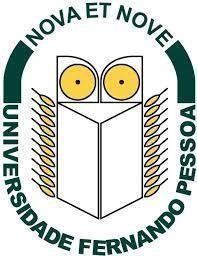
Sistema de Detecção de Intrusões na Rede (NIDS)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

LABORATÓRIO DE PROJETO INTEGRADO



Patrick Silva nº 2022102873 | Vasco Martins nº 2022121836

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

UNIVERSIDADE FERNANDO PESSOA | LICENCIATURA ENGENHARIA INFORMÁTICA

ANO LETIVO 2024/2025

# Índice

[Índice 2](#_heading=h.3gerg4j141rv)

[Resumo 4](#_heading=h.1n7xsspt0jzk)

[Introdução 5](#_heading=h.k6t5q06cpbt5)

[Objetivo 6](#_heading=h.8zhbdb5tidla)

[Âmbito 7](#_heading=h.u9a2ul157kn7)

[Mapeamento de Imagens 8](#_heading=h.cqaehzsv0va)

[Contexto 9](#_heading=h.wd7qh1xzdnp8)

[Detecção por NIDS (Suricata) 9](#_heading=h.ytg3y0t51lhj)

[Exemplos Recentes e Estatísticas de Incidentes 9](#_heading=h.njx3r0xmz0mp)

[Evolução das Técnicas de Ataque e Resposta de Segurança 10](#_heading=h.vt1tiepuxa9n)

[Machine Learning e Técnicas Avançadas de Deteção 10](#_heading=h.kl3bame56kr2)

[Planeamento 11](#_heading=h.3vf7fatdvwtz)

[Topologia 13](#_heading=h.irh84bz2vr3l)

[Ferramentas 14](#_heading=h.ipj79co1b6to)

[GNS3 14](#_heading=h.5nzxoaklkftf)

[OPNsense 16](#_heading=h.43jyazhc0ors)

[Regras de Firewall e NAT 16](#_heading=h.9a4w2xp0i34n)

[Suricata 19](#_heading=h.rj9ejhkiiui6)

[Regras Personalizadas no Suricata 20](#_heading=h.g5myowwfktuf)

[Sintaxe de uma Regra Suricata 21](#_heading=h.20u0rxgcyc9x)

[Análise de Logs de Tráfego 22](#_heading=h.81l8ndj978k5)

[NetFlow 22](#_heading=h.xwcpn52scpdi)

[IPFIX 22](#_heading=h.uv1mn1mko0ou)

[Ativação do NetFlow no OPNsense 22](#_heading=h.7ocslzjwzkj6)

[Utilidade dos Logs para Detecção e Correlação 23](#_heading=h.tq8txjlm7rzx)

[Docker Container 24](#_heading=h.rzqaaihqgjik)

[Ataques 26](#_heading=h.jugipo9up9hz)

[Port Scanning 26](#_heading=h.vv27n5bd3xyu)

[Descrição Técnica 26](#_heading=h.gl4fzkwe71ju)

[Ferramenta Utilizada 27](#_heading=h.pkvqqjtw4prm)

[Tráfego Gerado 27](#_heading=h.8rdtf5av0aaj)

[Regra Suricata 27](#_heading=h.bjhrxvo52fhj)

[Deteção 27](#_heading=h.uu46k39wc0xw)

[SYN Flood 28](#_heading=h.n4vcs1qv2zf)

[Descrição Técnica 28](#_heading=h.dun0sshd98qj)

[Ferramenta Utilizada 28](#_heading=h.323md3sdw0w6)

[Tráfego Gerado 29](#_heading=h.wj2iuiyv1sbm)

[Regras Suricata 29](#_heading=h.hptdrju4cbvm)

[Deteção 29](#_heading=h.yoztxzsyof3j)

[UDP Flood 30](#_heading=h.87ss03e0ydzm)

[Descrição técnica 30](#_heading=h.qa8wslyee0ub)

[Ferramenta Utilizada 30](#_heading=h.jgmrfk9gdd6l)

[Tráfego Gerado 31](#_heading=h.fqawo5o5n1id)

[Regras Suricata 31](#_heading=h.glt0kmbf01ms)

[Avaliação do Sistema e Métricas 32](#_heading=h.bk5leocqxpqe)

[Métricas Recolhidas 32](#_heading=h.4kazb0gki7j6)

[Análise Pós-Ataque e Desafios Identificados 33](#_heading=h.46oi38mdeq0s)

[Melhorias Futuras e Integração de Inteligência Artificial 36](#_heading=h.o2l900us1gs6)

[Integração com Sistemas SIEM 36](#_heading=h.b7eg90i39as1)

[Automatização da Resposta a Incidentes 36](#_heading=h.89lra8qaowel)

[Fortalecimento da Segurança Perimetral (Hardening) 36](#_heading=h.iol3irpplvv0)

[Vantagens da IA em Sistemas de Deteção 37](#_heading=h.ahry4993hawv)

[Exemplos de Algoritmos Relevantes: 37](#_heading=h.3kucg3oun24g)

[Aplicação Prática 38](#_heading=h.qw16qn2mjs3s)

[Papel dos Componentes numa Arquitetura Integrada 38](#_heading=h.24p2gnw9vlno)

[Conclusão 40](#_heading=h.8cfmdfwt9wfs)

[Agradecimentos 41](#_heading=h.7rxij4qshnge)

# Resumo

Este projeto teve como objetivo configurar e implementar um Sistema de Detecção de Intrusões em Rede (NIDS) utilizando OPNsense e Suricata para monitorizar o tráfego e identificar possíveis atividades maliciosas.

Foi criada uma topologia de laboratório em GNS3, com hosts, routers e uma firewall OPNsense, onde o Suricata foi instalado e configurado para inspecionar pacotes em tempo real.

Regras personalizadas foram desenvolvidas para detectar ataques de Port-Scanning, SYN Flood, UDP Flood.

Como resultado, o protótipo demonstrou alta eficácia na detecção dos diferentes tipos de ataque, com baixas taxas de falsos positivos, atendendo às metas do projeto.

# Introdução

Nos últimos anos, o aumento exponencial da interconectividade entre dispositivos e a proliferação de serviços baseados em rede tornaram as infraestruturas digitais cada vez mais vulneráveis a ataques cibernéticos.

Organizações de todos os setores, desde instituições financeiras a ambientes industriais, dependem da disponibilidade, confidencialidade e integridade dos seus dados para operar com eficiência e garantir a confiança dos seus clientes. Sendo por isso fundamental detetar precocemente atividades maliciosas para mitigar riscos e evitar danos significativos.

Os Sistemas de Detecção de Intrusão em Rede (NIDS) surgem como uma camada de defesa proativa, capaz de monitorizar o tráfego em tempo real, identificar padrões suspeitos e gerar alertas que auxiliam as equipas de segurança na resposta rápida a incidentes. Entre as inúmeras soluções disponíveis, destacam-se ferramentas de código aberto, como Snort, Zeek e Suricata, que oferecem flexibilidade para cenários variados.

Este relatório descreve o desenvolvimento de um protótipo de NIDS baseado em OPNsense e Suricata, implementado em ambiente de laboratório utilizando o simulador GNS3. A abordagem adotada foca-se na criação de regras personalizadas para detectar e mitigar diferentes tipos de ataque específicos, como Port-Scanning, SYN Flood, UDP Flood e na análise de logs de Netflow/IPFIX para validação dos resultados.

## Objetivo

O objetivo principal deste projeto é projetar, configurar e implementar um Sistema de Detecção de Intrusões em Rede (NIDS) funcional, capaz de monitorizar continuamente o tráfego de rede e identificar e mitigar atividades maliciosas. Especificamente, pretende-se:

1. Estabelecer uma infraestrutura de testes em GNS3 que reproduza um ambiente de rede corporativa, incluindo hosts, routers e o firewall OPNsense.
2. Configurar o OPNsense para atuar como plataforma de gestão de tráfego e ponto de inspeção do Suricata.
3. Desenvolver e aplicar regras personalizadas no Suricata para detetar e mitigar diferentes tipos de ataque: Port-Scanning, SYN Flood, UDP Flood.
4. Recolher e analisar dados de tráfego usando Netflow/IPFIX, gerando alertas e relatórios que evidenciem a capacidade de detecção e a precisão do sistema.
5. Avaliar o desempenho do protótipo em termos de taxa de detecção, tempo de resposta a alertas e taxa de falsos positivos.

## Âmbito

A realização deste projeto de implementação de um Sistema de Detecção de Intrusões em Rede (NIDS) insere-se no contexto académico da disciplina de Laboratório de Projeto Integrado, do curso de Engenharia Informática da Universidade Fernando Pessoa. Este trabalho proporcionou uma aplicação prática dos conhecimentos teóricos adquiridos ao longo da licenciatura, reforçando competências de análise, design, configuração de redes e desenvolvimento de soluções de segurança.

No âmbito deste projeto, foi desenhada uma topologia de rede em GNS3 que replicasse um ambiente corporativo, permitindo testar diferentes vetores de ataque em condições controladas. O OPNsense foi instalado como firewall e plataforma de gestão de tráfego, enquanto o Suricata foi integrado para a inspeção profunda de pacotes em tempo real. Regras personalizadas foram desenvolvidas para detectar e mitigar Port-Scanning, SYN Flood, UDP Flood.

O trabalho em grupo revelou-se fundamental na partilha de conhecimentos, coordenação de tarefas e revisão mútua das configurações e regras. Adotaram-se práticas de desenvolvimento ágil, com ciclos iterativos de planeamento, implementação e testes, o que permitiu validar progressivamente o funcionamento do NIDS.

Em suma, o âmbito deste projeto académico ultrapassou a mera simulação de cenários de segurança, foi uma experiência enriquecedora para aplicar competências técnicas, fomentar o trabalho em equipa, enfrentar desafios reais e aprofundar o conhecimento prático em Engenharia Informática.

# 

# Mapeamento de Imagens

Figura 1 Planeamento…………………………………………………………………………...11

Figura 2 Diagrama de Gantt……………………………………………………………………..12

Figura 3 LogBook………………………………………………………………………………..13

Figura 4 Topologia do Projeto……………...……………………………………………………15

Figura 5 Regras Nat criadas………………………………………………………………….…18

Figura 6 Regras Firewall criadas…………………………………………………………………20

Figura 7 Ativação do Suricata no OPNsense…………………………………………………….22

Figura 8 Regras Suricata criadas………………………………………………………………...22

Figura 9 Criação do Docker com as ferramentas necessárias…………………………………..26

Figura 10 Construção do Docker……………………………………………………………….26

Figura 11 Download do Docker para o Ubuntu…………………………..……………………..27

Figura 12 Transferência do Docker para o GNS3………………………..………………………27

Figura 13 Ataque Port-Scanning………………………………………………………………...29

Figura 14 Regras Suricata contra Port-Scanning………………………………………………...29

Figura 15 Deteção no OPNsense………………………………………………………………..29

Figura 16 Ataque SYN-Flood…………………………………………………………………….30

Figura 17 Regras Suricata contra SYN-Flood…………………………………………………….31

Figura 18 Deteção no OPNsense………………………………………………………………..31

Figura 19 Ataque UDP-Flood……………………………………………………………………32

Figura 20 Regras Suricata contra UDP-Flood……………………………………………………33

# Contexto

Nos últimos anos (2023–2025), os ataques de port-scanning e de negação de serviço (DDoS) têm-se intensificado globalmente. Relatórios de inteligência de segurança indicam que a atividade de “*scanning”* de portas e vulnerabilidades aumentou drasticamente – por exemplo, a F5 Labs registou um crescimento de cerca de 91% no volume de tráfego de varrimentos (scans) de 2023 para 2024.

Em termos de ataques DDoS, fontes como a Qrator apontam que inundações UDP (UDP flood) lideram em frequência (~55% dos tipos de DDoS), seguidas pelas inundações SYN(~17.6%). Relatórios da Cloudflare confirmam tendência similar: no 1.º trimestre de 2025 foram bloqueados 20,5 milhões de ataques DDoS (+358% em ano), com 16,8 milhões na camada de rede (incluindo SYN/UDP floods).

Em contrapartida, dados nacionais (CNCS/CERT.PT) destacam que em 2023 os incidentes mais comuns em Portugal foram phishing (35%) e outras fraudes, não sendo publicados números específicos para ataques de “*scanning”* ou inundação de tráfego.

## Detecção por NIDS (Suricata)

Os NIDS (Sistemas de Detecção de Intrusão em Rede) tradicionais usam essencialmente assinaturas e limiares de tráfego para identificar estes ataques.

Por exemplo, Port-scanning é detectado quando um host faz muitas tentativas de conexão a múltiplas portas ou destinos num curto intervalo: o Suricata inclui pré-processadores ou regras de thresholding que disparam alertas se um scanner excede certo número de portas falhadas. De forma similar, um SYN-Flood é detectado pelo elevado número de pacotes SYN não concluídos, e uma UDP-Flood pelo tráfego atípico de UDP (muitas requisições a serviços não existentes), muitas vezes utilizando regras de anomalia ou *rate limiting*.

Em resumo, as NIDS modernas, como suricata, criam logs detalhados de tráfego e usam regras – baseadas em assinaturas conhecidas ou limiares estatísticos – para sinalizar padrões de porta-arbitrária (scans) e fluxos anômalos (floods).

## Exemplos Recentes e Estatísticas de Incidentes

Os ataques mencionados figuram em grandes incidentes globais recentes. No primeiro trimestre de 2024, a Cloudflare mitigou 4,5 milhões de DDoS (mais 50% que no ano anterior). Dos ataques à camada de rede, observa-se que ataques SYN-Flood foram o 2.º tipo mais comum e UDP-Flood o 4.º (6% do total). Notórios tipos emergentes incluíram ataques SSDP e Jenkins (um ataque DDoS que explora serviços DNS em servidores Jenkins cresceu +826% em três meses).

Um exemplo emblemático foi um DDoS de 2 Tbps em 2024 lançado pela botnet Mirai, composto por dispositivos IoT infetados – a Mirai permanece ativa, sendo responsável por uma parte significativa dos ataques de grande escala.

## Evolução das Técnicas de Ataque e Resposta de Segurança

As técnicas de ataque continuam a evoluir. Vimos um ressurgimento de DDoS comerciais multi‑tipo (mirroring ataques dirigidos a infraestruturas críticas), crescimento nos *botnets* distribuídos (como variantes de Mirai) e a exploração de novos tipos (DNS flooding, SSDP, CLDAP).

Em resposta, as ferramentas de segurança adaptam‑se: NIDS como Suricata atualizam‑se constantemente com novas regras para os tipos de ataques emergentes.

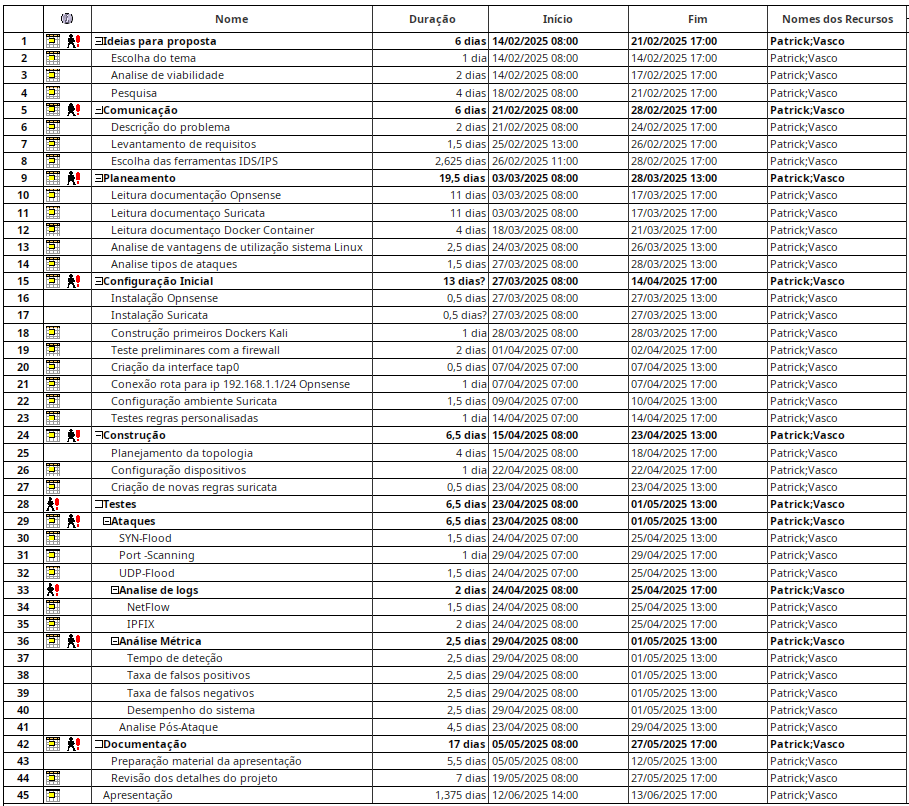
## Machine Learning e Técnicas Avançadas de Deteção

A integração de inteligência artificial em NIDS é uma tendência crescente. Em pesquisa, algoritmos de *anomaly detection* como isolação de floresta (*Isolation Forest*) ou redes neurais têm sido propostos para diferenciar tráfego legítimo de ataques de “scanning” ou DDoS.

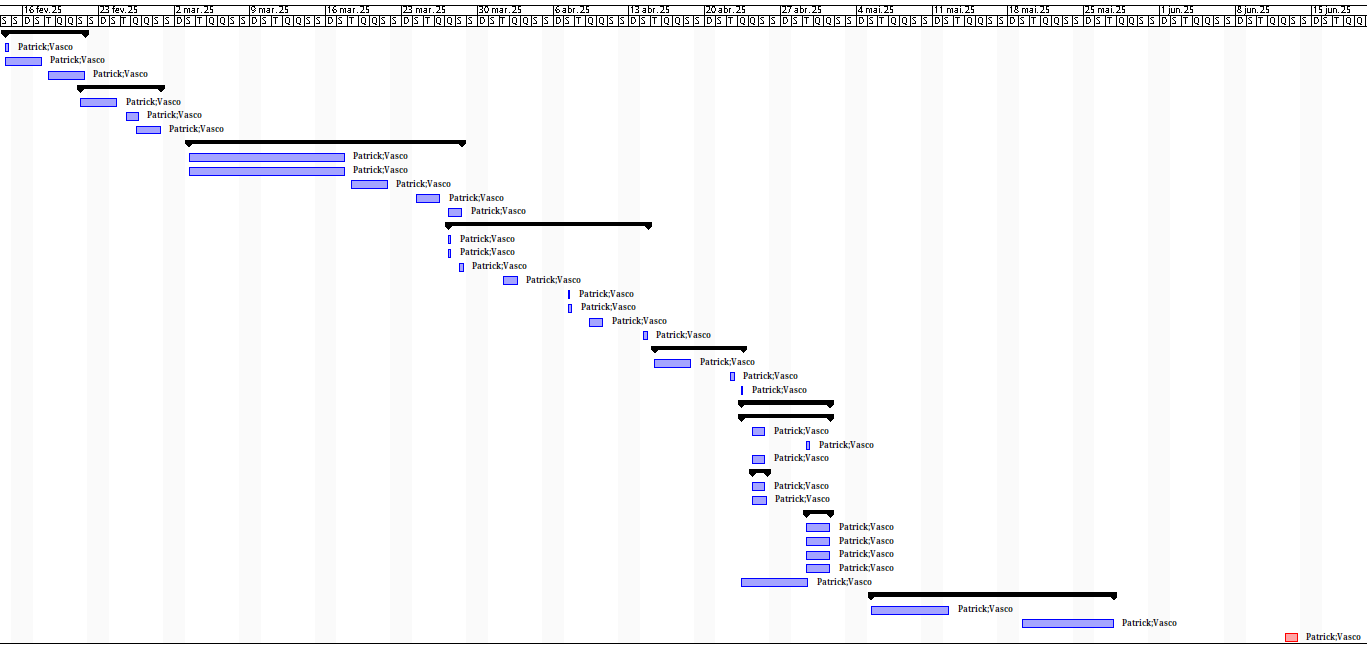
Em resumo, as ferramentas NIDS modernas combinam assinaturas tradicionais (fluxos, assinaturas de protocolo) com análise estatística e aprendizagem automática para detetar tanto ataques clássicos (port scanning, SYN/UDP floods) como variações mais sofisticadas, visando manter as redes mais seguras.

# 

# Planeamento

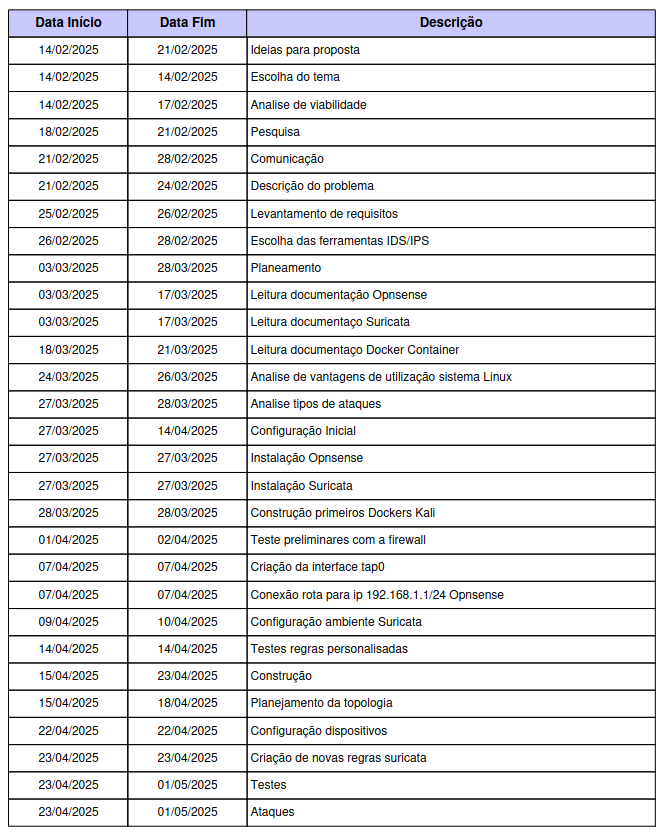


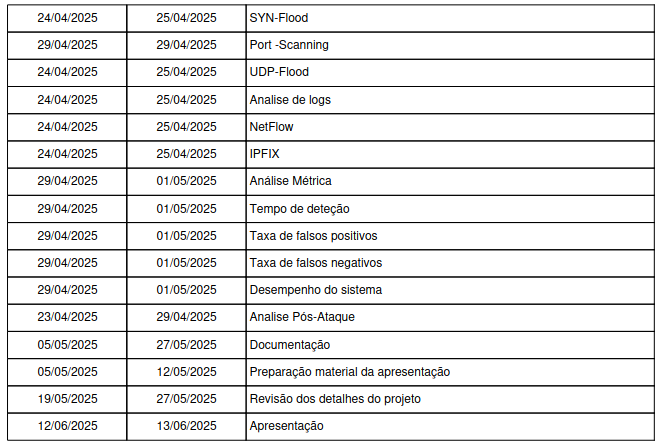
*Figura 1 Planeamento*



*Figura 2 Diagrama de Gantt*

# LogBook



*Figura 3 LogBook*

# Topologia



*Figura 4 Topologia do Projeto*

# Ferramentas

## GNS3

Planeamento da rede:

Para simular um ambiente realista de segurança de rede e ataque/defesa, foi desenhada uma topologia segmentada com redes internas, atacantes, acesso externo (GUI) e conectividade “WAN”. A distribuição de sub-redes permite simular ataques de diferentes origens (interna e externa), além de testar regras de IDS/IPS e firewall.

1. 1x OPNsense (Firewall)
2. 1x Router Cisco (R1)
3. 1x Ubuntu Host (representando a máquina real com acesso via tap0)
4. 1x PC1 (Docker)
5. 2x Atacantes (Docker, Kali + ferramentas)
6. 4x Switches Ethernet
7. 1x Nuvem Internet (Ethernet Switch)

Endereçamento IP e Interfaces:

| **Interface OPNsense** | **IP** | **Conectividade** |
| --- | --- | --- |
| em0 | 192.168.1.1/24 | Conectado à tap0 (Ubuntu host) |
| em1 | 192.168.2.1/24 | Rede Guest (PC1) |
| em2 | 209.20.1.1/24 | WAN (ligado a R1) |
| em3 | 192.168.3.1/24 | Rede Security (Atacante2) |

| **Interface R1** | **IP** | **Conectividade** |
| --- | --- | --- |
| e0/0 (WAN) | 209.20.1.2/24 | Ligado à em2 do OPNsense |
| e0/1 (LAN) | 10.0.0.1/24 | Rede externa atacante |

| **Outros Dispositivos** | **IP** | **Função** |
| --- | --- | --- |
| Atacante | 10.0.0.2 | Host externo (simula ataque externo) |
| Atacante2 | 192.168.3.2 | Host interno (ataque interno) |
| PC1 | 192.168.2.2 | Host legítimo interno |
| Ubuntu (tap0) | 192.168.1.2 | Acesso à interface web do OPNsense |

Configuração das Interfaces no GNS3:

Para cada ligação de switch, foi utilizado um Ethernet switch simples com portas e0–e1, ligando os dispositivos às suas respectivas interfaces. Os endereços IP foram atribuídos manualmente.

A interface tap0 foi criada no sistema anfitrião com:

sudo ip addr add 192.168.1.2/24 dev tap0

sudo ip link set tap0 up

sudo ip route add 192.168.1.2 dev tap0

Roteamento e NAT:

O router R1 desempenha o papel de gateway para a rede 10.0.0.0/24, onde se encontra o atacante externo, encaminhando o tráfego proveniente dessa rede em direção à firewall. Por sua vez, o OPNsense está configurado com o endereço IP 209.20.1.1 na sua interface WAN e utiliza como gateway padrão o endereço 209.20.1.2, correspondente à interface do R1. A firewall OPNsense está equipada com regras de NAT e filtragem que controlam o fluxo de pacotes entre as diferentes zonas da rede, permitindo não só a passagem seletiva de tráfego como também a atuação do sistema de deteção de intrusões Suricata. Esta infraestrutura possibilita a simulação de tráfego real entre segmentos de rede distintos, incluindo comunicações legítimas e tentativas de intrusão, garantindo a deteção e eventual bloqueio de atividades maliciosas através da geração de alertas.

## OPNsense

O OPNsense é uma plataforma de segurança de redes open source baseada em FreeBSD.

Funciona como firewall/router de última geração, oferecendo filtragem de pacotes *stateful* (IPv4/IPv6), NAT, balanceamento de carga multi-WAN e VPN integrado (IPsec, OpenVPN, WireGuard, etc.).

Dispõe de uma interface web intuitiva e recursos de relatório e monitorização de tráfego (RRD, NetFlow, etc.). As funcionalidades principais incluem:

1. Firewall stateful: regras detalhadas de filtragem de tráfego por origem/destino/porta, com ações *Pass* (permitir), *Block*(bloquear silenciosamente) ou *Reject* (bloquear com retorno de erro).
2. Multi-WAN: suporte a várias conexões WAN com balanceamento de carga.
3. VPNs: suporte nativo a IPsec (incluindo modos *route-based*), OpenVPN, Tinc (malha) e WireGuard para interligação de redes seguras.
4. Alta disponibilidade: redundância de hardware via CARP e sincronização de estado, minimizando tempo de inatividade.
5. Relatórios e monitorização: gráficos RRD, análise NetFlow, DNS reverso (Unbound) e serviços adicionais via plugins (suricata, proxy, etc.).

O OPNsense é bastante usado tanto em ambientes de laboratório de segurança (pela sua flexibilidade e licença aberta) como em infraestruturas reais (empresas ou residenciais avançadas). O seu modelo de desenvolvimento colaborativo assegura atualizações frequentes e permite estender funcionalidades através de plugins (p. ex. Suricata para IDS/IPS)

### Regras de Firewall e NAT

O OPNsense implementa um firewall de filtragem de pacotes *stateful*: cada regra é aplicada por fluxo de conexão e pode, por defeito, reter estado para agilizar o processamento.

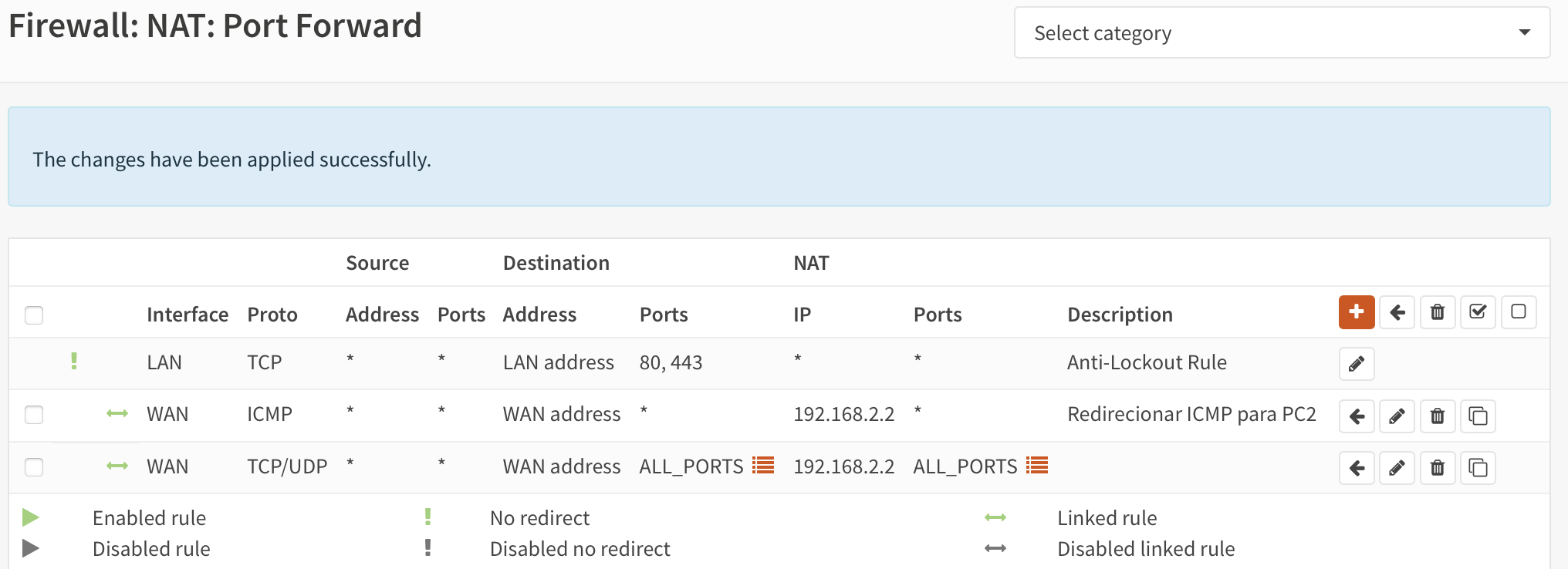
As regras são definidas por interface (LAN, WAN, etc.) e avaliadas em ordem (first-match ou last-match, via *quick*).

As ações disponíveis são Pass (permitir tráfego), Block(bloquear sem notificar) e Reject (bloquear enviando resposta de erro). Uma nota importante é que as regras NAT são sempre processadas antes das regras de Firewall. Ou seja, antes de aplicar bloqueios, o sistema traduz endereços conforme especificado nas regras NAT.

1. Regras NAT:

Permitem traduzir endereços entre redes internas e externas. O OPNsense suporta NAT estático(BINAT), NAT dinâmico de origem (Outbound NAT, para navegar na Internet com IP público) e Port Forwarding (Destination NAT). A NAT é usada para fazer um servidor interno responder sob um IP público: por exemplo, para tornar um servidor web interno acessível, criam-se regras de redirecionamento de portas (normalmente TCP 80/443) para o IP interno correspondente. As regras DNAT (“port forwarding”) convertem endereços de destino e portas de tráfego externo para o servidor interno.

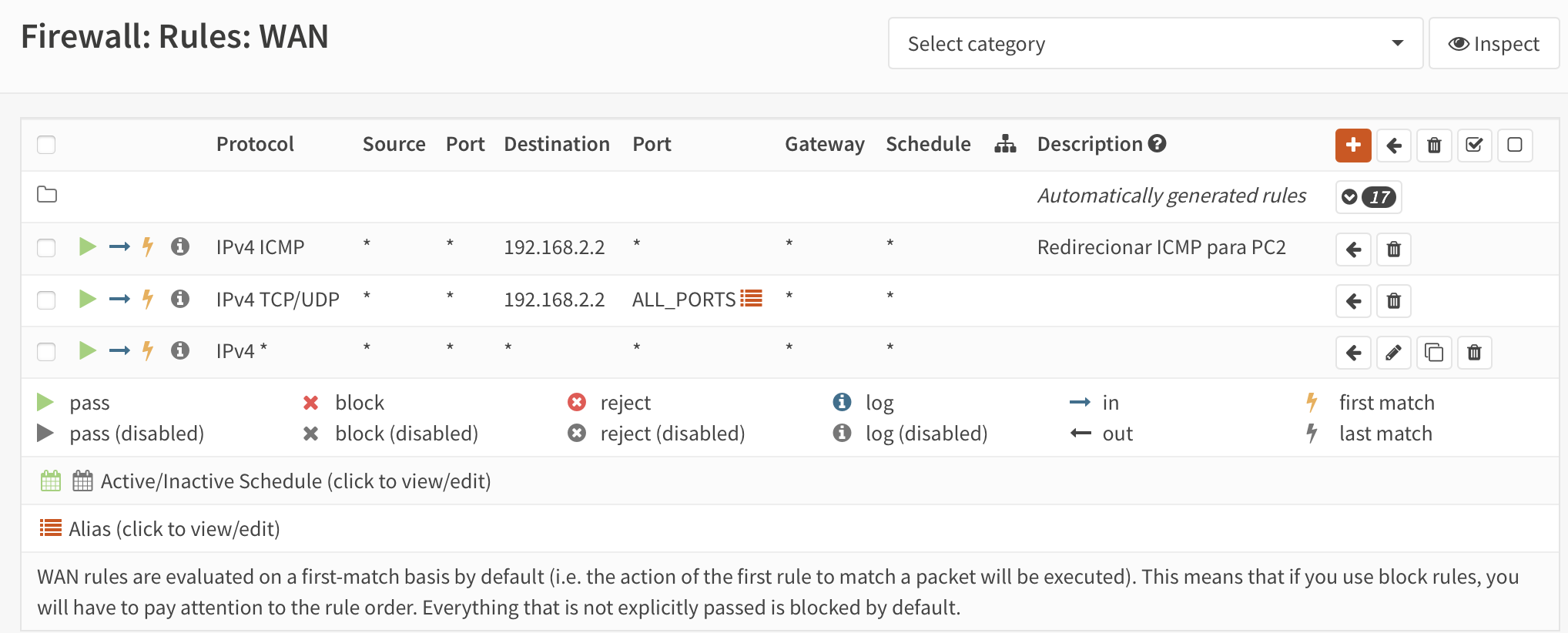
Em resumo, a NAT “separa” as redes WAN (externa) e LAN (interna) compartilhando um IP público entre múltiplos hosts internos. Uma regra de NAT no OPNsense geralmente cria automaticamente uma regra de firewall associada, garantindo consistência entre traduções e filtragens.



*Figura 5 Regras Nat criadas*

1. Regras de Firewall:

Especificam quais pacotes são autorizados a entrar ou sair em cada interface. O OPNsense inclui regras automáticas básicas (e.g. *Block bogon*, NAT reflexivo) que podem ser visíveis para inspeção. Em laboratórios ou implantação real, definem-se regras específicas por zona de rede para isolar segmentos, permitir serviços necessários e negar o resto.



*Figura 6 Regras Firewall criadas*

## Suricata

O Suricata é um motor de detecção e prevenção de intrusões (IDS/IPS) open source de alto desempenho. Trata-se de uma ferramenta de monitorização de tráfego de rede que inspeciona cada pacote em busca de padrões maliciosos conhecidos. Graças ao seu design multi-thread, pode analisar tráfego em paralelo sem degradar significativamente o desempenho do sistema. As suas principais características são:

1. Assinaturas e regras customizáveis:

Suricata compara o tráfego de rede com bases de assinaturas de ataques (regras) pré-definidas e definidas pelo utilizador. Isso permite identificar atividades maliciosas conhecidas (explorações, malware, scans, etc.).

1. Inspeção profunda de pacotes:

Analisa o conteúdo dos pacotes além dos cabeçalhos, podendo identificar ameaças encobertas em protocolos de aplicação e dados encriptados.

1. Análise de protocolos:

Suporta nativamente dezenas de protocolos (HTTP, TLS, DNS, SMTP, SMB, etc.), permitindo detetar comportamentos anómalos específicos de cada protocolo.

1. Deteção baseada em anomalias:

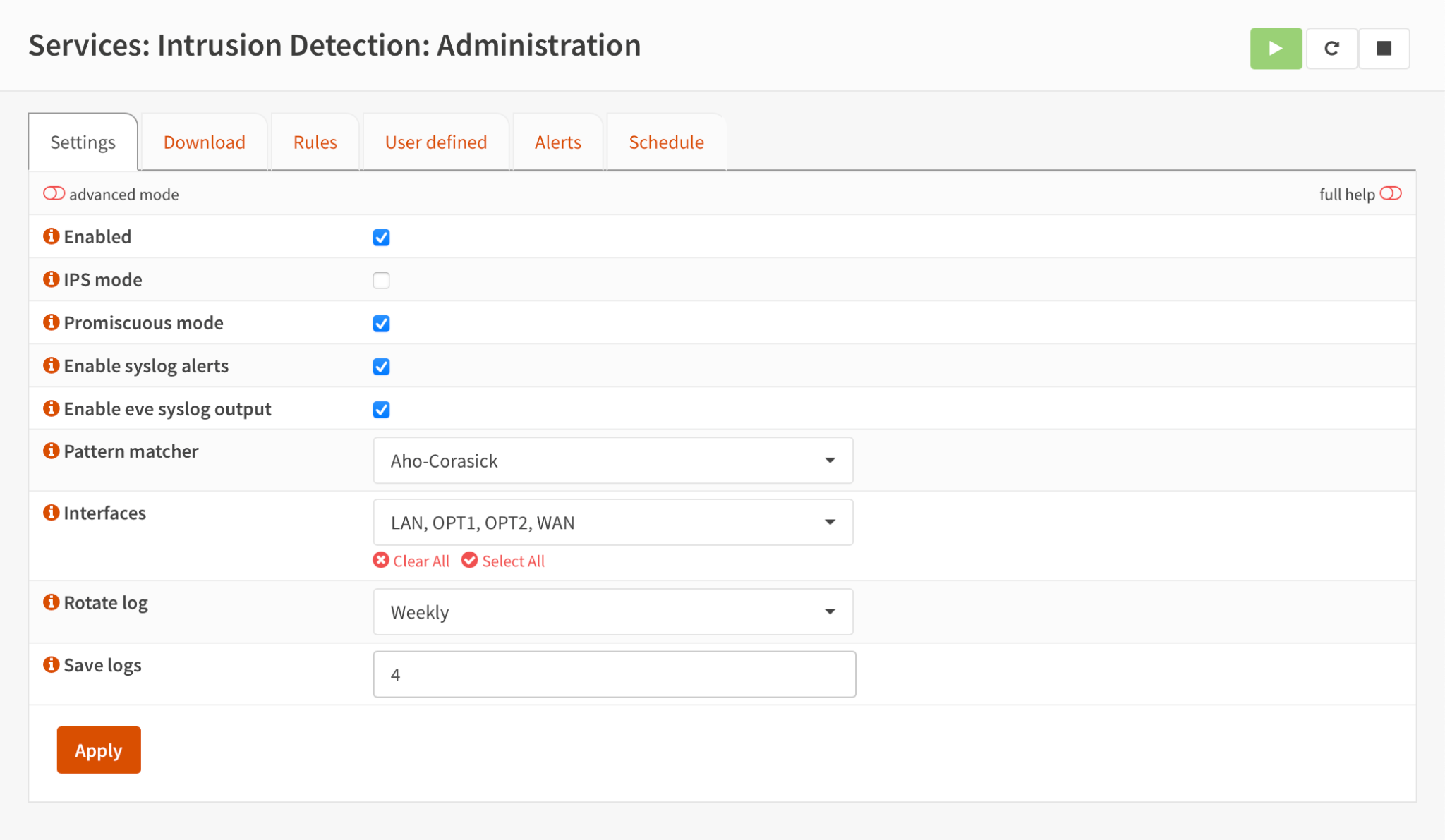
Permite definir limiares de tráfego (thresholds) ou usar motores de aprendizagem para sinalizar padrões incomuns. Por exemplo, picos súbitos de tráfego TCP ou múltiplas tentativas de conexão podem disparar alertas.

1. Modos IDS/IPS/NSM:

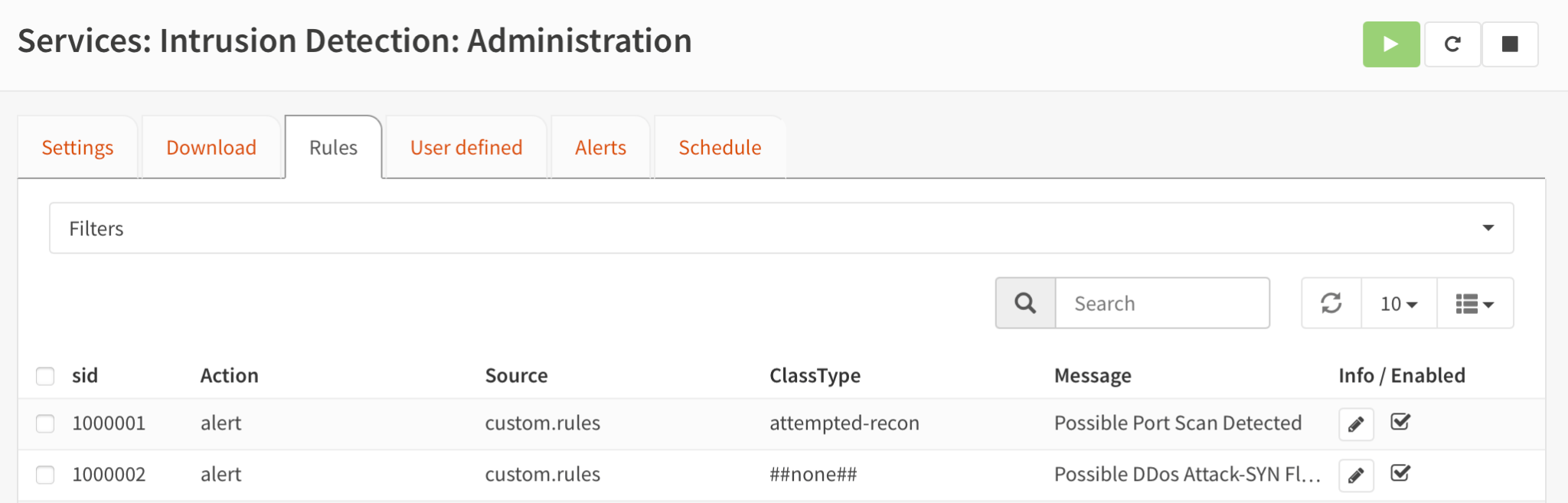
Além de funcionar como IDS (modo passivo, só alerta) e IPS (modo ativo, bloqueio de tráfego), o Suricata pode gerar logs detalhados (Network Security Monitoring) para análise forense.

Em conjunto com o OPNsense, o Suricata é disponibilizado como plugin de Intrusion Detection. No OPNsense, o sistema IPS baseia-se no Suricata e utiliza o Netmap para otimizar o desempenho e reduzir carga da CPU, permitindo inspeção profunda de pacotes em velocidade de rede.

O administrador pode instalar conjuntos de regras e escolher interfaces para monitorização, configurando o Suricata em modo IDS ou IPS conforme necessário. Dessa forma, tráfego suspeito (p.ex. botnets, exploits conhecidos ou floods) é detectado e opcionalmente bloqueado em linha pelo OPNsense.



*Figura 7 Ativação do Suricata no OPNsense*



*Figura 8 Regras Suricata criadas*

### Regras Personalizadas no Suricata

A configuração de regras personalizadas no Suricata permite adaptar o sistema de deteção de intrusões a cenários específicos e a padrões de tráfego observados na rede alvo. Embora o Suricata possua conjuntos de regras amplamente utilizados, como o Emerging Threats, a criação de regras próprias oferece maior controle sobre o comportamento do IDS/IPS, permitindo detectar ataques direcionados ou comportamentos anómalos muito específicos.

No âmbito deste projeto, foram desenvolvidas regras específicas para identificar ataques como SYN Flood, Port Scanning e UDP Flood. Esta secção descreve a estrutura sintática dessas regras, os objetivos de cada uma, os testes realizados e os resultados obtidos, incluindo eventuais falsos positivos e negativos.

### Sintaxe de uma Regra Suricata

O formato geral de uma regra no Suricata é:

ação protocolo origem porta\_origem -> destino porta\_destino (opções)

Componentes principais:

1. Ação: Pode ser alert, drop, reject, pass, ou log.
2. Protocolo: tcp, udp, icmp ou ip.
3. IP(origem,destino) e portas: Definem origem e destino da regra.
4. Opções: Instruções adicionais entre parênteses — mensagem, limiares, conteúdo, etc.

### Análise de Logs de Tráfego

A análise de logs de tráfego é uma componente essencial de um Sistema de Deteção de Intrusões em Rede (NIDS). Para além das regras de assinaturas que permitem a detecção imediata de padrões maliciosos, a análise contínua de fluxos de rede permite identificar comportamentos anómalos ao longo do tempo, como aumentos súbitos de tráfego, comunicação entre hosts inesperados, ou utilização excessiva de portas incomuns.

Neste projeto, foi utilizada a funcionalidade integrada do OPNsense para recolher dados de NetFlow e IPFIX (IP Flow Information Export), que são protocolos padronizados para exportação de dados sobre os fluxos de tráfego numa rede.

#### NetFlow

NetFlow é um protocolo originalmente desenvolvido pela Cisco, que permite a monitorização de fluxos IP numa rede. Cada "fluxo" representa uma comunicação unidirecional entre dois endpoints (IP de origem/destino, portas, protocolo, número de pacotes e bytes, etc.).

#### IPFIX

IPFIX é uma evolução e padronização do NetFlow (RFC 7011), permitindo maior flexibilidade e suporte a campos personalizados.

##### Ativação do NetFlow no OPNsense

A funcionalidade de exportação de fluxos foi ativada diretamente no OPNsense, na secção Reporting > NetFlow:

1. Foram selecionadas as interfaces:  
   1. em1 (LAN)
   2. em2 (WAN)
   3. em3 (OPT1)
2. O fluxo de dados foi armazenado localmente para análise direta ou exportado para ferramentas externas como o nfdump, em formato compatível com IPFIX.

#### Utilidade dos Logs para Detecção e Correlação

A análise de logs revelou-se valiosa para:

1. Corroborar alertas do Suricata com estatísticas reais de tráfego.
2. Detetar anomalias de volume (picos de pacotes/bytes) antes mesmo de ativar alertas por assinatura.
3. Identificar hosts suspeitos com padrões de comunicação incomuns (ex: múltiplas conexões curtas a vários destinos).
4. Gerar gráficos históricos de tráfego por protocolo, IP ou porta, ajudando na resposta e mitigação.

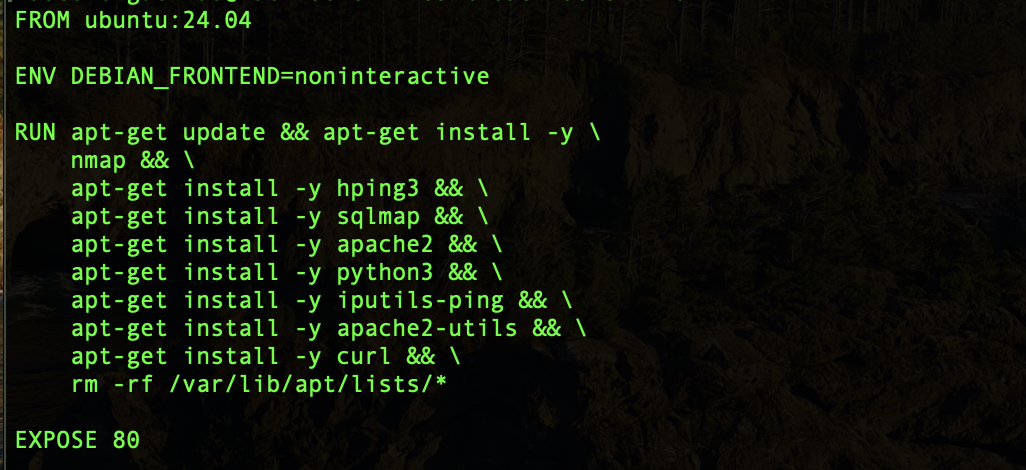
Ao combinar Suricata (deteção por assinatura) com NetFlow/IPFIX (análise comportamental), o sistema alcançou uma visibilidade mais abrangente da rede. Esta abordagem híbrida permitiu validar ataques simulados, reduzir falsos positivos e compreender a origem, escala e natureza do tráfego malicioso.

## Docker Container

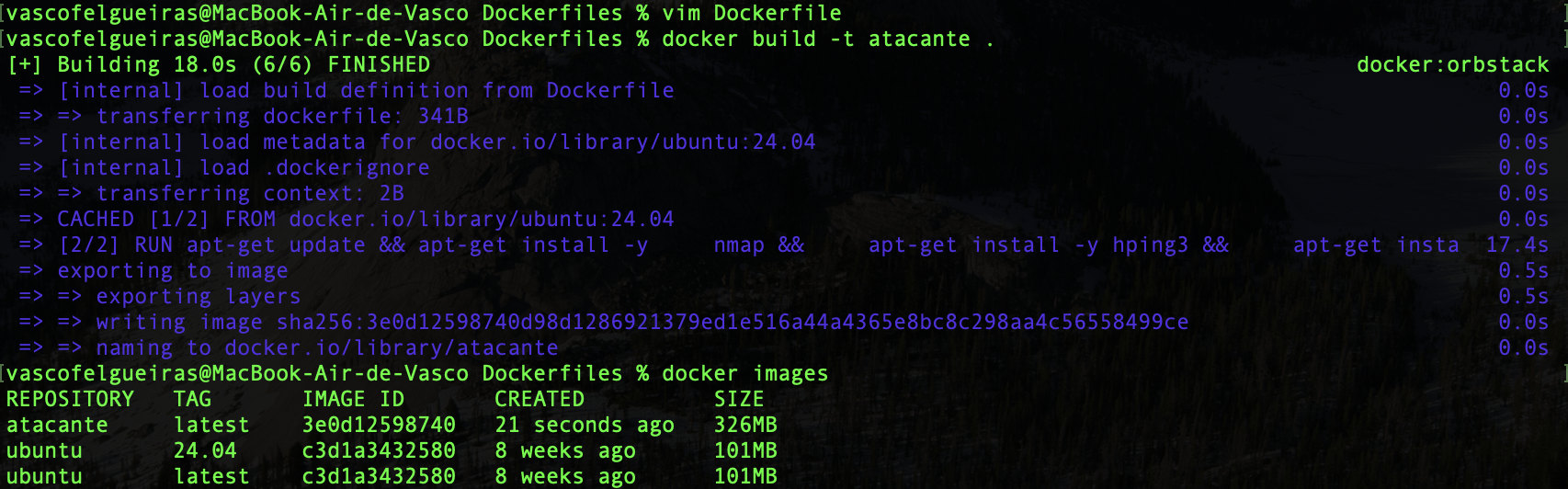
Um Docker Container personalizado é uma imagem construída pelo utilizador, com o sistema operativo base e as ferramentas necessárias para uma aplicação específica. Para criar uma imagem personalizada usa-se um *Dockerfile*: um ficheiro de texto contendo instruções (como FROM, RUN, COPY, etc.) que definem o ambiente, dependências e configuração do container.

Por exemplo, para o nosso projeto criamos um Dockerfile baseado em Ubuntu e instalamos ferramentas de segurança de rede como o *nmap*, o *hping3* eo *Apache2*.

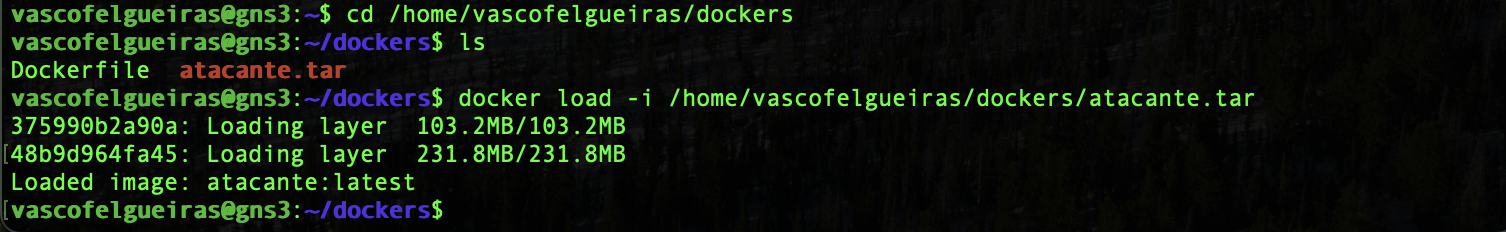
O Docker permite criar ambientes de teste isolados e reproduzíveis: em laboratórios de cibersegurança, fornece um ambiente leve, portátil e consistente para executar ferramentas de segurança e simular cenários de rede.



*Figura 9 Criação do Docker com as ferramentas necessárias*



*Figura 10 Construção do Docker*



*Figura 11 Download do Docker para o Ubuntu*



*Figura 12 Transferência do Docker para o GNS3*

A utilização do Docker neste projeto trouxe diversas vantagens específicas para o cenário de deteção de intrusões:

1. Ambientes isolados: cada container funciona como uma máquina independente (atacante, servidor, etc.), sem interferência entre si.
2. Facilidade de replicação: os containers podem ser reconstruídos a qualquer momento a partir de um Dockerfile, garantindo consistência.
3. Testes reversíveis: qualquer alteração indesejada pode ser descartada, recomeçando o ambiente em segundos.
4. Menor uso de recursos: containers consomem menos memória e CPU que máquinas virtuais, facilitando testes simultâneos em GNS3.

# Ataques

Esta seção apresenta a análise detalhada de três tipos de ataques simulados no ambiente de laboratório: Port Scanning e DDoS SYN Flood e UDP Flood. Para cada ataque, são descritas as suas características técnicas, ferramentas utilizadas, modo de detecção pelo sistema Suricata e respetiva resposta do sistema.

## Port Scanning

### Descrição Técnica

O Port Scanning é uma técnica amplamente utilizada por atacantes para identificar portas abertas em sistemas de destino. Através desta varredura, é possível descobrir quais serviços estão a escutar, constituindo um primeiro passo para exploração de vulnerabilidades.

Este tipo de ataque opera ao nível da Camada de Transporte do modelo OSI (camada 4), utilizando pacotes TCP ou UDP para testar respostas dos serviços remotos. Entre os métodos mais comuns estão o TCP Connect Scan, o SYN Scan (também conhecido como stealth scan) e o UDP Scan.

O SYN Scan, também conhecido como "half-open scan", funciona da seguinte forma:

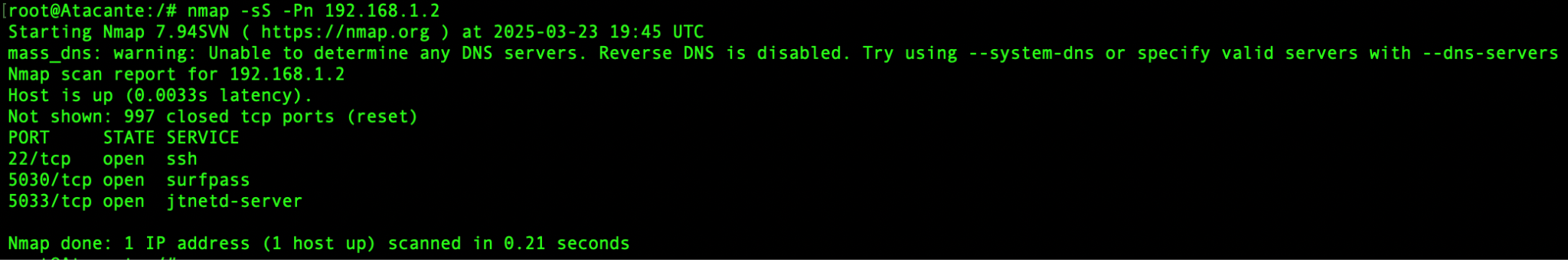
1. O atacante envia um pacote SYN (início do handshake TCP) para uma determinada porta.
2. Se a porta estiver aberta, o destino responde com um pacote SYN-ACK.
3. O atacante então não responde com ACK, interrompendo o processo e evitando a criação de uma ligação completa.
4. Se a porta estiver fechada, o destino responde com RST (Reset).

Este comportamento permite que o atacante analise rapidamente centenas ou milhares de portas sem completar sessões TCP.

#### 

### Ferramenta Utilizada

Para a execução deste ataque, foi utilizada a ferramenta Nmap, reconhecida pela sua versatilidade e capacidade de escanear grandes intervalos de portas.

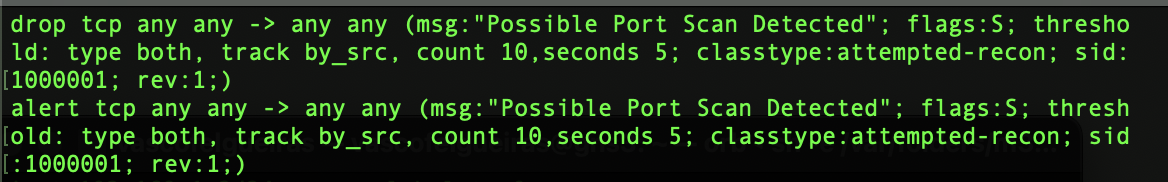


*Figura 13 Ataque Port-Scanning*

### Tráfego Gerado

Durante o scan, são enviados pacotes SYN de forma contínua, e registadas as respostas SYN-ACK (porta aberta), RST (porta fechada) ou ausência de resposta (porta filtrada).

### Regra Suricata



*Figura 14 Regras Suricata contra Port-Scanning*

### Deteção



*Figura 15 Deteção no OPNsense*

## SYN Flood

### Descrição Técnica

O SYN Flood é uma técnica de negação de serviço (DoS) que visa esgotar os recursos de uma máquina alvo, impedindo que ela atenda novas conexões legítimas. Este ataque é classificado como volumétrico e atua na camada 4 (transporte).

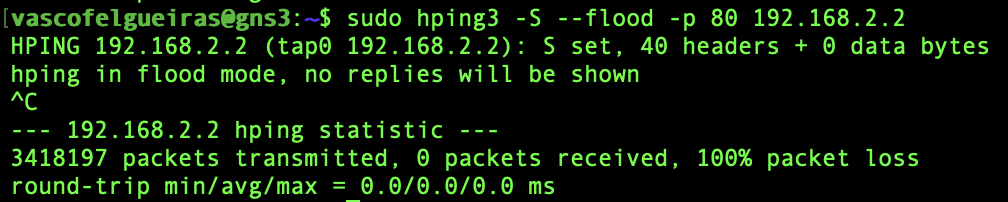
O funcionamento baseia-se na exploração do mecanismo de estabelecimento de conexão TCP, conhecido como handshake de três vias:

1. O atacante envia milhares de pacotes SYN a um servidor, fingindo iniciar conexões.
2. O servidor responde com SYN-ACK, esperando um ACK final para completar o handshake.
3. O atacante não envia o ACK final, mantendo o servidor com conexões "semiabertas".
4. À medida que estas se acumulam, a tabela de conexões do servidor atinge o limite, impedindo novas conexões legítimas.

Em ataques DDoS, este padrão é ampliado com múltiplos hosts distribuídos (botnets), tornando o tráfego mais difícil de filtrar.

### Ferramenta Utilizada

Foi utilizado o hping3 com o seguinte comando:

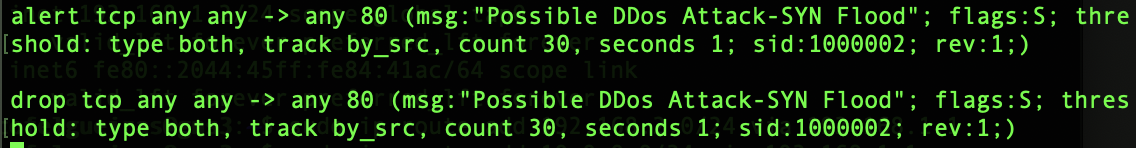


*Figura 16 Ataque SYN-Flood*

### Tráfego Gerado

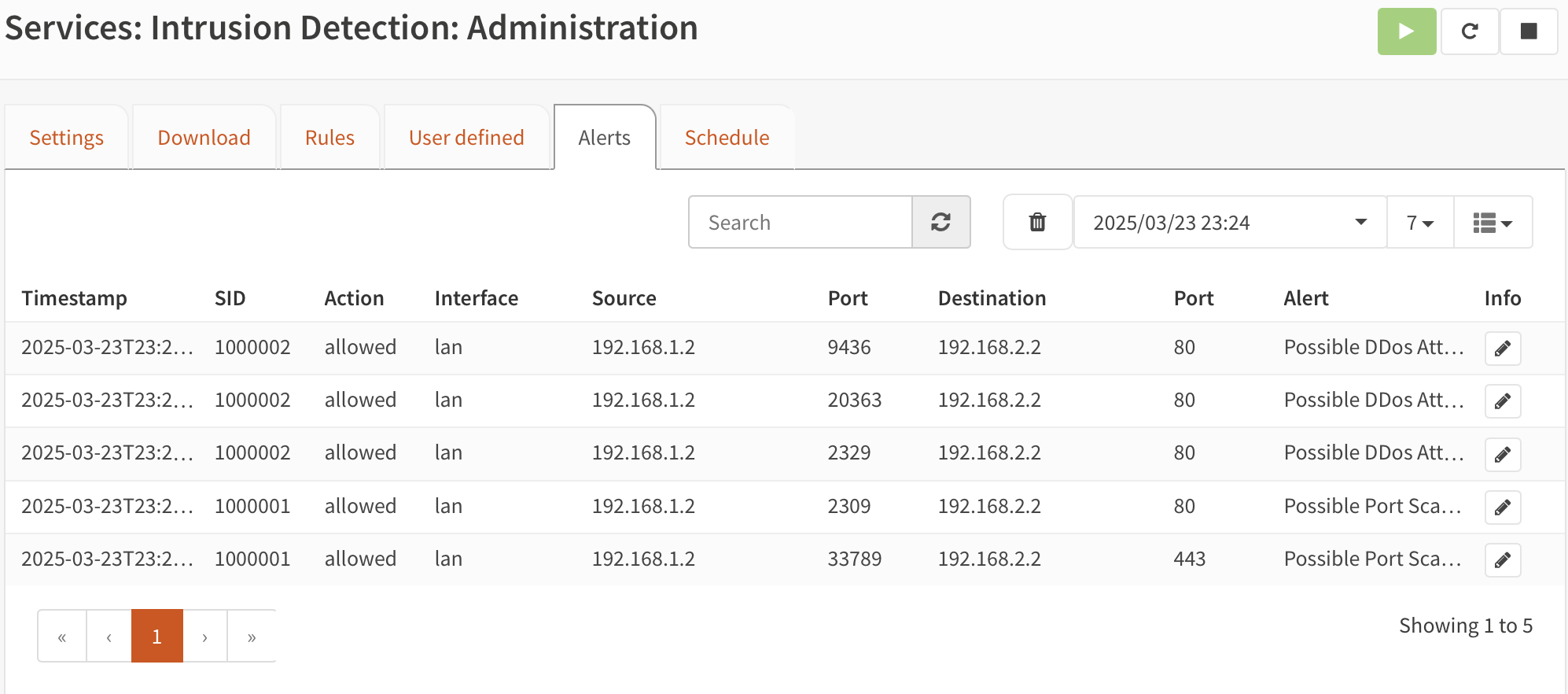
O tráfego gerado é composto por milhares de pacotes SYN por segundo, todos com destino à mesma porta do mesmo host, sem resposta ao SYN-ACK. Isso gera um consumo crescente de memória e CPU no sistema de destino.

### Regras Suricata



*Figura 17 Regras Suricata contra SYN-Flood*

### Deteção



*Figura 18 Deteção no OPNsense*

## UDP Flood

### Descrição técnica

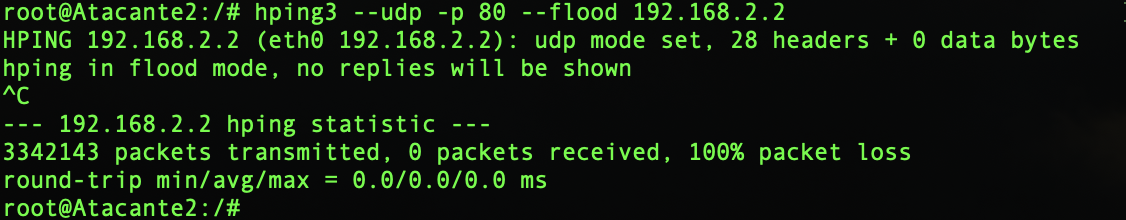
O UDP Flood é um ataque volumétrico de negação de serviço (DoS/DDoS) que atua na camada 4 (transporte). O atacante envia um grande volume de pacotes UDP para o alvo, muitas vezes para portas aleatórias ou inexistentes, sem estabelecer qualquer estado de conexão. O alvo, incapaz de descartar rapidamente estes pacotes, consome recursos de CPU e largura de banda, podendo responder com mensagens ICMP "Destination Unreachable", agravando a sobrecarga.

O funcionamento baseia-se em:

1. O atacante dispara milhares de datagramas UDP por segundo, com endereços de origem falsificados (IP spoofing).
2. O servidor, ao receber os pacotes para portas não abertas, tenta responder com ICMP, gerando tráfego de retorno.
3. A acumulação de pacotes de entrada e de respostas ICMP esgota recursos de rede e de sistema, bloqueando o serviço.
4. Em cenários DDoS, botnets distribuídas amplificam o volume, dificultando o bloqueio.

### Ferramenta Utilizada

Foi utilizado o hping3 com o seguinte comando:

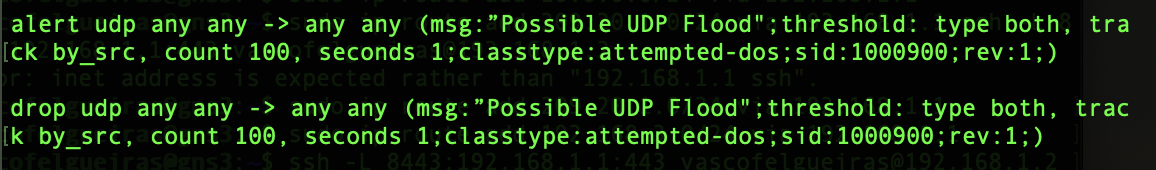


*Figura 19 Ataque UDP-Flood*

### Tráfego Gerado

O tráfego gerado consiste em milhares de pacotes UDP por segundo, todos dirigidos ao mesmo porto do host alvo, sem resposta ao pacote ICMP gerado pelo mesmo. Isto provoca consumo elevado de CPU, memória e largura de banda.

### Regras Suricata



*Figura 20 Regras Suricata contra UDP-Flood*

# Avaliação do Sistema e Métricas

Para avaliar a eficácia do sistema NIDS implementado, foram observadas métricas chave durante os testes:

### Métricas Recolhidas

1. Tempo de deteção: O sistema foi capaz de identificar ataques em alguns segundos após o início do tráfego malicioso.
2. Taxa de falsos positivos: Muito baixa, com exceções pontuais em regras de varrimento agressivas.
3. Taxa de falsos negativos: Moderada, especialmente em ataques lentos ou stealth.
4. Desempenho do sistema:  
   1. CPU: 40–65% de uso médio durante ataques.(variando em situações onde o GNS3 é executado em ambiente virtualizado)
   2. Memória: Estável (uso controlado com Docker e Netmap ativo no Suricata).
5. Geração de logs: Em média, 120–150 entradas de log por ataque simulado.

# Análise Pós-Ataque e Desafios Identificados

Após a realização das simulações de ataques e da respetiva monitorização por parte do sistema NIDS implementado, foi conduzida uma análise pós-ataque com o objetivo de avaliar a eficácia das medidas de deteção, identificar limitações operacionais e propor recomendações para reforçar a segurança da rede. Esta etapa revelou cinco áreas críticas de atenção, que serão analisadas individualmente a seguir.

1. Aumento da Sofisticação dos Ataques
   1. Um dos maiores desafios enfrentados atualmente é a crescente sofisticação dos ataques cibernéticos, que frequentemente recorrem a técnicas de disfarce, fragmentação de pacotes, uso de protocolos legítimos e variação de padrões. Durante os testes, por exemplo, foram identificadas dificuldades adicionais na detecção de varrimentos “stealth” ou ataques de negação de serviço fragmentados. Este tipo de ameaça consegue, por vezes, passar despercebido por regras baseadas apenas em assinaturas, exigindo mecanismos complementares, como análise estatística ou modelos baseados em inteligência artificial.
2. Regras Incompletas ou Desatualizadas
   1. Embora o Suricata disponibilize conjuntos de regras atualizadas (como o Emerging Threats), é comum que estas não cubram todas as variantes de ataque, especialmente as mais recentes ou personalizadas. A experiência do projeto evidenciou a necessidade de manter regras constantemente atualizadas e adaptadas ao contexto específico da rede monitorada.
   2. A eficácia das regras personalizadas depende diretamente do conhecimento do analista que as define, o que implica risco de lacunas, especialmente em ambientes dinâmicos ou heterogêneos.
   3. Uma das grandes mais-valias da comunidade é a disponibilização de regras públicas continuamente atualizadas, como o conjunto ET Open (Emerging Threats Open). Estas regras abrangem uma vasta gama de ameaças, incluindo malwares, tentativas de exploração, varrimentos de portas, botnets, ataques DDoS, entre outros. As regras são escritas em formato Suricata e podem ser descarregadas gratuitamente.
3. Má Configuração do Sistema
   1. A eficácia de um NIDS depende não apenas das regras, mas também da correta configuração das interfaces, gateways, roteamento e limiares de detecção. Durante o projeto, foi observado que erros simples, como a monitorização incompleta de interfaces ou a não aplicação de limiares adequados, podem comprometer totalmente a capacidade de detecção.
   2. Por exemplo, regras com thresholds demasiado altos não geram alertas em tempo útil, enquanto regras mal calibradas podem originar falsos positivos que distraem o analista de segurança.
4. Limitações de Desempenho
   1. Sistemas de detecção como o Suricata realizam inspeção profunda de pacotes, o que implica um consumo significativo de CPU e memória, especialmente em redes com tráfego intenso. Durante ataques de alta volumetria, como o SYN Flood, foi possível observar picos de utilização do sistema, que, se não forem mitigados, podem comprometer a análise de tráfego em tempo real.
   2. Essas limitações são particularmente relevantes quando se pretende ativar o Suricata em modo IPS, onde o bloqueio de pacotes em linha requer latência mínima.
5. Falsos Negativos
   1. Apesar dos bons resultados obtidos, a ocorrência de falsos negativos — isto é, tráfego malicioso que não foi identificado como tal — constitui um risco crítico para qualquer sistema de detecção. Esse tipo de falha é mais provável quando o ataque imita padrões de tráfego legítimo, ou quando explora canais menos monitorizados (como protocolos internos, DNS, ou portas não usuais).
   2. Durante os testes, simulações de port scanning com baixa cadência, não foram detectados em tempo útil, evidenciando a limitação de regras baseadas em volume ou frequência.
6. Rede Segura e Comunicação com o Exterior
   1. Um dos maiores desafios conceptuais foi equilibrar a proteção interna com a necessidade de se comunicar com redes externas. A rede monitorada neste projeto precisa manter acesso à internet, interações entre zonas e serviços internos abertos, o que a torna naturalmente exposta a riscos.
   2. É nesta complexidade que os sistemas NIDS ganham relevância, pois permitem monitorizar, alertar e bloquear tráfego malicioso sem impedir a funcionalidade da rede. No entanto, isso só é eficaz se a configuração for precisa e as regras forem adequadamente mantidas e ajustadas.

A análise pós-ataque demonstrou que, apesar da eficácia geral do Suricata e da topologia configurada, a segurança perfeita não existe. A eficácia de um NIDS depende de uma estratégia contínua de calibração, atualização e resposta, apoiada por métricas reais e conhecimento técnico atualizado. Por isso, além da detecção reativa,é aconselhável complementar o sistema com monitorização contínua, inteligência de ameaças e práticas de hardening da rede.

# Melhorias Futuras e Integração de Inteligência Artificial

Apesar dos resultados positivos obtidos com a implementação do NIDS baseado em OPNsense e Suricata, é evidente que a segurança de rede é um processo contínuo e dinâmico, exigindo atualizações regulares e adaptação às novas ameaças emergentes. Com base na experiência adquirida durante este projeto, identificamos diversas oportunidades de melhoria e evolução do sistema.

## Integração com Sistemas SIEM

Uma melhoria significativa seria a integração do Suricata com uma plataforma de SIEM (Security Information and Event Management), como Wazuh, Splunk ou o stack ELK (Elasticsearch, Logstash, Kibana). Essa integração permitiria uma centralização da gestão de eventos, maior visibilidade sobre incidentes, correlação entre alertas e automatização da resposta.

#### 

## Automatização da Resposta a Incidentes

Com base nos alertas gerados pelo Suricata, podem ser desenvolvidos scripts ou políticas automáticas de mitigação, como:

1. Bloqueio temporário de IPs suspeitos via regras de firewall.
2. Notificações por e-mail ou Telegram a administradores.
3. Integração com honeypots, para desviar e isolar atacantes.

## Fortalecimento da Segurança Perimetral (Hardening)

O OPNsense pode ser ainda mais reforçado através de práticas de hardening, como:

1. Limitação de serviços acessíveis a partir da WAN.
2. Implementação de VPN para gestão remota (ex: WireGuard).
3. Utilização de DNS seguro (DNS-over-TLS/HTTPS).
4. Segmentação mais rígida entre zonas internas (Guest/Security).

Uma das direções mais promissoras para evoluir este NIDS é a aplicação de técnicas de Inteligência Artificial (IA) e Aprendizagem Automática (Machine Learning). Estas abordagens não dependem exclusivamente de assinaturas ou limiares manuais, mas aprendem com o comportamento da rede e conseguem identificar anomalias com base em padrões estatísticos e históricos.

## Vantagens da IA em Sistemas de Deteção

1. Detecção de ataques “zero-day”, que ainda não possuem assinaturas conhecidas.
2. Redução de falsos positivos, ao distinguir tráfego legítimo de comportamentos maliciosos.
3. Capacidade adaptativa, onde o modelo melhora ao longo do tempo com mais dados.

### Exemplos de Algoritmos Relevantes:

| **Algoritmo** | **Aplicação** |
| --- | --- |
| Isolation Forest | Detetar outliers em fluxos de tráfego com base em métricas de dispersão. |
| K-Means Clustering | Agrupamento de fluxos semelhantes e deteção de grupos anómalos. |
| Redes Neuronais LSTM | Análise de séries temporais em tráfego de rede, ideal para identificar padrões de ataques lentos e progressivos. |
| Naive Bayes | Classificação probabilística de tráfego com base em etiquetas históricas. |

### Aplicação Prática

É possível capturar estatísticas NetFlow (como número de pacotes, bytes, duração de conexões, IPs e portas) e alimentar um modelo IA treinado para reconhecer padrões normais de utilização da rede. Sempre que uma conexão apresenta comportamentos estatisticamente improváveis, um alerta é gerado, mesmo que não exista uma assinatura associada.

### Papel dos Componentes numa Arquitetura Integrada

A integração entre o OPNsense, o Suricata e um sistema de Inteligência Artificial (IA) baseia-se na distribuição de responsabilidades bem definidas entre os componentes, garantindo eficiência na monitorização, análise e resposta a incidentes de segurança.

O sistema de IA opera fora do OPNsense, mas está integrado funcionalmente através da análise de dados exportados. Este componente é responsável por recolher e interpretar os logs gerados pelo Suricata, bem como dados de fluxo de rede (NetFlow/IPFIX) provenientes do OPNsense. Com base nesses dados, são treinados e aplicados modelos de machine learning capazes de:

1. Identificar anomalias comportamentais, como ataques lentos ou varreduras camufladas;
2. Reduzir a incidência de falsos positivos, aprendendo com o histórico da rede;
3. Detetar padrões emergentes ou evasivos que não são reconhecidos por regras tradicionais.

Além da análise, a IA pode também interagir com o OPNsense através da sua API REST, realizando ações automatizadas como:

1. Criação de regras temporárias de firewall;
2. Notificação aos administradores em tempo real;
3. Atualização dinâmica de listas de IPs bloqueados com base nos resultados da análise.

Apesar de ter sido identificada a viabilidade técnica e estratégica da integração entre o sistema NIDS implementado (OPNsense + Suricata) e um motor externo de Inteligência Artificial, esta fase do projeto não foi concluída dentro do tempo disponível. A prioridade foi garantir uma infraestrutura funcional de deteção de intrusões com base em regras tradicionais e personalizadas, bem como realizar testes de eficácia com ataques reais em ambiente controlado.

No entanto, a arquitetura desenvolvida já foi preparada para suportar futuras expansões, incluindo a exportação de logs e dados de tráfego compatíveis com sistemas de análise avançada. A utilização de ferramentas como NetFlow, a estrutura modular do OPNsense e o suporte à API REST abrem caminho para a implementação gradual de modelos de machine learning no futuro.

Assim, embora a integração com IA não tenha sido materializada nesta versão do projeto, a sua implementação é plenamente possível e representa uma evolução natural e altamente recomendável para reforçar a capacidade de deteção de ameaças desconhecidas, reduzir falsos positivos e adaptar o sistema a cenários de ataque cada vez mais complexos.

# 

# Conclusão

O projeto demonstrou com sucesso a implementação de um sistema de deteção de intrusões funcional, com base em ferramentas open source como OPNsense e Suricata. Foi possível simular cenários realistas de ataques externos e internos, definir regras personalizadas, monitorar fluxos de tráfego com NetFlow/IPFIX e registar alertas em tempo real.

A arquitetura adotada, incluindo roteador, firewall, redes segmentadas e máquinas virtuais via Docker, permitiu um ambiente de testes robusto e controlado. A capacidade do Suricata em identificar ataques clássicos como Port Scanning e SYN Flood confirmou a utilidade da solução para contextos corporativos ou académicos.

Foram também identificadas limitações importantes, como falsos negativos em tráfego stealth e desafios de desempenho sob carga elevada. No entanto, essas limitações são comuns em ambientes de segurança reais e abrem espaço para melhorias, nomeadamente com o uso de inteligência artificial e integração com plataformas SIEM.

Em suma, este projeto permitiu não só validar o funcionamento de um NIDS, como também reforçar conhecimentos práticos em redes, segurança, simulação de ataques e análise de tráfego. Representa um ponto de partida sólido para futuras implementações em ambientes reais com requisitos de segurança mais exigentes.

# Agradecimentos

Gostaríamos de expressar o nosso agradecimento ao professor responsável pela cadeira de LPI: Paulo Rurato, e aos orientadores: Pedro Sobral e Bruno Gomes, pelo acompanhamento e orientações técnicas ao longo da realização deste projeto. Agradecemos também à Universidade Fernando Pessoa pelo suporte e pelas infraestruturas disponibilizadas, bem como aos colegas que contribuíram com sugestões e testes durante a fase de simulação.

Este trabalho foi fruto de colaboração contínua, dedicação prática e espírito de equipa, aos quais deixamos aqui os nossos mais sinceros agradecimentos.