada e saída da máquina da grade pode ser frequente, não é claro que essa estratégia possa trazer economias similares.

Há também estudos que visam investigar estratégias de escalonamento ciente do consumo de energia. Essas estratégias visam minimizar o consumo de energia com o mínimo de impacto no desempenho das aplicações. Sharma e Aggarwal [27] analisam um escalonamento ciente do consumo de energia em grades de *desktops*. No entanto, eles analisam aplicações que fazem uso intensivo de memória, além disso, os *desktops* estão sempre disponíveis para a grade. De outro modo, Lammie, Brenner e Thain [18] analisam estratégias de escalonamento de cargas de trabalho de grades computacionais em *clusters multicores*. O objetivo do escalonamento também é minimizar o consumo de energia e maximizar o desempenho, no entanto, apenas o uso do processador foi considerado. São propostas três técnicas para atingir este objetivo: desativar as máquinas que se encontram subutilizadas, utilizando um temporizador de 300 segundos; prover um escalonamento inteligente das tarefas de modo a minimizar o número de máquinas ativas; e fazer um dimensionamento dinâmico da frequência, de modo a ativar e a desativar as máquinas de acordo com a demanda. Ambos os estudos indicam redução significativa no consumo de energia e pouco impacto no desempenho.

Zong et al. [30] propõem um *framework* para simulação e avaliação da eficiência energética de algoritmos de escalonamento de tarefas que fazem uso intensivo de dados. Para avaliar o *framework*, utilizou-se uma política de escalonamento que consiste em escalonar tarefas para nodos que economizam mais energia (energeticamente mais eficientes). Assim, o escalonador dá prioridade a escalonar tarefas que fazem uso intensivo de dados a recursos mais eficientes para este tipo de tarefa, podendo retirar demais tipos de tarefas desses recursos, reservarem recursos para alocar a tipos específicos de tarefas, e dá preferência por alocar em um mesmo recurso tarefas com dependências entre si, a fim de evitar consumo de energia com carga extra de comunicação. A grade utilizada considera total disponibilidade dos recursos e o escalonamento apenas de aplicações que fazem uso intensivo de dados, o que difere do ambiente e do tipo de tarefas submetidas a grades computacionais oportunistas.

O presente trabalho, diferente dos demais apresentados nesta seção, visa analisar estratégias de economia de energia no contexto de grades computacionais oportunistas. Essas grades apresentam um fator não investigado pelos demais trabalhos em grades computacionais que é aplicar essas estratégias em um cenário onde a disponibilidade das máquinas varia ao longo do tempo. A semelhança com os demais trabalhos está na utilização de estratégias de redução do consumo de energia durante os ciclos de ociosidade.

Este artigo é uma versão estendida do trabalho de Ponciano et al. [24]. Ponciano et al. avalia as estratégias Sobreaviso e Hibernação por meio de um estudo de caso realizado com um rastro da grade computacional OurGrid [6]. Os resultados mostram que ambas as estratégias possibilitam economizar energia na infraestrutura e impactam o tempo de resposta das tarefas. As estratégias são avaliadas em apenas dois cenários de contenção: alta e baixa. Ponciano et al. não avalia o uso de TI para decidir quando as estratégias devem ser utilizadas. Este trabalho avalia o uso das estratégias Sobreaviso e Hibernação em diversos cenários de contenção de recursos [24] e em conjunto com diferentes valores para TI. Além disso, este trabalho utiliza duas novas métricas: tempo de resposta das aplicações e número de transições realizadas pelas máquinas.

3. Estratégias para economia de energia em Grades Oportunistas

Nossa avaliação do uso de estratégias de economia de energia em grades computacionais oportunistas utiliza um modelo simulado para avaliar o impacto das estratégias Sobreaviso, Hibernação e diferentes valores de TI na economia de energia da infraestrutura, tempo de resposta das aplicações e número de transições realizadas pelas máquinas.

Nosso modelo de grade computacional baseia-se no *middleware* OurGrid [6]. A grade é composta por três componentes: *broker*, *workers* e *peer*. O *broker* é o com-

```
J1−1  1050208880   218
J2−1  1050208892   1107
J2−2  1050208892   723
```

(a) Demanda

```
V4−linux  1104904800   1636
V2−linux  1104904800   3611
V4−linux  1104914800   3666
```

(b) Disponibilidade

```
V2−linux  110700
V3−linux  177120
V4−linux  114810
```

(c) Processamento

Figura 1. Exemplos dos rastros utilizados nas simulações

ponente da grade que provê uma interface para que os usuários submetam aplicações para serem executadas nos recursos da grade. Outra função do *broker* é escalonar as tarefas das aplicações para os *workers* disponíveis. Os *workers* são agentes da grade que executam nos recursos. Os workers recebem e executam tarefas escalonadas pelos *brokers*. Os *workers* também monitoram e reportam o estado do recurso ao *peer*, a fim de informar quando ele está ou não disponível para executar uma tarefa. O *peer* é o componente responsável por gerenciar os recursos de um domínio administrativo e por se comunicar com outros *peers* da grade a fim de obter ou doar *workers*.

Por simplicidade, nosso modelo de simulação não considera o mecanismo de priorização utilizado pelo OurGrid para doação de recursos entre diversos *peers* quando a grade se encontra em alta contenção. Deste modo, nosso modelo de simulação foca em apenas um domínio administrativo (ou *site*) da grade, i.e., há apenas um *peer* e todas as máquinas da grade são gerenciadas por ele. Do ponto de vista de nossa avaliação, essa simplificação não tem um impacto grande nos resultados, haja vista que as decisões para aumentar a eficiência da grade são tomadas de forma autônoma por cada domínio administrativo.

O modelo de simulação é de eventos discretos guiados por rastros. O simulador recebe dois rastros como entrada: um rastro de submissão de aplicações e um rastro que descreve a variação na disponibilidade das máquinas para a grade. A Figura 1(a) apresenta um modelo do rastro de submissão de aplicações. O rastro contém uma tarefa em cada linha. As tarefas são constituídas de um identificador que indica o código da aplicação e o código da tarefa, um instante de submissão, e um valor estimado do tempo total de execução, respectivamente.

O rastro de variação na disponibilidade das máquinas para a grade descreve o oportunismo da grade, em que um recurso só é utilizado pela grade quando o usuário local não está utilizando-o. Um exemplo de rastro de disponibilidade das máquinas é apresentado na Figura 1(b). Em cada linha, tem-se uma máquina. Cada máquina é constituída, respectivamente, pelo instante de tempo em que ela se torna disponível para a grade e por quanto tempo ficou disponível. Um arquivo de configuração define o número de ciclos por segundo que cada máquina é capaz de executar, como mostrado na Figura 1(c). O processamento de uma tarefa da grade é interrompido se o recurso que está executando a tarefa for requisitado pelo usuário local. Nesse caso, o modelo simulação considera a implementação de *checkpoint*, de modo que, se uma máquina tornar-se indisponível durante a execução de determinada tarefa, todo o processamento já realizado é salvo e a tarefa é submetida novamente quando houver uma máquina disponível e ela reinicia a execução a partir do ponto em que foi interrompida. Neste trabalho o uso de *checkpoints* visa principalmente agilizar o tempo de simulação.

No momento em que uma máquina que se encontra disponível para a grade fica ociosa, i.e., não há tarefa da grade para ser executada, é inicializado um temporizador (TI) que define o tempo máximo em que a máquina permanecerá ociosa aguardando a chegada de uma nova tarefa. Durante esse tempo, se uma tarefa for submetida, a máquina pode iniciar a execução imediatamente. De outro modo, se o temporizador expirar sem que nenhuma tarefa seja submetida, a máquina transita para um estado de economia de energia, que pode ser Sobreaviso ou Hibernação. Esses estados são utilizados apenas nas máquinas que executam um agente *worker*.

Uma máquina em Sobreaviso opera em uma potência P_s e em Hibernação ela opera em uma potência P_h, esses estados podem reduzir o consumo de energia da máquina uma vez que essas potências são menores do que a potência P_i em que a máquina opera quando está ociosa. A potência P_i é menor que P_o, potência em que a máquina opera quando está executando uma aplicação. A redução da potência provida pelos estados Sobreaviso e Hibernação ocorre em razão da desativação de alguns componentes. De modo geral, P_s corresponde à potência da memória RAM, e outros dispositivos utilizados para acordar a máquina via WoL, teclado ou *mouse*. Por sua vez, a potência P_h corresponde apenas à potência dos dispositivos utilizados no WoL. Dado que a máquina permanece em um estado de economia de energia (v) durante Δt unidades de tempo a economia gerada em relação ao estado ocioso é dado por $\xi = (P_i - P_v) * \Delta t$, onde v pode ser Sobreaviso (s) ou Hibernação (h).

É necessário um tempo para que uma máquina transite completamente entre um estado de economia de energia e um estado ativo. Esse tempo é definido como a latência do estado. Durante essa latência a máquina é considerada ativa e, portanto, ela consome uma potência correspon-

dente ao estado ativo. No entanto, durante a latência, a máquina não pode ser utilizada para executar uma tarefa para grade, mas tão logo a transição tenha sido completada a máquina pode ser acordada ou uma tarefa pode ser escalonada para ser executada por ela. A latência do estado ocioso é desprezível ($L_i = 0$). Em Sobreaviso a latência (L_s) corresponde ao tempo gasto para ativar ou desativar alguns componentes, como o disco rígido e o processador. A latência do estado Hibernação (L_h) é maior que a do estado Sobreaviso, uma vez que envolve movimentar dados entre a memória RAM e o disco rígido, além de desativar os componentes.

Deste modo, o uso de Sobreaviso e Hibernação implica em um compromisso entre o benefício em termos de redução da potência provida pelo estado ($P_h < P_s < P_i$) e um custo associado em termos do aumento da latência para acordar a máquina ($L_h > L_s > L_i$). Um problema gerado pela latência é que, caso uma nova tarefa seja submetida à grade, será necessário aguardar esse tempo até que a máquina seja acordada e a tarefa possa iniciar a execução.

A cada instante, um recurso da grade pode estar em um de 4 estados possíveis (Figura 2): *(i)* `Ocioso`, caso não exista uma tarefa da grade alocada e nenhuma estratégia de economia de energia está sendo utilizada; *(ii)* `Grade`, caso esteja executando alguma tarefa da grade; *(iii)* `Usuário`, caso o usuário a esteja utilizando, neste caso o recurso não está disponível para a grade; *(iv)* ou em um estado de `economia de energia`, que pode ser uma das duas estratégias consideradas: `Sobreaviso` ou `Hibernação`. A Figura 2 apresenta um diagrama com os estados em que as máquinas da grade podem estar. Para facilitar a compreensão, esse diagrama apresenta exemplos de valores típicos do consumo de energia e do tempo gasto na transição de estados referentes aos estados Sobreaviso, Hibernação e Ocioso.

Para avaliar o uso das estratégias Sobreaviso e Hibernação em grades computacionais oportunistas são consideradas três métricas: *economia de energia*, *tempo de resposta* e *número de transições*. Para medir a economia de energia provida por Sobreaviso e Hibernação em relação ao estado Ocioso, calculamos o consumo de energia de todos os recursos durante o tempo do experimento. O consumo de energia de um recurso é dado pela Equação 1, onde se têm o tempo que o recurso permaneceu (t) e a potência em que operou (p) em cada uma das seguintes situações: (i) fazendo transição de estado (m); (ii) em estado de economia de energia (e) ou (iii) executando uma tarefa na grade (g). O estado Usuário e as transições para ele não são considerados no cálculo do custo energético da grade. A economia de energia é a diferença entre a energia consumida pelos recursos em uma configuração em que as máquinas são mantidas ociosas e a mesma configuração, mas utilizando uma estratégia de economia de energia.

$$C = t_m \times p_m + t_e \times p_e + t_g \times p_g \quad (1)$$

O número de transições realizadas por cada máquina da grade consiste na contagem do número de transições realizadas entre qualquer estado da grade e os estados de economia de energia. Esse cálculo considera, inclusive, as transições realizadas entre os estados de economia de energia e o estado Usuário.

Por fim, consideramos o impacto das estratégias no tempo de resposta das aplicações. O tempo de resposta de uma aplicação do tipo saco-de-tarefas é o tempo decorrido entre a submissão da aplicação e o tempo de término da execução da última tarefa da aplicação. Para avaliar o atraso no tempo de resposta gerado pelo uso das estratégias Sobreaviso e Hibernação em relação à configuração em que as máquinas são mantidas no estado Ocioso, utilizou-se a métrica *slowdown*.

O *slowdown* é a razão entre o tempo de resposta em uma configuração que utiliza uma estratégia de economia de energia e o tempo de resposta em uma configuração em que as máquinas são mantidas ociosas. Quando o *slowdown* é menor que 1, tem-se um ganho no tempo de resposta, de outro modo, quando o *slowdown* é maior que 1 ocorre um atraso.

De um modo geral, a dinâmica da simulação de eventos discretos funciona como segue. Quando uma nova aplicação é submetida à grade, o *peer* escolhe aleatoriamente, no conjunto de máquinas disponíveis, um subconjunto de máquinas para escalonar todas as tarefas da aplicação. Caso esse subconjunto de máquinas não seja suficiente para suprir toda a demanda da aplicação, o *peer* escolhe aleatoriamente entre as máquinas que se encontram em Sobreaviso ou Hibernação quais devem ser reativadas para atender a essa demanda. Uma vez escolhidas as máquinas, o *peer* fornece-as ao *broker* que submeteu a aplicação. Quando o *broker* recebe as máquinas, ele escalona as tarefas por ordem de criação para serem executadas nas máquinas ordenadas por ordem de chegada (FCFS, do inglês *first-come first-served*). Quando uma máquina termina de executar a tarefa ela inicia um temporizador (TI). Durante o tempo do temporizador ela aguarda a chegada de uma nova tarefa para ser executada. Caso o temporizador expire e nenhuma tarefa tenha sido submetida, a máquina transita para um estado de economia de energia. A qualquer instante da simulação uma máquina pode ser preemptada pelo usuário. A simulação termina quando o rastro de submissão de aplicações termina. Caso o rastro de variação na disponibilidade das máquinas termine antes que o rastro de submissão de aplicações, o rastro de variação na disponibilidade das máquinas retorna ao início, com os instantes de mudanças de estados atualizados. O simulador calcula o tempo de

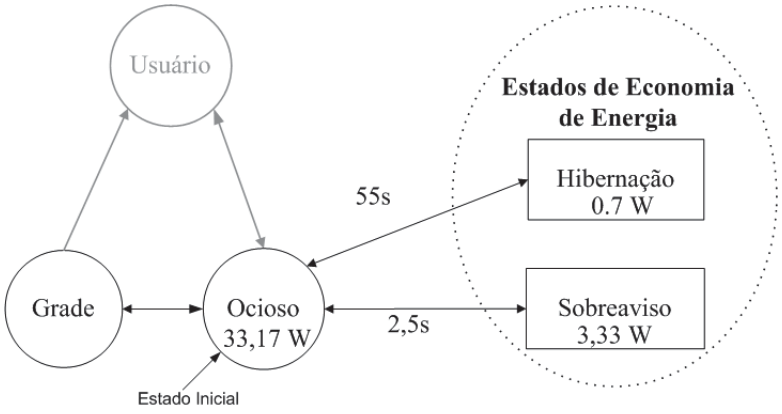


Figura 2. Estados das Máquinas

resposta de cada tarefa e de cada aplicação, o número de transições realizadas, o tempo gasto realizando transições e a energia consumida por cada máquina da grade.

4. PROJETO DOS EXPERIMENTOS

Nesta seção apresentamos o projeto dos experimentos, descrevemos os rastros, arquivos de configuração e os cenários avaliados. Para representar a variação da disponibilidade das máquinas ao longo do tempo, utilizou-se um rastro de *desktop* que apresenta dados sobre o uso das máquinas, identificando os períodos em que as máquinas estão ociosas (em todos os rastros utilizados neste trabalho, as informações referentes a tempo são medidas em segundos). O rastro utilizado foi o DEUG [17], que possui um período não contínuo de 1 mês, com informações sobre máquinas *desktop* da Universidade de París-Sud. Como as informações sobre disponibilidade nesse rastro não formam um período contínuo, foi feito um tratamento dos dados para gerar um novo rastro sem interrupções. O tratamento consistiu na replicação de períodos similares para os períodos em que não havia informações.

Do rastro DEUG também foram obtidas informações sobre o poder de processamento das máquinas. Nesse rastro, o poder de processamento é descrito pelo número de operações por segundo que a máquina pode executar. Há informações desse valor para as máquinas ao longo do tempo, e o valor varia em função da utilização da máquina no momento em que a medição foi realizada. Deste modo, utilizamos o maior desses valores para representar a capacidade de processamento de cada máquina, como apresentado na Figura 1(c).

Para estimar a frequência dos processadores, utilizamos a relação: $1,5$ GHz equivalente a 110.700 operações por segundo (como apresentado em Kondo et al. [17]). O tempo de execução das tarefas no rastro

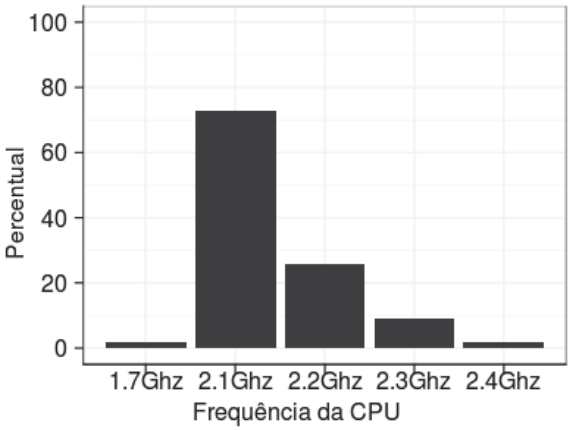


Figura 3. Histograma das frequências de CPU das máquinas utilizadas nas simulações

equivale à execução na máquina mais rápida da amostra (2.4 GHz). A variação da frequência da CPU da amostra de máquinas utilizada nas simulações é descrita pelo histograma apresentado na Figura 3.

Avaliamos valores de TI de 0, 300, 600, 900 e 1.200 segundos. Quando um tempo igual ao valor de TI passa sem que nenhuma tarefa seja submetida, a máquina inicia a transição para um estado de economia de energia. Como mencionado anteriormente, avaliamos dois estados de economia de energia: Sobreaviso e Hibernação. Estes estados são comparados com o estado ocioso, em que não é utilizada uma estratégia de economia de energia. O tempo de transição e o consumo de energia no estado de economia de energia são dados pela estratégia utilizada. A Tabela 1 apresenta os valores considerados nas simulações.

O valor de referência para o consumo de energia das máquinas quando estão executando uma tarefa foram obtidos com base na lista de máquinas avaliadas pela En-

Tabela 1. Parâmetros utilizados nas Simulações

Estratégia	Tempo de transição	Custo de energia
Ocioso	0 s	33, 17 W
Sobreaviso	2, 5 s	3, 33 W
Hibernação	55 s	0.7 W

ergyStar [10]. Procuramos os valores das máquinas que possuíam uma mesma configuração geral do sistema, mas com diferentes frequências de CPU. Os tempos de transição de estados são os mesmos utilizados por Orgerie et al. [22]. Durante o período de transição, a máquina opera em sua potência máxima [21].

Para simular a demanda de aplicações na grade, utilizamos um rastro de submissão de aplicações gerado pelo modelo de aplicações do tipo saco-de-tarefas proposto por Iosup et al. [14]. Saco-de-tarefas são aplicações formadas por tarefas intendentes, i.e., que não se comunicam entre si. Esse é o tipo de aplicação mais comum em grades computacionais oportunistas. Utilizamos aplicações submetidas ao longo de dois dias e as tarefas cujo tempo de execução não ultrapassava 35 minutos, tempo máximo de tarefas comuns em grades computacionais oportunistas [16]. Variamos o número de máquinas de 10 a 100 de modo a gerar cenários de baixa e alta contenção de recursos, ou seja, alguns cenários em que as máquinas permanecem ociosas durante muito tempo e outros em que as máquinas permanecem ociosas durante pouco tempo. Utilizamos o limite de 100 máquinas, que garante que nenhuma máquina permanece ociosa durante toda simulação. Para cada configuração foram executadas 30 simulações com diferentes rastros de demanda de modo a obter um resultado com confiança estatística de 95%.

Todos os rastros de submissão de aplicações, de variação na disponibilidade das máquinas e os arquivos de configuração utilizados nas simulações realizadas neste trabalho estão disponíveis na página: `http://redmine.lsd.ufcg.edu.br/projects/list_files/green-grid`.

5. Apresentação e Análise dos Resultados

Nesta seção apresentamos os resultados da avaliação do uso das estratégias Sobreaviso e Hibernação, com diferentes valores de TI, em grades computacionais oportunistas. Avaliamos o impacto dessas estratégias nas métricas economia de energia da infraestrutura, número de transições das máquinas e slowdown das aplicações.

A Figura 4 apresenta a economia de energia e o *slowdown* provida pelas estratégias Sobreaviso e Hibernação quando são utilizados diferentes valores de TI. As Figuras 4(a) e 4(b) mostram que os diferentes valores de TI impactam de forma semelhante na economia de energia provida por Sobreaviso e por Hibernação. Quando maior o valor de TI menor é a economia de energia. Isso é ocasionado pelo aumento do tempo em que cada máquina permanece no estado ocioso aguardando a chegada de uma nova tarefa, como mostra a Figura 5, o resultado é semelhante para a estratégia Sobreaviso. Quando a grade se encontra em alta contenção as máquinas permanecem pouco tempo executando o temporizador. Esse tempo aumenta à medida que a contenção da grade é reduzida.

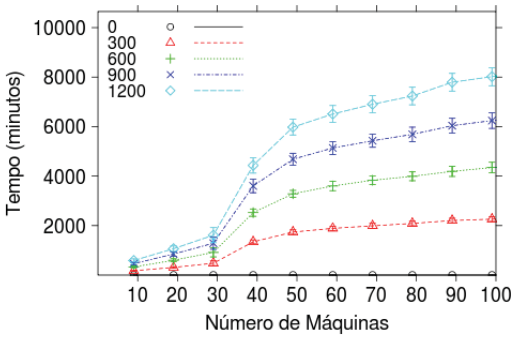
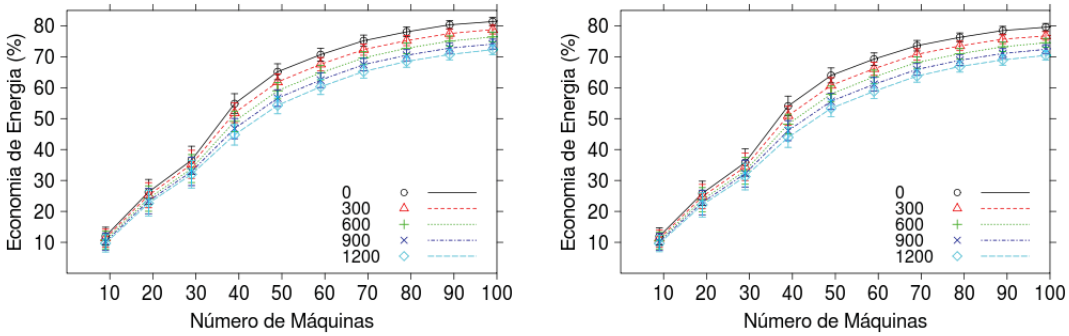


Figura 5. Total de tempo em que as máquinas permaneceram executando o temporizador com a estratégia Hibernação

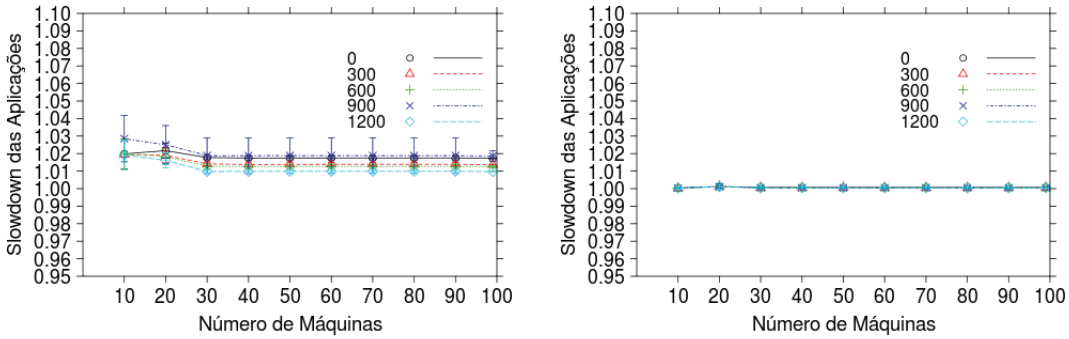
As Figuras 4(c) e 4(d), mostram que as estratégias Sobreaviso, Hibernação e o uso de diferentes valores para TI não têm impacto significativo no tempo de resposta das aplicações. Isso porque o impacto que as estratégias geram no tempo de resposta é referente ao tempo gasto para acordar a máquina do estado de economia de energia, que é de 2, 5 segundos em Sobreaviso e 55 segundos em Hibernação. Esse tempo é muito pequeno comparado ao tempo de execução de aplicações do tipo saco-de-tarefas em grades computacionais oportunistas, que pode chegar a aproximadamente 35 minutos [16]. O maior *slowdown* observado foi de aproximadamente 1, 04, gerado pela estratégia Hibernação em um cenário de alta contenção (10 máquinas na grade). Esse *slowdown* equivale a um aumento de 4% no tempo de resposta das aplicações. Sobreaviso gera baixo *slowdown* independente da contenção da grade e do temporizador utilizado, o maior *slowdown* obtido foi de 1.01, que equivale a um aumento de 1% no tempo de resposta das aplicações.

A Figura 6 apresenta o número médio de transições realizadas pelas máquinas da grade para os estados de economia de energia. Esse resultado corresponde à simulação de dois dias de operação da grade computacional. O maior número de transições foi obtido na configuração com 50 máquinas utilizando a estratégia Hibernação e TI igual a 0. Nessa configuração cada máquina realizou em média 15 transições, o que equivale a aproximadamente 8



(a) Economia de Energia obtida com Hibernação

(b) Economia de Energia obtida com Sobreaviso



(c) Slowdown obtido com Hibernação

(d) Slowdown obtido com Sobreaviso

Figura 4. Slowdown e Economia de Energia obtidos com as estratégias Sobreaviso e Hibernação com valores de TI definidos como: 0, 300, 600 e 900 segundos.

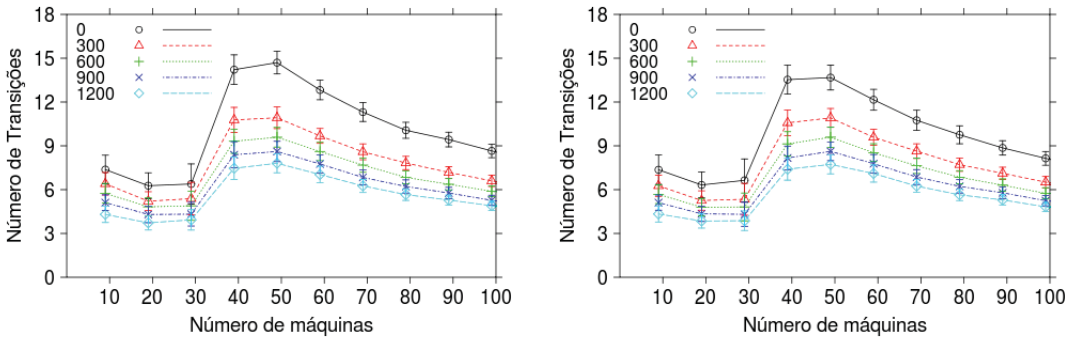
transições por dia. Esse resultado é semelhante ao obtido por Reich et al. [26], em que as máquinas realizam em média 7 transições por dia. Esse número de transições não reduz a vida útil, por exemplo, de discos rígidos SATA que podem tolerar até 500.000 transições de estado em 5 anos [15], i.e, aproximadamente 273 transições por dia.

Pela Figura 6 também se pode notar que valores de TI maiores geram menor número de transições, portanto, eles podem ser aplicados em cenários em que reduzir o número de transições for um objetivo. Pode-se observar também que quando o número de máquinas na grade é menor que 40 as máquinas realizam poucas transições, uma vez que permanecem mais tempo ocupadas executando o temporizador ou tarefas da grade. Quando o número de máquinas é entre 50 e 60 aumentam-se os períodos de ociosidade de modo que, com valor de TI igual a 0, há um aumento superior a 100% no número de transições realizadas. Por fim, quando o número de máquinas torna-se maior que 60, o número de transições tende a reduzir, uma vez que o número de máquinas na grade torna-se grande o suficiente para que nem todas as máquinas precisem ser acordadas sempre que surge uma nova demanda.

De um modo geral, as estratégias Sobreaviso e Hibernação apresentaram significante economia de energia em relação ao estado Ocioso. Não há diferença significativa na economia de energia gerada por essas estratégias. No entanto a estratégia Sobreaviso resultou em um menor impacto no tempo de resposta das aplicações quando comparado à estratégia Hibernação. Pode-se concluir, também, que o valor de TI igual a 0 é mais adequado aos contextos de contenção de recursos avaliados. Esse valor de TI aumenta pouco o número de transições, resulta em maior economia de energia e não impacta de modo significativo o tempo de resposta das aplicações.

6. Conclusões e Trabalhos Futuros

Neste trabalho avaliamos o impacto que as estratégias Sobreaviso, Hibernação e diferentes valores de TI têm na redução do custo de energia, no tempo de resposta das aplicações e no número de transições realizadas pelas máquinas em uma grade computacional oportunista sujeita a essas estratégias de economia de energia. Os resultados mostram que Sobreaviso e Hibernação reduzem



(a) Hibernação

(b) Sobreaviso

Figura 6. Número médio de transições para os estados Sobreaviso e Hibernação

o consumo de energia da infraestrutura em mais de 80% em cenários de baixa contenção de recursos. Essas estratégias têm pequeno impacto no tempo de resposta das aplicações, com um aumento de até 5% com o uso da estratégia Hibernação e um aumento menor que 1% com o uso da estratégia Sobreaviso. Na maior parte dos cenários avaliados, Sobreaviso apresentou economia de energia similar a Hibernação e menor impacto no tempo de resposta das aplicações.

Analisamos o uso de diferentes valores de TI associados às estratégias Sobreaviso e Hibernação. Foram avaliados cinco valores utilizados em outros trabalhos: 0 (sem espera), 300 [12, 18, 26], 600 [20, 5], 900 [8] e 1.200 segundos (temporizador mais agressivo). Os resultados obtidos mostram que quanto maior é o valor de TI, menor é a economia de energia. Além disso, nos cenários de contenção de recursos que foram avaliados, observamos que variar o valor de TI não impacta significativamente no tempo de resposta das aplicações. No entanto, valores de TI maiores resultam em menor número de transições entre o estado ativo e os estados de economia de energia.

Como trabalhos futuros, pode-se analisar a sensibilidade dos tempos de transição e das potências de Sobreaviso e Hibernação a fim de verificar o impacto da variação de seus valores no tempo de resposta, economia de energia e número de transições realizadas pelas máquinas em uma grade computacional oportunista. Além disso, mostra-se necessário investigar novas estratégias para escolher que subconjunto de máquinas deve ser reativado para executar as tarefas de uma aplicação quando o número de máquinas demandas é menor que o número de máquinas que se encontram em um estado de economia de energia. Alguns esforços foram dedicados nesse sentido [23], mas ainda há lacunas a serem exploradas. As estratégias de economia de energia analisadas neste artigo serão implantadas na grade computacional oportunista da Universidade Federal de Campina Grande (GridUFCG), que agregará mais de 1.000 *desktops* executando o *middleware* OurGrid [6] (http://www.ourgrid.org/).

Referências

[1] Susanne Albers. Energy-efficient algorithms. *Communications of the ACM*, 53:86–96, May 2010.

[2] John Augustine, Sandy Irani, and Chaitanya Swamy. Optimal power-down strategies. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 530–539, Washington, DC, USA, 2004. IEEE Computer Society.

[3] Luiz André Barroso and Urs Hölzle. The case for energy-proportional computing. *Computer*, 40:33–37, December 2007.

[4] Luca Benini, Alessandro Bogliolo, and Giovanni De Micheli. A survey of design techniques for system-level dynamic power management. In *Readings in hardware/software co-design*, pages 231–248, Norwell, MA, USA, 2002. Kluwer Academic Publishers.

[5] Canonical Ltd. Power management in ubuntu. Disponível em `https://wiki.ubuntu.com/power-management-in-Ubuntu` Último acesso em dezembro 2010.

[6] 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.

[7] Condor Project. Condor version 7.4.4, 2010. Disponível em: `http://www.cs.wisc.edu/condor/`. Último acesso em dezembro de 2010.

[8] Hewlett-Packard Corporation, Intel Corporation, Microsoft Corporation, Phoenix Technologies Ltd., and Toshiba Corporation. Advanced configuration and power interface specificatio, 2010. Disponível em: `http://www.acpi.info/spec.htm`. Último acesso em janeiro de 2010.

[9] The Economist. Going green. *The Economist*, Mar 2007.

[10] Energy Star. Computer-desktops & integrated computers qp list, 2009. Disponível em: `www.energystar.gov/ia/products/prod_lists/computers_prod_list.xls`. Último acesso em: setembro de 2009.

[11] Indiana University. Green computing project points to potential for energy savings, 2009. Disponível em: `http://newsinfo.iu.edu/news/page/normal/11142.html`. Último acesso em: acesso em agosto de 2009.

[12] Intel and U.S. Environmental Protection Agency. Energy star* system implementation whitepaper. Disponível em `www.intel.com/cd/channel/reseller/asmo-na/eng/339085.htm` Último acesso em dezembro 2010.

[13] Alexandru Iosup, Catalin Dumitrescu, Dick Epema, Hui Li, and Lex Wolters. How are real grids used? the analysis of four grid traces and its implications. In *Proceedings of the 7th IEEE/ACM International Conference on Grid Computing, 2006*, pages 262–269, Washington, DC, USA, 2006. IEEE Computer Society.

[14] Alexandru Iosup, Ozan Sonmez, Shanny Anoep, and Dick Epema. The performance of bags-of-tasks in large-scale distributed systems. In *Proceedings of the 17th international symposium on High performance distributed computing*, pages 97–108, New York, NY, USA, 2008. ACM.

[15] Rini T Kaushik, Milind Bhandarkar, and Klara Nahrstedt. Evaluation and analysis of greenhdfs: A self-adaptive, energy-conserving variant of the hadoop distributed file system. In *CloudCom 2010 : Proceedings of the 2th IEEE International Conference on Cloud Computing*, pages 1–12. IEEE Computer Society, 2010.

[16] Derrick Kondo, Andrew Chien, and Henri Casanova. Scheduling task parallel applications for rapid turnaround on enterprise desktop grids. *Journal of Grid Computing*, 5:379–405, 2007.

[17] Derrick Kondo, Gilles Fedak, Franck Cappello, Andrew A. Chien, and Henri Casanova. Characterizing resource availability in enterprise desktop grids. *Future Generation Computer Systems*, 23(7):888–903, 2007.

[18] M. Lammie, P. Brenner, and D. Thain. Scheduling grid workloads on multicore clusters to minimize energy and maximize performance. In *Proceedings of the 10th IEEE/ACM International Conference on Grid Computing, 2009*, pages 145 –152, october 2009.

[19] David Meisner, Brian T. Gold, and Thomas F. Wenisch. Powernap: eliminating server idle power. In *ASPLOS '09: Proceeding of the 14th international conference on Architectural support for programming languages and operating systems*, pages 205–216, New York, NY, USA, 2009. ACM.

[20] Microsoft Corporation. Windows power management. isponível em `http://www.microsoft.com/whdc/archive/winpowmgmt.mspx` Último acesso em dezembro 2010.

[21] A. C. Orgerie, L. Lefevre, and J. P. Gelas. Chasing gaps between bursts: Towards energy efficient large scale experimental grids. In *Parallel and Distributed Computing, Applications and Technologies, 2008. PDCAT 2008. Ninth International Conference on*, 2008.

[22] Anne C. Orgerie, Laurent Lefèvre, and Jean P. Gelas. Save watts in your grid: Green strategies for energy-aware framework in large scale distributed systems. *Parallel and Distributed Systems, International Conference on*, 0:171–178, 2008.

[23] Lesandro Ponciano and Francisco Brasileiro. On the impact of energy-saving strategies in opportunistic grids. In *Energy Efficient Grids, Clouds and Clusters Workshop, proceedings of the 11th ACM-IEEE International Conference on Grid Computing (Grid 2010),*, pages 282 – 289, Bruxelas, Bélgica, 2010. ACM-IEEE.

[24] Lesandro Ponciano, Jaindson Santana, Marcus Carvalho, Matheus Gaudencio, and Francisco Brasileiro. Analise de estrategias de computacao verde em grades computacionais oportunistas. In *Anais do XXVIII Simposio Brasileiro de Redes de Computadores e Sistemas Distribuidos*, pages 307–320, Porto Alegre, Brasil, may 2010. SBC.

[25] David Przybyla and Mahmoud Pegah. Dealing with the veiled devil: eco-responsible computing strategy. In *SIGUCCS '07: Proceedings of the 35th an-*

nual ACM SIGUCCS conference on User services, pages 296–301, New York, NY, USA, 2007. ACM.

[26] Joshua Reich, Aman Kansal, Michel Gorackzo, and Jitendra Padhye. Sleepless in seattle no longer. In *USENIX ATC'10: USENIX Annual Technical Conference*, 2010.

[27] Kamal Sharma and Sanjeev Aggarwal. Energy aware scheduling on desktop grid environment with static performance prediction. In *SpringSim '09: Proceedings of the 2009 Spring Simulation Multiconference*, pages 1–8, San Diego, CA, USA, 2009. Society for Computer Simulation International.

[28] Mujtaba Talebi and Thomas Way. Methods, metrics and motivation for a green computer science program. *SIGCSE Bull.*, 41(1):362–366, 2009.

[29] Joseph Williams and Lewis Curtis. Green: The new computing coat of arms? *IT Professional*, 10(1):12–16, 2008.

[30] Ziliang Zong, Xiao Qin, Xiaojun Ruan, Kiranmai Bellam, Yiming Yang, and Adam Manzanares. A simulation framework for energy efficient data grids. In *WSC '07: Proceedings of the 39th conference on Winter simulation*, pages 1417–1423, Piscataway, NJ, USA, 2007. IEEE Press.

Comunicação de Dados baseada no Receptor para Redes de Sensores Sem Fio

Max do Val Machado[1] **Raquel A. F. Mini**[2] **Antonio A. F. Loureiro**[1]

[1]Departamento de Ciência da Computação
Universidade Federal de Minas Gerais
{maxm, loureiro}@dcc.ufmg.br

[2]Departamento de Ciência da Computação
Pontifícia Universidade Católica de Minas Gerais
raquelmini@pucminas.br

Abstract

This work proposes the Receiver-based Medium Access Control (Rb-MAC) protocol to be used in cross-layer designs for the receiver-based communication. These models are appropriated for Wireless Sensor Networks deal with energy constrains and dynamic topology. Simulation results reveal that using Rb-MAC, the evaluated receiver-based model increases the delivery ratio, reduces the number of transmissions and energy consumption when compared with a sender-based model. Moreover, Rb-MAC allows a dynamic duty cycle and maintains the delivery ratio for reduced duty cycle values.

Keywords: Wireless sensor networks, data communication, receiver-based, MAC, routing, duty cycle

Resumo

Este trabalho apresenta o protocolo Receiver-based Medium Access Control (Rb-MAC) para ser utilizado em projetos integrados para a comunicação baseada no receptor. Esses modelos são apropriados para as Redes de Sensores Sem Fio por lidarem com restrições de energia e topologia dinâmica. Resultados de simulação mostram que usando o Rb-MAC, o modelo receptor avaliado aumenta a taxa de entrega, reduz o número de transmissões e o consumo de energia quando comparado com um modelo baseado no emissor. Além disso, o Rb-MAC permite um ciclo de trabalho dinâmico e mantém a taxa de entrega para valores reduzidos de ciclo de trabalho.

Palavras-chave: Redes de sensores sem fio, comunicação de dados, receptor, MAC, roteamento, ciclo de trabalho

1. INTRODUÇÃO

O principal desafio para transformar as Redes de Sensores Sem Fio (RSSFs) [1] na tecnologia de sensoriamento do futuro é projetar soluções eficientes em termos de energia. Outro desafio crucial é a topologia dinâmica dessas redes que geralmente é consequência dos nós adormecerem para economizar energia. Nesse contexto, a comunicação de dados vem recebendo atenção especial dos pesquisadores porque, normalmente, corresponde à tarefa com o maior custo de energia nas RSSFs e trata das mudanças topológicas. As camadas de rede e de acesso ao meio (MAC) são responsáveis pela comunicação em RSSFs.

A camada de rede é responsável pela escolha sistemática do próximo nó durante o processo de roteamento, o que leva à definição de rotas. Essa escolha, chamada de decisão de propagação, é baseada no emissor ou no receptor. Na primeira, quando um nó recebe um pacote não destinado a ele, o nó verifica se ele foi o escolhido para continuar o roteamento. Se sim, o nó escolhe um subconjunto de vizinhos para continuar o roteamento e, depois, propaga o pacote. Na outra abordagem, quando um nó recebe um pacote não destinado a ele, o nó decide localmente se deve ou não propagar o pacote.

A camada MAC é responsável por entregar o pacote para o próximo nó e, por isso, ela gerencia o uso do canal para evitar/tratar colisões e controla o ciclo de trabalho dos nós para tratar da comunicação quando o próximo nó estiver adormecido. O ciclo de trabalho é a porcentagem de tempo em que um nó permanece acordado durante um

ciclo de vida. O tempo de vida dos nós pode ser dividido em ciclos de vida sendo que cada um desses ciclos é composto por períodos de tempo em que o nó permanece acordado e por períodos em que ele dorme. O término de um ciclo de vida implica no início do próximo. Assim, quando o ciclo de trabalho de um nó é, por exemplo, 10%, o nó fica acordado durante 10% do tempo de cada ciclo de vida. A tarefa de controle do ciclo de trabalho é crucial em RSSFs, pois a melhor forma de economizar energia nessas redes é desativar os nós sempre que possível [3]. Dada a importância da energia em RSSFs, o principal desafio da camada MAC nessas redes é determinar o escalonamento de dormir/acordar dos nós para que o roteamento não seja prejudicado quando o próximo nó estiver adormecido.

Uma característica dos protocolos propostos para a camada MAC das RSSFs é que eles são focados no roteamento baseado no emissor. Isso significa que quando o protocolo MAC recebe um pacote a ser propagado, o próximo destino do pacote foi escolhido pela camada de rede e cabe à MAC entregar o pacote quando o escolhido estiver acordado. A maioria dos protocolos MAC faz com que nós vizinhos compartilhem os seus respectivos escalonamentos de dormir/acordar para que quando um nó enviar um pacote, ele saiba quando o vizinho escolhido estará acordado. A limitação dessa técnica é que o ambiente das RSSFs é normalmente dinâmico e os nós possuem restrições de hardware que levam a desatualização dessas informações. Normalmente, o compartilhamento é atualizado periodicamente através da troca de pacotes de controle (custo de energia). Além disso, o custo-benefício das informações compartilhadas é minimizado pelo fato das RSSFs normalmente possuírem longos períodos de inatividade em que nenhum dado é roteado e as informações sobre vizinhos são atualizadas. Alguns protocolos não compartilham tais informações e para um nó emissor descobrir quando o nó escolhido estará acordado, ele envia vários pacotes de controle. A limitação dessa técnica é o custo de energia com o envio desses pacotes. Dadas as limitações das técnicas existentes, novas soluções para a comunicação quando o próximo nó estiver adormecido devem ser investigadas.

Os modelos de comunicação baseados no receptor podem ser investigados como outra solução para o desafio proposto. O roteamento baseado no receptor é mais robusto às falhas no próximo nó que o baseado no emissor, pois qualquer vizinho que recebe o pacote pode se escolher como o próximo nó. No baseado no emissor, apenas o vizinho escolhido pode ser o próximo nó. O fato da decisão de propagação ser tomada pelo nó receptor faz com que a informação sobre vizinhos seja normalmente desnecessária, o que elimina o seu custo de atualização. Outro ponto relevante é que no roteamento baseado no receptor, o número de opções para o próximo nó é maior que o do baseado no emissor e, por isso, a estratégia de retransmissão para descobrir vizinhos acordados pode ser eficiente em termos de energia. Além disso, como o objetivo da camada MAC é entregar pacotes para o próximo nó, um protocolo MAC para a comunicação baseada no receptor não pode se limitar a entregar pacotes para um vizinho qualquer e, sim, ele deve aguardar a decisão de propagação dos nós receptores para garantir que um vizinho acordado será o próximo nó.

Este trabalho apresenta o *Receiver-based Medium Access Control* (Rb-MAC), um protocolo para projetos integrados de comunicação baseada no receptor. Um projeto integrado consiste em uma camada ser capaz de violar a arquitetura da pilha de protocolos para acessar informações disponíveis em outra camada a fim de explorar vantagens e evitar pontos fracos. O Rb-MAC elimina o compartilhamento de informações sobre vizinhos e aborda o desafio proposto através de um mecanismo de retransmissão. Esse mecanismo é baseado no escalonamento de dormir/acordar dos nós, no fato de qualquer vizinho pode ser o próximo nó e na importância do nó emissor confirmar que um vizinho decidiu ser o próximo nó. Resultados de simulação revelam que com o Rb-MAC, o modelo receptor avaliado aumenta a taxa de entrega (porcentagem de pacotes entregues aos nós destinos), e reduz o número de transmissões e o consumo de energia quando comparado com um modelo baseado no emissor. O Rb-MAC também garante uma taxa de entrega elevada até mesmo quando os nós são configurados com valores reduzidos de ciclos de trabalho. Contudo, a latência e o número de colisões do modelo emissor avaliado são menores que essas métricas no modelo receptor avaliado. Para um cenário específico em que uma mesma rota é utilizada mais vezes, o Rb-MAC possui uma técnica de ciclo de trabalho dinâmico que proporciona uma latência inferior a do outro modelo. Este trabalho também efetua a verificação formal de algumas propriedades do Rb-MAC.

O restante deste trabalho é dividido da seguinte forma. A seção 2 mostra os trabalhos relacionados. A seção 3 apresenta o protocolo Rb-MAC. A seção 4 mostra os resultados de simulação. A seção 5 descreve as conclusões e as direções futuras deste trabalho.

2. TRABALHOS RELACIONADOS

Esta seção mostra alguns protocolos de roteamento e MAC. Um exemplo de protocolo de roteamento baseado no receptor é o *Trajectory and Energy-based Dissemination* (TEDD) [2] e um baseado no emissor é o *Trajectory Based Forwarding* (TBF) [7]. Ambos utilizam o conceito de "roteamento em curva" que consiste em inserir uma equação de curva no pacote e fazer com que os nós localizados próximos à curva propaguem o pacote. O TEDD e

o TBF são utilizados nos modelos de comunicação avaliados neste trabalho.

A maioria dos protocolos MAC para RSSFs tais como o Sensor MAC (S-MAC) [9] e o Battery Aware MAC [5] compartilha informações de vizinhos, ao contrário do Rb-MAC. Como parte deste trabalho, o S-MAC é utilizado em um modelo de comunicação baseado no emissor. No S-MAC, os nós vizinhos efetuam uma sincronização para que eles acordem e adormeçam ao mesmo tempo. Para evitar colisões, o S-MAC utiliza o mecanismo *Request To Send/Clear To Send/Data/Acknowledge* (RTS/CTS/DATA/ACK) que é restrito à comunicação em que o nó emissor conhece o próximo nó – roteamento baseado no emissor. O S-MAC pode ser usado na comunicação baseada no receptor, contudo, nesse caso, ele evita colisões apenas escutando o meio.

Alguns protocolos não compartilham informações de vizinho [6, 8] e para um nó emissor descobrir quando o nó escolhido estará acordado, ele envia vários pacotes de RTS. Quando o vizinho escolhido acordar e receber um RTS, o esquema CTS/DATA/ACK tradicional é utilizado. Esses protocolos se diferenciam entre si pela forma como enviam o pacote de RTS e se diferenciam do Rb-MAC por serem focados na comunicação baseada no emissor. Esses protocolos não foram avaliados neste trabalho, pois como mostrado por seus autores, normalmente (incluindo o cenário simulado neste trabalho), as técnicas de compartilhamento consomem menos energia. Em [10], tem-se um projeto integrado baseado no receptor em que o esquema de RTS/CTS considera a decisão de propagação do nó receptor para enviar pacotes de CTS. Nesse caso, o RTS é destinado a qualquer vizinho e o primeiro nó que enviar um CTS será o próximo nó. Apesar do nó emissor confirmar que um vizinho tomou a decisão de propagação, o intervalo dos pacotes de RTS não considera o escalonamento de dormir/acordar dos nós o que causa muitas transmissões de RTS. Esse protocolo não foi avaliado neste trabalho devido ao seu custo de energia com transmissões.

3. O PROTOCOLO RECEIVER-BASED MEDIUM ACCESS CONTROL (RB-MAC)

Esta seção apresenta o Rb-MAC, um protocolo para projetos integrados de comunicação baseada do receptor. A seção 3.1 apresenta o funcionamento básico do Rb-MAC, onde o acesso ao meio é aleatório e baseado na técnica *Carrier Sense Multiple Access* (CSMA) p-persistente e os nós dormem/acordam de forma aleatória. A seção 3.2 mostra o mecanismo de retransmissão do Rb-MAC. A seção 3.3 mostra um projeto integrado em que a camada de rede atualiza dinamicamente o ciclo de trabalho dos nós sensores. A seção 3.4 efetua a verificação formal de algumas propriedades do Rb-MAC.

3.1. FUNCIONAMENTO BÁSICO

O funcionamento básico do Rb-MAC é descrito na máquina de estados finitos da figura 1, onde as transições são eventos gerados por outras camadas ou pelo término de temporizadores do Rb-MAC. Quando um evento acontece e uma transição é realizada, o Rb-MAC muda o seu estado corrente ou efetua uma ação. A mudança de estado e as ações só acontecem devido às transições, contudo, existem transições sem um deles. Quando um nó se torna ativo, ele vai para o estado *escutar* e permite que a camada física escute o meio.

A fase de recepção do Rb-MAC começa quando a camada física identifica que um nó vizinho vai começar uma transmissão. Nesse caso, a camada física gera uma requisição de receber e o Rb-MAC vai para o estado *receber*, onde o nó receberá o pacote. O Rb-MAC permanece nesse estado até que ele receba uma requisição de escutar que será gerada pela camada física quando o meio estiver livre. Quando isso acontece, o Rb-MAC processa o pacote recebido e retorna ao estado *escutar*. No processamento do pacote, se ele foi recebido com sucesso, o Rb-MAC o envia para a camada de rede, caso contrário, uma colisão/falha aconteceu e o Rb-MAC descarta o pacote.

A fase de transmissão do Rb-MAC é baseada na abordagem CSMA p-persistente e começa quando o protocolo está no estado *escutar* e a camada de rede possui um pacote a ser enviado. Nesse caso, a camada de rede gera uma requisição de aguardar para que Rb-MAC vá para o estado *aguardar*, como ilustrado na figura 1. Quando tal transição acontece, o Rb-MAC realiza a ação de aguardar na qual ele permite que a camada física comece a transmitir o pacote com probabilidade p. Com probabilidade $q = 1 - p$, ele inicializa o temporizador aguardar. Se camada física começar a transmitir, ela gera uma requisição de transmitir para que o Rb-MAC vá para o estado *transmitir*. Quando a transmissão termina, a camada física gera uma requisição de escutar para que o Rb-MAC retorne ao estado *escutar*. Por outro lado, se o temporizador aguardar foi inicializado, o seu término significa uma requisição de aguardar e essa faz com que o Rb-MAC execute a ação de aguardar novamente. O Rb-MAC permanece no estado *aguardar* até que ele receba uma requisição de transmissão ou uma de receber. Enquanto o Rb-MAC estiver no estado *aguardar*, a camada física pode identificar que algum vizinho vai começar uma transmissão. Nesse caso, essa camada gera uma requisição de receber para que o Rb-MAC interrompa o seu temporizador aguardar e vá para o estado *receber*. Quando a recepção termina, a camada física gera uma requisição de escutar e essa faz com que o Rb-MAC retorne ao estado de *escutar*. Como existe um pacote para ser enviado, o Rb-MAC inicializa o seu temporizador aguardar para gerar uma re-

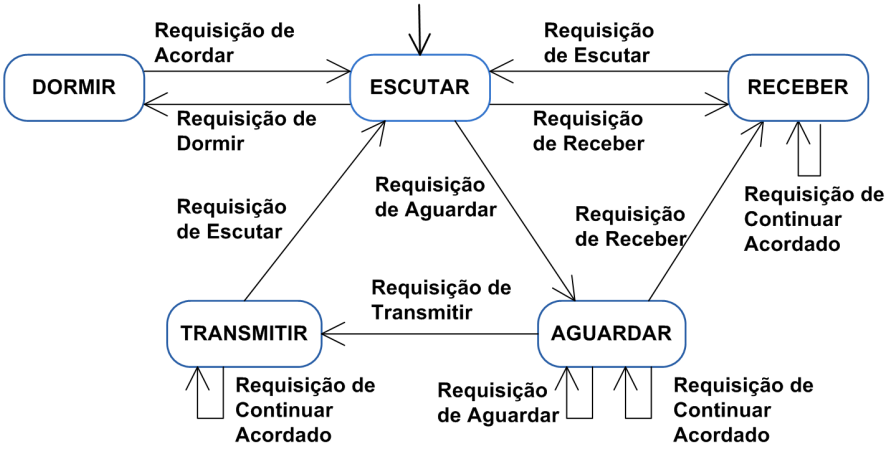


Figura 1. Funcionamento básico do Rb-MAC

quisição de aguardar. Um ponto importante desta fase é que quando o Rb-MAC estiver no estado *transmitir*, o nó não é capaz de receber pacotes, pois o seu rádio é normalmente *half-duplex*. Se dois nós vizinhos começarem a enviar pacotes ao mesmo tempo, as colisões não são detectadas/tratadas por esses nós; isso pode ser feito apenas nos nós que receberem o pacote.

O Rb-MAC também permite que os nós adormeçam para economizar energia. Como mostrado na figura 1, quando um nó estiver no estado *escutar* e receber uma requisição de dormir, ele vai para o estado *dormir*. O Rb-MAC permanece nesse estado até receber uma requisição de acordar para retornar ao estado *escutar*. O procedimento para a geração das requisições de dormir e acordar é cíclico e divide o tempo de vida dos nós em ciclos que são compostos por um período de tempo em que o nó permanece acordado seguido por outro em que ele dorme. O término de um ciclo implica no início do próximo. A duração dos períodos acordado A e dormindo S (do inglês, *sleeping*) são variáveis aleatórias com distribuição uniforme como mostrado nas equações 1 e 2, respectivamente. Os valores mínimo e máximo dessas variáveis são definidos nas equações 3 até 6 em que c é o tempo mínimo de duração de um ciclo, α é o fator de desvio e d é o ciclo de trabalho desejado (do inglês, *duty cycle*). O parâmetro c é um limite inferior para cada período. Quando um nó acorda, é interessante que ele permaneça acordado por um tempo mínimo em que o nó pode participar de tarefas da rede. De forma similar, se um nó dormir por um tempo mínimo após a realização de uma tarefa, o tempo de vida de sua bateria tende a ser maior. Segundo [5], isso é uma propriedade das baterias. O fator de desvio α determina o valor máximo das variáveis A e S. O parâmetro d é uma constante que indica o valor de ciclo de trabalho desejado pelo projetista da rede.

O processo de geração de requisições de dormir/acordar é descrito a seguir. Quando um nó se torna ativo, ele liga o seu rádio e inicializa o temporizador acordado para expirar após A unidades de tempo. Quando ele expira e o Rb-MAC está no estado *escutar*, esse término significa uma requisição de dormir e essa faz com que o Rb-MAC desligue o rádio, inicialize o temporizador dormir para expirar após S unidades de tempo e vá para o estado *dormir*. O término do temporizador dormir significa uma requisição de acordar e essa faz com que o Rb-MAC ligue o rádio, inicialize o temporizador acordado para expirar após A unidades de tempo e retorne ao estado *escutar*. Por outro lado, quando o temporizador acordado expirar e o Rb-MAC não estiver no estado *escutar*, isso significa uma requisição de continuar acordado e essa faz com que o Rb-MAC inicialize novamente o temporizador acordado para expirar após mais algumas unidades extras de tempo e continue no estado corrente. Esse processo é repetido sempre que o temporizador acordado expirar e o Rb-MAC não estiver no estado *escutar*. Todo o tempo extra acumulado é descontado na duração dos próximos períodos de dormir e acordar. Até que todo o tempo acordado extra seja compensado, a duração deles será S_{max} e A_{min}, respectivamente.

$$A \sim U(A_{min}, A_{max}) \quad (1)$$

$$S \sim U(S_{min}, S_{max}) \quad (2)$$

$$A_{min} = d \times c \quad (3)$$

$$S_{min} = (1 - d) \times c \quad (4)$$

$$A_{max} = \alpha \times A_{min} \quad (5)$$

$$S_{max} = \alpha \times S_{min} \quad (6)$$

O Rb-MAC também permite que as camadas superiores gerem requisições de dormir/acordar, pois elas conhecem situações em que o rádio pode ser ligado/desligado. Por exemplo, quando a aplicação utiliza apenas dados noturnos, a camada de aplicação gera uma requisição de acordar no início da noite e outra de dormir no final. As camadas superiores também podem suspender a transição para o estado *dormir*, pois existem situações em que o nó deve ficar acordado por mais tempo. Por exemplo, no roteamento baseado no receptor, um nó não deve adormecer durante um processo de decisão.

3.2. MECANISMO DE RETRANSMISSÃO

A camada MAC das RSSFs é responsável pelo comportamento de dormir dos nós e, por isso, ela deve tratar as desconexões causadas pelos nós adormecidos. No Rb-MAC, essas desconexões são tratadas através de um mecanismo de retransmissão baseado na confirmação salto a salto. O objetivo desse mecanismo é similar ao dos pacotes de confirmação do protocolo IEEE 802.11; contudo, o mecanismo proposto confirma que um vizinho se escolheu como o próximo nó e o protocolo 802.11 confirma apenas que o próximo nó recebeu o pacote. Além disso, diferente do protocolo 802.11, o Rb-MAC não envia pacotes de controle. No mecanismo proposto, quando um nó transmite um pacote, ele permanece acordado por certo tempo para aguardar a propagação do pacote por algum nó vizinho. Após esse tempo, se o pacote não for recebido, o mesmo é retransmitido.

A definição do tempo em que um nó aguarda para realizar uma retransmissão e o número máximo de retransmissões é baseada no comportamento de dormir do Rb-MAC que proporciona um limite inferior A_{min} para o tempo em que um nó permanece acordado e outro superior S_{max} para o tempo em que o nó dorme. O limite inferior acordado é utilizado na definição do intervalo de tempo entre as retransmissões e o limite superior dormindo, na definição de quando a última retransmissão será executada. O número máximo de retransmissões é $\lceil \frac{S_{max}}{A_{min}} \rceil$, pois durante S_{max} unidades de tempo, uma retransmissão é executada a cada A_{min} unidades de tempo.

Teorema 1: Seja o intervalo entre retransmissões A_{min} e o número máximo de retransmissões $n = \lceil \frac{S_{max}}{A_{min}} \rceil$, quando um nó envia um pacote e todos os seus vizinhos estão adormecidos, cada vizinho estará acordado em pelo menos uma retransmissão.

Prova: Dado o intervalo e o número máximo de retransmissões, a prova consiste em mostrar que, para cada vizinho, se esse dorme no instante de tempo T_A, acorda em T_B e retorna a dormir em T_C, existe uma retransmissão $T_i (1 \leq i \leq n)$ tal que $\boxed{T_B < T_i < T_C}$.

- 1^o passo: Sabe-se que:

  – T_0 é a primeira transmissão.

  – $T_A = T_B - S$, pois a variável aleatória S é o intervalo de tempo entre T_A e T_B.

  – $T_A < T_0$, pois o vizinho estava dormindo na transmissão T_0.

  – $T_0 = T_n - S_{max}$, pois S_{max} é o intervalo entre a primeira e a última retransmissões.

  – $S \leq S_{max}$, pois S_{max} é o maior valor da variável aleatória S.

- 2^o passo: Logo, $T_B - S = T_A < T_0 = T_n - S_{max} \Rightarrow T_B < T_n$ e pode-se afirmar que existe pelo menos uma retransmissão T_i tal que $\boxed{T_B < T_i}$.

- 3^o passo: Supondo que T_i é a primeira retransmissão tal que $T_B < T_i$. Se a suposição for falsa, $T_{j<i}$ é a primeira retransmissão maior que T_B o que é uma contradição.

- 4^o passo: Sabe-se também que:

  – $T_B > T_i - A$, pois T_i é a primeira retransmissão tal que $T_B < T_i$ e se T_B for subtraído de T_i, esse resultado será menor à variável aleatória A.

  – $T_B = T_C - A$, pois a variável aleatória A é o intervalo entre T_B e T_C.

- 5^o passo: Logo, $T_C - A = T_B > T_i - A \Rightarrow \boxed{T_i < T_C}$.

Um aspecto crucial do mecanismo proposto é o número esperado de retransmissões, pois quando um nó retransmite um pacote diversas vezes, o custo de energia dessa tarefa pode tornar o Rb-MAC inviável para as RSSFs. No esquema proposto, um nó efetua uma retransmissão se ele transmitiu um pacote e não recebeu uma propagação do mesmo. Supondo a existência de vizinhança, os nós vizinhos não propagam um pacote se todos estiverem dormindo no seu envio ou devido às colisões/falhas. Logo, o número esperado de retransmissões depende da probabilidade de colisões/falhas e da dos nós vizinhos estarem dormindo. O evento ocorrência de colisões/falhas pode ser modelado como uma variável aleatória com distribuição uniforme e com probabilidade constante p_{col}. O evento os vizinhos estarem dormindo é uma variável aleatória cuja probabilidade reduz à medida que as retransmissões são executadas. Cada vizinho acorda pelo menos uma vez durante um processo de retransmissão (Teorema 1) e, à medida que as retransmissões são executadas e nenhum vizinho acorda, o final do processo se aproxima assim como a chance de cada vizinho acordar. A esperança da variável aleatória S na retransmissão i corresponde à média aritmética dos valores mínimo e máximo de S em i, como mostrado na equação 7. Nesse caso, como todos os vizinhos dormem A_{min} unidades de tempo por retransmissão, os valores mínimo e máximo de S são reduzidos em A_{min} unidades a cada retransmissão, como mostrado nas equações 8 e 9, respectivamente.

$$E[S_i] = \frac{S_{min}(i) + S_{max}(i)}{2} \tag{7}$$

$$S_{min}(i) = \begin{cases} S_{min} - i * A_{min} & \text{se } S_{min} > i * A_{min} \\ 0 & \text{caso contrário} \end{cases} \tag{8}$$

$$S_{max}(i) = \begin{cases} S_{max} - i * A_{min} & \text{se } S_{max} > i * A_{min} \\ 0 & \text{caso contrário} \end{cases} \tag{9}$$

A equação 10 mostra a probabilidade de um vizinho estar dormindo na retransmissão i, onde E[A] é igual para todas as retransmissões. A equação 11 mostra a probabilidade de um vizinho não propagar com a retransmissão i. Um vizinho não propaga o pacote se ele estiver dormindo ou uma colisão/falha acontecer, eventos independentes. A probabilidade da retransmissão i acontecer é o complemento de "*todos os nn vizinhos não propagarem com a retransmissão i*", como mostrado na equação 12. Contudo, a retransmissão i acontece somente quando todas as $i - 1$ retransmissões anteriores acontecerem e nenhuma propagação for recebida pelo nó que efetua as retransmissões. A função de distribuição de probabilidade das retransmissões é mostrada na equação 13. A esperança da

variável aleatória R representando o número de retransmissões é o somatório de cada retransmissão multiplicada por sua probabilidade, como mostrado na equação 14.

$$probDormir(i) = \frac{E[S_i]}{E[S_i] + E[A]} \quad (10)$$

$$probNaoTx(i) = probDormir(i) + (1 - probDormir(i)) * p_{col} \quad (11)$$

$$prob(i, nn) = 1 - probNaoTx(i)^{nn} \quad (12)$$

$$fdp(i, nn) = prob(i, nn) \prod_{k=0}^{i-1} (1 - prob(k, nn)) \quad (13)$$

$$E[R] = \sum_{i=0}^{\lceil \frac{S_{max}}{A_{min}} \rceil} i * fdp(i, nn) \quad (14)$$

A principal vantagem do mecanismo proposto é eliminar as desconexões causadas por nós adormecidos o que eleva a taxa de pacotes entregues. Essa vantagem é mais interessante para os valores reduzidos de ciclo de trabalho em que os nós dormem mais tempo e o número de desconexões é maior. A sua desvantagem é o número de retransmissões e a latência, pois um pacote pode ser retransmitido mais de uma vez e cada retransmissão representa um atraso. Contudo, o custo de energia das retransmissões pode ser minimizado pelo ganho de energia obtido com a redução do ciclo de trabalho. Outro ponto importante é que o mecanismo proposto não garante que todos os pacotes serão entregues aos seus destinos devido às colisões/falhas. Contudo, ele reduz o efeito das mesmas. Quando uma colisão/falha acontece, nenhum vizinho propaga o pacote, causando uma retransmissão.

3.3. CICLO DE TRABALHO DINÂMICO

Esta seção apresenta um projeto integrado em que o Rb-MAC permite que a camada de rede atualize o ciclo de trabalho dos nós para criar caminhos especiais de roteamento compostos por nós que não dormem (ciclo de trabalho de 100%). O modelo proposto também considera que a camada de rede pode alterar o seu processo de decisão para que os nós do caminho transmitam pacotes sem atrasos. Os protocolos de roteamento baseados no receptor costumam inserir atrasos nesse processo de decisão [2]. Uma vantagem do modelo proposto é que a maioria dos nós não pertence ao caminho e é configurada com valores reduzidos de ciclo de trabalho para economizar energia. Outra vantagem é que o roteamento nos caminhos especiais é efetuado com latência reduzida e a sua limitação consiste em ser válido apenas se a rota é utilizada mais vezes.

O funcionamento básico da técnica proposta é descrito nos algoritmos 1 e 2 que são executados pela camada de rede. Após um nó enviar um pacote, o algoritmo 1 insere o nó no caminho especial. No primeiro passo (Linha 1), a camada de rede verifica se o nó pertence a um caminho. Se falso, a camada de rede viola a arquitetura da pilha de protocolos e aumenta o ciclo de trabalho (Linha 2) e elimina o atraso da decisão de roteamento para todos os pacotes a serem roteados pelo caminho (Linha 3). Depois, a camada de rede escalona o tempo de retorno para identificar quando o caminho expira (Linha 4). Por outro lado, quando um nó pertence ao caminho (Linha 5), a camada de rede reescalona o tempo de retorno desse nó para que ele permaneça por mais tempo em um caminho (Linha 6). Quando o tempo de retorno expirar, o algoritmo 2 é executado e, assim, a camada de rede retorna os valores originais da decisão (Linha 1) e do ciclo de trabalho (Linha 2).

Algoritmo 1 : Início do Ciclo de Trabalho Dinâmico

1: **se** eu não pertenço a um caminho especial de roteamento **então**
2: Aumentar o meu ciclo de trabalho no Rb-MAC para 100%
3: Eliminar o meu atraso para a propagação de pacotes do caminho
4: Escalonar o tempo de retorno
5: **senão**
6: Reescalonar o tempo de retorno
7: **fim se**

Algoritmo 2 : Término do Ciclo de Trabalho Dinâmico

1: Retornar o meu atraso original para a propagação de pacotes do caminho
2: Retornar o meu ciclo de trabalho original no Rb-MAC

3.4. VERIFICAÇÃO FORMAL

Esta seção efetua a verificação formal de algumas propriedades de um modelo de comunicação baseado no receptor que utiliza o Rb-MAC. O modelo de verificação foi o NuSMV e o conjunto φ de propriedades verificadas é listado abaixo.

- φ_1: Cada estado s é alcançável a partir do estado inicial s_0 e s_0 é alcançável a partir de cada estado s. Essa propriedade garante a inexistência de estados de erro ou inalcançáveis.
- φ_2: Se o Rb-MAC estiver no estado *dormir*, as camadas MAC e de rede não recebem nem enviam pacotes. Essa propriedade garante que o rádio está desligado no estado *dormir*.
- φ_3: Os protocolos MAC e de rede podem receber/enviar pacotes simultaneamente. Essa propriedade, φ_4 e φ_5 garantem (ou limitam) a independência das ações de receber/enviar.
- φ_4: O Rb-MAC não pode receber e enviar pacotes simultaneamente. Essa propriedade garante a limitação física dos nós sensores em que o rádio é geralmente *half-duplex*.
- φ_5: A camada de rede pode receber e enviar pacotes simultaneamente, pois a camada MAC armazena os pacotes ainda não processados.

- φ_6: A camada de rede recebe um pacote se e, somente se, o Rb-MAC o receber anteriormente. Essa propriedade e φ_7 garantem a ordem em que um pacote é processado.

- φ_7: O Rb-MAC envia um pacote se e, somente se, a camada de rede já o enviou.

4. RESULTADOS DE SIMULAÇÃO

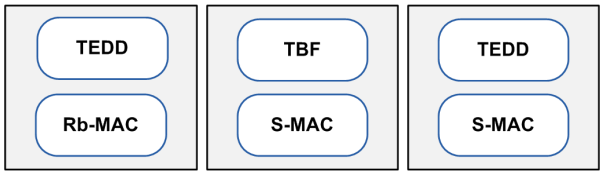
Esta seção tem como objetivo comparar modelos de comunicação baseados no emissor e no receptor em cenários de disseminação para RSSFs em que um nó monitor deseja enviar dados para os nós sensores. Esse tipo de comunicação é crucial em RSSFs, pois o nó monitor realiza tarefas específicas tais como alterar o modo de funcionamento dos sensores, ativar/desativá-los, e enviar requisições ou interesses para eles.

4.1. PARÂMETROS DE SIMULAÇÃO

Em todas as simulações, utilizou-se uma rede com 500 nós estáticos, homogêneos e com uma capacidade de energia finita e não renovável. O consumo de energia de cada nó foi baseado no Mica2 e a energia inicial foi suficiente para nenhum nó morrer por falta de energia. O raio de comunicação dos nós foi de 100 m, eles foram depositados de forma aleatória em uma área 1000×1000 m^2 e cada nó conhece a sua localização.

Um nó monitor sem restrição de recursos foi colocado no canto inferior esquerdo da rede e realizou várias disseminações intercaladas uniformemente e destinadas a 25 nós. Em um cenário de simulação (seção 4.2), cada disseminação foi destinada a um subconjunto aleatório de nós. No outro (seção 4.3), um subconjunto aleatório de destinos foi escolhido e utilizado em todas as disseminações. O tempo de simulação foi de 1000 s, contudo, a primeira disseminação foi efetuada apenas com 50 s e a última com 600 s. O atraso inicial é necessário para a inicialização do S-MAC. A última disseminação acontece antes do final da simulação para garantir a execução de todo o roteamento.

Os modelos de comunicação comparados são ilustrados na figura 2 e descritos abaixo. O modelo receptor avaliado combina o TEDD que é um protocolo de roteamento baseado no receptor específico para a disseminação de dados com o Rb-MAC. Este modelo apresenta uma interação entre as duas camadas em que o TEDD é capaz de proibir que um nó adormeça quando esse estiver em decisão de propagação. O modelo emissor avaliado combina o TBF e o S-MAC. O primeiro foi escolhido por ser um protocolo de roteamento baseado no emissor específico para a disseminação de dados que trabalha como o TEDD e, o segundo, por prover informações atualizadas de vizinhos que são exploradas pelo TBF para a escolha do próximo nó através de interações entre as duas camadas. Este trabalho também avalia um modelo de comunicação híbrido que combina o S-MAC e o TEDD. Esse modelo é denominado híbrido por combinar um protocolo MAC que compartilha informações de vizinho e um de roteamento baseado no receptor. O modelo híbrido foi proposto porque os outros apresentam diferentes protocolos MAC e de rede, o que dificulta identificar os ganhos em cada camada. Nesse modelo, o funcionamento básico do S-MAC foi alterado para permitir uma interação entre camadas em que o TEDD é capaz de proibir que um nó adormeça quando estiver em decisão de propagação.



(a) Receptor (b) Emissor (c) Híbrido

Figura 2. Modelos Avaliados para a Comunicação de Dados

O Rb-MAC e o S-MAC foram utilizados na camada MAC dos modelos avaliados. O ciclo de trabalho deles foi estático e avaliado para diferentes valores. Contudo, no segundo cenário de simulação, o Rb-MAC utilizou a sua técnica de ciclo de trabalho dinâmico onde os nós do caminho especial possuem um ciclo de trabalho de 100% e os demais de 10%. A frequência com que os nós atualizam as informações de vizinhos no S-MAC foi definida da forma mais justa possível, balanceando o número de transmissões e a taxa de pacotes recebidos. Se essa frequência aumentar, maior será o número de transmissões (logo, o consumo de energia) e a taxa de entrega obtidos pelo modelo emissor. Por outro lado, se ela for reduzida, o resultado oposto é verificado. A frequência de atualização utilizada no S-MAC foi de 150 ciclos de vida. No S-MAC, o tempo de vida dos nós é divido em ciclos de vida e a frequência utilizada significa que a cada 150 ciclos de vida, os nós trocam informações com os seus vizinhos. Além disso, considerou-se que todos os pacotes (de controle ou dados) possuem o mesmo tamanho.

Todos os modelos foram implementados no *Network Simulator 2* e os resultados de simulação correspondem à média aritmética de n simulações, onde n é o menor tamanho de amostra que proporciona o intervalo de confiança desejado. O nível de confiança foi de 95% e o *teste-T* [4] com 0.05 de significância foi utilizado para afirmar que um protocolo é melhor/pior do que outro para uma dada métrica.

4.2. DISSEMINAÇÃO PARA UM SUBCONJUNTO DINÂMICO DE DESTINOS

Esta seção avalia o desempenho dos modelos propostos em um cenário de disseminação em que o subconjunto

de destinos é alterado a cada disseminação. Esse cenário acontece, por exemplo, quando o nó monitor deseja enviar dados para os sensores que possuem uma propriedade dinâmica tal como ter mais/menos energia e ser responsável por uma tarefa. O objetivo nesse cenário é entregar pacotes e economizar energia. A taxa de entrega é fundamental, pois pacotes podem ser perdidos durante o roteamento. A economia de energia é objetivo de qualquer solução para as RSSFs. Outros objetivos são as reduções de latência e colisões. A latência é crucial em aplicações dependentes do tempo tal como a detecção de incêndio. As colisões devem ser reduzidas, pois elas tendem a reduzir a taxa de entrega e aumentar o consumo de energia e a latência.

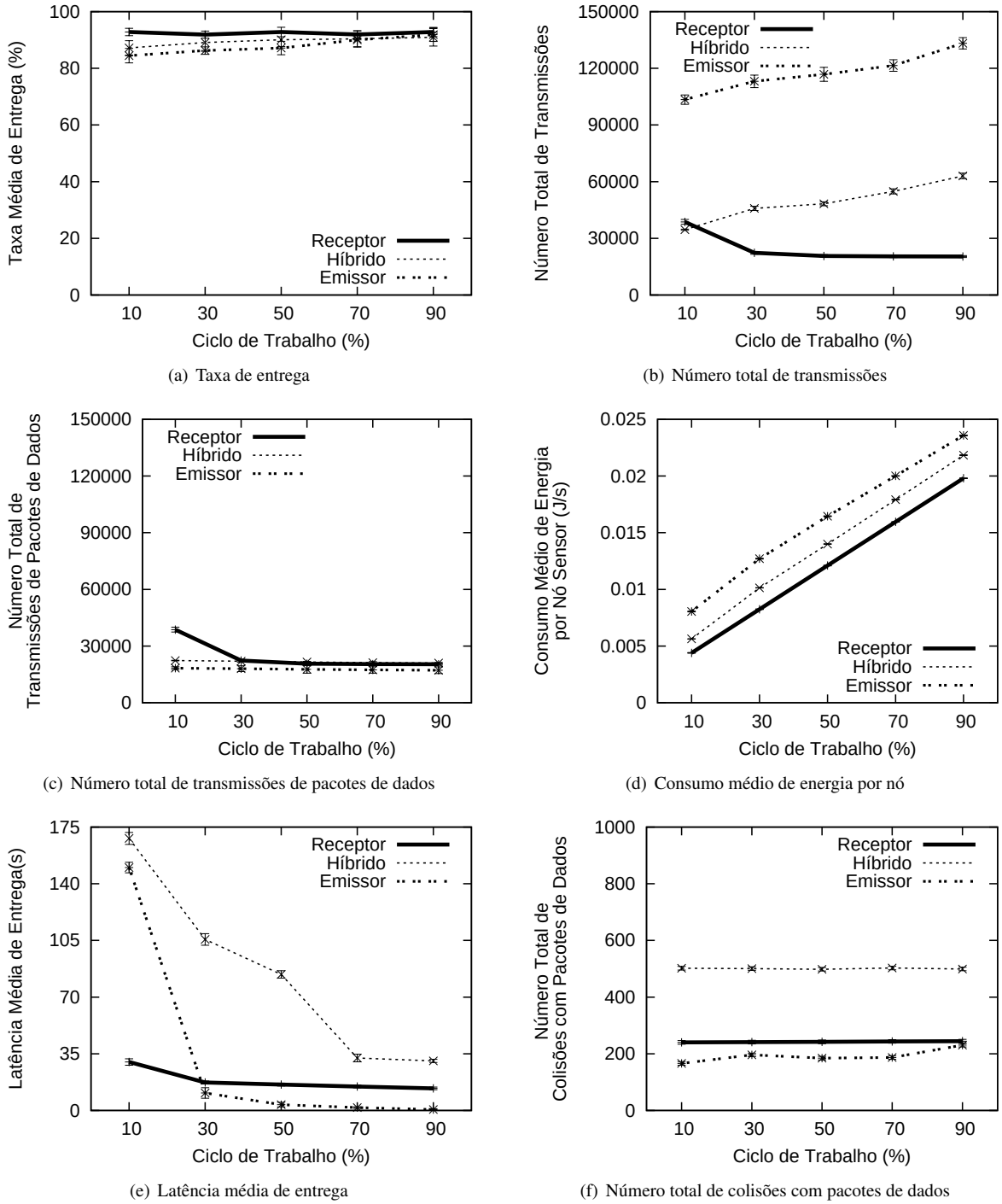
O desempenho dos modelos avaliados quando se aumenta o valor estático do ciclo de trabalho é mostrado na figura 3. A figura 3-a apresenta a taxa de entrega em que o modelo receptor avaliado entregou 1,05 e 1,03 vezes mais pacotes que os modelos emissor e híbrido, respectivamente. Esse resultado acontece porque o mecanismo de retransmissão do Rb-MAC elimina as perdas de pacote causadas pelos nós adormecidos e reduz o efeito das colisões. Além disso, ele mantém a taxa de entrega para os menores valores de ciclo de trabalho. As perdas de pacote do modelo receptor são consequência de colisões. No modelo emissor, além das colisões, as perdas foram causadas por nós adormecidos que foram selecionados para o roteamento. O aumento da entrega desse modelo foi porque, no S-MAC, o aumento do ciclo de trabalho implica em mais atualizações das informações de vizinhos e, assim, menos nós adormecidos são escolhidos para o roteamento. No modelo emissor, apesar do esquema RTS/CTS/DATA/ACK, as colisões acontecem, por exemplo, devido aos nós que dormem e acordam em tempos distintos e deturpam o esquema utilizado. No modelo híbrido, que não efetua retransmissões, as perdas são causadas pelas colisões e pelo modelo permitir que um nó efetue transmissões quando não existem vizinhos acordados. O aumento da taxa à medida que o ciclo de trabalho aumenta é porque a possibilidade de um nó efetuar uma transmissão sem vizinhos acordados é menor.

O número de transmissões quando se aumenta o valor estático do ciclo de trabalho é mostrado nas figuras 3-b e 3-c. A diferença entre elas é que a primeira considera as transmissões de pacotes de controle ou de dados e a outra, apenas as de dados. O modelo receptor apresenta o mesmo resultado nas duas figuras, pois o Rb-MAC não transmite pacotes de controle. Nos outros modelos, o S-MAC envia pacotes de controle para atualizar informações de vizinhos. No modelo emissor, ele também envia pacotes RTS/CTS/ACK. Os modelos emissor e híbrido efetuaram respectivamente 4,8 e 2,1 vezes mais transmissões que o receptor devido aos pacotes de controle do S-MAC. Neste ponto, observa-se que para os modelos avaliados, o custo de transmissão inserido pelas retransmissões do Rb-MAC foi menor que o inserido pelos pacotes de controle do S-MAC. O modelo emissor enviou 2,3 vezes mais pacotes que o híbrido devido aos pacotes de controle extras. Aumentando o ciclo de trabalho, o número de transmissões efetuadas pelo modelo receptor reduz conforme o funcionamento básico do mecanismo de retransmissão do Rb-MAC. Por outro lado, o número efetuado pelos outros modelos aumenta, pois, no S-MAC, o quão maior o ciclo de trabalho, mais frequente são os envios de pacote para atualizar as informações de vizinhos. Considerando apenas os pacotes de dados, como o modelo receptor efetua retransmissões, ele efetuou 1,4 e 1,1 vezes mais transmissões que os modelos emissor e híbrido, respectivamente.

O consumo de energia quando se aumenta o valor estático do ciclo de trabalho é apresentado na figura 3-d. À medida que o ciclo de trabalho aumenta, o consumo de energia aumenta, pois os nós sensores permanecem mais tempo acordados. Os modelos emissor e híbrido consomem 1,3 e 1,1 vezes mais energia que o modelo receptor avaliado, pois esses enviam mais pacotes que o último. O modelo emissor avaliado consome mais energia que o híbrido (1,2 vezes), pois o primeiro efetua mais transmissões.

A latência quando se aumenta o valor estático do ciclo de trabalho é apresentada na figura 3-e. A latência inicial do modelo emissor é 5,0 vezes maior que a do modelo receptor e, a partir do ciclo de trabalho de 30%, a do receptor torna-se 3,7 vezes maior que a do emissor. O principal responsável pela latência do modelo receptor é o TEDD que insere atrasos no roteamento para efetuar a decisão de propagação. No modelo emissor, esse papel é do S-MAC, pois ele proíbe que um nó propague um pacote no mesmo ciclo de vida em que esse é recebido. A latência reduz com o aumento do ciclo de trabalho, pois esse aumento provoca a redução do tamanho do ciclo de vida. A latência do modelo híbrido é 4,6 e 2,5 vezes maior que a dos modelos receptor e emissor, pois ele combina o S-MAC e o TEDD. Uma consideração crucial para a comunicação baseada no receptor é que como o Rb-MAC é capaz de manter a taxa de entrega para valores reduzidos de ciclo de trabalho, o projetista da rede deve definir o valor do ciclo de trabalho de acordo com o maior valor aceitável de latência. O quão maior esse valor, menor o ciclo de trabalho.

O número total de colisões com pacotes de dados quando se aumenta o valor estático do ciclo de trabalho é mostrado na figura 3-f. Todos os modelos foram eficientes em termos de colisões quando esse valor é comparado com o número de transmissões efetuadas por cada modelo. Os modelos receptor e emissor efetuam uma colisão a cada cem pacotes de dados transmitidos e, no modelo híbrido, essa relação é o dobro. O modelo emissor



(a) Taxa de entrega

(b) Número total de transmissões

(c) Número total de transmissões de pacotes de dados

(d) Consumo médio de energia por nó

(e) Latência média de entrega

(f) Número total de colisões com pacotes de dados

Figura 3. Parâmetros avaliados para o primeiro cenário de simulação

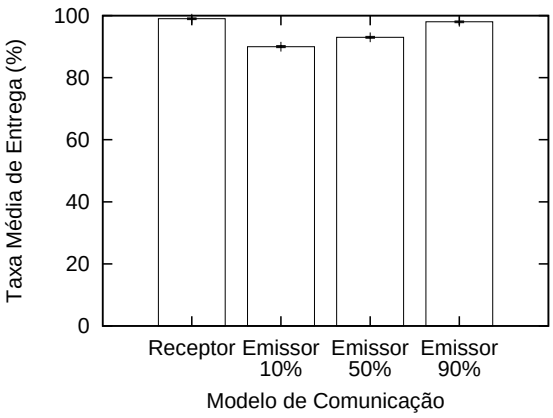
efetuou menos transmissões devido ao uso da técnica de RTS/CTS/DATA/ACK. O modelo receptor reduz as colisões através do esquema CSMA p-persistente. O modelo híbrido permite mais colisões por apenas escutar o meio como forma de reduzir colisões.

4.3. Disseminação para um Subconjunto Estático de Destinos

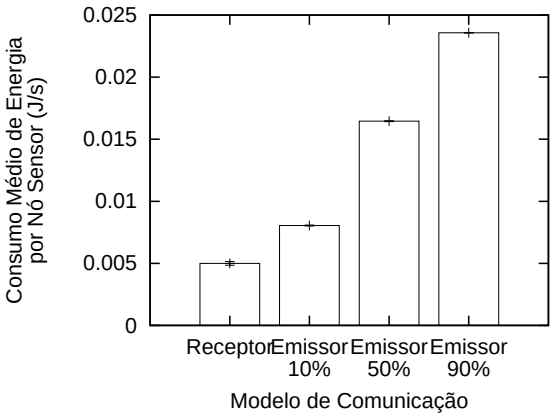
Esta seção avalia o desempenho dos modelos propostos em um cenário de disseminação em que todas as disseminações são destinadas para um mesmo subconjunto de nós definido aleatoriamente antes da simulação. Esse

cenário acontece, por exemplo, quando o nó monitor deseja enviar sistematicamente dados para nós localizados em posições estratégicas ou nós responsáveis por tarefas estáticas. Novamente, o objetivo é entregar pacotes e economizar energia. Outros objetivos relevantes são as reduções de latência e colisões. Além disso, o modelo receptor avaliado utiliza a técnica de ciclo de trabalho dinâmico do Rb-MAC. Os resultados de simulação obtidos pelos modelos emissor e híbrido são similares aos do cenário anterior, pois esses não se aproveitam das rotas serem repetidas. Os resultados do modelo híbrido não são mostrados e o modelo emissor avaliado utiliza os valores estáticos de ciclo de trabalho como 10%, 50% e 90%. Tanto no modelo receptor como no emissor, o protocolo de roteamento utilizou o mesmo conjunto de rotas em todas as disseminações de uma simulação.

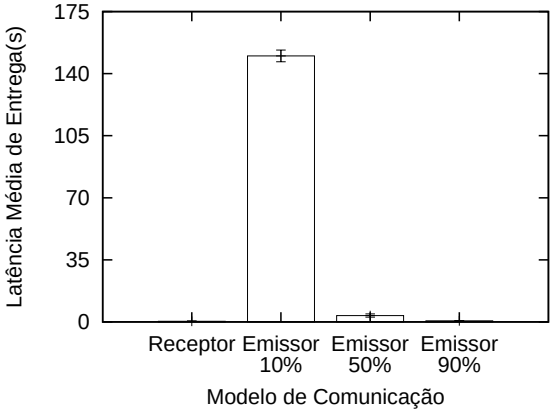
A taxa de entrega, o consumo de energia e a latência dos modelos receptor e emissor são mostrados na figura 4. O modelo receptor avaliado apresentou a maior entrega e os menores valores de consumo de energia e latência. Esse resultado é devido à técnica de ciclo de trabalho dinâmico do Rb-MAC. O modelo receptor entregou 1,1 mais pacotes que o emissor devido às retransmissões do Rb-MAC. O modelo emissor consome 9,6 vezes mais energia que o receptor, pois, nesse modelo, os nós não pertencentes ao caminho de roteamento são configurados com valores reduzidos de ciclo de trabalho. Além disso, o fato do modelo emissor transmitir pacotes de controle aumenta o seu consumo de energia. A latência do modelo emissor é 405,7 vezes maior que a do receptor, pois, nesse modelo, alguns nós ficam acordados e aguardando para rotear. Os números de transmissões e colisões não são mostrados por serem similares aos apresentados na seção anterior.



(a) Taxa de entrega



(b) Consumo médio de energia por nó



(c) Latência média de entrega

Figura 4. Parâmetros avaliados para o segundo cenário de simulação

5. Conclusões e Trabalhos Futuros

Este trabalho propõe o Rb-MAC, um protocolo para ser utilizado em soluções integradas com o roteamento baseado no receptor. O Rb-MAC explora o funcionamento básico desse tipo de roteamento para salvar recursos e permitir interações. Resultados de simulação mostram que o modelo receptor avaliado com o Rb-MAC aumentou a taxa de entrega, reduziu o número de transmissões e o consumo de energia quando comparado com um modelo baseado no emissor. Contudo, a latência e o número de colisões desse modelo foram menores. Resultados de simulação também revelaram que o Rb-MAC é capaz de manter a taxa de entrega para valores reduzidos de ciclo de trabalho. Além disso, o Rb-MAC possui uma técnica de ciclo de trabalho dinâmico que proporciona ao modelo receptor resultados de latência menores que os do modelo emissor avaliado, mas tal técnica é aplicada apenas em um cenário específico de disseminação.

A comunicação baseada no receptor é um tópico de pesquisa promissor, pois apresenta taxa de entrega elevada e consumo de energia reduzido. Um tópico de pesquisa que deve ser avaliado é o projeto integrado de

outras funções de rede (por exemplo, fusão de dados) com o modelo de comunicação baseados no receptor. Trabalhos futuros devem definir o ciclo de trabalho em função do maior valor aceitável para a latência.

Referências

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: A survey. *Computer Networks*, 38(4), 02.

[2] O. Goussevskaia, M. Machado, R. Mini, A. Loureiro, G. Mateus, and J. Nogueira. Data dissemination based on the energy map. *IEEE Communications Magazine*, 43(7), Jul. 05.

[3] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for networked sensors. *SIGPLAN Not.*, 35(11), 00.

[4] R. Jain. *The Art of Computer Systems Performance Analysis: techniques for experimental design, measurement, simulation and modeling*. 91.

[5] S. Jayashree, B. S. Manoj, and C. Murthy. On using battery state for medium access control in ad hoc wireless networks. *ACM/IEEE Mobicom*, 04.

[6] W. Liao and H. Wang. An asynchronous mac protocol for wireless sensor networks. *J. Netw. Comput. Appl.*, 31(4):807–820, 08.

[7] D. Niculescu and B. Nath. Trajectory-based forwarding and its applications. *ACM/IEEE International conference on Mobile computing and networking*, 03.

[8] S. Rashwand, J. Misic, V. Misic, S. Biswas, and M. Haque. A novel asynchronous, energy efficient, low transmission delay mac protocol for wireless sensor networks. *29th IEEE ICDCS*, Jun. 09.

[9] W. Ye, J. Heidemann, and D. Estrin. Medium access control with coordinated adaptive sleeping for wireless sensor networks. *IEEE/ACM Trans. Netw.*, 12(3), 04.

[10] M. Zorzi and R. Rao. Geographic random forwarding (geraf) for ad hoc and sensor networks: Energy and latency performance. *IEEE Transactions on Mobile Computing*, 2(4), 03.

Dois Pesos, Duas Medidas: Gerenciamento de Identidades Orientado a Desafios Adaptativos para Contenção de *Sybils*

Gustavo Mauch, Flávio Santos, Weverton Cordeiro, Marinho Barcellos, Luciano Gaspary

Instituto de Informática
Universidade Federal do Rio Grande do Sul (UFRGS)
Porto Alegre – RS – Brasil
{ghmauch, frsantos, wlccordeiro, marinho, paschoal}@inf.ufrgs.br

Abstract

The Sybil attack consists on the indiscriminate creation of counterfeit identities by a malicious user (attacker). An effective approach to tackle such attack consists of establishing computational puzzles to be solved prior to granting new identities. Despite its potentialities, solutions based on such approach do not distinguish between identity requests from correct users and attackers, and thus require both to afford the same cost per identity requested. To tackle this problem, in this paper we propose the use of adaptive computational puzzles to limit the spread of Sybils. We estimate a trust score of the source of identity requests in regard to the behavior of others. The higher the frequency a source requests identities, the lower its trust score and, consequently, the higher the complexity of the puzzle to be solved by the user(s) associated to that source. Results achieved by means of an experimental evaluation evidence our solution's ability to establish more complex puzzles to potential attackers, while minimally penalizing legitimate users.

Keywords: Peer-to-Peer Networks, Identity Management, Sybil Attack, Weak Identity Schemes, Computational Puzzles

Resumo

O ataque Sybil consiste na criação indiscriminada de identidades forjadas por um usuário malicioso (atacante). Uma abordagem promissora para mitigar esse ataque consiste em conceder novas identidades mediante a resolução de desafios computacionais. Apesar de suas potencialidades, as soluções baseadas em tal abordagem não distinguem solicitações de usuários corretos das de atacantes, fazendo com que ambos paguem o mesmo preço por identidade solicitada. Para lidar com esse problema, neste artigo propõe-se o uso de desafios adaptativos como limitante à disseminação de Sybils. Estima-se um grau de confiança da fonte de onde partem as solicitações de identidade em relação às demais. Quanto maior a frequência de solicitação de identidades, menor o grau de confiança e, consequentemente, maior a complexidade do desafio a ser resolvido pelo(s) usuário(s) associado(s) àquela fonte. Resultados obtidos por meio de experimentação mostram a capacidade da solução de atribuir desafios mais complexos a potenciais atacantes, penalizando minimamente usuários legítimos.

Palavras-chave: Redes Par-a-Par, Gerenciamento de Identidades, Ataque *Sybil*, Soluções Baseadas em Identidades Fracas, Desafios Computacionais

1. INTRODUÇÃO

O ataque *Sybil* [9] representa um dos mais elementares ataques de autenticidade em redes P2P, e consiste na criação de múltiplas identidades falsas, denominadas identidades (ou pares) *Sybil*. A idéia motivadora desse ataque é que um atacante possa controlar a maioria, ou pelo menos uma parte significativa, das identidades presentes na rede. Deste modo, toda interação entre pares terá grande chance de ser mediada por uma das identidades controladas e alterada da forma que mais aprouver ao seu controlador [2]. Um atacante com várias identidades falsas pode também

subverter algoritmos baseados em votação, manipulando com isso a reputação de pares ou conteúdos compartilhados na rede. Mais ainda, o ataque *Sybil* serve como base para o lançamento de outros ataques em redes P2P, tais como Eclipse [13] e *Free-riding* [10].

Uma abordagem bastante promissora para mitigar ataques *Sybil* consiste em atribuir ou renovar a concessão de identidades aos usuários solicitantes mediante a resolução de desafios computacionais [4]. A idéia por trás da exigência da resolução de desafios é que pares legítimos provem suas boas intenções com a rede, comprometendo uma parte de seus recursos. Ao mesmo tempo, pares maliciosos interessados em criar múltiplas identidades serão obrigados a passar grande parte de seu tempo processando desafios e, portanto, consumindo recursos, o que reduz seu poder de assumir um número elevado de identidades.

Diversos trabalhos foram publicados propondo o emprego de desafios computacionais para o gerenciamento de identidades em redes P2P [5, 4, 12]. Apesar de suas potencialidades, as propostas que adotam tal abordagem não fazem distinção entre solicitações de identidades oriundas de usuários corretos e de atacantes. A medida que ambos estão sujeitos ao pagamento do mesmo preço (computacional) por cada identidade solicitada, essas propostas podem não ser efetivas quando os recursos computacionais dos atacantes são muito superiores aos que os usuários legítimos dispõem. Assumindo desafios de uma determinada dificuldade, atacantes com *hardware* de maior capacidade conseguiriam resolver um conjunto muito superior de desafios e, com isso, obter um número elevado de identidades. Aumentar uniformemente a dificuldade dos desafios poderia, no outro extremo, tornar proibitivo o ingresso de pares legítimos à rede.

Para lidar com essa limitação, neste artigo propõe-se o uso de desafios adaptativos como estratégia de contenção contra a disseminação de *Sybils*. Em contraste com as propostas existentes na literatura, nossa solução estima um grau de confiança da *fonte* de onde parte a solicitação de identidade em relação ao comportamento das demais fontes. No contexto deste trabalho, *fonte* pode referir-se à estação de um usuário (identificada pelo seu endereço IP), à rede local a qual a estação pertença, a um sistema autônomo (*Autonomous System*, AS), etc. Essa decisão depende essencialmente da granularidade que se deseje ou seja possível adotar para a fonte (por exemplo, no caso de usuários posicionados atrás de redes usando NAT, a granularidade a ser considerada é associar todos os usuários daquela rede a uma única fonte). À medida que aumenta a frequência com que novas solicitações por identidades partem de uma dada fonte, diminui a confiabilidade da mesma. Consequentemente, maior será a complexidade do desafio computacional a ser resolvido antes que a identidade solicitada seja obtida pelo(s) usuário(s) associado(s) àquela fonte. Para avaliar a eficácia da solução proposta na contenção de ataques *Sybil*, foram realizadas simulações considerando traços históricos de solicitação de identidades em uma comunidade P2P. Os resultados obtidos mostram a capacidade da solução em atribuir desafios computacionais mais complexos a potenciais atacantes, ao passo que usuários legítimos são minimamente penalizados.

O restante do artigo está organizado como segue. A Seção 2 discute alguns dos principais trabalhos relacionados ao gerenciamento de identidades em redes P2P. A Seção 3 apresenta o mecanismo proposto para o uso de desafios adaptativos como uma proteção ao ataque *Sybil*, enquanto que a Seção 4 descreve a avaliação conduzida para avaliar a eficácia da mesma. A Seção 5 discute questões relacionadas ao emprego da solução proposta em arcabouços P2P. Por fim, a Seção 6 conclui o artigo com as considerações finais e possíveis desdobramentos para pesquisas futuras.

2. TRABALHOS RELACIONADOS

As investigações conduzidas para lidar com ataques *Sybil* podem ser classificadas de acordo com o mecanismo utilizado para garantir a autenticidade dos pares. As principais classes são *(i)* soluções baseadas em *identidades fracas*, e *(ii)* soluções baseadas em *identidades fortes*. A primeira reúne soluções em que cada par possui ampla autonomia para criar sua própria identidade. Nesse caso, são estabelecidas estimativas para número de identidades *Sybil* que são *aceitas* pelos demais pares. Tais estimativas podem ser úteis, por exemplo, para aplicações que tolerem uma fração previsível de pares *Sybil*. As soluções propostas por Yu Haifeng *et al.* [15, 14] e George Danezis *et al.* [8] são exemplos. Ambas exploram redes sociais para estimar o limite máximo de identidades *Sybil* presentes na rede em um dado momento. No entanto, elas apresentam problemas de violação de anonimidade e incapacidade de garantir autenticidade dos pares.

A segunda classe, por sua vez, reúne soluções nas quais os pares apenas podem obter identidades junto a entidades certificadoras. A principal vantagem é a dificuldade de um par criar e controlar várias identidades. No entanto, tais soluções podem diminuir a escalabilidade da rede P2P e inserir um ponto único de falha. Mais ainda, podem exigir que usuários confiem em entidades certificadoras desconhecidas, além de tornar inviável o acesso de potenciais usuários (por exemplo, quando for necessário informar dados pessoais ou pagar taxas para obter uma identidade). As propostas que se enquadram nessa classe buscam minimizar algumas das consequências danosas da introdução de uma entidade central. Por exemplo, em [11] e [1] os autores focaram na descentralização da infra-estrutura de distribuição de chaves públi-

cas (*Public-Key Infrastructure*, *PKI*). No entanto, essas propostas requerem a troca de uma grande quantidade de mensagens entre os pares, além de dependerem da colaboração de um número mínimo de pares para funcionarem como esperado.

No artigo que descreve o ataque *Sybil* [9], Douceur argumenta que não é possível resolver o problema de autenticação sem fazer uso de algum grau de centralização. Com base nessa afirmativa, e considerando as severas limitações decorrentes do uso de entidades certificadoras, tem ganhado força a corrente de soluções em que a concessão de identidades é feita mediante a resolução de desafios computacionais. O principal objetivo é causar a diminuição da capacidade que usuários maliciosos tem de criar identidades falsas, sem abrir mão das características "natas" de redes P2P (por exemplo, escalabilidade, descentralização e autonomia dos pares).

As soluções baseadas em desafios computacionais tem apresentado bons resultados ao utilizarem desafios que são criados ou verificados de forma distribuída. Nikita Borisov [4], por exemplo, mostrou a viabilidade da utilização de desafios gerados de forma distribuída e periódica, propondo uma solução na qual os pares que participaram da geração do desafio são capazes de validar a resolução dos mesmos. Em [12], por sua vez, propõe-se um esquema com múltiplas entidades geradoras de desafios. Esse esquema requer que, para a obtenção de identidades, usuários contatem uma dessas entidades e resolvam uma sequência de desafios propostos.

Apesar de promissoras, as soluções existentes não abordam a questão do dimensionamento da complexidade dos desafios. Ao utilizarem desafios com mesma complexidade computacional para todos os usuários, o problema que surge é buscar o melhor ponto de equilíbrio entre usar desafios mais complexos para coibir atacantes com alto poder computacional, e não tão complexos, para não penalizar usuários legítimos com *hardware* menos capacitado. Nesse contexto, a utilização de um peso e uma medida na atribuição dos desafios tenderá a favorecer os atacantes em detrimento dos usuários legítimos. O diferencial da proposta deste artigo é parametrizar a dificuldade dos desafios de acordo com o comportamento que cada fonte apresentar na rede. Os usuários associados a fontes cujo comportamento seja mais similar ao comportamento médio das demais fontes serão beneficiados com desafios menos complexos. Por outro lado, usuários associados a fontes com comportamento atípico deverão arcar com desafios mais custosos para a obtenção de identidades.

3. Proposta de Solução para Combater Ataques *Sybils*

A solução proposta neste artigo visa estabelecer o uso de desafios computacionais adaptativos para o gerenciamento de identidades em redes P2P. De uma forma geral, há três questões chave associadas à adoção de desafios adaptativos: (*i*) como caracterizar o comportamento das fontes (ou dos usuários associados às mesmas), (*ii*) como calcular o custo de um desafio a partir dos comportamentos observados, e (*iii*) como adaptar os desafios considerando a dinâmica do comportamento dos usuários da rede, observados pelo sistema como fontes de requisições de identidades. Cada uma dessas questões é abordada nas subseções a seguir.

3.1. Empregando *Taxas de Recorrências* para Caracterizar Comportamentos

Para permitir a caracterização do comportamento das diversas fontes de solicitação de identidades, duas métricas são introduzidas no contexto deste artigo: *taxa de recorrência da fonte* (ϕ) e *taxa de recorrência da rede* (Φ). A primeira reflete a frequência com que os usuários associados a uma determinada fonte solicitam novas identidades ao serviço de *bootstrap* da rede P2P em um intervalo de tempo t_w (com $t_w > 0$). A segunda, por sua vez, reflete a frequência média com que as fontes recorrem ao serviço de *bootstrap* para solicitar novas identidades.

O valor da taxa de recorrência da rede é calculado segundo a Equação 1, a qual utiliza a média harmônica das taxas de recorrências das fontes. Nessa equação, ϕ_i representa a taxa de recorrência da *i-ésima* fonte da rede P2P. Note que o caso em que $\phi_i = 0$ equivale à situação em que nenhum usuário da fonte i solicitou alguma identidade; nesse caso, tal fonte não é conhecida e, portanto, não é considerada para o cálculo da taxa de recorrência da rede.

$$\Phi = \frac{n}{\sum_{i=0}^{n} \frac{1}{\phi_i}} \tag{1}$$

É importante frisar que a média harmônica foi escolhida em detrimento de outras medidas estatísticas (média simples, média geométrica e a mediana) por ser especialmente resistente a alterações causadas por recorrências muito discrepantes do padrão observado (*outliers*). Essa característica é desejável e extremamente importante, uma vez que torna mais difícil para atacantes manipularem o comportamento da rede (por exemplo através de um ataque em conluio) para tornarem-se menos suspeitos. A resistência da taxa de recorrência da rede a ataques em conluio é avaliada em maior profundidade na Seção 4.

3.2. Calculando o Grau de Confiança a partir dos Comportamentos Observados

Considerando que o objetivo de um ataque *Sybil* é controlar uma fração significativa de identidades na rede P2P, para executá-lo o atacante deverá solicitar um grande número de identidades ao serviço de *bootstrap*. A consequência direta desse comportamento é um aumento da taxa de recorrência da fonte associada ao atacante. Por outro lado, é esperado que as fontes com usuários legítimos recorram minimamente para solicitar identidades (por exemplo, no momento que se registrarem na rede P2P). Logo, a idéia principal para conter ataques *Sybil* é atribuir desafios mais complexos ao(s) usuário(s) associado(s) às fontes cujas taxas de recorrência se tornarem superiores à taxa de recorrência da rede.

A partir da comparação entre o comportamento de cada fonte (inferido a partir de ϕ) e do comportamento considerado padrão para a rede (inferido a partir de Φ), é calculada a *relação entre as taxas de recorrências da fonte e da rede* (ρ). Obtida de acordo com a Equação 2, ela assume valores menores que zero para indicar quantas vezes a taxa de recorrência da fonte *i* é menor que a da rede, e maiores que zero para indicar quantas vezes a taxa de recorrência da fonte *i* é maior.

$$\rho = \begin{cases} -\frac{\Phi(t)}{\phi_i(t)} & \text{se } \phi_i(t) < \Phi(t) \\ \frac{\phi_i(t)}{\Phi(t)} & \text{se } \phi_i(t) \geq \Phi(t) \end{cases} \quad (2)$$

A relação entre as taxas de recorrências da fonte e da rede (ρ) serve como base para o cômputo do *grau de confiança da fonte* origem das solicitações de identidade (C). Esse grau, estimado para cada instante t de acordo com a Equação 3, assume valores no intervalo $[0, 1]$: em um extremo, valores mais próximos de 1 indicam uma maior confiança sobre a legitimidade do(s) usuário(s) associado(s) à fonte em questão; no outro extremo, valores mais próximos de 0 indicam maior desconfiança, isto é, uma maior probabilidade de que o(s) usuário(s) associados à fonte em questão está(ão) lançando um ataque *Sybil*. A complexidade do desafio computacional aplicado ao(s) usuário(s) será determinada pelo grau de confiança da fonte ao(s) qual(is) ele(s) está(ão) associado(s), no momento da solicitação de uma nova identidade. A Equação 3 é normalizada de modo que os extremos 0 e 1 representem total desconfiança e total confiança sobre uma determinada fonte, respectivamente.

$$C(t) = 0.5 - \frac{\arctan(a \times (\rho - c)^{(1+2\times b)})}{\pi} \quad (3)$$

A Figura 1 mostra quatro diferentes configurações que ilustram como varia o grau de confiança obtido para uma determinada fonte em função de ρ. Nessas configurações, os termos a, b e c da Equação 3 assumem valores arbitrários e desempenham um importante papel no controle da agressividade com que as configurações decrescem, da amplitude e da translação das mesmas, respectivamente.

A partir da Figura 1 é possível observar duas propriedades importantes que a Equação 3 apresenta. A primeira reside no fato do valor de confiança ter variações mínimas para valores de ρ mais próximos de 0 (situação em que a fonte se comporta de forma semelhante ou igual à média da rede), proporcionando assim uma certa tolerância na avaliação dos comportamentos das fontes. Considerando por exemplo a configuração ($a = 0,1$, $b = 2$, $c = 0$) na Figura 1, dentro do intervalo $-2 \leq \rho \leq 2$, variações são pouco consideradas, por serem ligeiramente semelhantes ao padrão observado na rede. Os comportamentos que desviam significativamente desse intervalo, no entanto, terão atribuídos menores (ou maiores) valores de confiança, como pode ser observado pelas súbitas variações da configuração ($a = 0,1$, $b = 2$, $c = 0$) nos intervalos $-5 \leq \rho \leq -2$ e $2 \leq \rho \leq 5$. A segunda propriedade reside no fato de ser assintótica em 0 e 1. Desse modo, para $\rho \to -\infty$ ou $\rho \to +\infty$, sempre haverá um valor de confiança associado.

3.3. Lidando com a Dinâmica do Comportamento dos Usuários da Rede

Uma característica importante de redes P2P é a ampla autonomia concedida aos pares. Desse modo, os pares podem entrar e sair da rede de acordo com seus interesses e disponibilidade, sem depender de entidades externas. Um dos possíveis desdobramentos dessa dinamicidade é a ocorrência de variações constantes (e eventualmente significativas) do padrão de comportamento tanto das fontes quanto da rede P2P como um todo. A seguir, é discutido como o mecanismo proposto lida com a dinâmica dos comportamentos observados.

A Equação 3, embora seja capaz de determinar a confiança de uma determinada fonte no instante t, não considera o histórico de comportamento da mesma. Com o objetivo de representar de modo apropriado o grau de confiança de uma determinada fonte, ao mesmo tempo considerando o histórico do comportamento da mesma, é inserido na solução um parâmetro β, o qual permite o cálculo da confiança suavizada, C_s, conforme apresentado na Equação 4. O parâmetro β é um fator de suavização que determina o peso do passado no cálculo do valor de confiança para o instante atual (t), e assume valores no intervalo (0,1]. Em um extremo, valores de β mais próximos de 0 conferem um peso maior ao comportamento histórico da fonte em questão. Em outro extremo, valores de β mais próximos de 1 dão um peso maior ao comportamento atual da fonte. No caso especial em que $\beta = 1$, o valor de confiança atual (tal como calculado pela Equação 3) é considerado integralmente, sendo o passado totalmente desconsiderado.

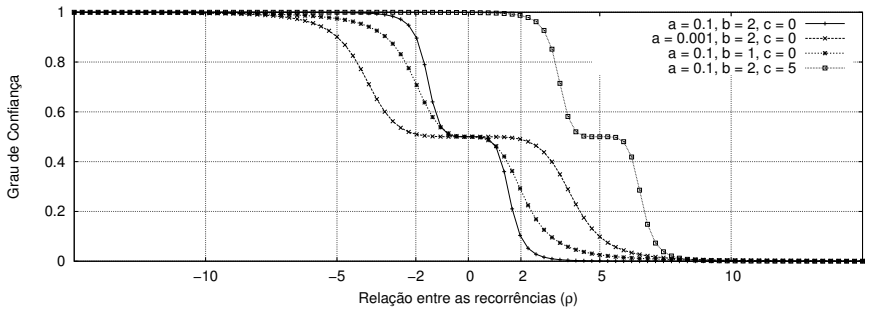


Figura 1. Exemplos de valores para os parâmetros a, b, e c da Equação 3 para cálculo do grau de confiança da fonte

$$C_s(t) = \beta \times C(t) + (1 - \beta) \times C(t - 1) \qquad (4)$$

A adição do parâmetro β ao cálculo da confiança é importante para tratar adequadamente as alterações no comportamento de cada fonte de solicitação de identidades. Em particular, as alterações intencionais e repentinas de comportamento, de usuários interessados em obter benefícios, como os *traidores*, são capturadas e refletidas no grau de confiança da fonte à qual o mesmo está associado. Um traidor é um atacante que busca angariar altos valores de confiança em sistemas de reputação e passa, então, a se aproveitar dela para prejudicar outros pares, ou obter vantagens indevidas. O correto dimensionamento do valor de β, nesse contexto, pode impedir que um traidor manipule a solução proposta de modo que a fonte em que se situa consiga (ou recupere) rapidamente uma alta confiança do sistema. Na medida em que o passado é considerado para determinar o presente, somente aquelas fontes cujos usuários apresentem bom comportamento histórico serão considerados confiáveis.

Outra questão importante, ainda em relação à dinâmica do comportamento da rede, reside no fato de que as taxas de recorrência podem variar em épocas diferentes. Por exemplo, é razoável esperar que em determinados períodos mais usuários estejam interessados em ingressar na rede P2P e, consequentemente, mais requisições por identidades sejam realizadas. Por outro lado, também é razoável esperar uma queda no número de usuários que ingressam na rede em outros períodos, o que se reflete em menos requisições. Sem considerar essa sazonalidade no comportamento dos usuários (e da rede como um todo), usuários legítimos podem ser considerados suspeitos pela solução proposta caso suas fontes solicitem identidades com maior frequência, mesmo que estejam acompanhando o comportamento dos demais usuários. Por outro lado, caso todas as solicitações desde o início ($t = 0$) fossem consideradas, seria mais fácil para um atacante lançar mão de ataques *Sybils*. Isso seria possível visto que a quantidade de requisições cresceria indefinidamente, consequentemente ofuscando as altas taxas de recorrência de fontes suspeitas.

Para acomodar questões de sazonalidade no comportamento dos pedidos de identidades, optou-se por utilizar uma *janela deslizante* – um intervalo de tempo t_w, que se inicia no passado e termina no momento presente – para restringir a quantidade de requisições a serem consideradas no cálculo das taxas de recorrência de cada fonte e da rede. Note que t_w corresponde ao tempo considerado para calcular a taxa de recorrência ϕ de cada fonte no sistema e, consequentemente, a taxa de recorrência da rede, Φ (tal como discutido na Subseção 3.1). A medida em que o tempo passa, a janela avança em passos com duração t_d (com $t_d \leq t_w$); com isso, as solicitações de identidade mais antigas vão sendo desconsideradas, dando lugar à solicitações mais recentes, as quais são mais representativas do estado atual da rede P2P.

4. Avaliação da Solução Proposta

Para avaliar a viabilidade técnica, a eficácia e a eficiência do uso de desafios adaptativos para combater *Sybil* em redes P2P, foi realizada a implementação prototípica de um serviço de *bootstrap*. Por meio desse protótipo, foram executados diversos experimentos, considerando solicitações sintéticas de identidades baseadas em traços históricos de uma comunidade P2P. Como resultados da avaliação conduzida, procurou-se observar que (*i*) os desafios computacionais propostos para usuários legítimos penalizam minimamente os mesmos, (*ii*) desafios atribuídos a potenciais atacantes possuem maiores complexidades computacionais, e (*iii*) a solução proposta é robusta e resiliente mesmo na presença de uma fração significativa de atacantes, bem como sob a ocorrência de ataques em conluio.

O restante desta seção está organizado como segue. A Subseção 4.1 descreve a configuração do ambiente considerado na análise (características dos traços históricos de solicitações de identidades usados, parâmetros da

solução, etc.). A Subseção 4.2, por sua vez, apresenta os resultados obtidos pela solução proposta na contenção de ataques *Sybil*.

4.1. Configuração do Ambiente de Experimentação

A Tabela 1 apresenta um resumo das características do traço utilizado nos experimentos, dos valores para os parâmetros envolvidos na solução, e a característica dos ataques *Sybil* considerados na análise da eficácia e eficiência da proposta. Cada um destes é discutido em detalhes a seguir.

Os experimentos foram feitos com base em traços históricos de solicitações de identidades na comunidade P2P fechada Bitsoup [3]. Uma vez que a admissão na comunidade é feita mediante autenticação por usuário e senha, e a criação de novas contas é moderada, assumiu-se para os fins da avaliação que o traço não contém registros de ataques *Sybils*. Essa premissa baseia-se na idéia de que o custo necessário para criar e manter diversas identidades falsas em uma comunidade fechada com moderação é de várias ordens de grandeza maior do que em uma comunidade aberta sem moderação.

Os traços considerados registram atividades de solicitação de identidades durante 15 dias consecutivos. Durante esse período, foram obtidas 625.079 identidades, solicitadas por 44.315 fontes distintas, perfazendo uma média de $14,21$ solicitações de identidades por fonte, e uma taxa global de 1 solicitação a cada $2,08$ segundos.

Os parâmetros envolvidos na experimentação foram definidos como segue. O valor de β foi definido como $0,125$, isto é, o comportamento histórico do grau de confiança possui um peso de 87,5% sobre o valor atual. Esses valores mostraram-se adequados, após sucessivas experimentações (omitidas neste artigo por questões de restrição de espaço), para impedir que fontes com histórico de *mal comportamento* alcançassem valores elevados de confiança ao tornarem-se repentinamente *bem comportadas*. Os valores de a, b e c, por sua vez, foram definidos como $0,1$, 2 e 5, respectivamente, visando controlar a forma como a relação entre as recorrências se reflete no grau de confiança obtido pela fonte. Com esses valores, uma taxa de recorrência da fonte similar ou igual a da rede ($\phi \simeq \Phi$) fará com que a fonte alcance um grau de confiança de aproximadamente 1 (por exemplo, vide configuração $a = 0,1$, $b = 2$ e $c = 5$ na Figura 1). A janela deslizante tem duração de 8 horas ($t_w = 8 \times 60$ *min*) e desliza de hora em hora ($t_d = 1 \times 60$ *min*). A duração de 8 horas mostrou ser adequada considerando as características da comunidade Bitsoup.org (materalizadas nos traços históricos estudados), sendo capaz de capturar adequadamente o comportamento passado de cada usuário, e ao mesmo tempo desconsiderando solicitações que não refletem mais o estado atual da rede. O deslizamento de hora em hora, por sua vez, mostrou-se adequado para capturar a evolução nos comportamentos dos participantes da rede, sem impor uma sobrecarga maior ao processo de cálculo do grau de confiança.

Para avaliar cenários em que a rede encontra-se sob ataque *Sybil*, foram injetadas artificialmente solicitações maliciosas de identidades, considerando duas estratégias diferentes. Na primeira, o atacante lança um ataque *Sybil* a partir de uma única fonte. A segunda estratégia, por sua vez, considerou que o atacante possui a sua disposição um determinado número de fontes. Nesse caso, o ataque *Sybil* realizado é distribuído, com solicitações partindo de cada uma das fontes sob o controle do atacante – cada fonte solicita uma quantidade pequena de identidades, de modo que não sejam classificadas como suspeitas. Em ambos os casos, busca-se avaliar a quantidade de identidades que um atacante consegue solicitar por meio do ataque *versus* a dificuldade do desafio que o sistema atribui para cada nova solicitação vinda de uma das fontes envolvidas no ataque.

4.2. Resultados Obtidos e Análise

Para organizar a discussão dos resultados obtidos, primeiramente é discutida a sobrecarga causada a usuários legítimos, em situações em que não há ocorrência de ataques *Sybil* na rede P2P. Em seguida, é avaliada a efetividade da solução na contenção de ataques *Sybil*, e o impacto que estes causam nos desafios com os quais usuários legítimos terão de arcar. Por fim, é analisada a sua resiliência em situações em que diversos atacantes agem em conluio, com o propósito específico de atacar a própria solução.

Sobrecarga Causada a Usuários Legítimos na Ausência de Ataques *Sybil*: A Figura 2 exibe a função cumulativa complementar de distribuição (*Complementary Cumulative Distribution Function*, *CCDF*) das confianças calculadas para as solitações de identidades partindo das fontes (consideradas legítimas) do traço estudado. É importante ressaltar que esse resultado refere-se somente às solicitações de identidade presentes no traço original, não tendo sido perturbado pela ocorrência de ataques *Sybil*.

É possível notar no gráfico da Figura 2 que a maioria das solicitações de identidade geradas pelas fontes é de alta confiança. Por exemplo, aproximadamente 45% das solicitações de identidades foram realizadas por usuários oriundos de fontes com confiança maior ou igual a $0,9$. Em outras palavras, existe um grau de confiança igual ou maior do que $0,9$, para aproximadamente 45% das solicitações, de que as mesmas não estejam relacionadas a um ataque *Sybil*. Esse percentual aumenta para 60% se considerarmos as solicitações com confiança maior ou igual a $0,7$, e para aproximadamente 75% se considerarmos

Tabela 1. Informações sobre o ambiente considerado na avaliação experimental.

Características do Traço Empregado			
Duração		15 dias	
Quantidade de identidades solicitadas		625.079 identidades	
Número de fontes distintas		44.315 fontes	
Intervalo médio entre requisições		2,08 segundos	
Quantidade média de requisições por fonte		14,21 identidades	
Parâmetros da Solução			
a	0,1	b	2
c	5	β	0,125
Duração da Janela (t_w)	8 horas	Passo da Janela (t_d)	1 hora
Estratégias de Ataque			
Taxa de requisição por fonte atacante		1; 1,25; 1,5; 2; e 2,5 requisições/hora	
Quantidade de fontes atacantes		1; 100; 500; 1.000; e 2.000	

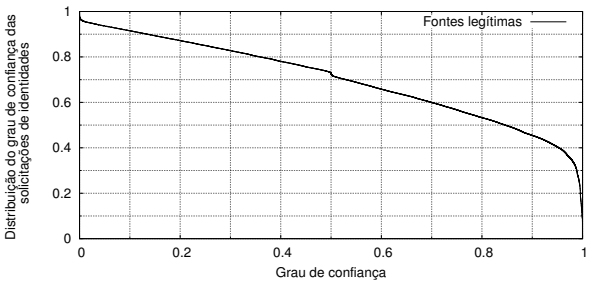


Figura 2. *CCDF* do grau de confiança de solicitações de identidades originadas por fontes legítimas

aquelas com confiança maior ou igual a $0,5$. Com esses valores de confiança obtidos, uma significativa fração das fontes obterá desafios computacionais de menor complexidade, logo causando mínimo impacto para os respectivos usuários.

Um aspecto importante a ser discutido sobre os resultados da Figura 2 diz respeito aos 25% das fontes com confiança menor que $0,5$. Embora as fontes contidas no traço sejam presumidamente legítimas (isto é, não lançaram algum ataque *Sybil* contra a rede P2P), existem casos em que as fontes podem recorrer mais vezes que a média da rede para solicitar identidades. Esse é o caso, por exemplo, em que vários usuários acessam a Internet através de redes utilizando o mecanismo de NAT, o qual faz com que os mesmos sejam associados à uma única fonte. De qualquer forma, o número de usuários afetados no experimento executado foi mínimo. Pouco mais de 10% das fontes alcançou valores de confiança menores ou iguais a $0,2$.

Impacto Causado a Potenciais Atacantes: A Figura 3 apresenta os resultados obtidos a partir do desdobramento do cenário ilustrado na Figura 2 em cinco novos cenários, cada um sob os efeitos de um ataque *Sybil* gerado artificialmente. Os ataques, em cada um dos cenários, são orquestrados por uma única fonte (maliciosa). A principal diferença entre os mesmos reside nas taxas de solicitação de identidades adotadas: 1; $1,25$; $1,5$; 2; e $2,5$ solicitações por hora, respectivamente.

Uma observação importante em relação aos resultados apresentados na Figura 3 corresponde à influência do ataque *Sybil* sobre o grau de confiança obtido pelas fontes legítimas. Independente da taxa de solicitação de identidades adotada pelo atacante, as curvas que mostram a distribuição do grau de confiança dos pares legítimos mantêm-se inalteradas e idênticas. Tal se deve à resistência da média harmônica – medida empregada para calcular a taxa de recorrência da rede, conforme discutido na Seção 3.1 – à presença de taxas de recorrência com desvio significativo em relação as das demais fontes. Por questões de legibilidade, apenas uma curva é apresentada na Figura 3 para ilustrar a distribuição do grau de confiança das fontes legítimas.

Analisando os resultados da Figura 3 por uma perspectiva diferente, é possível observar que um aumento gradual na taxa de recorrência da fonte maliciosa é suficiente para que a mesma sofra quedas significativas no seu grau de confiança. Por exemplo, quando a taxa de recorrência da fonte maliciosa corresponde a 1 solicitação por hora (cenário 1), aproximadamente 100% das solicitações de identidades partindo daquela fonte obteve grau de confiança igual a $0,5$ (isto é, um grau de confiança de $0,5$ de que a solicitação não está relacionada a um ataque *Sybil*). Para a taxa de $1,25$, por sua vez, apenas 10% das solicitações de identidades obteve grau de confiança maior ou igual a $0,3$. No cenário 5, o mais extremo ilustrado, a taxa de $2,5$ faz com que todas as solicitações de identidades partindo da fonte maliciosa sejam consideradas como parte de um ataque *Sybil* (uma vez que 100% das solicitações de identidade obteve grau de confiança 0). A consequência direta das quedas observadas é a

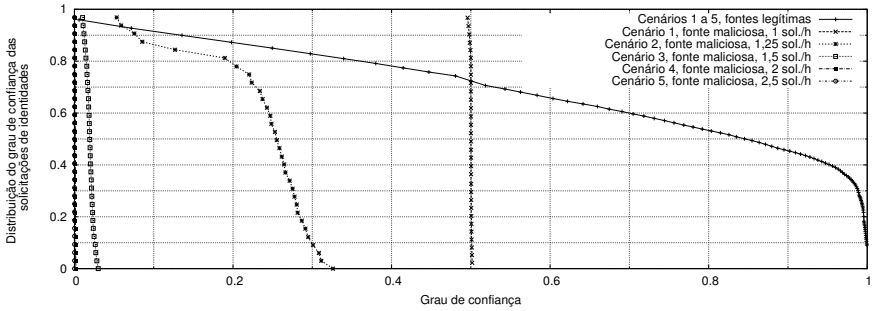


Figura 3. Resistência da solução proposta à ataques *Sybil* partindo de uma única fonte maliciosa, considerando diferentes taxas de recorrência

imposição, aos usuários associados à fonte maliciosa, de desafios computacionais de complexidade computacional extrema.

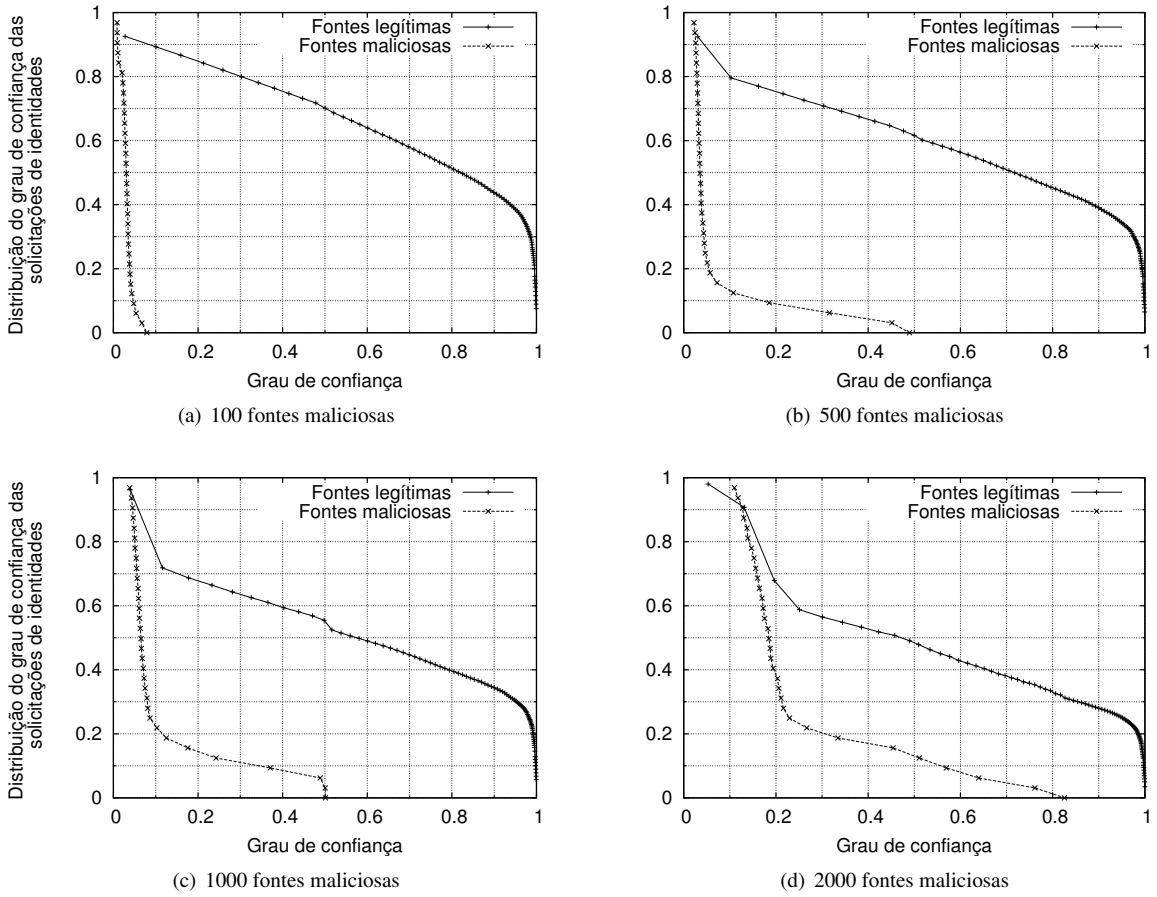
Os resultados apresentados levam a duas conclusões distintas, dependendo da perspectiva pela qual são analisados. Por um lado, evidenciam que a solução proposta reage adequadamente ao aumento na taxa de recorrência das fontes, penalizando severamente aquelas que recorrem a uma taxa muito maior que a média observada na rede. Por outro lado, mostra que a solução compele as fontes a se "comportarem adequadamente" – isto é, recorrendo harmonicamente em relação às demais fontes – caso não desejem ser penalizadas com desafios computacionais mais complexos.

Resiliência da Solução Proposta a Ataques em Conluio: Após ter-se analisado o efeito de um ataque *Sybil* considerando uma única fonte, avaliou-se o efeito do ataque realizado de forma distribuída, isto é, considerando múltiplas fontes como origem das solicitações de identidade. Nesse caso, ao invés de aumentar a taxa de recorrência para obter mais identidades falsas, o atacante age em conluio com outros atacantes (ou lança mão de uma *botnet* formada por várias estações zumbi conectadas à Internet). São dois os objetivos do lançamento de um ataque *Sybil* em conluio. Primeiro, busca-se aumentar a velocidade com que o atacante obtém identidades falsas na rede P2P, sem ter de arcar com desafios mais complexos. Segundo, procura-se alterar a percepção de normalidade da rede. Em outras palavras, parte-se da idéia de que mais fontes maliciosas atuando com o mesmo comportamento tende a mudar a percepção sobre qual é, efetivamente, o comportamento da maioria das fontes na rede.

A Figura 4 apresenta os resultados obtidos considerando a nova estratégia de ataque. Quatro cenários distintos são considerados, cada um com um número distinto de fontes maliciosas à disposição do atacante: 100; 500; 1000; e 2000 fontes. Em todos os cenários, cada fonte maliciosa atua a uma taxa de $1,5$ solicitações de identidades por hora. Essa taxa foi escolhida porque permite ao atacante obter um número significativo de identidades e, ao mesmo tempo, passar mais despercebido como um atacante na rede (conforme evidenciado na análise anterior).

Observe na Figura 4 que, mesmo utilizando uma quantidade extremamente alta de fontes para lançar o ataque *Sybil*, o efeito que o atacante consegue exercer sobre o padrão de normalidade da rede é relativamente limitado. Por exemplo, na Figura 4 (a), 70% das solicitações de identidades foram realizadas por fontes com confiança maior ou igual a $0,5$. Esse percentual decresce para 61% na Figura 4 (b), 56% na Figura 4 (c) e aproximadamente 50% na Figura 4 (d).

Em contrapartida, as fontes associadas aos atacantes continuam a apresentar um comportamento discrepante em relação às demais fontes. Apesar de os atacantes conseguirem algum sucesso no ataque em conluio, estes continuam a obter baixíssimos valores de confiança (logo, desafios computacionais mais complexos). Com 100 fontes, nenhuma solicitação obtém grau de confiança maior ou igual a $0,05$. Embora haja um ganho considerável no ataque para o caso em que 500 fontes são empregadas, apenas 13% das solicitações obtiveram confiança maior ou igual a $0,1$. Para o caso com 1000 fontes, 15% das solicitações obtiveram confiança maior ou igual a $0,2$, e para o caso com 2000 fontes, 35% das solicitações. Esses resultados evidenciam, ao mesmo tempo, a robustez e a eficácia da solução proposta frente a ataques *Sybil*, mesmo quando estes ocorrem em conluio. Mais importante, mostra que o atacante precisa dedicar uma gigantesca quantidade de recursos para obter sucesso no ataques, tanto em termos de *fontes* distribuídas (para despistar o esquema de diferenciação por fontes de solicitação), como em termos de capacidade computacional (para resolver os desafios propostos).



(a) 100 fontes maliciosas

(b) 500 fontes maliciosas

(c) 1000 fontes maliciosas

(d) 2000 fontes maliciosas

Figura 4. Resiliência da solução proposta à ataques *Sybil* partindo de várias fontes maliciosas, considerando uma mesma taxa de recorrência

5. Discussões sobre a Solução Proposta

Conforme apresentado na Seção 3, nossa solução baseia-se na exigência da resolução de desafios computacionais antes que potenciais usuários obtenham identidades que os permitam ingressar na rede P2P. É importante frisar que a implantação da solução requer mínimas alterações nas *entidades* que compõem a rede. Considerando a sua instanciação em um arcabouço P2P tal como o BitTorrent [6], por exemplo, uma sequência de passos para a obtenção de identidades seria: (*i*) usuário solicitar uma identidade juntamente ao *tracker*; (*ii*) *tracker* requisitar a resolução de um desafio computacional ao usuário solicitante; (*iii*) usuário responder ao *tracker* o desafio computacional proposto; e (*iv*) *tracker* conferir a resolução e continuar o processamento segundo o protocolo BitTorrent original, caso a resolução esteja correta. Nesse caso, as modificações mais significativas para instanciar a solução ficariam restritas ao *tracker* – de modo que este passasse a manter informações sobre taxas de recorrência dos usuários, calcular seus respectivos valores para grau de confiança, e definir a complexidade do desafio computacional a ser resolvido em função do grau de confiança atual.

Sobre os parâmetros da solução proposta (a, b, c, β, t_w, e t_d), atualizações na valoração dos mesmos são necessárias para refletir mudanças de mais longo prazo nas características da rede (por exemplo, aumento do número de usuários ou mudanças no perfil da comunidade). Nesse caso, ajustes nos parâmetros de tempos em tempos (na ordem de meses, por exemplo) pode ser feito pelo administrador da rede, considerando sua própria experiência ou com o apoio de heurísticas e/ou mecanismos automatizados. Por outro lado, é importante mencionar que ajustes para adaptar a solução à mudanças de mais curto prazo não são necessários, uma vez que as métricas *taxa recorrência da fonte* e *taxa de recorrência da rede* são capazes de capturar variações no comportamento na rede que ocorrem nesse período de tempo.

Focando agora no mapeamento do grau de *confiança* em complexidade do desafios, tal depende essencialmente da natureza do mesmo. Por exemplo, considere o desafio

apresentado em [9]: dado um número aleatório suficientemente grande y, encontrar dois números x e z em um período de tempo limitado tal que a concatenação $x|y|z$, após processada por uma função *hash* segura, leva a um número cujos n bits menos significantes são todos 0. Uma vez que o tempo para resolver esse desafio é proporcional a 2^{n-1}, e o tempo para verificar a resolução é constante, uma estratégia de mapeamento seria adotar uma função $f(x)$, que recebe como parâmetro o grau de confiança, e retorna um número inteiro n que define a complexidade do desafio.

Por fim, em relação a materialização da noção de fonte, uma estratégia é considerar um endereço IP, uma subrede ou um sistema autônomo como uma fonte distinta. Outra estratégia seria o uso de sistemas de coordenadas de rede, por exemplo o Vivaldi [7], para distinguir solicitações vindas de determinadas regiões, cidades, estados, ou mesmo países.

6. Considerações Finais

O emprego de desafios computacionais é uma alternativa que tem se mostrado promissora para combater a ocorrência de ataques *Sybils* em redes P2P, e que tem recebido massiva atenção da comunidade de pesquisa. Entretanto, a falta de mecanismos que permitam lidar adequadamente com situações em que existe significativa disparidade de poder computacional entre usuários legítimos e atacantes impede o seu uso mais efetivo e disseminado. Para lidar com essa limitação, nesse artigo foi proposto o uso de desafios computacionais adaptativos como limitante à disseminação de *Sybils*.

Os experimentos realizados, embora não exaustivos, evidenciaram a capacidade da solução proposta em diminuir a capacidade dos atacantes de criarem identidades falsas de forma indiscriminada, ao mesmo tempo sendo favorável a usuários legítimos, os quais foram, em geral, penalizados minimamente. Ao calcular valores de confiança menores a fontes com taxas de solicitação de identidade mais altas, os usuários (maliciosos) atrelados a essas fontes tiveram de arcar com desafios computacionais de maior complexidade. Por outro lado, os usuários associados a fontes presumidamente legítimas (e que recorreram menos vezes para solicitar identidades), receberam desafios computacionais menos complexos (dados os maiores valores de grau de confiança que as fontes em questão possuiam perante a rede P2P).

Como trabalhos futuros, pretende-se (*i*) estender a solução proposta para capturar o comportamento das fontes face o atraso associado à resolução dos desafios, posto que hoje tal não é considerado por questões de simplicidade; (*ii*) investigar um mecanismo que apóie a valoração mais adequada dos parâmetros da solução proposta considerando comunidades com características distintas; e (*iii*) instanciar a solução proposta em um arcabouço P2P, por exemplo o BitTorrent.

Agradecimentos

Agradecimentos ao Prof. Nazareno Andrade (UFCG), pela concessão dos traços históricos do Bitsoup.org utilizados na avaliação experimental apresentada no artigo.

Referências

[1] Karl Aberer, Anwitaman Datta, and Manfred Hauswirth. A decentralized public key infrastructure for customer-to customer e-commerce. In *International Journal of Business Process Integration and Management*, pages 26–33, 2005.

[2] Marinho Pilla Barcellos and Luciano Paschoal Gaspary. Fundamentos, tecnologias e tendencias rumo a redes p2p seguras. *Jornadas de Atualizações em Informática*, pages 187–244, July 2006.

[3] Bitsoup.org. Bitsoup.org – the number one site for your torrent appetite. http://bitsoup.org/, 2009.

[4] N Borisov. Computational puzzles as sybil defenses. In *6th IEEE International Conference on Peer-to-Peer Computing (P2P 2006)*, pages 171–176, September 2006.

[5] Miguel Castro, Peter Drushel, Ayalvadi Ganesh, Antony Rowstron, and Dan S. Wallach. Secure routing for structured peer-to-peer overlay networks. In *5th Usenix Symposium on Operating Systems Design and Implementation (OSDI 2002)*, pages 299–314, 2002.

[6] Bram Cohen. Incentives Build Robustness in BitTorrent. http://citeseer.ist.psu.edu/579364.html, 2003.

[7] Frank Dabek, Russ Cox, Frans Kaashoek, and Robert Morris. Vivaldi: a decentralized network coordinate system. *SIGCOMM Comput. Commun. Rev.*, 34(4):15–26, October 2004.

[8] George Danezis, Chris Lesniewski-Laas, Frans M. Kaashoek, and Ross Anderson. Sybil-resistant dht routing. pages 305–318. 2005.

[9] John R. Douceur. The sybil attack. In *1st International Workshop on Peer-to-Peer Systems (IPTPS 2002)*, pages 251–260, 2002.

[10] Michal Feldman, Christos Papadimitriou, John Chuang, and Ion Stoica. Free-riding and whitewashing in peer-to-peer systems. *IEEE Journal on Selected Areas in Communications*, 24(5):1010–1019, 2006.

[11] Ruggero Morselli, Bobby Bhattacharjee, Jonathan Katz, and Michael A. Marsh. Keychains: A decentralized public-key infrastructure. 2006.

[12] H. Rowaihy, W. Enck, P. McDaniel, and T. La Porta. Limiting sybil attacks in structured p2p networks. In *26th IEEE International Conference on Computer Communications (INFOCOM 2007)*, pages 2596–2600, Anchorage, Alaska , USA, May 2007.

[13] Atul Singh, Tsuen-Wan Ngan, Peter Druschel, and Dan S. Wallach. Eclipse attacks on overlay networks: Threats and defenses. In *25th Conference on Computer Communications (INFOCOM 2006)*, pages 1–12, Barcelona, Catalunya, Spain, 2006.

[14] Haifeng Yu, Phillip B. Gibbons, Michael Kaminsky, and Feng Xiao. Sybillimit: A near-optimal social network defense against sybil attacks. In *IEEE Symposium on Security and Privacy*, pages 3–17. IEEE Computer Society, 2008.

[15] 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.

Revista Brasileira de Redes de Computadores e Sistemas Distribuídos

LARC - Laboratório de Redes de Computadores
SBC - Sociedade Brasileria de Computação
Universidade Federal do Rio de Janeiro
Núcleo de Computação Eletrônica