Detecção eficiente de comportamento parasita em sistemas de difusão entre-pares.

João Silva, Xavier Vilaça, Luís Rodrigues, Hugo Miranda

INESC-ID, Instituto Superior Técnico, Universidade de Lisboa
{joao.roque,xavier.vilaca,ler}@tecnico.ulisboa.pt,
Universidade de Lisboa, Faculdade de Ciências, LASIGE,
hamiranda@ciencias.ulisboa.pt

Resumo Neste artigo estudamos técnicas que permitem evitar comportamentos parasitas em sistemas de difusão entre-pares. Para este efeito, desenvolvemos uma variante de uma rede sobreposta em que todos os nós possuem vistas parciais simétricas, de tamanho semelhante. Estas vistas são usadas para promover interacções frequentes entre vizinhos, que facilitam a monitorização e a classificação dos parceiros de forma localizada e eficiente. Esta classificação é usada para aplicar sistemas de penalização simples, que permitem detectar e excluir os nós parasitas em sistemas de transmissão contínua de dados (streaming). A avaliação mostra que este mecanismo limita a assimetria na utilização de recursos mesmo na presença de comportamentos mais sofisticados.

1 Introdução

A comunicação descentralizada entre-pares é uma técnica interessante para explorar os recursos sub-utilizados que existem na periferia da rede. Infelizmente, existem evidências de que, em sistemas entre-pares, uma fracção dos nós (designados por nós parasitas) pode não contribuir para o sistema, explorando abusivamente o esforço dos restantes os quais cooperam incondicionalmente (designados por nós altruístas) [1, 3, 9]. O comportamento dos nós parasitas impõe uma carga injusta nos nós altruístas e degrada a execução de tarefas. Mecanismos que permitem detectar e limitar o comportamento parasita têm, portanto, grande relevância prática.

Neste artigo abordamos o comportamento parasita na transmissão contínua de dados (streaming) em sistemas entre-pares. As vantagens da difusão entre pares para suportar este tipo de aplicações têm sido confirmadas por diversos exemplos reais de larga escala, incluindo PPLive [8] entre outros [16]. Neste contexto, o trabalho relacionado [7, 14, 15] tem tratado comportamentos parasitas assumindo que todos os nós são racionais. Isto é, os nós não contribuem porque tentam maximizar a sua utilidade, a qual diminui com a quantidade de recursos fornecidos aos restantes nós. Como os nós racionais se conseguem desviar do protocolo de formas arbitrárias, é necessário fornecer incentivos sofisticados, que geralmente acarretam uma penalização no desempenho do sistema ou requerem

algum tipo de controlo centralizado. No entanto, na prática, não é realista esperar que cada nó seja capaz de calcular a melhor estratégia que maximize a sua utilidade. Desta forma, fazemos as seguintes hipóteses que, acreditamos, representam um cenário mais realista. Primeiro, assumimos que uma grande fracção de nós é altruísta. Segundo, assumimos que a maior parte dos nós que se desviam do protocolo são parasitas puros, isto é, não partilham qualquer tipo de recurso. Terceiro, consideramos que os nós podem atacar o sistema através de lavagem de identidade *(white-washing)*, isto é, podem sair e re-entrar no sistema com uma nova identidade. Quarto, admitimos que apenas uma pequena fracção de nós é racional. Com esta configuração, basta assegurar que: *i*) os parasitas são detectados e expulsos de forma eficiente e não obtêm beneficios quando tentam re-entrar no sistema com nova identidade; *ii*) nós racionais precisam de contribuir para o sistema com uma quantidade de recursos razoável para conseguirem receber os conteúdos; *iii*) os nós altruístas têm mecanismos para detectar que as hipóteses sobre o comportamento do sistema foram violadas (por exemplo, no caso onde, contra as expectativas, existe uma grande fracção de nós racionais entre a população) de forma a poderem activar mecanismos de controlo mais robustos (e também com custo mais elevado), mas apenas quando necessário.

Tal como na maioria das concretizações de transmissão contínua de dados, no nosso sistema é necessário que os nós se juntem a uma rede sobreposta para receberem o conteúdo. Em vez de suportar topologias arbitrárias da rede sobreposta, o nosso protocolo incentiva os nós a manterem ligações simétricas com um grupo pequeno de vizinhos. Sendo assim, é possível usar estratégias do tipo "olho-por-olho" para detectar nós parasitas, em vez de mecanismos complexos de controlo e reputação que podem acarretar uma grande carga adicional. Com base nos princípios acima, desenvolvemos o FastRank, um sistema que faz gestão integrada da topologia e um esquema de monitorização dos nós que minimiza eficazmente o impacto de nós parasitas em aplicações de transmissão contínua de dados. O FastRank implementa um esquema de classificação de vizinhos que permite aos nós altruístas ficarem ligados a outros nós altruístas, enquanto os nós parasitas são rapidamente penalizados por não contribuirem para o sistema e não podem evitar esta penalização mudando de identidade.

Mostramos também que a nossa abordagem é robusta a comportamentos mais sofisticados, tais como às tentativas de manipulação da topologia para benefício próprio, ou esforços de contribuição minimalista. Em particular, mostramos que para estes comportamentos terem sucesso, o atacante tem de contribuir com uma quantidade razoável de recursos. Estes resultados são suportados por uma extensa avaliação do FastRank, com diferentes parâmetros de topologia da rede sobreposta e percentagens de nós parasitas.

O resto do artigo está organizado da seguinte forma. A Secção 2 aborda o trabalho relacionado. A Secção 3 descreve a arquitectura e algoritmos do FastRank. Os modelos de comportamentos dos nós são discutidos na Secção 4. A Secção 5 reporta os nossos resultados experimentais. Finalmente, a Secção 6 conclui o artigo.

2 Trabalho Relacionado

Existem duas abordagens principais para suportar a transmissão contínua de dados em sistemas de larga escala: uma é construir árvores de disseminação e a outra consiste em usar protocolos epidémicos. Em condições ideais, i.e. em sistemas estáveis e com um número desprezável de nós parasitas, as abordagens em árvore oferecem uma estrutura eficiente para disseminar a informação, porque evitam qualquer tipo de redundância. No entanto, em ambientes não controlados como a Internet, a qualidade das árvores rapidamente decresce devido à necessidade de reconstrução sempre que um nó se desliga ou demonstra um comportamento parasita.

Um protocolo de disseminação epidémico pode ser decomposto em dois componentes. O componente de filiação é responsável pela definição e manutenção da topologia de rede. Um grande desafio está na definição de uma topologia que seja resiliente, mitigando o impacto das entradas e saídas. Exemplos de algoritmos com aproximações distintas a este problema são o SCAMP [6] e o HyParView [13]. A segunda componente de um protocolo epidémico consiste no processo de disseminação. Tipicamente, os nós trabalham por turnos, e seleccionam aleatoriamente um subconjunto da vista, cuja dimensão é denominada *fanout*, para enviarem a informação. Esta estratégia de difusão é eficaz e resiliente em cenários onde todos os nós cooperam. No entanto, nós racionais, não irão contribuir para a disseminação da informação beneficiando assim da economia de resursos conseguida. Do ponto de vista do protocolo, nós racionais são comparáveis a nós que falham, e por isso o seu impacto pode, até certo nível, ser atenuado pela robustez e redundância que caracterizam os protocolos epidémicos. Ainda assim, é necessário concretizar mecanismos para prevenir que este problema se agrave e resulte num rácio de retransmissões insuficiente.

De forma a mitigar o impacto de participantes racionais, o BAR Gossip [15] utiliza um mecanismo de trocas balanceadas. A informação é cifrada antes de ser enviada e o emissor apenas publica a chave após ter recebido, do mesmo nó, uma quantidade de informação semelhante. O mecanismo de trocas balanceadas pode ser visto como uma materialização da aborgadem "olho-por-olho" que é praticada no BitTorrent [3]. Infelizmente, a cifra adiciona um custo considerável na troca de informação. O FlightPath [14] utiliza duas aproximações para atenuar o custo da cifra imposta pelo BAR Gossip. Por um lado, utiliza um mecanismo de controlo de fluxo, onde os nós negoceiam antecipadamente que pacotes vão trocar. O controlo de fluxo permite balancear o tráfego a cada ronda para diminuir a redundância. Em segundo lugar, rastreia as trocas entre vizinhos, o que substitui o modelo do BAR Gossip por um mecanismo mais eficiente onde os nós têm uma margem que lhes permite não enviar informação. O LiFTinG [7] implementa mecanismos distribuídos para detectar comportamentos racionais em ambientes assimétricos, i.e. quando é esperado que um nó envie mais informação do que aquela que recebe desse mesmo nó. O LiFTinG exige que os nós confirmem o comportamento de outros nós através de validações cruzadas ao histórico das interacções passadas. Infelizmente, o cruzamento de informação pode impor uma sobrecarga no sistema que se aproxima dos 10%.

3 FastRank

Descrevemos agora o FastRank, um serviço de difusão resiliente a comportamentos parasitas. O FastRank tem três componentes principais: construção e manutençao da topologia da rede, um esquema de classificação localizado, e uma estratégia de disseminação. Estes componentes cooperam entre si para evitar que os nós do sistema possam beneficiar de comportamentos parasitas. Em particular, a manutenção da rede estabelece mecanismos que encorajam os nós a manterem um número pequeno de ligações estáveis e bi-direccionais. Isto obriga os nós a interagirem entre si frequentemente. Estas interacções podem ser facilmente monitorizadas localmente, sem ser necessário a disseminação de tráfego de controlo adicional. Desta forma, a monitorização local torna-se muito eficiente, e é usada por nós altruístas para expulsarem da sua vista nós que aparentem ter comportamentos parasitas. Os nós parasitas podem tentar re-estabelecer ligações com outros membros da rede. No entanto, os protocolos de gestão da topologia incluem mecanismos para assegurar que estas operações são morosas, o que prejudica a recepção dos conteúdos pelos nós parasitas.

3.1 Organização da Rede

A ideia principal do FastRank consiste em utilizar uma rede sobreposta estável, com ligações simétricas, para efectuar a disseminação de mensagens. Dois nós altruístas, após estabelecerem uma ligação, irão preservá-la até que um deles falhe. O número de vizinhos de cada nó é deliberadamente pequeno (i.e, logaritmico ao tamanho do sistema) [11] de forma a que os vizinhos tenham de interagir frequentemente durante o processo de disseminação das mensagens, permitindo a detecção rápida e eficiente de nós parasitas.

O HyParView [13] é um protocol que possui características que se aproximam bastante do nosso objectivo. Os autores do HyParView demonstraram experimentalmente que a rede suporta eficazmente a difusão fiável de mensagens [12]. Infelizmente, não é possível utilizar o HyParView no desenvolvimento do FastRank sem alterar alguns dos seus mecanismos. Isto porque o HyParView foi desenvolvido assumindo que todos os nós do sistema são altruístas. Por esta razão, também não possui mecanismos para prevenir a "lavagem de identidades" (do Inglês, *white washing)*, onde um nó estabelece constantemente novas ligações a diferentes nós. Como os nós demoram algum tempo a identificar o comportamento de um novo vizinho, um nó parasita irá receber informação até a ligação ser quebrada. Se um nó parasita consegue estabelecer novas ligações ao mesmo ritmo que perde as antigas, então ele consegue receber todos os conteúdos, mesmo que os seus vizinhos o identifiquem como parasita. Por esta razão, o FastRank implementa uma variante do HyParView. Esta adaptação do HyParView, a que chamamos *HyParView com Restrições*, implementa mecanismos que limitam o ritmo a que um nó consegue estabelecer novas ligações.

HyParView O objectivo do HyParView consiste na criação de uma rede onde cada nó está ligado a um número pequeno de vizinhos que representam uma

amostra aleatória da população total. O conjunto de vizinhos de cada nó é designado por *vista activa*. O HyParView assegura que as vistas parciais são *simétricas*, i.e., se n_1 está na vista parcial de n_2, então, n_2 também está na vista parcial de n_1. A topologia resultante tem várias propriedades interessantes para os nossos objectivos: *i)* existe um limite máximo para o grau incidente de cada nó, e *ii)* se cada nó tiver uma vista parcial completa (*i.e.* se o grau-divergente de todos os nós for igual ao tamanho da vista parcial), então cada nó do sistema tem exactamente o mesmo número de ligações. Para aumentar a robustez da rede, cada nó tem também uma vista *passiva* com os identificadores de nós adicionais que são usados para substituir as ligações perdidas na vista activa (devido a falhas ou disconexões). A vista activa é usada para suportar as aplicações que usam o HyParView, como disseminação de mensagens. Como a vista activa é pequena e baseada em ligações simétricas, os vizinhos interagem frequentemente. No HyParView original, isto é usado para detectar falhas. No FastRank, isto é também usado para detectar comportamentos parasitas.

HyParView com Restrições O HyParView com restrições é uma adaptação do protocolo original HyParView, que foi desenvolvido para resolver os requisitos do FastRank. Esta variante inclui mecanismos que limitam o ritmo a que um nó consegue estabelecer novas ligações. Deste modo, quando um nó i contacta outro nó j, para estabelecer uma nova ligação entre i e j, o nó i recebe uma tarefa morosa que tem de resolver antes do pedido de ligação ser aceite. Na prática, é introduzido um *período de quarentena* antes de uma nova ligação ser estabelecida. Este período de quarentena deverá ser suficientemente longo de forma a que várias tramas sejam perdidas e que a lavagem de identidade deixe de compensar. Além disso, a tarefa dada ao nó i deverá impossibilitar a sua resolução em paralelo por um único nó durante o período de quarentena, i.e., se um nó tenta estabelecer 2 novas ligações, ele deverá ser forçado a sofrer um atraso correspondente a 2 períodos de quarentena.

Para atingir os objectivos acima, no FastRank, optámos por usar puzzles criptográficos. Mais precisamente, quando um nó i contacta um nó j para estabelecer uma nova ligação, o nó j prepara um puzzle criptográfico que tem de ser resolvido por i para que j aceite a ligação. O tempo médio para resolver cada puzzle criptográfico corresponde ao tempo de quarentena pré-definido, considerando que todos os recursos computacionais de um nó estejam dedicados à sua resolução. Existem na literatura vários exemplos de puzzles criptográficos com estas propriedades [2, 17, 20]; no FastRank optámos por usar o mecanismo descrito em [5]. Uma vantagem desta abordagem é que um nó que se queira juntar à rede é limitado pelos seus próprios recursos, independentemente do número de nós diferentes a que ele se tenta ligar ou do número de identidades diferentes que tenta usar, porque o número de desafios que pode resolver por unidade de tempo é limitado pela capacidade do hardware. Desta forma, o FastRank mitiga também o impacto de outros comportamentos, como o ataque de Sybil [4, 19].

Os mecanismos acima descritos são assimétricos: enquanto o nó que tenta estabelecer a ligação tem de resolver um puzzle criptográfico, o nó que a aceita não

tem. A razão para este comportamento é que o FastRank é feito para detectar, e expulsar da rede, nós parasitas. Estes nós, que são expulsos, irão tentar estabelecer novas ligações, contactando diferentes membros. Para minimizar o impacto negativo que este requisito tem nos nós altruístas, se um nó recebe múltiplos pedidos de ligação concorrentemente, ele irá submeter um puzzle criptográfico diferente para cada nó que tente estabelecer a ligação e irá apenas ligar-se ao primeiro nó que termine a tarefa. Desta forma, se um nó parasita tentar resolver múltiplos puzzles criptográficos ao mesmo tempo, e competir com um nó altruísta pelas ligações, corre o risco de não ser aceite, dado que nós que dediquem todos os recursos a resolver um único puzzle têm maior probabilidade de responder primeiro e obter a ligação. Como um nó é obrigado a participar em repetidas interacções com os seus vizinhos imediatamente após a ligação ser estabelecida, caso se limite a consumir pacotes dessa ligação, irá ser detectado e irá perder a ligação antes de adquirir outra nova.

Mudámos também o comportamento do protocolo HyParView quando uma ligação é perdida. No protocolo original, um nó tenta pró-activamente manter a sua vista completa. Portanto, se um vizinho falha, um nó da sua vista passiva é escolhido e imediatamente usado para tentar substituir essa ligação. No entanto, no HyParView com Restrições, substituir uma ligação tem um custo. Além de que, se um nó tem poucas vagas na sua vista activa, é provável que múltiplos nós concorrentemente tentem ocupar essas vagas. Isto agrava o custo de um nó se juntar a uma rede onde todos os nós tentem encher a sua vista o mais cedo possível. Por outro lado, um nó altruísta pode optar por esperar novos pedidos de ligação por nós que se queiram juntar à rede para completar a sua vista activa. Neste caso, o nó altruísta não só evitará o custo de inicializar uma ligação, mas, também, facilitará o procedimento de ligação a novos nós altruístas que se queiram integrar. No HyParView com Restrições usamos um limite inferior, chamado *linha-de-água*, que tem de ser atingido antes de um nó pró-activamente procurar novos vizinhos. Se um vizinho falha ou é expulso (após ser detectado como parasita) mas o tamanho da vista activa é superior à *linha-de-água*, o nó simplesmente espera por pedidos, e irá aceitar ligações até que o tamanho máximo da *vista activa* seja alcançado.

A fonte da stream é tratada de forma diferente de todos os outros nós, na medida em que não mantém nenhuma vista activa explícita. Em vez disso, usa uma grande vista passiva para escolher de forma aleatória um número de pontos de contacto, cada vez que uma trama é enviada. Os nós altruístas que recebem a informação da fonte irão reenviá-la para os seus vizinhos (uma descrição pormenorizada é fornecida abaixo). O objectivo é distribuir a carga uniformemente pelos membros da rede, de modo a que não haja um grupo fixo de nós que estejam mais próximos da fonte (e que tenham sempre de retransmitir a informação) e outro grupo que se comporte sempre como folhas (e apenas receba informação).

3.2 Algoritmo de Classificação

O FastRank utiliza as propriedades da topologia gerada pelo HyParView com Restrições para executar um algoritmo localizado e eficiente, do tipo "olho-por-

olho", que consegue eficazmente detectar nós parasitas e, expulsá-los das vistas activas dos nós altruístas. O mecanismo é baseado na observação de que, normalmente, dois nós altruístas trocam a mesma quantidade de informação entre si. Se este balanço for bastante assimétrico, com grande probabilidade, o nó que está a receber e não está a enviar informação é um parasita.

O balanço das trocas com cada vizinho é capturado pelo FastRank através de uma classificação numérica atribuída ao vizinho [10]. Cada nó i mantém, para cada vizinho j da sua vista activa, uma classificação denominada $peso_{ij}$. O valor do peso de um novo vizinho é iniciado com um valor predefinido, chamado *peso base* e depois é mantido usando uma regra simples que consiste no incremento do peso do vizinho por uma unidade sempre que uma mensagem é recebida e no decremento do peso também por uma unidade quando uma mensagem é enviada para esse vizinho. No FastRank, o valor do *peso base* é 0. Isto significa que um vizinho que envia mais mensagens do que recebe tem uma pontuação positiva, e um parasita irá ter uma pontuação negativa. Além disso, o algoritmo de classificação também define um valor mínimo para a pontuação, chamado *peso mínimo*, abaixo do qual um nó é considerado parasita. Na Secção 5, discutimos que valor de *peso mínimo* deverá ser escolhido.

3.3 Disseminação das Mensagens

No FastRank, a disseminação das mensagens é concretizada da seguinte forma. A fonte escolhe, para cada pacote, um número f de contactos entre os nós presentes na rede (isto é conseguido mantendo uma vista passiva de grande dimensão) e depois envia a informação para esses nós. Quando um nó recebe um pacote, directamente da fonte ou dos seus vizinhos, primeiro confirma que a trama não é duplicada (para este propósito, cada nó guarda informação com os identificadores dos últimos pacotes que recebeu). Informação repetida é descartada e nunca re-enviada para nenhum vizinho. Se a trama é nova, o nó irá re-enviá-la para cada um dos seus vizinhos com uma probabilidade que é uma função do peso desse vizinho. O tamanho da vista activa é deliberadamente pequeno mas suficientemente grande para tolerar uma fracção de nós não altruístas. Como resultado, se todos os nós forem altruístas, recorrer a uma estratégia de inundação poderá originar bastantes mensagens redundantes. Deste forma, um nó altruísta irá enviar uma mensagem para outro nó altruísta com probabilidade inferior a 1 que se chama *probabilidade base de retransmissão* (ou simplesmente *pbr*). O valor de *pbr* é escolhido de forma a que a fiabilidade da disseminação seja alta mas com um custo reduzido que é imposto quando a inundação é usada. Naturalmente, o valor de *pbr* depende do tamanho da vista activa. Na Secção 5, discutimos como *pbr* pode ser configurado.

A probabilidade base de retransmissão é usada para disseminar mensagens para outros nós altruístas. Por outro lado, se um vizinho foi detectado como parasita (i.e., se o seu peso for inferior ao *peso mínimo*), então nenhuma mensagem é retransmitida para esse vizinho. Além disso, para vizinhos que têm um peso negativo mas superior ao limite *peso mínimo*, as mensagens são retransmitidas com uma probabilidade inferior a *pbr* mas superior a 0. Mais precisamente, o

FastRank usa a seguinte fórmula para calcular a probabilidade de retransmissão de um nó i para um vizinho j, denomidada $pr_i(j)$:

$$pr_i(j) = \begin{cases} pbr & \text{se } peso_{ij} \geq peso\ base \\ pbr \left| \frac{peso\ mínimo - peso_{ij}}{peso\ mínimo} \right| & \text{se } peso\ base > peso_{ij} \geq peso\ mínimo \\ 0 & \text{se } peso\ mínimo > peso_{ij} \end{cases}$$

4 Comportamentos Considerados

O FastRank foi concebido para sistemas onde a maioria dos nós é altruísta e apenas uma pequena fracção de nós são parasitas ou racionais. O caso em que todos os nós são racionais está fora do âmbito deste artigo. Como discutido no trabalho relacionado, este cenário requer protocolos consideravelmente mais onerosos. Na secção de avaliação, analizaremos a qualidade do sistema para diferentes fracções de nós da população que adoptam diferentes comportamentos, considerando que os nós pertencem a uma das seguintes categorias:

- *Nós altruístas:* Um nó altruista segue o protocolo especificado sem nunca adoptar um comportamento diferente.
- *Parasitas:* São nós que nunca reenviam qualquer mensagem. Recebem informação dos seus vizinhos até serem expulsos. Além disso, tentam maximizar o número de vizinhos que têm.
- *Nós Racionais:* São nós que se tentam desviar do protocolo para maximizarem a sua utilidade. Este comportamento é discutido abaixo.

4.1 Comportamento Racional

Consideramos que um nó racional obtém benefício sempre que recebe um novo pacote e tem um custo sempre que envia um pacote ou resolve um puzzle criptográfico. Portanto, para maximizar a sua utilidade, um racional tenta aumentar o rácio entre os pacotes recebidos e enviados. É claro da descrição do algoritmo que, se um nó não mantém as trocas de informação balanceadas com um vizinho, eventualmente será detectado como parasita e expulso por esse vizinho. Existem duas possíveis estratégias que um nó racional pode ter para aumentar a sua utilidade:

- *Diminuição da vista activa:* Um nó racional pode optar por diminuir a sua vista activa, de modo a que receba os conteúdos mas tenha de reenviar informação para um número inferior de vizinhos. Com este ataque o nó aparenta ser altruísta para os seus vizinhos, que não têm maneira de detectar que este está a usar uma vista reduzida. Algoritmos mais sofisticados (e.g. LiFTinG [7]) que não dependem somente de informação local, foram desenvolvidos para detectar este ataque.

– *Serviço mínimo (para manter a sua pontuação estritamente acima do limite):* Um nó racional pode reduzir a taxa à qual reenvia pacotes de maneira a que seja capaz de manter a sua pontuação acima do *peso mínimo* e, portanto, nunca ser expulso da vista activa dos nós altruístas. Além disso, pode tentar arranjar tantos vizinhos quanto possível. A ideia é que se tiver vizinhos suficientes, poderá receber a informação com boa qualidade, apesar de raramente reenviar a informação .

Vamos mostrar que, face aos desvios acima, o FastRank apresenta as seguintes propriedades: i) a qualidade recebida por nós altruístas é apenas parcialmente afectada, mesmo para uma grande fracção de nós com comportamento não altruísta; ii) os nós racionais ainda têm de contribuir com uma quantidade razoável de recursos para conseguirem receber os dados contínuos com qualidade aceitável; iii) se existir uma grande fracção de nós com comportamento racional, isto poderá ser detectado por nós altruístas.

5 Avaliação

Nesta secção, iremos reportar uma extensa avaliação do FastRank. Todas as experiências foram feitas usando o simulador PeerSim [18]. A fonte utiliza um protocolo baseado em ciclos, enquanto os restantes nós, utilizam um protocolo baseado em eventos. As simulações usaram 1000 nós e consistiram na disseminação de 20000 pacotes, sendo cada um injectado pela fonte em 7 nós escolhidos aleatoriamente. Os resultados apresentados nesta secção são a média de 100 testes independentes, usando as mesmas configurações. A avaliação está dividida em quatro partes. Na primeira, motivamos a escolha dos valores dos diferentes parâmetros do sistema. A segunda parte ilustra a operação quando todos os nós são altruístas. A terceira discute o efeito dos nós parasitas e, finalmente, a quarta o efeito dos nós racionais no sistema.

5.1 Configuração do FastRank

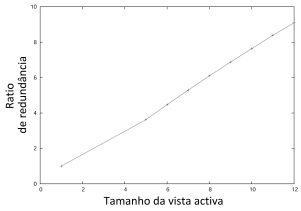
Os parâmetros que afectam a operação do FastRank são os seguintes: o tamanho da vista activa; a classificação mínima de um vizinho (*peso mínimo*); a probabilidade base de retransmissão *pbr*; e o *período de quarentena* (i.e., o tempo médio necessário para resolver um puzzle criptográfico quando se estabelece uma nova ligação). A Tabela 1 mostra os valores por omissão da configuração do FastRank. Discutimos a motivação para estes parametros nas subsecções seguintes.

Tempo de Criação da Rede. O tempo de criação da rede é o tempo necessário para que todos os nós acabem de resolver os puzzles criptográficos e estabeleçam um número de ligações igual ou acima da linha-de-água.

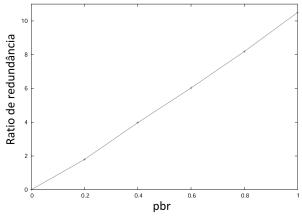
Tamanho da Vista Activa. Discutimos agora qual o tamanho da vista activa. Começamos por assumir que usariamos uma técnica de inundação para propagar as mensagens na rede (na secção seguinte discutimos porque é que não

parâmetro	**valor**	**parâmetro**	**valor**
vista activa	15	*pbr*	0.4
linha-de-água	12	período de quarentena (pacotes)	220
peso mínimo	-15		

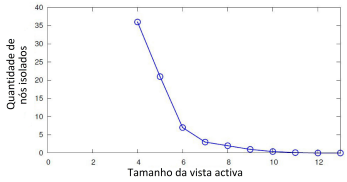
Tabela 1: Configuração por omissão doFastRank



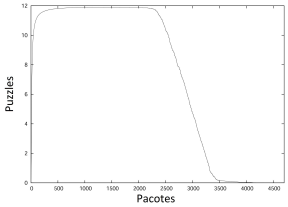
(a) Redundância vs tamanho vista.



(b) Redundância vs *pbr*.



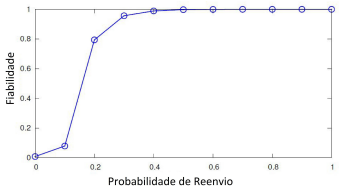
(c) Fracção de nós isolados após 30% de falhas.



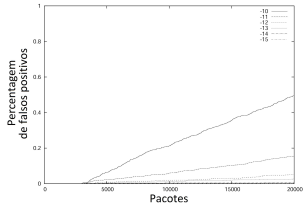
(d) Tempo para construir a rede.

Figura 1: Redundância, efeito de faltas e tempo para construir a rede

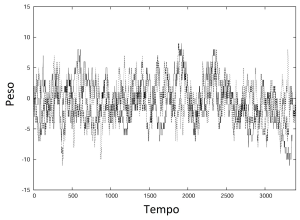
usamos inundação no FastRank). Se recorrêssemos à inundação, enquanto os nós altruístas estivessem ligados, todos estes nós iriam receber todas as mensagens. Desta forma, o tamanha da vista activa não teria impacto na fiabilidade. Portanto, o tamanho da vista activa deve ser escolhido de modo a que a probabilidade de que um nó altruísta fique isolado, na presença de nós parasitas, seja muito pequena. Optamos por configurar o FastRank para que pelo menos 30% de nós parasitas sejam tolerados com o mínimo dano para os nós altruístas. A Figura 1(c) mostra a percentagem de nós altruístas que ficam isolados após a construção da rede para diferentes tamanhos da vista activa. Como pode ser visto, se este tamanho for igual ou superior a 11, apenas 0.1% de nós altruístas ficam isolados. Este valor motivou o uso de uma abordagem mais conservativa e foi escolhido o valor 12.



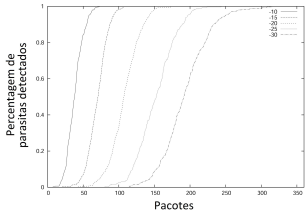
(a) Fiabilidade da difusão.



(b) Falsos-positivos em função do valor do peso mínimo.



(c) Flutuação do peso.

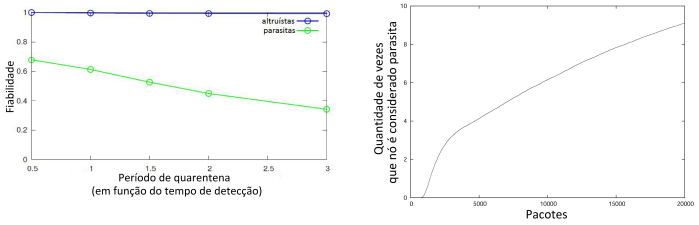


(d) Tempo necessário para detectar um parasita.

Figura 2: Parâmetros relevantes para a classificação dos vizinhos

Probabilidade de Reenvio. Na subsecção anterior foi escolhido um tamanho de vista activa em função da fracção de parasitas que pretendemos tolerar. Agora explicamos a motivação para escolher a probabilidade base de retransmissão *pbr*. A Figura 1(a) retrata quantas vezes, em média, um nó recebe a mesma mensagem se for usada inundação no sistema, para diferentes tamanhos da vista activa. Como pode ser visto, se a inundação for usada, existe bastante redundância, o que acarreta um desperdício significativo de recursos. Tal como nos protocolos epidémicos, evitamos esta redundância reenviando as mensagens com uma probabilidade inferior a 1. Fixando o valor da vista activa em 12, a Figura 2(a) retrata a fiabilidade do protocolo numa rede como a do FastRank, em função da probabilidade base de retransmissão e a Figura 1(b) retrata a redundância resultante. Como pode ser visto, escolhendo uma probabilidade base de retransmissão de 0.4 numa rede com uma vista activa de 12, é possível atingir valores de fiabilidade altos com uma redução significativa de redundância no processo de disseminação.

Classificação dos Vizinhos. O objectivo do mecanismo de classificação é detectar os nós parasitas o mais cedo possível, sem causar um número significativo de falsos positivos. Devido à aleatoriedade dos procedimentos de disseminação, o peso do vizinho poderá variar ao longo do tempo. Portanto, o valor de *peso mínimo* deverá considerar uma margem de segurança, para que estas variações



(a) Efeito do período de quarentena.

(b) Responde a um.

Figura 3: Configuração da quarentena

não marquem nós altruístas como parasitas. A Figura 2(c) mostra as variações do valor do peso entre dois nós altruístas para diferentes probabilidades base de retransmissão. Destes gráficos, é claro que o valor de *peso mínimo* não deverá ser superior a -15. De notar que, quanto mais baixo este valor for, menos provável é de gerar um falso positivo mas mais tempo demorará a detectar um nó parasita. Como discutimos na Secção 3, o protocolo penaliza nós com baixa pontuação, enviando-lhes mensagens com probabilidade inferior a *pbr*, prejudicando nós que enviam menos mensagens e conseguem manter a sua pontuação acima do *peso mínimo*. Um efeito secundário desta estratégia, é que o tempo necessário para detectar um nó parasita aumenta linearmente com o *peso mínimo*. A Figura 2(d) mostra o número de pacotes que um nó parasita recebe antes de ser detectado para diferentes valores de **peso mínimo**.

Período de quarentena. Como discutido antes, o objectivo do período de quarentena é assegurar que um nó parasita não consegue renovar as suas ligação mais rapidamente do que demora a ser detectado. A Figura 2(d) demonstra que são necessários 110 pacotes até detectar o último nó parasita, para um *peso mínimo* de -15. Desta forma, o mecanismo de entrada na rede deverá utilizar um puzzle criptográfico que, em média, demore mais tempo a ser resolvido do que demora a enviar esse número de pacotes. A Figura 3(a) mostra a fiabilidade de um nó com o seguinte comportamento: nunca reenvia nenhum pacote e tenta estabeleber quantas ligações quanto possível. Sabendo que enquanto estiver em período de quarentena não consegue estabelecer outra ligação, a figura retrata a fiabilidade obtida por um nó parasita para diferentes períodos de quarentena. Pode ser observado que se o período de quarentena for superior ao tempo que demora à sua detecção, então a proporção de pacotes recebidos por um nó parasita, decresce significamente.

Resolução do Puzzle Criptográfico. O período de quarentena assegura que um nó demora mais tempo a resolver um puzzle criptográfico do que aquele que um vizinho demora a detectar um nó como sendo parasita. Uma consequência desta abordagem é que se um nó apenas tem uma ligação e está a resolver puzzles para outros vizinhos, ele será considerado como parasita para o seu único vizinho,

antes de conseguir estabelecer novas ligações. A Figura 3(b) retrata quantas vezes, em media, um nó é marcado como parasita se este resolver um puzzle e logo após a sua resolução, envia a resposta. Como pode ser observado, cada nó, em média é considerado como parasita por 9 dos seus vizinhos.

5.2 Operação Livre de Desvios

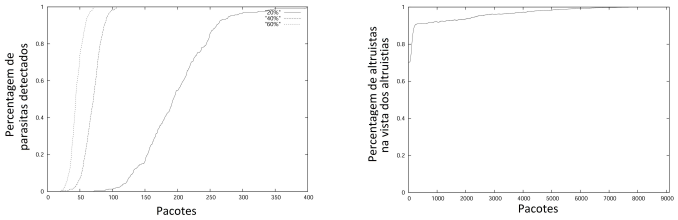
Nesta subsecção, fornecemos informação adicional sobre a operação num cenário livre de desvios, i.e., numa configuração onde todos os nós são altruístas. Para isto, considerámos uma transmissão contínua de dados de 24 tramas por segundo, geradas por uma única fonte, durante uma sessão completa de 14 minutos. A sessão começa com 1000 nós e a meio da transmissão (ao minuto 7), 100 novos nós são adicionados (i.e., neste momento, provocamos um aumento de 10% da população da rede). Com esta configuração, mostrámos: o tempo que demora a configurar a rede do FastRank, a fiabilidade recebida por um nó; o número médio de retransmissões de cada pacote; a percentagem de falsos-positivos durante a sessão; a quantidade de puzzles criptográficos resolvidos por cada nó; e, finalmente, o tempo que demora a um nó para se juntar a uma transmissão até atingir 90% de fiabilidade. Os resultados estão expostos na Tabela 2. Como pode ser visto, os valores obtidos coincidem com os previstos para a configuração do FastRank. Com uma vista activa de 12 e uma probabilidade base de retransmissão de 0.4, é esperado que cada trama seja envidada, em média, para 8.4 vizinhos, o que pode ser medido experimentalmente.

5.3 Tolerar Nós Parasitas

Nesta secção, fornecemos informação adicional sobre a operação do FastRank na presença de nós parasitas. Nestas experiências, o sistema corre com 100% de nós altruístas até ao momento em que uma fracção dos nós assume comportamento parasita. Estes nós deixam de reenviar informação e tentam estabelecer tantas ligações tantas possível. A Figura 4(b) mostra a evolução da composição das vistas activas dos nós após os parasitas terem sido injectados. A figura mostra que após 110 tramas os parasitas começam a ser detectados e o sistema começa a reconfigurar. Após, aproximadamente, 2500 tramas, o sistema atinge numa configuração onde 95% dos membros da vista activa de um altruísta são outros altruístas. Os parasitas ficam desligados da rede. Também medimos quantos puzzles criptográficos os altruístas e os parasitas têm de resolver durante a reconfiguração. Em média, um altruísta tem de resolver 6.75 puzzles antes de

Tamanho inicial da rede	1000	Falsos positivos	0.030
Nós adicionais	100	Fiabilidade global	0.999
Puzzles resolvidos pelos novos nós	12.2	Fiabilidade dos novos nós	0.995
Tempo para se juntar (pacotes)	3116		

Tabela 2: Operação com nós altruístas



(a) Tempo necessário para detectar um parasita para diferentes valores da *pbr* usando peso mínimo = -15.

(b) Recuperando de comportamentos parasitas.

Figura 4: Efeitos dos parasitas

estabilizar a sua vista activa com outros nós altruístas. Naturalmente, os parasitas continuam constantemente a resolver puzzles para substituir as ligações perdidas, mas isto não lhes permite receber a sequência de tramas.

5.4 Tolerar Nós Racionais

Esta subsecção, mostra os comportamento do FastRank na presença de nós racionais. Consideramos dois tipos de ataques descritos na Secção 4. Primeiro, discutimos o impacto de existirem nós que deliberadamente cancelam as ligações e aparentam comportarem-se correctamente para os nós que escolherem manter na sua vista. Experimentalmente, aferimos que os nós precisam de manter pelo menos 7 membros na sua vista activa para conseguerem ter uma fiabilidade aceitável. Desta forma, devido ao valor da *linha-de-água* usado pelos altruístas ser de 12 e o tamanho da *vista activa* ser de 15, os nós racionais conseguem poupar até 60% dos recursos. Na Tabela 3, mostramos a composição da vista dos nós altruístas e dos nós racionais, a fiabilidade da transmissão após o ataque, e o aumento na latência da transmissão devido ao ataque. Uma vez que os nós racionais limitam o número dos seus vizinhos, os nós altruístas substituem essas ligações por ligações a outros altruístas. Desta forma a rede mantém-se conexa e a fiabilidade não é afectada. No entanto, o diâmetro da rede aumenta, e isso tem um impacto observável na latência.

Finalmente, discutimos o impacto de existir um nó racional a enviar mensagens suficientes para manter a sua pontuação no limite de forma a nunca ser detectado. A Tabela 3 mostra o efeito deste tipo de comportamento. Como pode ser visto, se um nó racional utiliza este comportamento, mas não aumenta o número de ligações, é severamente penalizado. Por outro lado, mesmo com 30% de nós racionais, este ataque não tem impacto na fiabilidade observada pelos altruístas.

Também analisámos uma variante deste comportamento, onde não só os nós racionais mantêm as suas ligações no limite de *peso mínimo*, mas tentam esta-

Ataque de diminuição da vista activa: efeitos no sistema (30% de nós racionais)					
Vista activa dos nós racionais	-	*3*	*4*	*5*	*6*
Salto máximo	1	1.27	1.45	1.37	1.09
Salto médio	1	1.07	1.08	1.13	1.11
Fiabilidade altruístas	0.99	0.99	0.99	0.99	0.99
Fiabilidade racionais	-	0.03	0.06	0.67	0.88

Ataque de serviço mínimo: efeitos no sistema (30% de nós racionais)	
Fiabilidade dos altruístas após o ataque	0.97
Faibilidade dos racionais após o ataque	0.50
Fracção de mensagens enviadas pelos racionais	0.2

Ataque de serviço mínimo: efeitos no nó racional						
Vista activa dos nós racionais	*12*	*20*	*25*	*30*	*35*	*40*
Rácio de puzzles resolvidos	1	5	10	17	45	49
Rácio médio de mensagens	0.24	0.3	0.40	0.48	0.66	0.80
Fiabilidade do racional	0.71	0.80	0.86	0.98	0.94	0.99

Tabela 3: Efeitos de ataques racionais

belecer novas ligações. A Tabela 3 mostra quanto tempo um nó demora a ter uma fiabilidade semelhante à de um altruísta. Pode ser observado que um nó com um rank de *peso mínimo* em todas as suas ligações, precisa de manter uma vista activa constante de, pelo menos, 25. No entanto, tem de resolver 22 vezes mais puzzles criptográficos do que os nós altruístas para atingir esse ponto. Além disso, como tem de manter todas as ligações acima do limite, ainda é obrigado a reenviar informação: aproximadamente 80% de um altruísta. Desta forma, o ataque de serviços mínimos é menos lucrativo do que encolher o tamanho da vista activa. Estes resultados mostram que, apesar dos nós racionais beneficiarem de usarem estratégias sofisticadas num sistema como o FastRank, ainda têm de cooperar ajudando no processo de disseminação.

6 Conclusões

Neste artigo apresentámos o FastRank, um protocolo de transmissão contínua de dados entre-pares que se baseia na utilização de uma rede sobreposta com ligações simétricas para mitigar o efeito de nós parasitas de uma maneira eficiente e eficaz. O FastRank opera com baixo custo nos cenários favoráveis em que a percentagem de nós parasitas ou racionais é mínima (o que acontece regularmente na prática). O FastRank também consegue tolerar e detectar uma fracção de nós com comportamento racional, de forma a permitir reconfigurar o sistema quando o número de nós racionais se torna excessivo.

Agradecimentos Este trabalho foi parcialmente suportado pela Fundação para a Ciência e Tecnologia (FCT) através dos projectos com referência PEPITA (PTDC/EEI-SCR/2776/2012) e UID/CEC/50021/2013.

Referências

1. E. Adar and B. Huberman. Free riding on gnutella. *First Monday*, 5(10), October 2000.
2. Nikita Borisov. Computational puzzles as sybil defenses. In *IEEE P2P 2006*, pages 171–176, 2006.
3. Bram Cohen. Incentives build robustness in bittorrent. In *Workshop on Economics of Peer-to-Peer systems*, pages 68–72, 2003.
4. John R Douceur. The sybil attack. In *Peer-to-peer Systems*, pages 251–260. Springer, 2002.
5. Wu-chi Feng, E Kaiser, and Antoine Luu. Design and implementation of network puzzles. In *IEEE INFOCOM 2005*, volume 4, pages 2372–2382, 2005.
6. Ayalvadi J. Ganesh, A.-M. Kermarrec, and Laurent Massouli. Scamp: Peer-to-peer lightweight membership service for large-scale group communication. pages 44–55. Springer-Verlag, 2001.
7. Rachid Guerraoui, Kévin Huguenin, Anne-Marie Kermarrec, Maxime Monod, and Swagatika Prusty. Lifting: lightweight freerider-tracking in gossip. In *Middleware*, pages 313–333, 2010.
8. Yan Huang, Tom Fu, Dah-Ming Chiu, John Lui, and Cheng Huang. Challenges, design and analysis of a large-scale p2p-vod system. *SIGCOMM Comput. Commun. Rev.*, 38(4):375–388, 2008.
9. D. Hughes, G. Coulson, and J. Walkerdine. Free riding on gnutella revisited: The bell tolls? *IEEE Distributed Systems Online*, 6(6):1–, June 2005.
10. Murat Karakaya, Ibrahim Korpeoglu, and Ozgur Ulusoy. Free riding in peer-to-peer networks. *Internet Computing, IEEE*, 13(2):92–98, 2009.
11. A-M Kermarrec, Laurent Massoulié, and Ayalvadi J. Ganesh. Probabilistic reliable dissemination in large-scale systems. *IEEE Trans. on Parallel and Distributed Systems,*, 14(3):248–258, 2003.
12. J. Leitão, J. Pereira, and L. Rodrigues. Epidemic broadcast trees. In *IEEE SRDS*, pages 301–310, Beijing, China, October 2007.
13. João Leitão, José Pereira, and Luís Rodrigues. Hyparview: A membership protocol for reliable gossip-based broadcast. In *IEEE DSN*, pages 419–428, 2007.
14. Harry C Li, Allen Clement, Mirco Marchetti, Manos Kapritsos, Luke Robison, Lorenzo Alvisi, and Mike Dahlin. Flightpath: Obedience vs. choice in cooperative services. In *OSDI*, volume 8, pages 355–368, 2008.
15. Harry C Li, Allen Clement, Edmund L Wong, Jeff Napper, Indrajit Roy, Lorenzo Alvisi, and Michael Dahlin. Bar gossip. In *OSDI*, pages 191–204, 2006.
16. Xiaofei Liao, Hai Jin, Yunhao Liu, Lionel M Ni, and Dafu Deng. Anysee: Peer-to-peer live streaming. In *INFOCOM*, volume 25, pages 1–10, 2006.
17. Ralph C Merkle. Secure communications over insecure channels. *Comm. of the ACM*, 21(4):294–299, 1978.
18. Alberto Montresor and Mark Jelasity. Peersim: A scalable p2p simulator *.
19. Paul Resnick et al. The social cost of cheap pseudonyms. *Journal of Economics & Management Strategy*, 10(2):173–199, 2001.
20. XiaoFeng Wang and Michael K Reiter. Defending against denial-of-service attacks with puzzle auctions. In *Security and Privacy*, pages 78–92. IEEE, 2003.