# Minimização de Perda de Pacotes em Sistemas de Deteção de Intrusões

Tiago Picado, Paolo Romano e Carlos Ribeiro

INESC-ID/IST
tiago.picado@ist.utl.pt,
{romano,carlos.ribeiro}@inesc-id.pt

**Resumo** Neste trabalho foi investigada a possibilidade de minimizar falhas por omissão em sistemas de deteção de intrusões de rede, utilizando múltiplas réplicas, com base na ideia de que a probabilidade de um conjunto de réplicas perder um dado pacote deverá ser menor do que a de apenas uma instância. Foi desenvolvida uma camada de sincronização, que deteta desalinhamentos no tráfego recebido pelas réplicas e recupera os pacotes perdidos por cada uma, antes de os entregar à camada de deteção. Os resultados sugerem que em alguns cenários esta camada pode minimizar as falhas por omissão.

**Palavras-chave:** Sistemas de deteção de intrusões de rede; falhas por omissão; replicação; protocolo de sincronização; diff

## 1 Introdução

Para proteger sistemas e redes informáticas de atacantes, a abordagem convencional é a de implementar um conjunto de mecanismos protetores, por forma a procurar garantir a sua segurança. Esta abordagem tem contudo algumas limitações [1], que justificam a utilização de outras. Os sistemas de detecção de intrusões podem fornecer uma segunda linha de defesa, ao permitir a deteção atempada das atividades de intrusão iniciais, ao dissuadir as intenções dos intrusos ou ao permitir a recolha de informação acerca de técnicas de intrusão, que pode depois ser usada para fortalecer os mecanismos preventivos [2].

Para que cumpram o seu papel, é importante que seja dada aos sistemas de detecção de intrusões a oportunidade de analisar o máximo possível dos dados disponíveis. É no entanto sabido que os sistemas de deteção de intrusões de rede em particular, podem perder pacotes por diversos motivos, nomeadamente quando são incapazes de processar de forma suficientemente célere o fluxo de tráfego que lhes é apresentado para análise. Mesmo que a taxa de pacotes perdidos pelo sistema seja baixa, aqueles que são perdidos poderão ser importantes para a deteção de uma intrusão ou tentativa de intrusão relevante.

Neste trabalho, foi investigada a possibilidade de minimizar a ocorrência de falhas por omissão por parte de um sistemas de deteção de intrusões de rede, através da utilização de múltiplas réplicas desse sistema, sincronizadas entre si. A ideia subjacente é o facto de a probabilidade de perda de um dado pacote por

um conjunto de réplicas dever ser inferior à probabilidade de perda desse pacote por um sistema composto por uma única instância de um sistema de deteção de intrusões. O objetivo foi o de desenvolver um protocolo de sincronização, capaz de detetar réplicas desalinhadas, relativamente ao fluxo de pacotes recebidos por cada uma para análise, de determinar quais os pacotes específicos em falta em cada uma, e de recuperar os pacotes requeridos, para que todas as réplicas pudessem analisar o mesmo conjunto de pacotes.

Para conseguir este objetivo, foi implementada uma camada de sincronização, colocada entre a biblioteca de captura de tráfego de rede e a camada de análise. Para esta última, foi utilizado o popular sistema de código aberto Snort, que na atual versão é mono-tarefa, deixando por isso por aproveitar parte da capacidade existente nos atuais processadores, predominantemente multi-núcleo.

**Trabalho relacionado:** muitos autores têm referido a importância de os sistemas de deteção de intrusões serem tolerantes a faltas. No entanto, essas referências e os trabalhos desenvolvidos centram-se na capacidade de os sistemas serem capazes de lidar com faltas por paragem ou com componentes comprometidos por atacantes (ex.: [3], [4]). O trabalho apresentado é, tanto quanto sabemos, o primeiro que procura lidar com faltas por omissão, utilizando múltiplas réplicas para procurar minimizar a ocorrência deste tipo de faltas.

## 2 Solução Proposta

O sistema tolerante a faltas proposto foi desenhado considerando a configuração típica de um sistema de deteção de intrusões que escuta apenas uma interface de rede, da qual recebe os pacotes a analisar. Assume-se que essa interface é utilizada apenas para análise de tráfego, e em particular que o sistema não envia tráfego gerado por si mesmo, através dessa interface. Neste cenário, um sistema autónomo poderá ser substituído por um conjunto de réplicas, cada uma executando uma camada de sincronização, que por sua vez alimenta a sua camada de análise. Para isso, todas as réplicas deverão receber os mesmos pacotes de rede, enviados por uma dada interface de origem, e duplicada por um dispositivo que se comporte como um repetidor multi-porta. Uma segunda interface é utilizada para as tarefas de sincronização e quaisquer outras funções necessárias.

**Mecanismo de Sincronização:** Num sistema composto por múltiplas réplicas, cada uma recebendo e processado o mesmo tráfego de rede, as omissões, se ocorrerem, não acontecerão necessariamente da mesma forma em todas as réplicas, devido a diferenças na carga e ritmo de sistemas assíncronos e independentes, mesmo que o factor mais determinante para a marcação desse ritmo, o fluxo de pacotes, seja o mesmo para todas. O sistema proposto procura tirar partido deste facto, ao combinar os dados recebidos por cada réplica, e tentando detetar se alguns dos pacotes foram perdidos por uma ou mais réplicas. Se for capaz de detetar a ocorrência de perda de pacotes, procura inferir, da melhor forma possível, qual teria sido o fluxo de pacotes original.

Para isso, e para minimizar a sobrecarga com a sincronização, uma representação compacta de cada pacote recebido é gerada e armazenada, utilizando

uma função de síntese. A cada $k$ pacotes, uma representação compacta do conjunto correspondente de sínteses é gerada da mesma forma, sendo de seguida trocada e comparada por todas as réplicas. Uma vez que a função é determinística, na ausência de faltas, as sínteses calculadas serão iguais em todas as réplicas, e também as sínteses geradas a partir das sínteses. Isto também será verdade se todas as réplicas perderem exatamente o mesmo sub-conjunto de pacotes, mas neste caso a falha não poderá ser recuperada. Se, por outro lado, as perdas não envolverem exatamente o mesmo sub-conjunto, a síntese de $k$ sínteses diferirá, e as faltas tornam-se visíveis. Uma vez detetadas, as réplicas trocam as sínteses dos pacotes individuais e executam um algoritmo determinístico para determinar quais as sínteses que são comuns, quais as que diferem, e como deverão as diferenças ser fundidas. Tomando as sequências de sínteses como linhas de um ficheiro de texto, esta tarefa pode ser deixada ao estabelecido algoritmo `diff`.

**Realinhamento:** Uma vez desalinhadas as réplicas, torna-se necessário encontrar um novo ponto de sincronização, o que é equivalente a encontrar uma "âncora"e a transferir os pacotes em falta, anteriores a essa "âncora", entre réplicas, antes de retomar o processo de comparação apenas das sínteses das sínteses. Uma "âncora"é definida como um bloco de uma dada dimensão de $l$ pacotes contíguos, que estão presentes em todas as réplicas. Se tal bloco existir, considera-se que as réplicas estão alinhadas nesse bloco, e o algoritmo `diff` pode ser utilizado para determinar a melhor forma de fundir todas as sínteses de pacotes que o precedam. Se não existir, considera-se que as réplicas estão desalinhadas de tal forma que a utilização do `diff` não é válida, e torna-se necessário incluir as $k$ sínteses seguintes, antes de tentar realinhar as réplicas. Poderiam ser adicionadas mais sínteses uma a uma, até ser possível encontrar uma "âncora"e retomar então o processo de comparação de sínteses das sínteses; no entanto, na atual implementação, o mecanismo de sincronização progride sempre em blocos de $k$ pacotes. Se o primeiro bloco de $k$ pacotes desalinhado não puder ser realinhado, é adicionado um segundo bloco, e diz-se que ocorreu uma dessincronização de nível 1. No entanto, do ponto de vista do algoritmo de fusão, uma dessincronização de nível 1 é simplesmente equivalente a utilizar um tamanho de $2k$ para os blocos de sínteses. Se existirem múltiplas "âncoras", é utilizada a que esteja mais próxima do final do bloco de sínteses em avaliação.

**Consumo Pessimista/Otimista de Pacotes:** A camada de sincronização regista os pacotes capturados ou ressincronizados num segmento de memória partilhada. A camada de análise poderá consumir estes pacotes apenas até ao ponto em que o seu alinhamento esteja confirmado - modo pessimista - ou poderá também consumir pacotes até aos pertencentes ao bloco em sincronização, cujo alinhamento ainda não esteja confirmado - modo otimista. O modo otimista requer um mecanismo de salvaguarda de pontos de controle e de reversão do estado, para que a camada de análise possa ser revertida a um estado em que que apenas pacotes alinhados tenham sido consumidos, caso se verifique que consumiu otimisticamente pacotes desalinhados. Este modo pode ser relevante para minimizar a latência introduzida pela sincronização, quando o sistema de deteção de intrusões funciona de forma reativa ao detetar tentativas de intrusão.

## 3 Resultados e Discussão

Para avaliação do sistema, que atualmente suporta apenas duas réplicas, foram considerados dois cenários de aplicação típicos. Foi possível encontrar para ambos condições específicas em que a utilização do protocolo se mostrou vantajosa.

**Picos de Tráfego:** O primeiro cenário de aplicação considerado foi o caso em que ocorrem esporadicamente picos de tráfego no padrão habitual, com que o sistema é incapaz de lidar, perdendo alguns pacotes. Uma vez que esses picos são seguidos por períodos de tráfego muito menos intenso, o sistema tem a oportunidade de recuperar alguns dos pacotes que cada réplica perdeu, e de aliviar recursos de memória que ficaram saturados. Num teste com injeção de tráfego a 50 MBit/s, em que foram introduzidos alguns picos de 300 MBit/s, um sistema Snort não replicado e não modificado perdeu em média menos pacotes do que cada réplica sincronizada, mas as camadas de análise (Snort) dessas réplicas acabaram por processar mais pacotes, devido ao mecanismo de recuperação.

| **Pacotes perdidos** | **Média 2 réplicas** | **Sem replicação** |
|---|---|---|
| **Biblioteca de captura** | 1266 | 1070 |
| **Snort** | 552 | 1070 |

**Tabela 1.** Perda de pacotes média em tráfego com injeção de picos.

**Sistemas Sobrecarregados:** O segundo cenário de aplicação considerado foi o caso em que um sistema fica temporariamente sujeito a uma carga de processamento elevada, que poderá ser devida a algum processo independente do processo de deteção de intrusões, uma vez que existem quase sempre outros processos, mesmo num sistema dedicado à deteção de intrusões. Num teste com injeção de tráfego a 50 MBit/s, com a execução de um processo curto mas exigente em processamento numa das réplicas, foi possível nela provocar uma perda média de 385 pacotes, que foram recuperados a partir da outra réplica.

**Agradecimentos.** Este trabalho foi parcialmente suportado por fundos nacionais através da FCT, sob o projeto PEst-OE/EEI/LA0021/2011.

## Referências

1. Mukherjee, Biswanath and Heberlein, L. Todd and Levitt, Karl N.: Network Intrusion Detection. IEEE Network, 8(3), pp. 26-41, May/Jun 1994.
2. Stallings, William: Network Security Essentials: Applications and Standards. Prentice Hall Professional Technical Reference, 2002, ISBN 0130351288.
3. Crosbie, Mark and Spafford, Gene. Active defense of a computer system using autonomous agents. Technical Report 95-008, COAST Group, Department of Computer Sciences, Purdue University, West Lafayette, IN 47907-1398, Feb 1995.
4. Sen, Jaydip and Sengupta, Indrani: Autonomous agent-based distributed fault-tolerant intrusion detection system. In Proc. of the ICDCIT, pp. 125-131, Dec 2005.