

Autoconfiguração de Endereços Baseada em Filtros de Bloom para Redes Ad Hoc

Natalia Castro Fernandes e Otto Carlos Muniz Bandeira Duarte*

¹GTA/PEE/COPPE – Universidade Federal do Rio de Janeiro
Rio de Janeiro, RJ - Brasil

Abstract. *In this paper, we propose the Bloom Filter based address autoconfiguration protocol for ad hoc networks (FIBRA). The goal of our protocol is to dynamically allocate addresses in ad hoc networks with high efficiency in solving address collisions and reducing control traffic volume, even with packet losses. We present a probabilistic analysis of address collisions and discuss Bloom Filter functionalities in the address autoconfiguration. We evaluate the performance of FIBRA for mobile and static scenarios, and in the presence of network partitions. Simulation results show that FIBRA reduces up to 13 times the control traffic when compared to the other address autoconfiguration protocols analyzed and solves all the address collisions, using a few node resources.*

Resumo. *Este artigo propõe o protocolo para a autoconfiguração de endereços baseado em Filtros de Bloom para Redes Ad hoc (FIBRA). O protocolo proposto objetiva a alocação dinâmica de endereços em redes ad hoc com uma alta eficiência em termos de capacidade de solução de colisões de endereço, redução do volume de tráfego de controle e robustez à perda de pacotes. Uma análise da probabilidade de colisão de endereços é apresentada e são discutidas as funcionalidades do Filtro de Bloom na autoconfiguração de endereços. O desempenho do FIBRA é avaliado em cenários com nós móveis e estáticos e na presença de partições na rede. Os resultados da simulação demonstram que o FIBRA reduz em até 13 vezes a carga de controle quando comparado aos outros protocolos de autoconfiguração de endereços analisados, além de solucionar os casos de colisão, utilizando poucos recursos dos nós.*

1. Introdução

As redes ad hoc são redes sem infra-estrutura que podem ser utilizadas em diversos cenários, tais como sensoriamento, acesso à Internet para populações carentes e recuperação de desastres. Tipicamente, o endereçamento dos nós deve ser compatível com o *Internet Protocol* (IP), que é o protocolo utilizado pela maioria das aplicações e disponível em todos os sistemas operacionais. Dessa forma, cada nó deve ter um endereço IP único, contendo um prefixo de rede e um sufixo de máquina. Com a utilização do IP, a alocação dinâmica de endereços se torna fundamental para redes onde o espaço de endereçamento é menor que o número de dispositivos que podem ser interconectados. A alocação dinâmica também é importante para redes criadas para eventos e recuperação de desastres, nas quais não é possível fazer uma alocação estática de endereços, porque não se conhece informações sobre os dispositivos que irão utilizar a rede.

*Este trabalho foi realizado com recursos do CNPq, CAPES, FINEP, RNP e FAPERJ.

Nas redes com infra-estrutura, como as redes cabeadas e as redes sem-fio em malha, a alocação automática de endereços pode ser feita de forma centralizada por um servidor responsável por controlar os endereços disponíveis. Nas redes sem infra-estrutura, como é o caso das redes ad hoc, nenhum nó da rede está sempre disponível a todos os outros nós para atuar como servidor de alocação de endereços. Assim, se torna necessário um protocolo para a autoconfiguração dos endereços que funcione de forma distribuída. Esse protocolo deve consumir poucos recursos, pois muitos dispositivos sem-fio, como sensores, assistentes digitais pessoais (*Personal Digital Assistant* - PDA) e computadores portáteis, possuem restrições de processamento, memória, armazenamento e bateria.

Nesse artigo é proposto o protocolo de autoconfiguração de endereços baseado em Filtros de Bloom para Redes Ad hoc (FIBRA). No FIBRA, cada nó controla de forma distribuída os endereços disponíveis através do uso de Filtros de Bloom, que são estruturas que possuem a propriedade de representar informações de forma compacta. A utilização do Filtro de Bloom no FIBRA permite também observar a união de partições na rede, situação comum em redes ad hoc, além de reduzir o número de mensagens para solução de colisões de endereços. Simulações mostram que, mesmo com perdas de pacote, o protocolo proposto consegue distribuir os endereços e solucionar as colisões, pois mensagens de controle perdidas são identificadas no Filtro de Bloom, evitando que colisões de endereço não sejam detectadas. Além disso, o protocolo proposto atende às restrições de bateria, processamento e armazenamento comuns em dispositivos portáteis e móveis.

O restante do artigo está distribuído da seguinte forma. Na Seção 2 são apresentados trabalhos mais relevantes relacionados à alocação de endereços, enquanto que a Seção 3 apresenta uma descrição detalhada da proposta. Na Seção 4, é descrito o ambiente de simulação e, na Seção 5, são apresentados os resultados obtidos. Por fim, na Seção 6, são apresentadas as conclusões.

2. Trabalhos Relacionados

A utilização de protocolos de alocação de endereços baseados no paradigma cliente/servidor desenvolvidos para as redes infra-estruturadas, como o *Dynamic Host Configuration Protocol* (DHCP) [Droms, 1997], não é viável em redes sem infra-estrutura devido às frequentes desconexões e à ausência de nós com características específicas de servidores. Uma proposta para solucionar esse problema foi a adoção de endereçamento baseado em *hardware*, como sugerido na autoconfiguração de endereços sem estados do IPv6 [Thomson e Narten, 1998]. A proposta é utilizar um endereço formado por um prefixo conhecido de rede e um sufixo baseado no endereço MAC do dispositivo. Um fator restritivo para essa proposta é que, com esse tipo de configuração, o espaço de endereçamento disponível tem que ser maior ou igual ao número de máquinas. Outra questão relevante do uso do endereço MAC no endereço IP é a facilidade de rastreamento de um dispositivo que pode gerar um problema de privacidade, uma vez que com simples monitoramento do IP nas conexões é possível conhecer a movimentação do usuário entre os diferentes locais nos quais ele esteve com o dispositivo [Narten e Draves, 2001]. Além disso, sabe-se que os protocolos TCP/IP devem funcionar independentes da implementação da camada de enlace. De fato, nem todos os dispositivos em uma rede ad hoc utilizam cartões de interface de rede com endereços de MAC únicos distribuídos pelo IEEE com 48 bits. É possível também trocar manualmente o endereço MAC e existem casos conhecidos de placas do mesmo fabricante com o endereço MAC repetido.

Assim, a alocação dinâmica de endereços realizada de forma não centralizada é foco em diversos trabalhos de pesquisa recentes.

Um dos principais problemas da alocação distribuída de endereços é a garantia da unicidade do endereço. Supondo uma escolha aleatória, a probabilidade de colisões de endereço ($P(E_c)$) será determinada pelo espaço de endereçamento disponível e pelo número de nós disputando esses endereços. Essa probabilidade de colisão pode ser calculada com analogia ao conhecido paradoxo do aniversário [Parno et al., 2005] e pode ser estimada pela inequação dada por

$$P(E_c) > 1 - e^{-\frac{n(n-1)}{2r}}, \quad (1)$$

onde n representa o número de nós disputando endereços e r representa o espaço de endereços disponível. Suponha uma rede utilizando endereços IP versão 4 (IPv4) com endereços classe C. Pela Equação 1, é possível observar que, com um espaço de endereçamento de 256 endereços e com 42 endereços ocupados, o que representa cerca de um sexto do total, a probabilidade de acontecer colisões de endereço é maior que 0,96, ou seja, é quase certo que haverá colisão de endereço. Assim, fica claro que é necessária a utilização de um protocolo de alocação de endereços que solucione as colisões.

Perkins *et al.* propuseram um protocolo totalmente distribuído que provê a detecção de endereços duplicados (*Duplicate Address Detection - DAD*) [Perkins et al., 2000]. Este protocolo é baseado em mensagens de requisição de endereço (*Address Request - AREQ*) e resposta (*Address Reply - AREP*). Assim, cada nó, ao entrar na rede, escolhe um endereço e inunda a rede com AREQs por uma quantidade determinada de vezes. Se algum nó já possuir um endereço igual, deve enviar uma mensagem de AREP para o novo nó. Na ausência de respostas, o novo nó utiliza o endereço. Caso contrário, ao receber um pacote de resposta, o nó escolhe um novo endereço e repete o processo de inundação. A proposta de Perkins *et al.*, no entanto, não trata dos casos de união de partição, o que impede o uso do protocolo em redes com baixa conectividade.

Baseado na detecção de endereços duplicados (DAD), outros protocolos foram propostos, inserindo mensagens de HELLO e identificadores de rede para detectar partições na rede. Os identificadores de rede são números escolhidos aleatoriamente que devem ser trocados sempre que uma partição se formar e quando partições se unirem. Os identificadores de rede são importantes porque, quando a rede sofre partições, o prefixo de rede não deve ser modificado, já que as máquinas continuam ligadas ao mesmo domínio, embora estejam temporariamente desconectadas. Assim, diferentes identificadores de redes servem como uma identificação das partições formadas. Fan e Subramani propuseram um protocolo baseado no DAD que detecta união de partições sempre que um nó receber um HELLO com um identificador de rede diferente do seu ou ainda quando ocorrerem mudanças no conjunto de vizinhos [Fan e Subramani, 2005]. Fazio *et al.* propuseram um protocolo também baseado em identificadores de rede, mas que funciona de forma reativa [Fazio et al., 2006]. Ao invés de enviar AREQs e AREPs em todas as entradas de nós ou uniões de partições, o protocolo faz a identificação de colisões apenas quando existe a necessidade de troca de dados. Embora reduza o número de mensagens trocadas, esse protocolo, por ser reativo, acarreta um atraso na transmissão dos dados. Além disso, o mecanismo usado se aplica apenas a protocolos de roteamento reativos, pois, nos protocolos pró-ativos, como as rotas são formadas em avanço, colisões de endereço causam a

formação de rotas erradas.

Algumas propostas são baseadas na alocação de conjuntos de endereços para cada nó. O protocolo *Dynamic Address assignment Protocol in mobile ad-hoc networks* (DAP) subdivide o seu conjunto de endereços disponíveis à medida que novos nós entram na rede [Kim et al., 2007]. Quando um nó fica sem endereços extras para a alocação de novos nós, é feito um pedido que gera uma realocação dos endereços disponíveis. O inconveniente dessa proposta é a realocação de endereços, pois é difícil detectar que um determinado endereço não está mais sendo utilizado e pode ser disponibilizado para os novos nós. Outra proposta semelhante é o *Prophet*, que, para endereçar os nós, utiliza um gerador de números aleatórios com alta entropia, ou seja, uma baixa probabilidade de repetição do mesmo valor [Zhou et al., 2003]. O primeiro nó da rede, chamado de nó profeta, escolhe a semente da seqüência de números aleatórios e atribui endereços para os novos nós que o contatam. Esses novos nós passam a distribuir endereços a partir de pontos diferentes da seqüência de números aleatórios, formando uma árvore de distribuição. O protocolo gera poucas mensagens, mas, em compensação, não soluciona completamente a união de partições e também não trata a realocação de endereços.

Outro tipo de abordagem de autoconfiguração de endereços é a formação de células de alocação de endereço baseadas na localização dos nós [Giruka e Singhal, 2006]. Existem, ainda, propostas que utilizam informações de roteamento para detectar colisões [Eriksson et al., 2007]. Essas propostas, no entanto, acarretam atrasos significativos na detecção de colisões ou são soluções específicas para determinados protocolos.

3. Protocolo FIBRA

O FIBRA foi desenvolvido para redes sem infra-estrutura nas quais os nós conhecem o prefixo da rede e, assim, necessitam apenas determinar um sufixo de máquina. O objetivo do protocolo proposto é fazer a autoconfiguração de endereços de forma dinâmica, identificando e solucionando as colisões de endereço de forma eficiente. Dessa forma, pretende-se gerar uma baixa carga de mensagens de controle para manter o conjunto de endereços utilizados sem colisões mesmo quando novos nós entram na rede ou quando ocorrem uniões de partições. Para reduzir o número de mensagens a serem trocadas e a memória necessária, o protocolo proposto utiliza Filtros de Bloom, que são estruturas capazes de armazenar informações de forma compacta.

3.1. Filtros de Bloom

Um Filtro de Bloom é um vetor formado por m bits que representa um determinado conjunto $A = \{a_1, a_2, a_3, \dots, a_n\}$ formado por n elementos. Devido à sua alta capacidade de compressão, os Filtros de Bloom vêm sendo utilizados em diversos tipos de aplicações, como rastreamento de pacotes [Laufer et al., 2007] e *web cache* [Fan et al., 2000]. Para gerar o Filtro de Bloom, k funções *hash* independentes (h_1, h_2, \dots, h_k) com um alcance m são utilizadas. Inicialmente, o vetor de bits que forma o Filtro deve ser zerado, e, em seguida, cada um dos elementos $a_i \in A$ deve ser aplicado a cada uma das k funções *hash*. O resultado de cada função *hash* representa uma posição do vetor que deve ser trocada de 0 para 1, como mostrado na Figura 1(a). Para verificar se um elemento a_j pertence ao conjunto A de elementos inseridos no filtro, deve-se aplicar as k funções *hash* ao elemento, e se alguma das posições $h_1(a_j), h_2(a_j), \dots, h_k(a_j)$

corresponder a um bit em 0 no filtro, o elemento certamente não pertence ao conjunto A . Caso todos os bits correspondam a bits em 1 no filtro, assume-se que o elemento pertence ao conjunto A . No entanto, existe uma probabilidade de ocorrerem falso-positivos, ou seja, de que elementos que não foram inseridos anteriormente apareçam como presentes após uma verificação. A partir da expressão da probabilidade de um bit permanecer em 0 após a inserção de n elementos ($P(bit_i = 0, n)$), é possível concluir que a probabilidade de falso-positivo é dada por

$$P(\text{falso-positivo}) = (1 - P(bit_i = 0, n))^k = \left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right)^k. \quad (2)$$

Pode-se observar que quanto menor o número de elementos, n , do conjunto A e quanto maior o tamanho do filtro, m , menor a probabilidade de falsos positivos. Para $n = 150$, $m = 900$ e $k = 4$, a probabilidade de falso-positivo do filtro é igual a 0,0561.

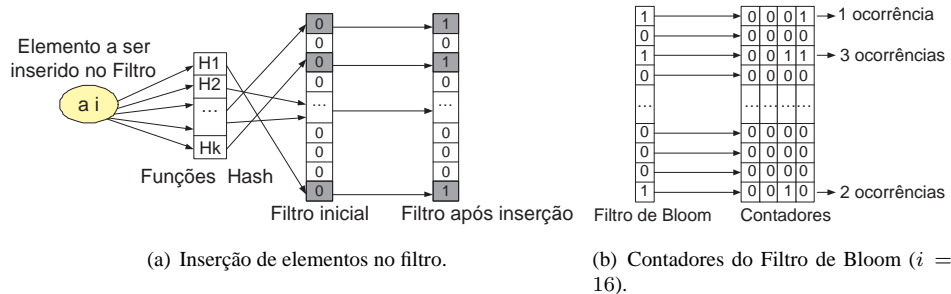


Figura 1. Filtro de Bloom.

Em Filtros de Bloom tradicionais, não é possível retirar elementos do Filtro, pois, se os bits relativos a um elemento fossem zerados, poderiam ocorrer falso-negativos. Isso ocorre porque os bits zerados podem ter sido mudados para 1 por mais que um elemento, e ao zerá-los, todos esses outros elementos também seriam excluídos do Filtro. Assim, para aplicações onde é necessário retirar elementos do Filtro é utilizado um contador para cada bit do Filtro (Figura 1(b)). Dessa forma, sempre que um elemento a_j for incluído, os contadores dos bits $h_1(a_j)$, $h_2(a_j)$, ..., $h_k(a_j)$ são incrementados, e, quando um elemento é excluído, os contadores são decrementados. Assim, é possível saber o número de vezes que um determinado bit foi selecionado por alguma das funções *hash*. Além disso, somando os valores de cada contador e dividindo pelo número de funções *hash*, é possível determinar o número exato de elementos no filtro. Em [Fan et al., 2000], os autores mostram que a probabilidade de um contador ser maior ou igual a i ($P(c \geq i)$) é dada por

$$P(c \geq i) \leq m \left(\frac{enk}{im}\right)^i. \quad (3)$$

Ao se usar contadores é importante que se garanta que não ocorra transbordo do contador, pois isto implicaria na geração de falso-negativos. Assim, supondo contadores com quatro bits ($i = 16$) e, como no exemplo anterior, $n = 150$, $m = 900$ e $k = 4$, a probabilidade de transbordo do contador é menor que $6,6 \cdot 10^{-13}$, o que pode ser considerado desprezível.

A idéia chave da utilização do Filtro de Bloom no protocolo de alocação de endereços proposto é a representação de forma compacta de todos os endereços já atribuídos. Assim, sempre que um novo nó entra na rede, ele solicita a um vizinho qualquer o Filtro de Bloom, escolhe aleatoriamente um endereço e verifica se o endereço já foi alocado. Caso o endereço já tenha sido alocado, o teste de pertinência deu positivo, ele

repete o processo. Caso o teste de pertinência seja negativo é certo que este endereço não foi inserido no filtro e, portanto, não foi ainda alocado a nenhum nó. Neste caso, ele aloca este endereço para si, insere este novo endereço no filtro e difunde o novo endereço alocado para todos os nós. Com isso, evita-se a ocorrência de mais que uma inundação para anunciar um novo endereço na entrada de novos nós, como pode acontecer nos protocolos baseados no DAD [Perkins et al., 2000].

A utilização de Filtros de Bloom também simplifica a detecção de partições. Uma união de partições é detectada sempre que nós vizinhos possuem Filtros de Bloom diferentes. Filtros diferentes indicam conjuntos de nós diferentes, já que a probabilidade de dois conjuntos de nós diferentes acarretarem no mesmo filtro é menor que $2^{-(m)}$ e pode ser considerada desprezível. Com os Filtros de Bloom, é possível, ainda, estimar qual é a menor partição através da observação da quantidade de nós inserida em cada Filtro. Com base na informação sobre o tamanho das partições, apenas os nós da partição menor cujos endereços estiverem presentes no Filtro da partição maior deverão escolher um novo endereço não utilizado como sua nova identificação. Portanto, os Filtros de Bloom otimizam o controle dos endereços na rede e reduzem o volume de tráfego de controle.

3.2. Inicialização da rede

O processo de inicialização da rede refere-se a como o conjunto inicial de nós consegue obter os seus endereços. A inicialização da rede pode ocorrer gradativamente, com longos intervalos entre a entrada de um nó e outro, ou todos os nós podem entrar simultaneamente. Os protocolos de alocação de endereços devem funcionar independentes da forma como os nós são ativados, embora a maioria dos protocolos suponha que exista um intervalo grande entre a entrada do primeiro nó e a entrada do segundo nó. Um exemplo é o protocolo de Fan e Subramani [Fan e Subramani, 2005], no qual o primeiro nó, após identificar que está sozinho, escolhe um identificador de rede e começa a emitir mensagens de HELLO. Os próximos nós a entrar na rede são tratados como um nó novo pelo primeiro nó da rede.

O FIBRA trata tanto a entrada gradativa como a simultânea, utilizando mensagens de HELLO e *Address Requests* (AREQs). Ao entrar na rede, o nó observa o meio por um período T_{Obs} . Caso não escute HELLOs de nós que já tenham obtido um Filtro de Bloom, o nó escolhe um endereço aleatoriamente, respeitando os bits do prefixo de rede e passa para a fase de inicialização da rede. Nessa fase, o nó deverá anunciar o seu endereço através da inundação de AREQs por um número Num_{Rep} de vezes e esperar por outros anúncios. Após um período T_{Init} sem escutar AREQs, o nó deixa a fase de inicialização e insere no seu Filtro de Bloom todos os endereços recebidos em AREQs. A partir daí, o nó passa a emitir HELLOs indicando que já possui um Filtro de Bloom. Se durante a fase de inicialização o nó receber um endereço igual ao seu em um AREQ, o nó deve esperar por um período T_{Troca} , escolher um novo endereço e enviar novamente os AREQs. O período T_{Troca} é importante para permitir que o nó receba mais AREQs informando outros endereços. Assim, mais colisões são evitadas, pois o nó terá um conhecimento mais amplo dos endereços que já foram alocados.

A Figura 2 mostra as mensagens de AREQ e HELLO do FIBRA. O bit I das mensagens de AREQ e HELLO indica se o nó está ou não na fase de inicialização da rede. Assim, se um nó A recebe um HELLO com o bit I em 0 de um nó B, o nó A sabe que o nó B ainda está na inicialização e que não possui Filtro de Bloom. Esse campo nas

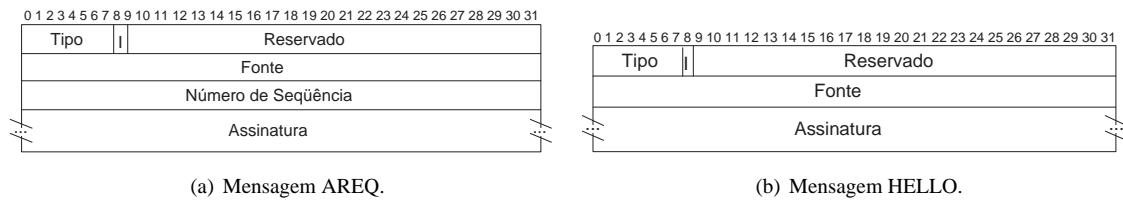


Figura 2. Mensagens do FIBRA.

duas mensagens é importante para que os novos nós saibam se eles devem passar para a fase de inicialização da rede ou se devem se comportar como novos nós na rede. O campo assinatura contém o *hash* do Filtro de Bloom, para identificar o Filtro atual.

3.3. Entrada de Novos Nós

Após a inicialização, os nós começam a emitir HELLOs com o bit *I* em 1, para indicar que já possuem um Filtro de Bloom. No FIBRA, qualquer nó que já possua um Filtro de Bloom pode receber um pedido de entrada de um novo nó.

Quando um novo nó fica ativo, ele deve escutar o meio durante o período T_{Obs} . Como a rede já está formada, o novo nó escutará algum HELLO com o bit *I* em 1 e escolherá o primeiro nó escutado para lhe transmitir o Filtro de Bloom da rede. Para obter o Filtro, o novo nó envia uma mensagem Requisição, representada na Figura 3(a), para o nó “escutado”. Ao receber uma Requisição, o nó observa o bit *I*, para identificar se o pedido contido na mensagem veio de um novo nó. Caso confirme que é um processo de entrada de novos nós, o nó envia como resposta à Requisição uma nova mensagem Requisição com o bit *R* igual a 1. Ao receber a Requisição em resposta, o novo nó deve guardar o Filtro de Bloom recebido e escolher um endereço que não esteja sendo utilizado. Em seguida, o novo nó envia um AREQ para a rede anunciando que um novo endereço foi alocado. Cada nó, ao receber essa mensagem, insere o endereço indicado no Filtro de Bloom e atualiza a assinatura de seu Filtro.

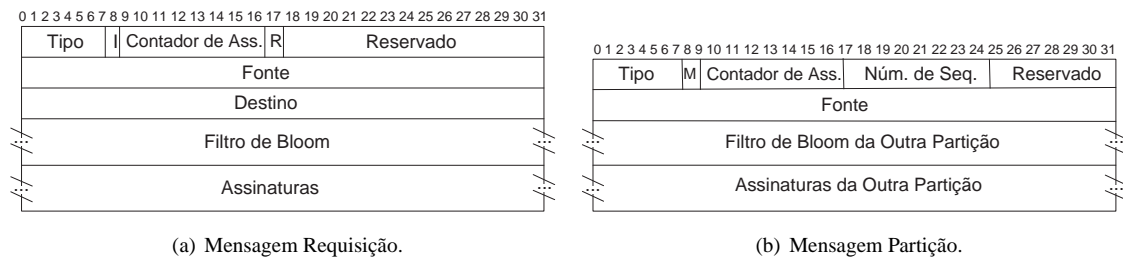


Figura 3. Mensagens do FIBRA.

A mensagem Requisição traz em seu corpo, além do Filtro de Bloom que está sendo utilizado na assinatura do HELLO, todas as assinaturas de Filtros de Bloom válidas atualmente para o nó emissor da Requisição. Essas assinaturas extras de Filtros são guardadas por pouco tempo e devem ser mantidas para evitar a falsa detecção de união de partições durante o processo de entrada de novos nós ou de uniões de partições. Como exemplo, suponha a situação demonstrada pela Figura 4(a). Nesta figura, o nó A recebeu um AREQ anunciando que o nó C entrou na rede em t_1 . O nó A deve encaminhar essa mensagem para o nó B. No entanto, antes de reencaminhar a mensagem, em t_2 , o nó A atualiza a sua assinatura de Filtro e envia um HELLO. O nó B, ao receber o HELLO de A, irá verificar que a assinatura do HELLO difere da sua própria assinatura, indicando

um processo de união de partição. Assim, em t_4 , um falso processo seria disparado, implicando em uma emissão desnecessária de mensagens.

Na Figura 4(b), o nó A, ao invés de atualizar a assinatura ao receber o AREQ de C, guarda a assinatura do Filtro K1 e insere em K1 o endereço de C. Dessa forma, o nó B não detecta partição e tem tempo para receber o AREQ de C. Portanto, o falso processo de partição é evitado. Para completar a transição de filtros, o nó A deve esperar um tempo T_{Filtro} para deixar de usar a assinatura de K1 nos HELLOS e passar a utilizar a assinatura de K2, que é o Filtro de Bloom mais novo. O tempo T_{Filtro} deve ser grande o suficiente para garantir que todos os vizinhos de A já receberam o AREQ e já criaram um próximo filtro igual a K2. Desta forma, após o nó A fazer a troca da assinatura de K1 pela assinatura de K2, o nó B não detectará partição, pois identificará que o filtro usado no HELLO de A, K2, está presente na sua memória como o seu próximo Filtro. Da mesma forma, após o nó A trocar a sua assinatura para K2, ele deve guardar a assinatura de K1 novamente por um tempo T_{Filtro} . Assim, o nó A poderá reconhecer que o HELLO de B também não é uma partição até que o nó B realize a troca para a assinatura do Filtro mais atualizado, K2. Cabe observar que cada nó guarda apenas o Filtro de Bloom mais atualizado. Assim, por exemplo, em t_4 , o nó A guarda o Filtro K2 e apenas a assinatura de K1, para evitar ocupar muitos recursos dos nós.

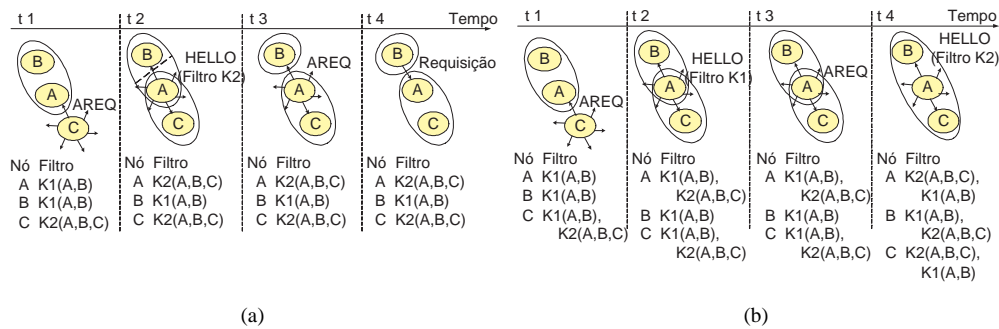


Figura 4. Exemplo de falso processo de partição na entrada de novos nós.

3.4. Detecção de Colisões em Uniões de Partições

Nós que estão em partições diferentes selecionam endereços baseados apenas no conjunto de endereços alocados na sua partição. Portanto, nós em diferentes partições podem selecionar o mesmo endereço, gerando colisões durante a união de partições. No FIBRA, a detecção das partições é feita através da assinatura nos HELLOS. Ao receber um HELLO, o nó deve verificar se a assinatura presente na mensagem corresponde à assinatura do seu Filtro de Bloom ou a alguma outra assinatura de Filtro guardada. Caso seja diferente, significa que aquele nó possui um conjunto de endereços utilizados diferente e, assim, um processo de união de partição deve ser iniciado.

Quando dois nós detectam a existência de uma partição através da mensagem de HELLO, eles devem trocar mensagens Requisição para enviar o filtro e as assinaturas de sua partição. Em seguida, ambos os nós devem inundar a sua partição com a mensagem Partição, representada na Figura 3(b), para atualizar os outros nós com as novas informações recebidas na Requisição. Com base no Filtro recebido, cada nó da rede pode determinar se está na menor ou na maior partição. Apenas os nós na menor partição devem verificar se existem réplicas do seu endereço e, caso existam réplicas, deve-se escolher um novo endereço não utilizado e enviar os AREQs para toda a rede, anunciando

uma nova alocação de endereços. Caso as partições tenham o mesmo tamanho, a partição do nó que enviou a Requisição com $R = 1$ verifica as colisões de endereços e isso é anunciado para as duas partições através do bit M da mensagem Partição. Após receber a mensagem Partição, os nós devem atualizar o Filtro, fazendo a união entre o seu Filtro e o Filtro da outra partição, e atualizar a assinatura do Filtro.

Para evitar que múltiplos processos de união de partição se iniciem ao mesmo tempo, deve-se observar uma regra. Se o nó A detecta uma partição com o nó B e se o endereço de A é maior que o de B, então A não deve iniciar o processo de união de partições. Isso evita que A e B enviem mensagens de Requisição simultaneamente.

3.5. Saída de Nós da Rede

A saída de nós da rede deve implicar na devolução do endereço utilizado para o espaço de endereços disponível. A saída pode acontecer através de uma notificação para toda a rede ou de forma repentina, causada, por exemplo, por uma falta de energia. Quando a saída não é repentina, cada nó, ao receber a notificação de saída, exclui o nó indicado do Filtro de Bloom. Quando a saída é repentina, o endereço permanece alocado no Filtro. Após diversas saídas repentinas, o Filtro pode passar a ter poucos ou nenhum endereço disponível, o que é identificado pelo percentual de bits em 1 no Filtro.

Para solucionar o problema da devolução de endereços, basta que, ao receber uma notificação de partição ou de entrada de novo nó, cada nó verifique o percentual de bits em 1 no seu Filtro de Bloom. Se o percentual chegar a um nível crítico, então todos os nós devem voltar à fase de inicialização, zerando os seus filtros e enviando novos AREQs. Embora essa situação pareça dispendiosa em termos de número de mensagens enviadas, ela corresponde apenas a uma união de partições nos protocolos sem estado, ou seja, que não guardam informações sobre o conjunto de endereços alocado, como é o caso do protocolo de Fan e Subramani [Fan e Subramani, 2005]. Para evitar constantes renovações do Filtro em redes com o espaço de endereços muito ocupado, deve-se esperar um período mínimo entre as renovações de Filtro.

4. Ambiente de Simulação

O desempenho do FIBRA foi avaliado utilizando o ns-2, que é um simulador amplamente usado na avaliação de desempenho de protocolos para redes ad hoc. O FIBRA foi implementado no módulo de roteamento do *Ad Hoc On-Demand Distance Vector routing protocol* (AODV), seguindo a descrição apresentada e as condições de exceção encontradas em [Fernandes, 2008]. Na camada MAC utilizou-se o IEEE 802.11g e o modelo de propagação utilizado foi o *Shadowing* com parâmetros relativos a uma rede comunitária. Foi assumido que os nós possuem um alcance médio de 18.5m e um limiar de interferência da portadora máximo de 108m, para representar valores comerciais típicos de equipamentos IEEE 802.11.

O FIBRA foi comparado com o protocolo proposto por Perkins *et al.* [Perkins et al., 2000], porque o mecanismo do *Duplicated Address Detection* (DAD) é a base para muitos outros protocolos, embora não solucione as uniões de partições. Além disso, o FIBRA também foi comparado com a proposta de Fan e Subramani [Fan e Subramani, 2005], um protocolo sem estado, que utiliza identificadores de rede para identificar partições e o DAD para permitir a entrada de novos nós e a solução

de colisões durante a união de partições. Ao comparar o FIBRA com a proposta de Fan e Subramani, é possível determinar o impacto do uso de Filtros de Bloom na redução do volume de tráfego de controle enviado e na detecção correta de partições da rede.

Os parâmetros do FIBRA utilizados na simulação estão na Tabela 1 e foram escolhidos com base em experimentos, de forma a maximizar a eficiência do protocolo. O número de retransmissões das mensagens inundadas, Num_{Rep} , foi escolhido como 4 para que a maioria das inundações alcançasse a todos os nós em todos os três protocolos. Os HELLOs utilizado pelo FIBRA e pela proposta de Fan e Subramani possuem os mesmos campos e o identificador de rede foi escolhido do mesmo tamanho que a assinatura do Filtro de Bloom, composta por 4 bytes. O intervalo entre HELLOs foi escolhido como 1s. O Filtro de Bloom foi escolhido de forma a não ocupar muito espaço e possuir uma baixa probabilidade de colisão de endereços. Assim, escolheu-se, para um ambiente com 150 endereços disponíveis, um Filtro de Bloom com contadores que ocupa, no total, 450 bytes e, para o preenchimento do Filtro, escolheu-se 4 funções *hash* independentes. Os resultados apresentados possuem um intervalo de confiança de 95% e o tempo de simulação para cada rodada foi de 50s nos cenários estáticos e 30s nos cenários móveis.

Tabela 1. Parâmetros do FIBRA.

Variável	Descrição	Valor
T_{Obs}	Tempo de observação do meio na entrada na rede	1s
T_{Troca}	Tempo de espera para troca de endereços	1s
T_{Init}	Tempo de espera por AREQs na inicialização	4s
T_{Filtro}	Tempo para troca de Filtro	1s

5. Resultados

Primeiramente, foi analisado o efeito da inicialização da rede. Na primeira simulação utilizou-se um cenário em grade com a densidade de uma rede comunitária ($\approx 0.0121nós/m^2$) [Campista et al., 2007] com 64 nós. Os nós são ligados simultaneamente e a rede não possui partições. Os endereços são escolhidos aleatoriamente por cada nó com base em geradores pseudo-aleatórios com uma semente diferente para cada nó. O número de colisões médio no conjunto inicial de endereços foi de aproximadamente 13 colisões e, ao final da simulação, todos os protocolos solucionaram todas as colisões.

A média dos resultados obtidos com esse cenário está representada na Tabela 2, na qual está contabilizado o total de *bytes* emitidos por transmissão ou retransmissão de mensagens de controle dos protocolos de autoconfiguração de endereço. Analisando os resultados, observa-se que todos os protocolos evitaram as colisões de endereço, embora o protocolo de Fan e Subramani tenha demorado mais tempo para se estabilizar, já que não previa a entrada simultânea de nós. De fato, o protocolo de Fan e Subramani supõe a existência de um intervalo significativo entre a entrada do primeiro e do segundo nó na rede. Como os nós entram simultaneamente, cada nó escolhe um identificador de rede próprio, e os nós passam a encarar os vizinhos como partições. Por essa razão, o protocolo de Fan e Subramani apresentou uma carga quase 13 vezes maior que o FIBRA.

Outro resultado interessante é a existência de mensagens de Requisição no FIBRA durante a inicialização, pois se observou que as quatro retransmissões de cada AREQ não foram suficientes para alcançar todos os nós em 100% dos casos. Quando uma determinada mensagem não consegue alcançar todos os nós mesmo após todas as retransmissões, os nós que não receberam essa mensagem ficam com um Filtro de Bloom diferente dos

demais. Assim, a perda dos pacotes afeta o FIBRA, que reage a ela através de um processo falso de união de partições. Essa falsa união de partições é importante para garantir que todos os nós, de fato, irão receber as mesmas informações sobre os endereços utilizados na rede. Como os outros protocolos não possuem esse tipo de robustez a perdas, uma colisão pode não ser identificada se a mensagem perdida for relativa a algum nó com endereço colidido. Essa propriedade do FIBRA de robustez à perda de mensagens dá maior confiabilidade ao protocolo, embora aumente a carga de controle. De fato, o protocolo de Perkins *et al.* obteve uma menor carga de controle apenas devido às falsas uniões de partições do FIBRA e por não emitir mensagens de HELLO.

Tabela 2. Efeito da Inicialização.

Protocolo	Total de MBytes	Atraso Médio(s)
Perkins <i>et al.</i>	1,20	0,18
Fan e Subramani	30,72	1,08
FIBRA	2,37	2,14

Além da carga de controle, a Tabela 2 mostra o tempo médio para a estabilização dos endereços. Foi considerado, nessa análise, o tempo até a última mudança de endereço. Os resultados mostram que o FIBRA possui um atraso um pouco maior que os outros protocolos, porque cada nó deve esperar o tempo T_{Obs} para determinar em qual estado está a rede e o tempo T_{Troca} para escolher um novo endereço, em caso de colisão. O protocolo de Fan e Subramani apresentou um resultado pior que a proposta de Perkins *et al.* porque também espera T_{Obs} antes de selecionar o endereço.

Com base nessa primeira análise, é possível afirmar o protocolo de Perkins *et al.* apresenta o melhor resultado na inicialização, tanto em termos de quantidade de mensagens quanto em atraso, pois o protocolo não emite mensagens de HELLO e a escolha de endereços é imediata quando o nó entra na rede.

O efeito da mobilidade também foi avaliado em dois cenários de 50x50m com baixa probabilidade de formação de partição e compostos por 30 nós. Os nós se movem segundo o modelo *random-way point*, com tempo de pausa de 1s.

No primeiro cenário, cujos resultados estão na Figura 5(a), os nós foram ligados simultaneamente. O FIBRA e o protocolo de Perkins *et al.* não sofrem efeitos com a mobilidade, devido à baixa probabilidade de ocorrerem partições, enquanto que o protocolo de Fan e Subramani tem sua eficiência reduzida com o aumento da mobilidade. Uma vez que o protocolo de Fan e Subramani não utiliza Filtros de Bloom, a detecção de partições deve ser feita apenas com base no encontro de diferentes identificadores de rede ou em determinadas mudanças na vizinhança de um nó. Assim, como o aumento da mobilidade leva a muitas mudanças nas vizinhanças dos nós, falsos processos de união de partições passam a ser detectados na rede, o que justifica o aumento do número de mensagens enviadas. Como o protocolo determina um intervalo mínimo entre processos de união de partições para cada nó, após atingir certa velocidade, o número de processos de união de partições na rede já é máximo e a quantidade de mensagens trocadas se mantém independente da mobilidade. No segundo cenário com mobilidade, os nós não entram de forma simultânea na rede. O que se observa nos resultados, representados na Figura 5(b), é que a carga do protocolo de Fan e Subramani diminui, pois o protocolo consegue se estabilizar mais rapidamente na inicialização. No entanto, devido à mobilidade, falsas partições continuam a ser detectadas e o comportamento do protocolo fica semelhante ao observado

no primeiro cenário com mobilidade. Mesmo sem a entrada simultânea, o protocolo de Fan e Subramani chega a apresentar um desempenho 2,33 vezes pior que o FIBRA, já que possui um mecanismo de detecção de partições que gera muitos falso-positivos.

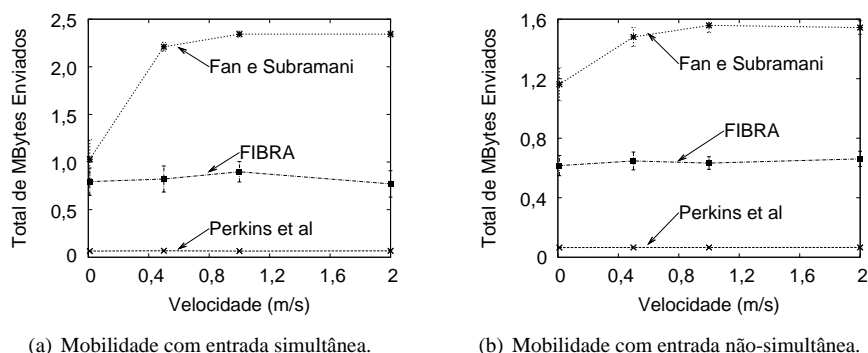


Figura 5. Efeito da Mobilidade.

A análise seguinte foi relativa à formação de partições. Em um cenário estático formado por 50 nós, variou-se a quantidade de uniões de partições através da ativação/desativação de determinados nós responsáveis por conectar as partições isoladas. A densidade em cada uma das partições é similar a de uma rede comunitária e os nós nas partições são ligados simultaneamente. O gráfico representando a quantidade de bytes transmitidos ou retransmitidos após o início das uniões de partições está na Figura 6(a). Por esse gráfico, é possível perceber que o protocolo de Perkins *et al.* não enviou nenhum tipo de mensagem de controle, pois esse protocolo não é capaz de detectar partições e solucionar colisões formadas durante a união de partições. Assim, o protocolo é ineficiente em ambientes com partições. O FIBRA obteve uma carga até 3.6 vezes menor que a do protocolo de Fan e Subramani, pois evita o excesso de AREQs durante a união de partições. De fato, com o FIBRA, apenas metade dos nós envolvidos em colisões troca de endereço e envia AREQs, enquanto que no protocolo de Fan e Subramani todos os nós verificam seus endereços com AREQs.

Na Figura 6(a) observa-se que o total de bytes transmitidos não aumenta com o aumento do número de uniões de partições. Isso se explica devido à carga de controle na união de partições ser composta, em sua maioria, por mensagens de AREQ e Partição, como pode ser observado na Figura 6(b), relativa ao funcionamento do FIBRA durante o processo de união de partições. Uma vez que essas mensagens são inundadas, quanto menor é o número de nós na partição, menor é a carga na rede. Assim, quando existem mais partições, o número de nós em cada partição diminui e o número de mensagens nas inundações também diminui. Além disso, com partições maiores, o número de colisões também cresce, o que leva o FIBRA a formação de falsas uniões de partições para solucionar a perda de mensagens, levando a emissão de um volume maior de mensagens Requisição, AREQ e Partição.

A última análise foi referente ao efeito do número de nós e da densidade. O primeiro cenário, utilizado para avaliar o impacto do número de nós, é formado por nós estáticos organizados em grade, com uma densidade similar a de uma rede comunitária. O momento de entrada de cada nó é escolhido aleatoriamente entre 1 e 25s, enquanto que o momento de saída, entre 25 e 50s. Já o segundo cenário, usado para avaliar o impacto da variação da densidade, segue o mesmo comportamento na ativação e desativação dos nós e é composto por um total de 49 nós distribuídos em grade. Os resultados dessa análise

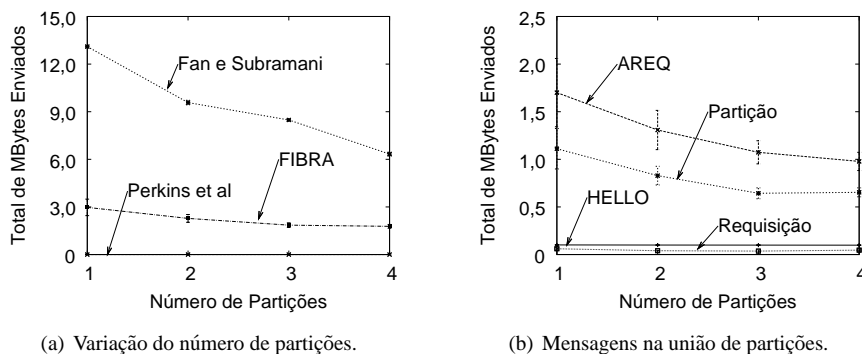


Figura 6. Efeito da união de partições.

estão representados nas Figuras 7(a) e 7(b). Por esses gráficos, é possível observar que o aumento do número de nós tem grande impacto sobre os protocolos, devido à utilização de inundações. O FIBRA é menos afetado pelo aumento do número de nós que a proposta de Fan e Subramani, que tem uma carga 3,9 vezes maior que a do FIBRA com 64 nós. Isso se deve ao FIBRA reduzir a emissão de mensagens que são inundadas na rede. Por outro lado, o aumento da densidade só traz impactos para o FIBRA, pois somente o protocolo proposto tem capacidade para detectar perdas de mensagens. Assim, com o aumento da densidade, o número de vizinhos por nó aumenta, assim como a disputa pelo meio durante as retransmissões, causando mais colisões na rede e levando o FIBRA a reagir até que todos os nós recebam todos os AREQs.

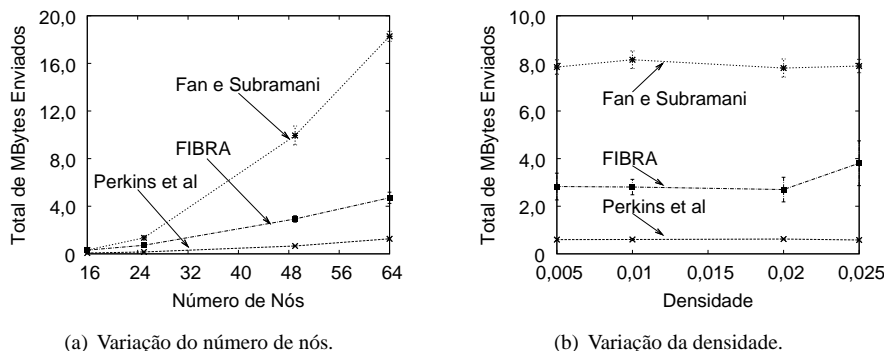


Figura 7. Efeito da variação do número de nós e da densidade.

6. Conclusões

Os nós de uma rede ad hoc devem receber um endereço único antes de poderem se comunicar. Idealmente, esse processo deve ser automático, rápido, sem erros e deve poupar recursos do nó. Nesse artigo, foram analisadas as principais propostas para a autoconfiguração de endereços em redes ad hoc, mostrando seus pontos positivos e negativos. O protocolo FIBRA foi proposto com o objetivo de atender melhor aos requisitos de uma alocação de endereços em redes ad hoc. A utilização de Filtros de Bloom trouxe grandes vantagens em termos de redução do número de mensagens de controle trocadas, ao custo de uma pequena quantidade de armazenamento em cada nó.

Os resultados das simulações mostram que o FIBRA é capaz de solucionar todas as colisões de endereço mesmo em situações de união de partições, enviando poucas mensagens de controle. Além disso, os resultados mostram que o FIBRA sofre um impacto menor com o aumento do número de nós na rede que os protocolos semelhantes à proposta de Fan e Subramani. A única desvantagem do FIBRA com relação às outras propostas é

o atraso médio para determinação do endereço. No entanto, essa desvantagem é pequena, uma vez que 1s a mais na obtenção do endereço não causa um impacto muito grande para a maioria das aplicações. Além disso, a economia de energia devido às poucas mensagens emitidas é de grande importância para a maioria dos dispositivos móveis e a robustez à perda de pacotes é essencial para garantir um processo de alocação seguro.

Referências

- Campista, M. E. M., Moraes, I. M., Esposito, P., Amodei Jr., A., Costa, L. H. M. K. e Duarte, O. C. M. B. (2007). The ad hoc return channel: a low-cost solution for brazilian interactive digital TV. *IEEE Communications Magazine*, 45(1):136–143.
- Droms, R. (1997). *Dynamic host configuration protocol*. RFC 2131.
- Eriksson, J., Faloutsos, M. e Krishnamurthy, S. V. (2007). DART: dynamic address routing for scalable ad hoc and mesh networks. *IEEE/ACM Transactions on Networking*, 15(1):119–132.
- Fan, L., Cao, P., Almeida, J. e Broder, A. Z. (2000). Summary cache: A scalable wide-area web cache sharing protocol. *IEEE/ACM Transactions on Networking*, 8(3):281–293.
- Fan, Z. e Subramani, S. (2005). An address autoconfiguration protocol for ipv6 hosts in a mobile ad hoc network. *Computer Communications*, 28(4):339–350.
- Fazio, M., Villari, M. e Puliafito, A. (2006). IP address autoconfiguration in ad hoc networks: design, implementation and measurements. *Computer Networks*, 50(7):898–920.
- Fernandes, N. C. (2008). Controle de acesso distribuído para redes ad hoc. Master's thesis, Universidade Federal do Rio de Janeiro.
- Giruka, V. C. e Singhal, M. (2006). A localized IP-address auto-configuration protocol for wireless ad-hoc networks. Em *4th international workshop on Wireless Mobile Applications and Services on WLAN Hotspots (WMASH'06)*, páginas 101–108. ACM.
- Kim, H., Kim, S. C., Yu, M., Song, J. K. e Mah, P. (2007). DAP: Dynamic address assignment protocol in mobile ad-hoc networks. Em *IEEE International Symposium on Consumer Electronics (ISCE 2007)*, páginas 1–6. IEEE.
- Laufer, R. P., Velloso, P. B., Cunha, D. O., Moraes, I. M., Bicudo, M. D. D., Moreira, M. D. D. e Duarte, O. C. M. B. (2007). Towards stateless single-packet IP traceback. Em *32nd IEEE Conference on Local Computer Networks (LCN'2007)*, páginas 548–555.
- Narten, T. e Draves, R. (2001). *Privacy Extensions for Stateless Address Autoconfiguration in IPv6*. RFC 3041.
- Parno, B., Perrig, A. e Gligor, V. (2005). Distributed detection of node replication attacks in sensor networks. Em *IEEE Symposium on Security and Privacy*, páginas 49–63.
- Perkins, C. E., Royers, E. M. e Das, S. R. (2000). *IP address autoconfiguration for ad hoc networks*. Internet Draft.
- Thomson, S. e Narten, T. (1998). *IPv6 stateless address autoconfiguration*. RFC 2462.
- Zhou, H., Ni, L. e Mutka, M. (2003). Prophet address allocation for large scale manets. Em *Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE INFOCOM 2003)*, volume 2, páginas 1304–1311. IEEE.