MicroSec Traffic: Utilizando Estratégias de Engenharia de Tráfego para Aprimoramento da Eficiência de Sistemas de Detecção de Intrusão

Mateus dos Santos Herbele[1], **Raphael Kaviak Machnicki**[1], **Vinicius Fulber-Garcia**[1]

[1]Departamento de Informática – Universidade Federal do Paraná (UFPR)
Caixa Postal 19.011 – 81.530-090 – Curitiba – PR – Brasil

{msh22,rkmachnicki,vinicius}@inf.ufpr.br

Abstract. *This work proposes MicroSec Traffic, an approach to improve the efficiency of traditional IDS solutions based on rule-defined signatures and anomalies by reducing the data load of network traffic without compromising threat detection. The technique requires no modifications to IDS tools such as Snort or Suricata; only adjustments to the rules in use. Evaluated in a controlled scenario with Snort, the approach proved effective in maintaining alert generation while reducing both processing time and data volume.*

Resumo. *Este trabalho propõe o MicroSec Traffic, uma abordagem para melhorar a eficiência de soluções IDS tradicionais baseadas em assinaturas e anomalias definidas por regras, por meio da redução da carga de dados do tráfego de rede, sem comprometer a detecção de ameaças. A técnica não exige modificações nas ferramentas IDS, como Snort ou Suricata, apenas ajustes nas regras utilizadas. Avaliada em um cenário controlado com o Snort, a abordagem demonstrou ser efetiva ao manter a geração de alertas com menor tempo de processamento e volume de dados.*

1. Introdução

Nas redes de computadores atuais, o crescente volume de tráfego e a complexidade arquitetural, além do número e da sofisticação de novas ameaças cibernéticas, impõem grandes desafios tanto à segurança quanto ao desempenho dos sistemas conectados [Asad et al. 2024]. Dentre as ferramentas disponíveis para identificar essas ameaças, os Sistemas de Detecção de Intrusão (*Intrusion Detection Systems* – IDS), como o Snort[1], desempenham um papel importante ao fornecer recursos para o monitoramento e a indicação de padrões suspeitos no tráfego de rede.

No entanto, soluções tradicionais de IDS realizam operações bastante custosas computacionalmente para viabilizar a detecção de ameaças. Em cenários com baixo volume de tráfego, essas soluções conseguem inspecionar por completo os quadros e pacotes recebidos. Porém, conforme o volume de tráfego aumenta, a eficácia das soluções de IDS pode ser severamente comprometida, uma vez que sua capacidade de processamento é excedida, resultando em atrasos na notificação, perda de pacotes, análises parciais de fluxos e, eventualmente, diminuição na capacidade de detecção e geração de alertas [Machnicki et al. 2024].

[1]https://www.snort.org

Dadas as dificuldades mencionadas, diversos autores têm empregado esforços no desenvolvimento de soluções de IDS de alto desempenho para processar grandes volumes de tráfego. Entre essas soluções, destacam-se aquelas implementadas utilizando eBPF [Wang and Chang 2022] [Machnicki et al. 2024], P4 [Lewis et al. 2019] e FPGA [Zhao et al. 2020]. No entanto, considerando a flexibilidade de instanciação, facilidade de configuração e uso das soluções de IDS tradicionais, em software executando em hardware de propósito geral, elas ainda são amplamente adotadas e desempenham um papel importante para a cibersegurança. Sendo assim, um importante desafio a ser enfrentado consiste no aprimoramento das condições de operação dessas soluções, reduzindo gargalos e aprimorando a eficiência e eficácia destas na identificação de ameaças.

Nesse contexto, o presente trabalho propõe o *MicroSec Traffic*, uma abordagem baseada em engenharia de tráfego, utilizando *packet washing* [Dong and Clemm 2021], para reduzir a carga de dados processados por soluções de IDS. O método visa ajustar as regras instaladas em um IDS e moldar o tráfego de rede ingressante para preservar apenas as informações essenciais para a detecção de ameaças, otimizando o desempenho da solução sem comprometer sua eficácia ou requisitar modificações técnicas. A abordagem proposta foi avaliada em um ambiente controlado, por meio da reprodução e detecção de ataques, avaliando os resultados relacionados à efetividade do IDS de teste. Os resultados demonstraram que a técnica *MicroSec Traffic* teve impacto positivo e relevante na inspeção de tráfego de rede (redução de 83% no tempo de execução, sem prejuízos relevantes na detecção de ataques). Os testes também permitiram realizar uma discussão profunda sobre as implicações práticas, desafios e oportunidades relacionados ao *MicroSec Traffic*. Assim, destacam-se as principais contribuições deste trabalho:

- Proposição e definição de uma nova técnica de engenharia de tráfego, baseada em *Packet Washing*, para aprimoramento da eficiência de soluções de IDS no processamento de cópias modificadas do tráfego de rede: o *MicroSec Traffic*;
  - Foco na minimização de dados transmitidos/processados, sem perda de informações de detecção.
- Avaliação do *MicroSec Traffic* utilizando um IDS amplamente conhecido e utilizado, o Snort, e um *dataset* com amostras reais de ataques;
  - Foco na compatibilidade integral com as soluções legadas, mantendo-as completamente inalteradas, sendo necessário modificar apenas o tráfego de entrada e as regras configuradas.
- Discussão das limitações, desafios e oportunidades relacionados à técnica proposta.
  - Identificação e justificativa da não aplicação da técnica com soluções de IPS, apenas com IDS;
  - Apresentação do problema de pesquisa relacionado à seleção, adaptação e conciliação de regras de IDS para aplicação ao *MicroSec Traffic*.

O restante deste artigo está organizado como segue: a Seção 2 apresenta os conceitos básicos necessários para o entendimento da proposta; a Seção 3 define o *MicroSec Traffic*, bem como os conceitos e ferramentas utilizados para sua implementação. A Seção 4 descreve os experimentos e apresenta os resultados obtidos. A Seção 5 identifica e discute limitações, desafios e oportunidades. Finalmente, a Seção 6 conclui o artigo.

2. Fundamentação Teórica

Esta seção apresenta o arcabouço conceitual básico utilizado neste trabalho. Na Subseção 2.1, fundamentos de engenharia de tráfego e de suas principais estratégias são apresentados. Na Subseção 2.2, os principais conceitos sobre IDS são destacados. A Subseção 2.3 descreve e analisa os principais trabalhos relacionados.

2.1. Engenharia de Tráfego

A popularização da Internet no século XXI trouxe os benefícios do acesso rápido à informação e ao entretenimento para cerca de 5,56 bilhões de usuários [Kepios et al. 2025]. Tal feito, embora um marco para as redes de computadores, também gerou grandes desafios para o gerenciamento e funcionamento dessas redes [Pedreno-Manresa et al. 2017]. Nesse contexto, a engenharia de tráfego se torna especialmente relevante, pois trata do processo de otimização de recursos em redes IP visando melhorar seu desempenho e eficiência [Wang et al. 2008]. Entre seus principais objetivos estão: balancear a distribuição de carga, minimizar o consumo de largura de banda, garantir qualidade de serviço e aumentar a resiliência da rede contra falhas.

Diversas estratégias podem ser adotadas para implementar engenharia de tráfego; no entanto, três se destacam: *traffic policing*, *traffic shaping* e *packet washing*. A primeira, o *traffic policing*, estabelece limites para o tráfego que passa pela rede, descartando os pacotes que excedam os limites [Flach et al. 2016]. A segunda, o *traffic shaping*, apresenta uma abordagem menos agressiva, porém mais custosa em termos de recursos computacionais. Também baseada na definição de limites para o tráfego, essa estratégia armazena os pacotes excedentes em uma fila, para poderem ser transmitidos posteriormente, em momento oportuno, conforme uma política de enfileiramento que também precisa ser determinada [Marcon et al. 2011]. Na primeira, o descarte de pacotes é o objetivo da estratégia; na segunda, o descarte de pacotes ocorre apenas como uma eventual consequência da utilização total da memória dedicada à fila.

Por fim, a terceira abordagem, conhecida como *packet washing*, introduz um modelo mais granular de gerenciamento de pacotes. Ao invés de descartar pacotes inteiros com base em limites, como no *traffic policing*, essa técnica realiza uma divisão da carga útil em pedaços (*chunks*), que recebem diferentes valores de prioridade de acordo com sua importância para a aplicação receptora. Isso permite que, em situações de sobrecarga, o descarte ocorra apenas para os pedaços de menor prioridade, preservando a baixa latência e a integridade dos dados mais relevantes. É importante destacar que, originalmente, a definição desses pedaços prioritários é responsabilidade da aplicação transmissora, a qual deve distinguir os dados que podem ser descartados sem prejuízo à comunicação daqueles cuja perda exigirá retransmissão do pacote [Dong and Clemm 2021].

2.2. Sistemas de Detecção de Intrusão

A detecção de intrusão, em geral, consiste em processos de monitoramento de eventos com o objetivo de classificá-los como normais ou maliciosos [Liao et al. 2013]. Os sistemas de detecção de intrusão podem processar tráfego de rede ou utilizar outros recursos para identificar ameaças, como *logs* de sistema. Além disso, um IDS pode ser dedicado ao monitoramento e geração de alertas para uma rede de computadores (*Network* IDS) ou dedicado a um sistema em particular (*Host* IDS) [Soniya and Vigila 2016,

Frasão et al. 2024]. Além disso, existem duas abordagens principais para a detecção de ameaças por meio de IDS: detecção baseada em anomalias e baseada em assinaturas [Shafi et al. 2025].

A primeira abordagem, de detecção baseada em anomalias, busca modelar o comportamento padrão da rede a partir do histórico de tráfego normal. Assim, alertas são gerados quando desvios significativos em relação ao padrão são detectados. Essa estratégia pode empregar métodos como aprendizado de máquina, técnicas estatísticas, conhecimento especializado ou mineração de dados. Em particular, destaca-se a estratégia baseada em regras [Garcia-Teodoro et al. 2009]. Nessa estratégia, definem-se explicitamente condições sobre atributos temporais (por exemplo, frequência de pacotes em janelas de tempo) e volumétricos (tamanho ou contagem de pacotes), para disparar alertas sempre que tais parâmetros ultrapassarem limites pré-estabelecidos.

A segunda abordagem, de detecção baseada em assinaturas, identifica ameaças conhecidas por meio da comparação do tráfego de rede com padrões de ataque previamente documentados. Cada assinatura descreve, de forma precisa, características estáticas ou comportamentais de uma ameaça, como sequências de bytes no *payload* de um pacote, combinações específicas de *flags* TCP/IP ou padrões específicos em campos de cabeçalho, que, quando simultaneamente satisfeitos, disparam um alerta.

Em termos de implementação, algumas soluções de IDS amplamente conhecidas e utilizadas incluem:

- Snort[2]: solução lançada em 1998 para análise de assinaturas em tráfego de rede. Em suas versões legadas, operava em um modelo *single thread*. Porém, em sua versão mais recente (o Snort 3, lançado em 2021) foi atualizado para operar em um modelo *multi thread*, resultando em aprimoramentos significativos na eficiência do processamento de tráfego de rede.
- Suricata[3]: diferentemente do Snort, o Suricata foi lançado em 2010 já adotando um modelo *multi thread* para detecção de intrusões. Assim, uma década antes de o Snort passar a operar com múltiplas *threads* na análise de assinaturas maliciosas no tráfego de rede, o Suricata já se consolidava como a principal alternativa para lidar com grandes volumes de tráfego.
- Zeek[4]: lançado em 1996 e inicialmente chamado de Bro, o Zeek é um analisador de tráfego de rede capaz de identificar assinaturas maliciosas. Ele emprega diversos mecanismos visando a geração de logs de alta fidelidade, que facilitam a tomada de decisões em Centros de Operações de Segurança.

Também é relevante destacar que as soluções de IDS são diferentes dos Sistemas de Prevenção de Intrusão (*Intrusion Prevention Systems* - IPS), mesmo que compartilhem muitas características operacionais e abordagens de detecção de ameaças. O IDS é planejado, essencialmente, para detectar e informar o resultado da detecção de ameaças, não realizando ações de prevenção ou mitigação para as ameaças detectadas, ao contrário dos IPS, que ativamente realizam essas ações [Ashoor and Gore 2011]. A consequência disso, essencialmente, é que um IPS precisa operar *inline* e *online* em relação ao tráfego de rede

[2]`https://www.snort.org`
[3]`https://suricata.io`
[4]`https://zeek.org`

e demais tipos de evento; já o IDS pode executar a partir de tráfego de rede espelhado e/ou registros de eventos já concluídos.

2.3. Trabalhos Relacionados

Esforços direcionados ao desenvolvimento de soluções de IDS de alto desempenho, principalmente relacionados à capacidade de processamento de grandes volumes de dados, têm sido realizados nos últimos anos. A maioria desses esforços se dedica a utilizar um conjunto de tecnologias específicas, adequadas para lidar com o volume de tráfego das redes e sistemas modernos.

Em [Machnicki et al. 2024], o IDS Sapo-Boi é apresentado, baseado em assinaturas, implementado utilizando XDP/eBPF e capturando pacotes ao nível da interface de rede, antes mesmo que o sistema operacional os reconheça e os entregue à aplicação. A análise do tráfego é feita em dois níveis: no *kernel* e no espaço de usuário, sendo os pacotes trocados entre esses espaços por meio de *sockets* XDP. Uma solução IDS com os mesmos padrões de operação, também implementada utilizando XDP/eBPF, é apresentada em [Wang and Chang 2022]; porém, esta utiliza *perf events* para trocar pacotes entre o espaço de *kernel* e o de usuário.

Outra alternativa para a implementação de *Network* IDS de alto desempenho consiste na execução desses sistemas diretamente em *switches* P4. Essa abordagem é explorada em [Lewis et al. 2019], onde um conjunto de regras, que normalmente seriam analisadas por um IDS tradicional, é transferido para ser processado por um *switch* P4, ainda antes de os pacotes chegarem ao ativo de segurança. Esse ativo, por sua vez, passa a executar apenas um subconjunto reduzido de regras mais complexas, não suportadas pelo programa implementado no *switch* P4.

As soluções baseadas em XDP/eBPF propõem a implementação de IDS completos. No entanto, para poderem ser executadas, exigem não apenas suporte à tecnologia pelo sistema operacional e pelo *driver* de rede, mas também, em alguns casos, modificações nesses componentes para viabilizar a cópia dos pacotes, sendo o original encaminhado às aplicações e a cópia analisada pelo IDS. Por sua vez, os NIDS baseados em P4 requerem equipamentos específicos com suporte à linguagem P4, o que limita sua aplicabilidade. Este trabalho, em particular, não pretende desenvolver uma nova solução de IDS, mas sim aprimorar soluções tradicionais, sem exigir modificações arquiteturais, operacionais ou de implantação. Embora a solução apresentada em [Lewis et al. 2019] compartilhe um objetivo semelhante, ao reduzir a quantidade de regras processadas por um IDS tradicional, ela difere da proposta deste artigo, que busca reduzir a carga de dados (tráfego de rede) processada pelo mesmo.

Finalmente, destaca-se que técnicas de engenharia de tráfego também têm sido usadas para dar suporte a soluções de cibersegurança. Em [Fulber-Garcia et al. 2018], um serviço para mitigação de ataques DDoS, chamado DeMONS, é apresentado. Esse serviço utiliza um módulo de engenharia de tráfego executando *traffic policing* para priorizar a passagem de pacotes de fluxos com maiores reputações no sistema. Já em [da Silveira et al. 2025], o uso da engenharia de tráfego no contexto do DeMONS é expandido com a adição de um novo módulo de *traffic shaping* ao serviço, destinado a reduzir a quantidade de tráfego potencialmente benigno descartado. No entanto, a abordagem de *packet washing* no contexto de implementação ou aprimoramento de soluções

de cibersegurança, no melhor de nosso conhecimento, é inexplorada.

3. *MicroSec Traffic*

A proposta deste trabalho, denominada *MicroSec Traffic*, consiste em uma abordagem baseada em engenharia de tráfego (*packet wash*) para otimizar o desempenho de sistemas de detecção de intrusão que analisam tráfego de rede. Apesar de poder ser ajustado para funcionar em múltiplos contextos de detecção, o *MicroSec Traffic* é inicialmente **modelado para operar com soluções de IDS baseadas em assinaturas e/ou baseadas em anomalias identificadas por regras**.

A abordagem proposta visa, em resumo, reduzir o volume de dados processados pelas soluções de IDS, extraindo dos pacotes ingressantes toda a carga de dados que não contribui para a potencial geração de alertas de ataque, com base no conjunto de assinaturas e regras em análise. Dessa forma, **há uma redução da quantidade de dados processados sem qualquer perda de informações úteis** para o IDS em execução. É importante destacar que essa abordagem é adequada para uso em conjunto com soluções de IDS (mas não com IPS), uma vez que os IDS operam sobre tráfego espelhado (copiado). Ou seja, não há qualquer prejuízo na entrega da íntegra dos dados ao destino original dos quadros e pacotes.

A redução de dados nos pacotes é benéfico ao desempenho da infraestrutura de monitoramento. Ao diminuir o volume de dados entregues ao IDS, reduz-se também o risco de sobrecarga na fila de pacotes do sistema subjacente, evitando o descarte dos mesmos por limitação de *buffer*. Isso gera, por consequência, um aumento significativo na capacidade de resposta, análise e alerta das soluções de IDS frente a picos de tráfego, cenário o qual é considerado sensível e desafiador para os IDS tradicionais e um dos principais motivadores dos esforços de substituição destes por soluções baseadas em tecnologias eBPF e P4 [Dang et al. 2017, Machnicki et al. 2024].

Um aspecto técnico importante é que não é necessário realizar nenhuma mudança nas características operacionais das soluções de IDS para utilizar a abordagem proposta: **o IDS é agnóstico ao *MicroSec Traffic***. No entanto, (i) as assinaturas e regras instaladas no IDS precisam ser adaptadas para que este seja capaz de processar corretamente o *MicroSec Traffic*. Além disso, (ii) é necessário um elemento de adaptação do tráfego de rede ingressante no IDS, responsável por realizar a redução e reorganização dos pacotes e quadros, criando, efetivamente, o *MicroSec Traffic*. Dessa forma, o conjunto original de assinaturas e regras do IDS determina quais dados podem ser descartados dos pacotes e onde eles se encontram; esse descarte e a nova organização do *payload* resultante definem, então, as adaptações necessárias no conjunto original de regras para o correto processamento do *MicroSec Traffic*.

Os processos de preparação e estabelecimento do *MicroSec Traffic* podem ser compreendidos em cinco estágios, conforme ilustrado na Figura 1. O primeiro estágio representa o modelo tradicional de execução de um IDS com análise de rede; o quinto estágio, por sua vez, mostra a execução desse mesmo IDS no contexto da abordagem proposta.

Os três estágios intermediários são de preparação. Para exemplificar cada um deles, considera-se o seguinte cenário: um IDS opera utilizando a íntegra do tráfego de

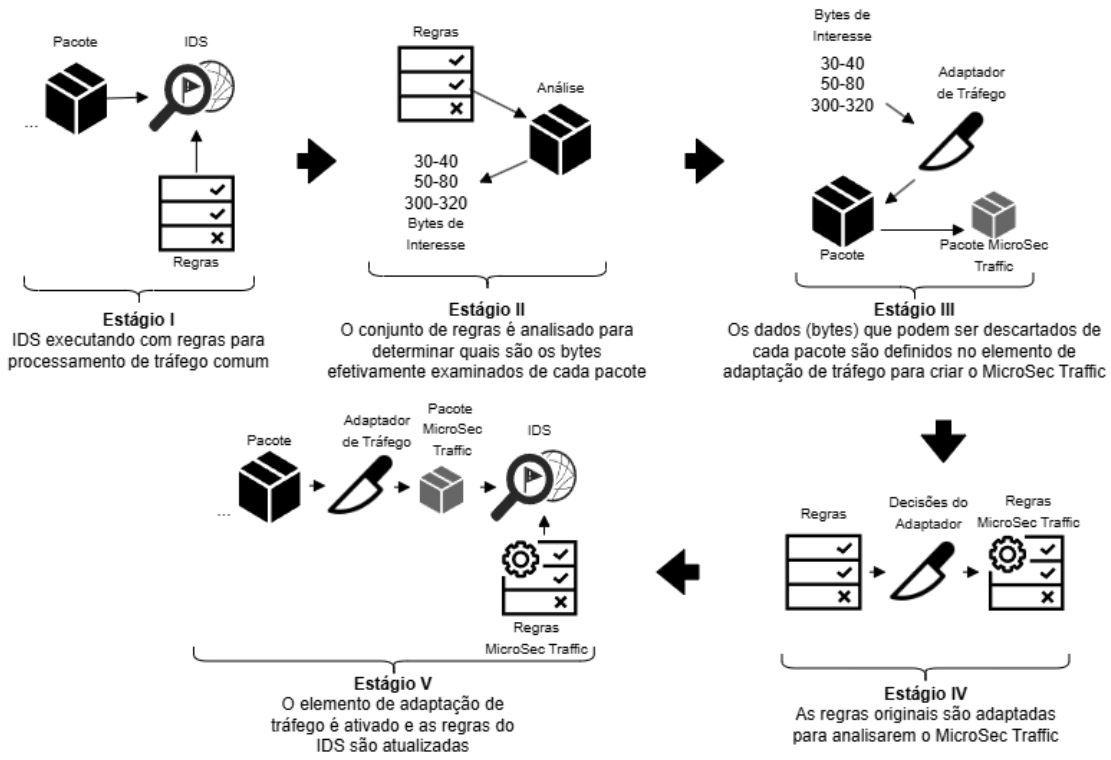

Figura 1. Estágios de Preparação do *MicroSec Traffic*

rede (modelo tradicional). Um alerta é disparado sempre que uma determinada assinatura de 50 bytes é encontrada entre os bytes 500 e 549 do *payload* do pacote (assinatura), ou quando um IP de origem envia mais de 100 pacotes por segundo (anomalia baseada em regra).

Assim, no segundo estágio, o processo consiste em analisar cada uma das regras utilizadas determinando quais são os dados úteis dos pacotes analisados pelo IDS. No exemplo, 50 bytes do *payload*, assim como o cabeçalho IP, são necessários para atender ambas as regras. Essas informações são passadas ao terceiro estágio, onde o elemento de adaptação de tráfego é configurado. No exemplo, um pacote qualquer, formado tipicamente por cabeçalhos de enlace, rede e transporte, seguido pelo *payload* de aplicação, manteria apenas os cabeçalhos de enlace e rede, além de 50 bytes no *payload* de aplicação, justamente aqueles que são analisados. Dessa forma, um pacote que originalmente apresenta 1500 bytes de dados passa a ter, após a adaptação e considerando um cabeçalho IP padrão, 84 bytes no *MicroSec Traffic*. Já pacotes com menos de 550 bytes de *payload* seriam encaminhados como pacotes vazios, ou seja, passariam a ter 34 bytes. É importante notar que os pacotes adaptados podem ser inválidos. No entanto, soluções de IDS operam em modo promíscuo, recebendo e analisando inclusive pacotes inválidos.

Ressalta-se que o elemento de adaptação de tráfego é o responsável por, operacionalmente, criar o *MicroSec Traffic* a partir de um tráfego tradicional. Esse elemento pode ser implementado e instalado de diversas maneiras, podendo estar acoplado ou desacoplado ao sistema subjacente do IDS. No caso de ser acoplado, ele pode operar ao nível de *kernel* (*e.g.*, eBPF/XDP), interceptando o tráfego de rede e realizando as modificações necessárias, ou ainda ao nível de usuário, recebendo e adaptando o tráfego antes de en-

caminhá-lo ao IDS por meio de uma interface virtual. Quando desacoplado do IDS, esse elemento pode ser implementado diretamente no *switch* que espelha o tráfego (*e.g.*, P4), como uma função virtualizada de rede compondo uma cadeia de serviço NFV, ou ainda como um ativo de rede específico (*e.g.*, *middlebox*) posicionado antes do IDS.

Então, no estágio IV, as regras instaladas no IDS são revisitadas e adaptadas para considerarem intervalos de comparação não mais do tráfego de rede tradicional, mas sim do *MicroSec Traffic* gerado pelo elemento de adaptação de rede. A instalação dessas novas regras no IDS e a ativação do elemento de adaptação de tráfego (estágio V) concluem, finalmente, a transição do modelo tradicional de execução de um IDS para o modelo que considera a abordagem proposta neste trabalho.

4. Avaliação Experimental

Esta seção descreve um estudo de caso para avaliar os ganhos de desempenho relacionados à utilização do *MicroSec Traffic* em um cenário específico. Inicialmente, na Subseção 4.1, o cenário de teste é descrito. Em seguida, na Subseção 4.2, são apresentados os materiais, métodos e tecnologias utilizados na implementação da abordagem proposta. Por fim, os testes e seus resultados são apresentados e discutidos na Subseção 4.3.

4.1. Cenário de Teste

Para validar a proposta deste trabalho, foi considerado um estudo de caso baseado na detecção e alerta de ataques de negação de serviço que exploram o protocolo HTTP [Liang et al. 2016], ou seja, ataques que se aproveitam de características da comunicação HTTP para sobrecarregar servidores, de forma que o tráfego malicioso não seja imediatamente identificado pelos nós da rede, afetando diretamente o alvo. Entre os ataques considerados estão o *Slowloris* [Shorey et al. 2018a], *Slow* HTTP [Hirakawa et al. 2016], *GoldenEye* [Shorey et al. 2018b] e HULK [Xavier et al. 2023]. Esses ataques realizam requisições em grande volume ou mantêm conexões abertas por longos períodos, utilizando técnicas que impedem o encerramento automático por parte do servidor, forçando o uso prolongado de seus recursos.

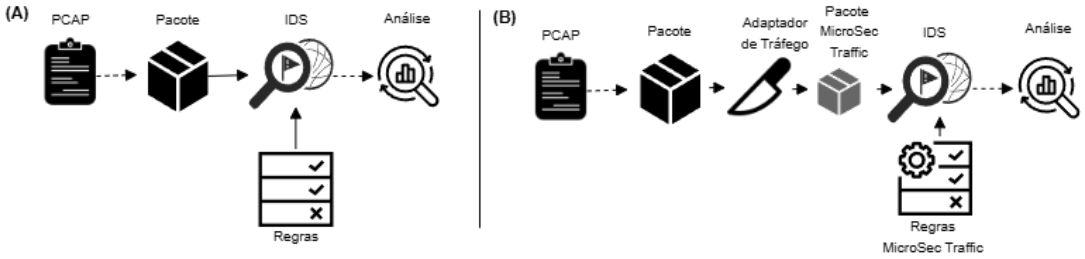

Figura 2. Cenário de Teste do *MicroSec Traffic*

O cenário de testes é apresentado na Figura 2. Em um primeiro momento (Figura 2-A), considera-se a implantação de um IDS tradicional, que processa a íntegra do tráfego de rede em busca de assinaturas e padrões relacionados aos ataques previamente citados. Em um segundo momento (Figura 2-B), as regras de detecção desses ataques são utilizadas para a criação do *MicroSec Traffic*, realizada pelo elemento de adaptação de tráfego implementado no contexto deste trabalho, reduzindo a quantidade de dados, mas preservando a carga de informações necessária ao processamento pelo IDS. Neste

segundo caso, as regras do IDS são adaptadas para operar com o *MicroSec Traffic*, em vez da íntegra do tráfego de rede.

Para os testes, o tráfego foi injetado no sistema por meio da reprodução de um arquivo PCAP de forma virtual. Isso significa que os *buffers* de tráfego podem se expandir dinamicamente, conforme o tráfego ingressante é recebido. Ou seja, não há perda de pacotes por enfileiramento, e o IDS pode operar em modo de melhor esforço. Durante os testes, foram analisadas métricas relacionadas ao tempo de análise dos pacotes, à quantidade de alertas gerados e à quantidade de dados processados.

4.2. Materiais, Métodos e Tecnologias

Para viabilizar a execução dos testes relacionados ao *MicroSec Traffic*, os recursos indicados na Figura 2 foram definidos conforme descrito nesta subseção[5].

PCAP (Tráfego de Rede Original). O conjunto de dados *Intrusion Detection Evaluation Dataset* (CIC-IDS2017) [Sharafaldin et al. 2018] foi escolhido para a execução dos testes. A escolha se justifica pelo tipo de dado e pela metodologia de construção do *dataset*, que busca representar fielmente o comportamento benigno de usuários em uma rede realista. Para isso, foi utilizado o sistema B-Profile [Sharafaldin et al. 2017], responsável por abstrair e reproduzir o comportamento de mais de 25 usuários interagindo com diferentes serviços e protocolos, como HTTP, HTTPS, FTP, entre outros.

O CIC-IDS2017 é organizado em cinco segmentos, cada um correspondente a um dia da semana. Em cada dia, são realizados ataques específicos, com horários previamente definidos. Essa estrutura permite selecionar com precisão os tipos de ataques a serem avaliados, alinhando-se aos objetivos do experimento. Dentre os dias disponíveis, foi escolhido o tráfego registrado na quarta-feira, 5 de julho de 2017, contendo 13.788.878 pacotes, por incluir ataques de negação de serviço, adequados aos objetivos de validação e avaliação do *MicroSec Traffic*. Além disso, a granularidade temporal e a rotulação clara dos ataques nesse dia específico favorecem análises detalhadas sobre detecção, desempenho e impacto das transformações aplicadas aos pacotes. Nas próximas seções, o conjunto de dados CIC-IDS2017 será denominado `PCAP original`.

Adaptador de Tráfego. Dois adaptadores de tráfego foram implementados no contexto deste trabalho. O primeiro realiza o ajuste do tráfego em espaço de usuário, utilizando a ferramenta Scapy[6] para a manipulação dos pacotes. Nessa estratégia, os cortes são feitos com atenção à integridade dos pacotes, especialmente à preservação de campos como o *checksum* do cabeçalho IP. Essa alternativa é ideal para a adaptação do tráfego *a posteriori*, ou seja, quando esta não é realizada de forma *online*.

A segunda abordagem atua em espaço de *kernel*, por meio da tecnologia eBPF (*Extended Berkeley Packet Filter*) XDP (*eXpress Data Path*) [Høiland-Jørgensen et al. 2018]. O XDP é um *framework* do eBPF que permite a execução de programas em conjunto com o *driver* da placa de rede, antes mesmo que os pacotes ingressem na pilha de rede convencional do Linux. Essa técnica elimina a necessidade de modificações no código-fonte do *kernel* ou do carregamento de módulos,

[5]Todos os recursos utilizados para a execução dos testes (artefatos) estão disponíveis em `https://github.com/MateusHerbele/SBSeg-2025-Herbele`

[6]`https://scapy.net`

operando em um estágio extremamente inicial do processamento de pacotes. No caso deste adaptador, o foco é o desempenho no processamento dos pacotes: ao processá-los no XDP, busca-se o ajuste mais rápido possível, reduzindo a latência associada à cópia de dados entre o *kernel* e o espaço de usuário. A alternativa com eBPF/XDP resulta em pacotes inválidos, visto que não há recálculo do *checksum*, porém permite a adaptação do tráfego de forma *online*, sem causar prejuízos ou perda de pacotes ingressantes no sistema.

Independente da implementação do adaptador de tráfego, dada a natureza desses ataques, é essencial manter a parte do pacote que determina o método HTTP requisitado, já que ele, com informações como a marcação temporal e a densidade do fluxo, é necessário para caracterizar comportamentos maliciosos. Além disso, os cabeçalhos IP contêm metadados relevantes para a geração de alertas, como os endereços de origem e destino, que permitem contabilizar o número de pacotes enviados por uma mesma fonte e classificá-la como maliciosa. Portanto, o corte dos pacotes é feito para remover o segmento TCP, preservando apenas o quadro Ethernet, o cabeçalho IP e a carga útil mínima necessária que contenha o método HTTP. Isso resulta em uma redução significativa no tamanho dos pacotes analisados, mantendo as informações essenciais para a detecção do ataque: **de uma média de 941 bytes por pacote no PCAP original para 34 bytes por pacote após a adaptação do tráfego**.

As condições gerais estabelecidas para a adaptação do tráfego foram as seguintes: pacotes que não contêm um quadro Ethernet ou que não pertencem ao protocolo IP são mantidos inalterados; pacotes que não são segmentos TCP, não transportam HTTP ou não correspondem a requisições HTTP têm todo o *payload* do protocolo IP removido; nos casos restantes, em que há uma mensagem HTTP caracterizada como requisição, os dados da requisição são os únicos mantidos no *payload* do pacote IP, reduzindo o tamanho do mesmo, mas preservando a informação referente ao método HTTP. Assim, foi possível gerar o tráfego adaptado, denominado no conjunto de artefatos como `PCAP processado`, com uma **redução de 13,42GB (PCAP original) para 692MB (PCAP processado), cerca de 94,9%**.

IDS. O IDS selecionado para a execução dos experimentos foi o **Snort 3.7.40**[7], executado em um contêiner Docker sobre um sistema hospedeiro Linux. O Snort é uma solução que opera majoritariamente por meio de inspeções baseadas em assinaturas ou em anomalias definidas por regras [Waleed et al. 2022]. Esse IDS foi lançado em 1998, sendo amplamente adotado e contando com uma comunidade ativa que promove atualizações frequentes para revisão, ajuste e aprimoramento da solução.

Regras Originais e Regras *MicroSec Traffic*. As regras utilizadas no Snort 3 seguem uma estrutura dividida em cabeçalho e corpo, como exemplificado na Figura 3. O cabeçalho corresponde à parte anterior à abertura dos parênteses e define os elementos básicos de detecção. Inicialmente, especifica-se a ação a ser tomada quando a condição da regra for satisfeita, como, por exemplo, *alert* (gerar um alerta). Em seguida, determina-se o protocolo de transporte sobre o qual a verificação será aplicada; no exemplo apresentado, o protocolo TCP. Posteriormente, configuram-se o endereço IP e a porta de origem e de destino dos pacotes que serão analisados. Esses campos permitem restringir a atuação

[7] `https://www.snort.org`

da regra a fluxos específicos de comunicação. Também é possível utilizar o valor especial *any* para indicar que qualquer endereço ou porta são aceitos naquele campo, flexibilizando o escopo da verificação.

O corpo da regra, localizado dentro dos parênteses, é composto por uma série de opções que detalham as condições específicas de inspeção do pacote e as ações associadas. No exemplo da Figura 3, o corpo inclui o campo *msg*, que define a mensagem a ser registrada quando a regra for acionada; o campo *content*, que especifica o padrão de bytes que deve ser encontrado no conteúdo do pacote; a opção *nocase*, que torna a busca pelo padrão insensível a diferenças entre letras maiúsculas e minúsculas; e o parâmetro *offset*, que indica o deslocamento inicial, em bytes, a partir do qual o Snort deve começar a busca pelo padrão. O único campo obrigatório no corpo da regra é o *sid* (Snort ID), que atribui um identificador único à regra, essencial para sua gestão e controle no sistema. Além dos elementos exemplificados, o Snort 3 oferece uma ampla variedade de outras opções que podem ser utilizadas no corpo da regra para refinar a análise, como verificações de tamanho de pacote, análise de protocolos de aplicação, entre outras.

```
alert tcp 10.24.54.123 any -> any 443
(msg:"Regra exemplo.  Alerta!";content:2b55e75e|",nocase,
offset 4;sid:3000493;)
```

Figura 3. Exemplo de Regra do Snort 3

Para executar os experimentos no cenário de teste proposto, **foram desenvolvidos dois conjuntos de regras, cada um composto por 4 regras**. O primeiro conjunto, voltado ao tráfego original, utiliza o protocolo TCP como base para a geração de alertas, enquanto o segundo conjunto, destinado ao *MicroSec Traffic*, emprega o protocolo IP como parâmetro inicial e considera os pacotes gerados pelo adaptador de tráfego. Para garantir que a avaliação não fosse influenciada por um conjunto reduzido de regras, foram adicionadas 500 regras de simulação (*dummy*, que geram um número insignificante de alertas; cerca de 0,1% do total em todos os casos), seguindo a mesma estrutura de variação entre TCP e IP. Essas regras adicionais buscam no conteúdo dos pacotes uma sequência hexadecimal gerada aleatoriamente, aumentando a carga de processamento do IDS sem interferir nos resultados finais, uma vez que eventuais alertas gerados por elas foram excluídos da análise.

A primeira regra do conjunto, para ataques *Slowloris*, estabelece uma ação de alerta para qualquer origem e destino. A opção `pcre` é utilizada para definir, no corpo da regra, uma expressão regular destinada à identificação dos métodos HTTP `GET`, `POST` ou `HEAD` nos pacotes inspecionados. A configuração `detection_filter` impõe critérios adicionais, exigindo uma sequência consistente de eventos para a geração do alerta. A análise é realizada com base na origem dos pacotes, especificada por meio da opção `by_src`. O parâmetro `count` determina que 20 pacotes sejam contabilizados dentro de um intervalo de 10 segundos, conforme definido pelo parâmetro `seconds`.

A segunda regra, para ataques *Slow HTTP*, estabelece um alerta para o envio de múltiplas requisições do tipo `POST`. A correspondência com o conteúdo é feita por meio da opção `content`, com a sensibilidade a maiúsculas e minúsculas desabilitada (`nocase`). A política de detecção, configurada com `detection_filter`, exige a observação de três pacotes provenientes de uma mesma origem dentro de um intervalo de

120 segundos para que o alerta seja gerado.

A terceira regra, para ataques *GoldenEye*, configura um alerta para o envio elevado de requisições com os métodos `GET`, `POST` ou `HEAD`. A opção `pcre` é empregada para localizar esses métodos no início ou no fim da linha, de forma insensível a maiúsculas e minúsculas. O `detection_filter` determina que o alerta seja acionado após a observação de 50 eventos provenientes da mesma origem em um intervalo de 10 segundos.

A quarta regra, para ataques *HULK*, define um alerta para atividades envolvendo o envio intensivo de requisições `GET`. O operador `content` é utilizado para identificar a palavra-chave "GET" a partir da posição inicial do pacote, especificada com `offset 0`. O alerta é configurado para ser disparado quando forem observados 100 pacotes provenientes de uma mesma origem em um intervalo de 5 segundos.

Análise. As análises, tanto em relação aos alertas gerados quanto ao tempo de processamento da íntegra do tráfego, foram realizadas com base nos arquivos de *log* de execução gerados pelo IDS Snort.

4.3. Testes e Resultados

O primeiro teste realizado tem como objetivo determinar o tempo total necessário para o processamento do tráfego original e do *MicroSec Traffic*, com o Snort operando em modo de melhor esforço, sem descarte de pacotes. Os testes de temporização foram executados 10 vezes, sendo os resultados individuais para ambos os casos apresentados na Figura 4. Com o intuito de eliminar variações decorrentes de tempos extras de *warm-up*, como o carregamento de regras e a inicialização da reprodução do tráfego, foram desconsiderados os dois maiores e os dois menores tempos medidos. A média e o desvio padrão foram então calculados com base nas seis execuções restantes.

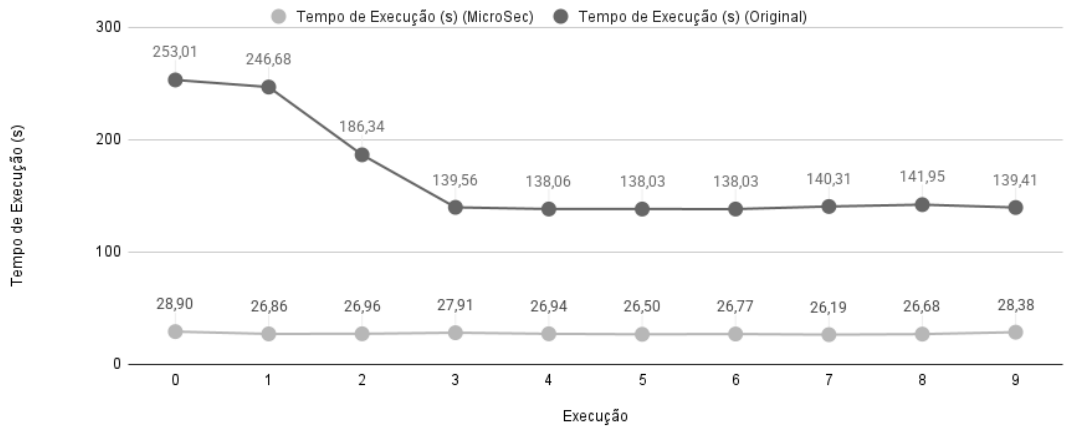

Figura 4. Tempo de Processamento do Tráfego Original e *MicroSec Traffic*

Para processar a íntegra do tráfego original, o IDS demandou, em média, 147,6s (±19,0s), enquanto o processamento da versão *MicroSec* do mesmo tráfego requereu, em média, 27,0s (±0,4s). A redução de aproximadamente 81% no tempo médio de processamento é consequência direta da menor carga de dados analisada, além da previsibilidade estrutural dos pacotes, que seguem um formato padronizado estabelecido pelo adaptador de tráfego.

Para a análise da geração de alertas, o tráfego foi dividido em 14 *chunks* com o mesmo número de pacotes, a fim de facilitar a visualização da distribuição dos alertas ao longo do tempo, uma vez que os ataques não estão uniformemente distribuídos no *dataset*. A Figura 5 apresenta o tamanho, em GB, de cada *chunk* gerada. Observa-se, mais uma vez, a estabilização e a previsibilidade do *MicroSec Traffic*, resultado da atuação do adaptador de tráfego, que garante uma correlação direta entre o número de pacotes e o volume de dados processados. Destaca-se, ainda, que a maior concentração de ataques ocorre entre o *chunk* 6 e o *chunk* 9.

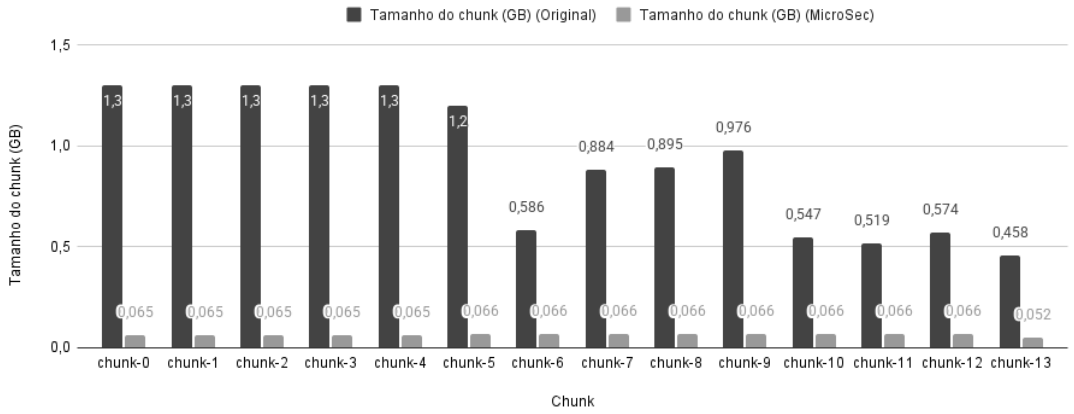

Figura 5. Tamanho dos *Chunks* do Tráfego Original e do *MicroSec Traffic*

As Figuras 6 e 7 apresentam, respectivamente, o tempo de processamento por *chunk* e os alertas gerados pelo Snort para o tráfego original e para o *MicroSec Traffic*. Em ambos os casos, todas as origens maliciosas foram corretamente detectadas. No entanto, observa-se a presença de alertas adicionais quando o *MicroSec Traffic* é utilizado. Isso ocorre porque, nesse cenário, a identificação da origem e do destino baseia-se unicamente na relação IP–IP, uma vez que os cabeçalhos da camada de transporte são descartados. Já no tráfego original, a distinção é feita com base na relação IP:Porta–IP:Porta, o que permite maior granularidade na identificação dos fluxos de comunicação.

A diferença no número de alertas pode ser mitigada de duas maneiras: (i) ajustando as regras que processam o tráfego original para considerarem apenas o par de endereços IP (protocolo de rede) como origem e destino; ou (ii) mantendo o cabeçalho da camada de transporte no *MicroSec Traffic*. No primeiro caso, evita-se que atacantes explorem mecanismos de NAT baseados em portas para ofuscar suas ações, embora isso possa aumentar a ocorrência de falsos positivos. No segundo caso, a probabilidade de falsos positivos é reduzida, mas cada pacote na *chunk* apresenta um acréscimo de 6 a 60 bytes no *MicroSec Traffic*. No pior cenário, isso elevaria a média do tamanho dos pacotes de 34 para 84 bytes, ainda assim permanecendo significativamente abaixo da média do tráfego original (941 bytes).

Para analisar especificamente o custo do adaptador de tráfego, foi avaliada uma implementação em eBPF/XDP responsável por ajustar pacotes HTTP de requisição com métodos variados (HEAD, GET e POST), contendo *payload* arbitrário de 1500 bytes, considerando a transmissão de um milhão de pacotes. A adaptação do tráfego segue o mesmo modelo utilizado nos experimentos anteriores: manutenção do cabeçalho Ether-

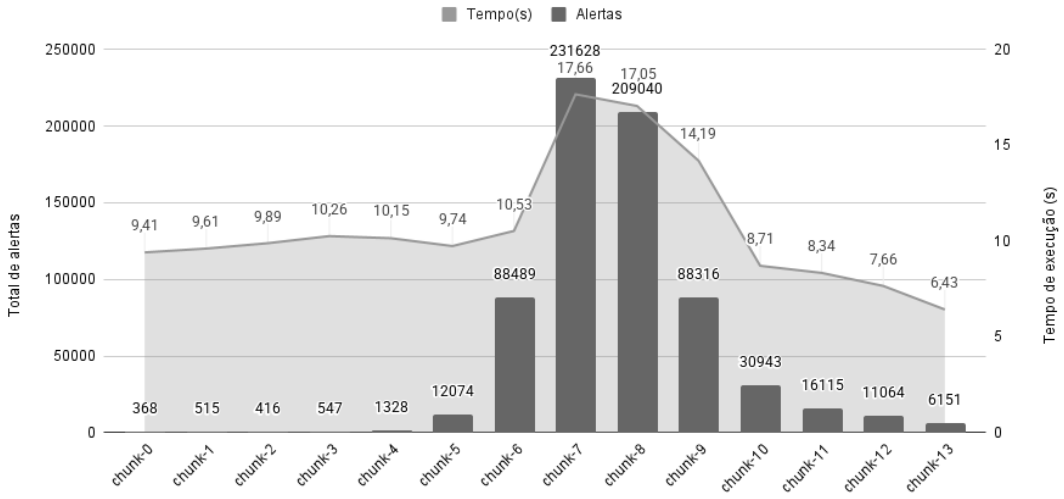

Figura 6. Tempo de Execução e Número de Alertas Snort (Tráfego Original)

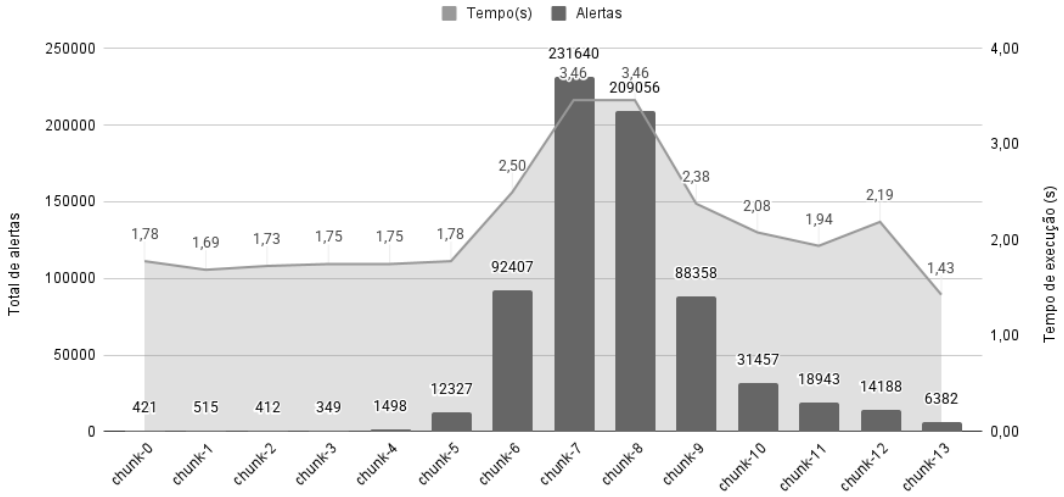

Figura 7. Tempo de Execução e Número de Alertas Snort (*MicroSec Traffic*)

net, cabeçalho IP e do cabeçalho HTTP. Os resultados de 10 execuções realizadas determinou que o tempo médio de processamento por pacote é de 0,7ms (ou 700 microssegundos), totalizando, em média, 0,7s para o processamento de aproximadamente 1,4GB de tráfego (um milhão de pacotes). Esse desempenho é adequado para atender, de forma *online*, ao tráfego típico de *datacenters* de pequeno e médio porte, como *datacenters* universitários.

5. Limitações, Desafios e Oportunidades

A abordagem do *MicroSec Traffic* mostrou-se promissora como alternativa para o aprimoramento do desempenho de soluções IDS tradicionais, implementadas em *software*, como o Snort. No entanto, alguns pontos de atenção, que representam desafios e oportunidades de pesquisa, merecem destaque. Esses aspectos são apresentados e discutidos a seguir.

Exclusividade de Uso em Soluções de IDS. O *MicroSec Traffic* não é aplicável

em contextos de soluções IPS, uma vez que a remoção de dados considerados irrelevantes para a segurança pode comprometer informações essenciais à aplicação final. Em outras palavras, essa abordagem é especialmente adequada para soluções IDS, que operam com cópias do tráfego de rede, geralmente obtidas por espelhamento de portas, e são destinadas exclusivamente à inspeção. Dessa forma, embora o *MicroSec Traffic* melhore o desempenho de soluções tradicionais como Snort e Suricata em modo IDS, o uso dessas ferramentas em modo IPS ainda representa um cenário desafiador, que demanda novas abordagens e propostas de pesquisa.

Limitação das Regras Utilizadas. Há um subconjunto de regras potencialmente utilizadas por soluções IDS baseadas em assinaturas e anomalias que não são compatíveis com o *MicroSec Traffic*. Esse subconjunto inclui regras cujos padrões podem ocorrer em qualquer parte do pacote ou quadro. Quando tais regras não podem ser descartadas ou substituídas, o uso do *MicroSec Traffic* torna-se inadequado como abordagem de otimização de desempenho.

Complexidade de Conciliação de Regras. A conciliação e transformação de regras de IDS para gerar o *MicroSec Traffic* é uma tarefa complexa, e a diversidade de regras pode reduzir a efetividade do *packet wash* como mecanismo de redução de dados. O desenvolvimento de ferramentas automatizadas para essa tarefa, que avaliem os potenciais ganhos decorrentes da adoção do *MicroSec Traffic*, representa um passo importante para consolidar essa abordagem como uma alternativa viável para a otimização do desempenho de soluções IDS. A criação dessas ferramentas pode se beneficiar de técnicas empregadas em compiladores, bem como do uso de recursos de aprendizado de máquina.

6. Conclusão

Este trabalho apresentou o *MicroSec Traffic*, uma abordagem voltada ao aprimoramento da eficiência de soluções IDS baseadas na identificação de assinaturas e anomalias definidas por regras. Os experimentos demonstraram que a técnica é capaz de reduzir significativamente o tempo de processamento do tráfego e a geração de alertas, por meio da diminuição da quantidade de dados analisados, sem comprometer as informações essenciais ao funcionamento do IDS. Como trabalhos futuros, propõe-se o desenvolvimento de estratégias automatizadas para a conciliação de regras e a adaptação do tráfego com vistas à geração do *MicroSec Traffic*, além da investigação de técnicas de previsão de ganho de eficiência por meio de simulações da redução da carga de dados processada pelo IDS.

Agradecimentos

Este trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES PROEX) e Centro de Computação Científica e Software Livre (C3SL - TED SAPS/MS). Os autores também agradecem o Programa de Pós-Graduação em Informática da Universidade Federal do Paraná.

Referências

Asad, H., Adhikari, S., and Gashi, I. (2024). A perspective–retrospective analysis of diversity in signature-based open-source network intrusion detection systems. *International Journal of Information Security*, 23(2):1331–1346.

Ashoor, A. S. and Gore, S. (2011). Difference between intrusion detection system (ids) and intrusion prevention system (ips). In *International Conference Advances in Network Security and Applications*, pages 497–501.

da Silveira, V. F. M., Muhlmann, J. M., and Fulber-Garcia, V. (2025). Demons++: Utilizando técnicas de modelagem de tráfego no combate de ddos via serviço demons. In *Computer On The Beach*.

Dang, H. T., Wang, H., et al. (2017). Whippersnapper: A p4 language benchmark suite. In *Symposium on SDN Research*, pages 95–101.

Dong, L. and Clemm, A. (2021). High-precision end-to-end latency guarantees using packet wash. In *IFIP/IEEE International Symposium on Integrated Network Management*, pages 260–267.

Flach, T., Papageorge, P., Terzis, A., Pedrosa, L., Cheng, Y., Karim, T., Katz-Bassett, E., and Govindan, R. (2016). An internet-wide analysis of traffic policing. In *ACM SIGCOMM Conference*, pages 468–482.

Frasão, A., Heinrich, T., et al. (2024). I see syscalls by the seashore: An anomaly-based ids for containers leveraging sysdig data. *IEEE Symposium on Computers and Communications*, pages 1–6.

Fulber-Garcia, V., de Freitas Gaiardo, G., da Cruz Marcuzzo, L., Nunes, R. C., and dos Santos, C. R. P. (2018). Demons: A ddos mitigation nfv solution. In *IEEE International Conference on Advanced Information Networking and Applications*, pages 769–776.

Garcia-Teodoro, P., Diaz-Verdejo, J., Maciá-Fernández, G., and Vázquez, E. (2009). Anomaly-based network intrusion detection: Techniques, systems and challenges. *Computers & Security*, 28(1):18–28.

Hirakawa, T., Ogura, K., Bista, B. B., and Takata, T. (2016). A defense method against distributed slow http dos attack. In *International Conference on Network-Based Information Systems*, pages 152–158.

Høiland-Jørgensen, T., Brouer, J. D., et al. (2018). The express data path: Fast programmable packet processing in the operating system kernel. In *International Conference on Emerging Networking Experiments and Technologies*, pages 54–66.

Kepios, Social, W. A., and Meltwater (2025). Digital 2025: Global overview report.

Lewis, B., Broadbent, M., and Race, N. (2019). P4id: P4 enhanced intrusion detection. In *IEEE Conference on Network Function Virtualization and Software Defined Networks*, pages 1–4.

Liang, L., Zheng, K., Sheng, Q., and Huang, X. (2016). A denial of service attack method for an iot system. In *International Conference on Information Technology in Medicine and Education*, pages 360–364.

Liao, H.-J., Lin, C.-H. R., Lin, Y.-C., and Tung, K.-Y. (2013). Intrusion detection systems: A comprehensive review. *Journal of Network and Computer Applications*, 36(1):16–24.

Machnicki, R. K., Correia, J., Penteado, U., Fulber-Garcia, V., and Grégio, A. (2024). Sapo-boi: Pulando a pilha de rede no desenvolvimento de um nids baseado em bpf/xdp.

In *Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais*, pages 538–553.

Marcon, M., Dischinger, M., Gummadi, K. P., and Vahdat, A. (2011). The local and global effects of traffic shaping in the internet. In *International Conference on Communication Systems and Networks*, pages 1–10.

Pedreno-Manresa, J.-J., Khodashenas, P. S., Siddiqui, M. S., and Pavon-Marino, P. (2017). Dynamic qos/qoe assurance in realistic nfv-enabled 5g access networks. In *International Conference on Transparent Optical Networks*, pages 1–4.

Shafi, M., Lashkari, A. H., and Roudsari, A. H. (2025). Toward generating a large scale intrusion detection dataset and intruders behavioral profiling using network and transportation layers traffic flow analyzer (ntlflowlyzer). *Journal of Network and Systems Management*, 33(2):44.

Sharafaldin, I., Lashkari, A. H., and Ghorbani, A. A. (2017). Intrusion detection evaluation dataset (cic-ids2017).

Sharafaldin, I., Lashkari, A. H., and Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *International Conference on Information Systems Security and Privacy*.

Shorey, T., Subbaiah, D., Goyal, A., Sakxena, A., and Mishra, A. K. (2018a). Performance comparison and analysis of slowloris, goldeneye and xerxes ddos attack tools. In *International Conference on Advances in Computing, Communications and Informatics*, pages 318–322.

Shorey, T., Subbaiah, D., Goyal, A., Sakxena, A., and Mishra, A. K. (2018b). Performance comparison and analysis of slowloris, goldeneye and xerxes ddos attack tools. In *International Conference on Advances in Computing, Communications and Informatics*, pages 318–322.

Soniya, S. S. and Vigila, S. M. C. (2016). Intrusion detection system: Classification and techniques. In *International Conference on Circuit, Power and Computing Technologies*, pages 1–7.

Waleed, A., Jamali, A. F., and Masood, A. (2022). Which open-source ids? snort, suricata or zeek. *Computer Networks*, 213:109116.

Wang, N., Ho, K. H., Pavlou, G., and Howarth, M. (2008). An overview of routing optimization for internet traffic engineering. *IEEE Communications Surveys & Tutorials*, pages 36–56.

Wang, S.-Y. and Chang, J.-C. (2022). Design and implementation of an intrusion detection system by using extended bpf in the linux kernel. *Journal of Network and Computer Applications*, 198:103283.

Xavier, B. M., Dzaferagic, M., Collins, D., Comarela, G., Martinello, M., and Ruffini, M. (2023). Machine learning-based early attack detection using open ran intelligent controller. In *IEEE International Conference on Communications*, pages 1856–1861.

Zhao, Z., Sadok, H., et al. (2020). Achieving 100gbps intrusion prevention on a single server. In *USENIX Symposium on Operating Systems Design and Implementation*, pages 1083–1100.