# Eficiência do Kitsune no Dataset AB-TRAP: Classificação e Separação de Tráfego de Rede

*Vinicius Silva Gonçalves (IC), Edvard Martins de Oliveira (PQ)*[1]

**Palavras-chave: KITSUNE;AUTOENCODERS;REDE NEURAL; FLUXO;DADOS.**.

## Introdução

A segurança cibernética tornou-se um pilar essencial no cenário global atual, impactando diretamente setores vitais como negócios, saúde, transporte e segurança pública. Com a crescente digitalização e a integração de tecnologias emergentes, como a Internet das Coisas (IoT) e a conectividade 5G, a proteção de dados e sistemas tornou-se uma prioridade de nível global. De acordo com a Cybersecurity Ventures, o custo dos crimes cibernéticos deve atingir impressionantes US$ 8 trilhões em 2023, consolidando o cibercrime como a terceira maior economia mundial, apenas atrás dos EUA e da China[1].

No Brasil, a promulgação da Lei Geral de Proteção de Dados Pessoais (LGPD) em 2018 sublinhou a importância da segurança cibernética, estabelecendo diretrizes rigorosas para a coleta e tratamento de dados pessoais. Este marco legal destaca a necessidade de sistemas robustos para proteger informações sensíveis, especialmente à luz de incidentes recentes. Em agosto de 2023, uma série de ataques cibernéticos comprometeu as operações de várias unidades de saúde da Prospect Medical Holdings nos EUA, afetando hospitais e clínicas em estados como Califórnia, Texas e Pensilvânia. Esses ataques ressaltam a vulnerabilidade dos sistemas de saúde e a necessidade crítica de proteger dados pessoais e operacionais[2].

A chegada da tecnologia 5G está transformando a conectividade global, possibilitando inovações como veículos autônomos e cirurgias remotas. No entanto, essa expansão também aumenta a superfície de ataque para cibercriminosos. Um exemplo marcante é a botnet Mirai, que infectou entre 200.000 e 300.000 dispositivos IoT, explorando vulnerabilidades em protocolos desatualizados e configurações de segurança inadequada[3]. Além dos dispositivos IoT, os sistemas de informação e entretenimento dos veículos modernos representam outra área crítica. Recentemente, hackers conseguiram comprometer o sistema de um veículo, manipulando funções como motor e rodas, o que aumentou o risco de acidentes e evidenciou a necessidade de proteção adicional. Nesse contexto, a implementação de Sistemas de Detecção de Intrusão de Rede (NIDS) é crucial para mitigar essas ameaças. Um NIDS monitora o tráfego de rede, identificando atividades maliciosas e alertando administradores. No entanto, a eficácia desses sistemas pode ser limitada pelos elevados requisitos computacionais, como os exigidos por redes neurais tradicionais. Essas redes necessitam de grande capacidade de processamento e atualizações manuais constantes, o que as torna impraticáveis para dispositivos com recursos limitados.

É neste cenário que o Kitsune, um NIDS baseado em aprendizado de máquina, se destaca. Operando de forma não supervisionada e online com baixa complexidade, o Kitsune é adequado para implementação em dispositivos embarcados, como roteadores e gateways. No entanto, o Kitsune ainda enfrenta desafios, como a dificuldade em distinguir pacotes específicos de dados maliciosos dos benignos em um fluxo contínuo de pacotes de rede.

Além disso, a segmentação introduzida no Kitsune aprimora a capacidade do NIDS na identificação de pacotes maliciosos específicos, superando limitações das versões anteriores. Essa melhoria não só aumenta a precisão na detecção de ameaças, mas também permite uma resposta mais eficiente a ataques cibernéticos. Com isso, o sistema se torna mais robusto para redes com recursos limitados, oferecendo uma proteção aprimorada para dispositivos IoT e veículos conectados. No entanto, durante a segmentação, foi observado que o Kitsune ainda enfrenta dificuldades na detecção de certos arquivos maliciosos em fluxos de rede. O objetivo deste artigo é apresentar os resultados do Kitsune após a realização da segmentação do fluxo de rede em arquivos PCAP, distinguindo entre tráfego malicioso e não malicioso, utilizando um conjunto de dados reais.

## Metodologia

O Kitsune[4] utiliza uma série de pequenas redes neurais conhecidas como autoencoders, treinadas para replicar padrões de tráfego de rede, melhorando seu desempenho ao longo do tempo. *Autoencoders* são um tipo específico de rede neural que tenta copiar a saída a partir da entrada, comprimindo os dados em uma representação de menor dimensão e depois expandindo-os para o formato original.

Diferente da maioria das redes neurais, o Kitsune emprega aprendizado de máquina não supervisionado, o que significa que os dados usados para o treinamento não são rotulados por humanos. A rede neural deve analisar e classificar os dados de forma autônoma. Esse processo se baseia em uma técnica de *clustering*, ou seja, agrupando os dados conforme suas semelhanças e diferenças. Um exemplo prático seria o de uma empresa que utiliza algoritmos de clusterização para segmentar seu mercado, formando grupos baseados em características como localização geográfica ou perfil de consumo. Esses grupos podem ajudar na reconstrução eficiente dos dados durante o processamento.

A eficiência do NIDS pode ser ajustada por um parâmetro chamado **m**, que define o tamanho máximo dos *autoencoders* na camada de processamento. *Autoencoders* menores são mais rápidos de treinar e operar.O *autoencoder* busca aprender uma função **h(W,b)(x) ≈ x** ou seja, ele tenta replicar **x** na saída. **W** e **b** são os pesos de cada camada, e a função hhh descreve a propagação direta pelas camadas da entrada **x**. Quando a rede é limitada, como com um número reduzido de unidades ocultas, ela aprende uma representação mais compacta dos dados. Por exemplo, imagine uma imagem de 10x10 pixels como entrada, com 50 unidades ocultas na camada intermediária. A rede deve então reconstruir a imagem original de 100 pixels a partir dessa representação compacta. Se os dados forem estruturados, a rede poderá detectar correlações, resultando em uma representação de menor dimensão.[5]

Além disso, a função de ativação no Kitsune desempenha um papel fundamental ao decidir se um neurônio deve ser ativado ou não, com base na relevância dos dados recebidos. Essas funções aplicam transformações não lineares aos sinais de entrada, permitindo que a rede resolva problemas mais complexos. Sem a função de ativação, a rede seria limitada a uma simples transformação linear. Uma função de ativação comumente usada no Kitsune é a sigmoide, que pode ser definida da seguinte forma:

$$\sigma(x) = \frac{1}{1+e^{x}}$$

O modelo completo do Kitsune pode ser visualizado na Figura 1.

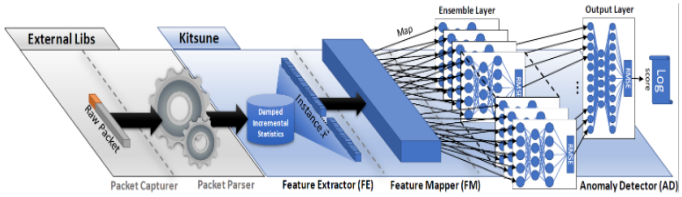


figura 1: Estrutura do Kitsune

**Os componentes do Kitsune são:**

**Packet Capture:** Primeiro componente, responsável por capturar os pacotes de rede e enviá-los ao Packet Parser.

**Packet Parser:** Extrai informações dos pacotes capturados e repassa os dados ao *Feature Extractor* (FE).

**Feature Extractor (FE):** Armazena estatísticas dos pacotes e gera um vetor com 115 *features* **x**, que é enviado ao *Feature Mapper* (FM).

**Feature Mapper (FM):** Agrupa os elementos em subespaços e clusters **v**, com cada cluster contendo até **m** *features*, valor definido pelo usuário.

**Anomaly Detector (AD):** Detecta pacotes anômalos com base na representação **v.**

Um dos principais desafios enfrentados foi a obtenção de fluxos de rede maliciosos reais, principalmente devido à dificuldade em encontrar datasets com gabarito adequado para a detecção de intrusões. Para superar essa limitação, utilizamos a estrutura AB-TRAP, amplamente aplicável no desenvolvimento de Sistemas de Detecção de Intrusão de Rede (NIDS). Essa estrutura não apenas permite o uso de tráfego de rede atualizado, como também aborda preocupações operacionais, facilitando a implantação completa de soluções de segurança.

A AB-TRAP segue cinco etapas principais:

1. **Geração do Conjunto de Dados de Ataque:** Nessa fase, são gerados os dados representando cenários de ataques tanto em redes locais (LAN) quanto em ambientes de Internet.
2. **Geração do Conjunto de Dados Genuíno:** O conjunto de dados genuíno, ou "bonafide", é

criado com base no conjunto de dados MAWILab, que captura fluxos de rede sem atividade maliciosa conhecida.

3. **Treinamento de Modelos de Aprendizado de Máquina:** Jupyter Notebooks são utilizados para pré-processamento dos dados e treinamento de modelos de aprendizado de máquina, adequados para os casos LAN e Internet.
4. **Realização dos Modelos:** Após o treinamento, os modelos de aprendizado de máquina são incorporados nos dispositivos de destino. Para o caso LAN, isso é feito no espaço do kernel usando módulos LKM, enquanto para o caso da Internet, a implementação é realizada no espaço do usuário com Python.
5. **Avaliação de Desempenho:** Por fim, o desempenho dos modelos é avaliado diretamente nos dispositivos de destino, garantindo que o sistema seja eficiente em ambientes operacionais reais.

Os exemplos fornecidos no repositório da AB-TRAP[6] oferecem suporte à virtualização, utilizando máquinas virtuais e contêineres para facilitar a geração dos conjuntos de dados e o treinamento dos modelos.

O conjunto de dados de ataques é particularmente relevante, pois inclui um arquivo .txt que associa IPs maliciosos aos tipos de ataques executados. Este conjunto é uma coletânea abrangente de ataques cibernéticos, abrangendo ferramentas e técnicas amplamente utilizadas em testes de penetração e exploração de redes.

O **Attack dataset** abrange uma série de ataques conduzidos por ferramentas como *network mapper* (Nmap), *Unicornscan*, *Hping*, *Zmap* e *Masscan*, todas amplamente utilizadas em testes de penetração e varreduras de rede. Essas ferramentas desempenham papéis cruciais na detecção de vulnerabilidades em redes e sistemas, sendo descritas como:

- **Nmap:** Ferramenta de código aberto popular para auditoria de segurança, utilizada para identificar hosts e serviços em uma rede.
- **Unicornscan:** Especializada em varredura de alta precisão e coleta de informações sobre redes e sistemas.
- **Hping:** Utilizada para escaneamento e ataques a partir de pacotes personalizados via linha de comando.
- **Zmap:** Notável por sua capacidade de realizar varreduras extremamente rápidas em redes de grande escala.
- **Masscan:** Conhecida por ser uma das ferramentas mais rápidas para varredura de portas, capaz de escanear grandes redes em alta velocidade.

Cada ataque registrado no *Attack dataset* está associado a uma etiqueta que descreve a ação específica realizada. Entre as técnicas mais frequentes, incluem-se:

- **Nmap TCP Null:** Um escaneamento em que todos os flags TCP são desativados, permitindo que o escaneamento passe por alguns firewalls sem ser detectado.
- **Nmap TCP Xmas:** Escaneamento em que as flags FIN, PSH e URG são ativadas, criando um padrão "Christmas tree" nos pacotes, usado para contornar mecanismos de defesa.
- **Nmap TCP Maimon:** Utiliza pacotes FIN/ACK para identificar portas abertas sem disparar alertas em firewalls.
- **Hping TCP Xmas:** Similar ao Nmap TCP Xmas, mas utilizando a ferramenta Hping.
- **Nmap TCP FIN:** Um escaneamento que define apenas a flag FIN, explorando especificidades de certos sistemas para detectar portas abertas.
- **Unicornscan TCP Connect:** Um escaneamento que utiliza a chamada connect() para estabelecer uma conexão completa.
- **Masscan TCP SYN:** Escaneamento TCP SYN, uma das técnicas mais rápidas e eficientes para varredura de portas.

Durante a execução do *attack dataset*, é gerado um arquivo output.pcap, enquanto o *bonafide dataset* gera um arquivo bonafide.pcap. Usando o Wireshark, esses pacotes TCP de ambos os datasets são combinados. No processo de treinamento do Kitsune, 90% dos pacotes são utilizados para treinamento e os 10% restantes para teste. Após essa fase, os pacotes são separados em arquivos pcap maliciosos e não maliciosos, que são então processados para verificar as métricas de desempenho do modelo.

Essa abordagem de mistura e separação dos pacotes, juntamente com a análise meticulosa das métricas, visa observar o quão eficiente o modelo Kitsune é ao diferenciar entre tráfego malicioso e não malicioso, abrindo espaços para possíveis melhorias aumentando sua precisão em detecções futuras.

## Resultados e discussão/Conclusão

Os resultados obtidos e apresentados na Figura 2 evidenciam tanto as capacidades quanto as limitações do Kitsune na detecção de tráfego malicioso em redes, utilizando arquivos de captura de pacotes (pcap). Embora o modelo tenha mostrado ser capaz de identificar uma parcela significativa das ameaças, com uma taxa de acerto de 66%, ele também apresentou uma quantidade considerável de falsos positivos, em torno de 44%. Esses falsos positivos refletem um dos desafios cruciais na aplicação prática de modelos baseados em autoencoders, onde qualquer falha na reconstrução do pacote, mesmo que mínima, leva à classificação como anômalo.

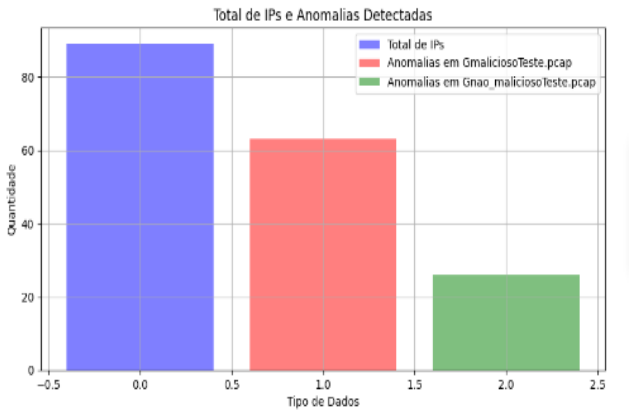


figura 2: Gráfico de resultados

O limiar utilizado para o erro médio quadrático (RMSE), configurado como maior que zero, foi eficaz em detectar pequenas variações, mas também resultou em um número elevado de falsos positivos. Isso sugere que, embora a metodologia de detecção de anomalias seja sensível, ajustes mais refinados no limiar ou nos parâmetros de reconstrução são necessários para evitar a penalização excessiva de pacotes legítimos.

Adicionalmente, alguns tipos de ataques, como os realizados por ferramentas como Unicornscan, Masscan e Hping, não foram corretamente identificados pelo Kitsune. Isso levanta a necessidade de melhorias no modelo, particularmente na arquitetura dos autoencoders, para capturar melhor os padrões gerados por essas ferramentas específicas. A falha em detectar esses ataques pode estar relacionada à natureza peculiar do tráfego gerado por essas ferramentas, que o Kitsune, na sua configuração atual, não conseguiu interpretar de maneira adequada.

Em termos de métricas de desempenho, o Kitsune apresentou uma precisão de 0,87 e um recall perfeito de 1,0, resultando em um F1-score de 0,931. Esses números indicam que o modelo possui um bom equilíbrio entre a detecção de pacotes maliciosos e a redução de falsos negativos. No entanto, o alto valor de RMSE e as variações significativas nos erros de reconstrução destacam a necessidade de ajustes adicionais, especialmente em ambientes de rede mais complexos.

Portanto, enquanto o Kitsune demonstrou ser uma ferramenta eficaz para detecção de intrusões em rede, ajustes na arquitetura do modelo e na configuração do RMSE são essenciais para reduzir a taxa de falsos positivos e aumentar a capacidade de detectar ataques realizados por ferramentas específicas, como Unicornscan, Masscan e Hping. Esse aprimoramento será fundamental para garantir que o Kitsune possa operar com maior consistência e precisão em cenários reais de rede, fortalecendo a segurança das redes monitoradas.

## Agradecimentos

Gostaria de expressar minha sincera gratidão ao Professor Edvar, meu coordenador, por seu apoio e orientação ao longo deste trabalho. Agradeço também ao Professor Bruno.

Agradeço à minha família pelo amor e incentivo constantes, que foram essenciais para minha jornada acadêmica. Por fim, agradeço à Universidade Federal de Itajubá (UNIFEI) e ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pela concessão da bolsa através do Programa Institucional de Bolsa de Iniciação Científica, que possibilitou a realização desta pesquisa.

## Referências

1. **A. Press, “Ataque hacker interrompe funcionamento de hospitais nos Estados Unidos e FBI abre investigação.” Disponível em: http://glo.bo/4exs6MJ. Acesso em: 13 de setembro de 2024, 2018.**
2. **J. Luz, “Qual é a real importância da cibersegurança?” Disponível em: https://exame.com/bussola/qual-e-a-real-importancia-da-ciberseguranca/. Acesso em: 14 de setembro de 2024, 2023.**

3. **P. A. Por Di Blasi, "IoT e 5G: Avanços e perspectivas da implementação." Disponível em: https://diblasiparente.com.br/iot-e-5g-avancos-e-perspectivas-da-implementacao/. Acesso em: 19 de agosto de 2024, 2022.**
4. **I. Mirsky, "Kitsune.py." Disponível em: https://github.com/ymirsky/Kitsune-py. Acesso em: 1 de setembro de 2024, 2020.**
5. **Y. E. Yisroel Mirsky, Tomer Doitshman e A. Shabtai, "Kitsune: An ensemble of autoencoders for online network intrusion detection," Ben-Gurion University of the Negev, 2018.**
6. **G. Bertoli, "AB-TRAP." Disponível em: https://github.com/c2dc/AB-TRAP/tree/main/1_Attack%20dataset. Acesso em: 21 de setembro de 2024, 2020.**