# Uma Heurística para Ofuscação de Tráfego em Internet das Coisas

**Francisco Thiago dos Santos Gonçalves[1], Antonio Janael Pinheiro[2], Críston Pereira de Souza[1], Jeandro de Mesquita Bezerra[1]**

[1]Universidade Federal do Ceará - Campus de Quixadá (UFC)
Quixadá, CE, Brazil

[2]Centro de Estudos e Sistemas Avançados do Recife (CESAR), Recife, PE, Brazil

franciscothiago@alu.ufc.br, ajp@cesar.org.br,

criston, jeandro{@ufc.br}

***Abstract.*** *Smart technologies are increasingly present in the population's daily lives due to the Internet of Things (IoT) advancement. Research shows that attackers can use statistics present in traffic attributes, such as packet size, to infer sensitive user information. In the literature, several padding approaches have been proposed to protect user privacy against attacks based on traffic analysis. The proposed solutions insert redundant data into the network, which can cause delays and overloads. This work presents a heuristic for filling packets to mitigate attacks based on traffic analysis that uses packet size to infer user information. To balance the trade-off between privacy and performance, the heuristic aims to minimize the amount of bytes added to packets. Compared to a similar approach, the padding method improved up to* $90\%$ *in trade-off.*

***Resumo.*** *O uso de tecnologias inteligentes está cada vez mais presente no cotidiano da população devido ao avanço da Internet das Coisas (IoT). Pesquisas demonstram que invasores podem utilizar estatísticas presentes em atributos do tráfego, como tamanho dos pacotes, para inferir informações sigilosas do usuário. Na literatura, diversas abordagens de preenchimento foram propostas para proteger a privacidade do usuário contra ataques baseados em análise de tráfego. As soluções propostas inserem dados redundantes na rede, o que pode causar atrasos e sobrecargas. Esse trabalho propõem uma heurística para preenchimento de pacotes com o objetivo de mitigar ataques baseados em análise de tráfego que utilizem o tamanho dos pacotes para inferir informações do usuário. Para balancear o trade-off entre privacidade e desempenho, a heurísitica tem o objetivo de minimizar a quantidade de bytes adicionados nos pacotes. O método de preenchimento desenvolvido obteve uma melhoria de até* $90\%$ *no trade-off comparada com a abordagem similar.*

## 1. Introdução

O avanço da indústria de semicondutores possibilitou embutir dispositivos eletrônicos, como sensores e microcontroladores, em objetos do cotidiano, logo permitindo obter e compartilhar via rede de computadores informações contextuais a respeito do ambiente

em que estão inseridos. Dessa forma, esses objetos podem oferecer experiências e funcionalidades personalizadas para cada usuário. A transmissão e o vazamento de dados relacionados a hábitos, comportamento, saúde, entre outras informações, expõem riscos em relação à privacidade dos usuários de tecnologias de *Internet of Things* (IoT) [Poh et al. 2021]. Dentre os domínios de IoT, os dispositivos de casas inteligentes destacam-se pela proximidade com o usuário para automação de tarefas e sensoriamento [Serror et al. 2018]. Porém, são mais vulneráveis a ciberataques por falta de conhecimento técnico dos usuários e atualizações de *firmware*.

Nesse contexto, protocolos seguros como *Security Socket Layer* (SSL) podem ser utilizados para garantir que agentes maliciosos não tenham acesso às informações enviadas na rede. Contudo, mesmo em um tráfego cifrado ainda é possível identificar eventos e interação desses dispositivos analisando estatísticas e padrões presentes nos pacotes enviados [Pinheiro et al. 2019, Skowron et al. 2020]. A privacidade do usuário pode ser violada mesmo que os dispositivos usem métodos de confidencialidade de dados, como criptografia. Logo, um invasor pode inferir informações sobre o usuário analisando padrões presentes nos envios realizados pelos dispositivos presentes na rede. Por exemplo, um invasor pode analisar eventos gerados por dispositivo como lâmpadas inteligentes para identificar a presença do usuário em sua residência.

A pesquisa realizada por [Pinheiro et al. 2019] demonstra ser possível diferenciar dispositivos IoT e não IoT utilizando modelos de classificação que analisam o tamanho dos pacotes, obtendo acurácias acima de 90%. O autor explora apenas ataques baseados no tamanho dos pacotes, entretanto, [Skowron et al. 2020] aponta que aspectos temporais, como intervalos de envio dos pacotes, também podem ser usados para identificar os dispositivos na rede.

Diferentes técnicas de ofuscamento foram propostas na literatura [Santos et al. 2022, Hafeez et al. 2019, Pinheiro et al. 2021] para mascarar estatísticas presentes no tráfego a fim de reduzir a taxa de acertos de modelos utilizados para classificar informações relevantes presentes no tráfego, como dispositivo de origem dos pacotes. A revisão desses trabalhos mostrou que geralmente esses métodos consistem no preenchimento dos pacotes ou envio de tráfego fictício na rede. Técnicas de preenchimento consistem em adicionar *bytes* redundantes nos pacotes transmitidos na rede, consequentemente adicionando um custo (*overhead*) que pode ocasionar atrasos ou sobrecargas.

Geralmente, aplicação de técnicas de ofuscação de tráfego ocasionam *trade-off*, isto é, a melhoria na privacidade provida ao usuário em contraste com o *overhead* adicionado no tráfego [Pinheiro et al. 2021]. Portanto, abordagens de preenchimento devem balancear o *trade-off* a fim de evitar atrasos ou transtornos ao usuário. Para isso, modelos de otimização podem ser utilizados para reduzir a quantidade total de *bytes* inseridos no tráfego, nesse contexto, a frequência de envio dos pacotes pode ser utilizada por esses modelos para que pacotes com uma maior taxa de envio recebam um menor preenchimento em comparação com pacotes com uma menor frequência de envio.

O objetivo geral deste trabalho é desenvolver e aplicar um método de preenchimento de pacotes que proteja a privacidade do usuário contra ataques baseados em análise de tráfego e que balanceie o *tradeoff* entre privacidade e desempenho. Para isso, a solução

proposta por [Pinheiro et al. 2021] é aprimorada para uma heurística de preenchimento para encontrar valores otimizados para a quantidade de *bytes* inseridos nos pacotes. Dessa forma, espera-se aperfeiçoar a segurança de usuários que utilizam tecnologias IoT em seu cotidiano. Os objetivos específicos desse objetivo são listados a seguir.

- Implementar uma heurística baseada na distribuição de probabilidade de envio dos pacotes para minimizar a quantidade de *bytes* inseridos;
- Comparar o desempenho e *overhead* gerado pelo método desenvolvido com abordagens de ofuscamento baseados em preenchimento disponíveis na literatura.

O trabalho está organizado da seguinte forma, a Seção 2 resume os principais conceitos teóricos usados pelo projeto, na Seção 3, trabalhos relacionados na literatura são analisados e comparados com a heurística proposta, a Seção 4 apresenta o modelo de otimização para minimizar o *overhead*, a Seção 5 apresenta a metodologia escolhida para implementar e avaliar a abordagem proposta. A Seção 6 apresenta a análise dos resultados e a Seção 7 apresenta as conclusões e trabalhos futuros.

## 2. Fundamentação Teórica

Métodos de aprendizado de máquina podem ser utilizados para explorar padrões característicos dos pacotes enviados pelos dispositivos presentes na rede e inferir informações sobre o usuário. Nesse contexto, métodos de ofuscamento alteram atributos do tráfego, como tamanho dos pacotes, a fim de reduzir o desempenho desses modelos. Essa seção resume os conceitos teóricos de aprendizado de máquina, classificação de tráfego e métodos de ofuscamento de tráfego.

### 2.1. Aprendizado de Máquina

Aprendizado de máquina é uma técnica de Inteligência Artificial que permite computadores, a partir de dados prévios, resolverem problemas de predição, regressão e classificação sem a necessidade de explicitamente programar uma solução para cada problema [Meng et al. 2020]. Aprendizado de máquina possui três categorias: supervisionado, não supervisionado e aprendizado por reforço, onde informações relevantes são obtidas a partir de dados rotulados e não rotulados, respectivamente [Liu and Lang 2019]. No contexto desse trabalho, aprendizado de máquina supervisionado é utilizado para identificar dispositivos presentes em uma rede IoT. A escolha dos dois modelos listados abaixo foi baseada no melhor desempenho apresentado em [Pinheiro et al. 2019].

- ***k-Nearest Neighbors* (KNN):** o algoritmo KNN analisa a diferença ou similaridade das entradas baseada na proximidade de seus atributos, calculada pela distância euclidiana, desse modo, uma instância é classificada a partir da classe com maior ocorrência nas k instâncias mais próximas a ela [Xin et al. 2018];
- ***Decision Tree* (DT):** *Decision Tree* ou Árvore de Decisão utiliza estrutura de árvore para classificar as entradas, onde cada nó representa um teste de um dos atributos, os ramos representam os resultados desse teste e as folhas representam a categoria que será atribuída a instância analisada [Xin et al. 2018].

### 2.2. Classificação de Tráfego e *Traffic Fingerprint*

O avanço das tecnologias IoT permitiu que objetos do cotidiano proporcionassem uma melhor interação com o usuário e com outros dispositivos, logo ampliando suas funcionalidades. Contudo, o uso dessas tecnologias traz preocupações em relação à privacidade,

visto que esses eletrônicos captam e compartilham informações sobre o ambiente e comportamento dos usuários, onde muitos dispositivos IoT não utilizam protocolos seguros em suas requisições. Entretanto, mesmo em uma comunicação cifrada, estatísticas características dos dispositivos continuam presentes no tráfego e podem ser utilizadas para expor informações do usuário.

Técnicas de classificação de tráfego visam categorizar o tráfego de rede em classes apropriadas [Rezaei and Liu 2019]. Classificação de tráfego é utilizada por diferentes aplicações para prover serviços como sistemas de detectores de intrusão, qualidade de serviço e *firewall* [Tahaei et al. 2020]. Nesse contexto, diferentes métodos, como inspecionamento das portas de origem dos pacotes ou seus conteúdos encapsulados, são utilizados para classificar informações referentes ao tráfego analisado.

*Traffic Fingerprint* é uma técnica de classificação de tráfego que consiste em comparar padrões presentes no tráfego com modelos de classificação já conhecidos a fim de identificar informações como endereços Web acessados, serviços utilizados e dispositivos presentes no tráfego [Skowron et al. 2020]. Algoritmos de aprendizado de máquina podem ser empregados para construir os modelos para classificar o tráfego, nesse contexto, características como tamanho dos pacotes, tempo de resposta, quantidade de envios entre outros atributos podem ser utilizados como entradas dos algoritmos de classificação.

### 2.3. Ofuscamento de Tráfego e Preenchimento de Pacotes.

A privacidade de usuários de dispositivos IoT pode ser violada por ataques baseados em técnicas de classificação de tráfego, nesse contexto, técnicas de ofuscação de tráfego visam impedir que invasores monitorem o comportamento do usuário a partir de padrões presentes nos envios dos dispositivos na rede [Shen et al. 2022]. Dessa forma, alterar características do tráfego é uma abordagem utilizada para prevenir que invasores relacionem padrões presentes na comunicação com as atividades do usuário [Mohit et al. 2021].

Pesquisas no contexto de métodos de ofuscação de tráfego focam em alterar as entradas de modelos de classificação de tráfego baseados em aprendizado de máquina de forma que reduza a acurácia desses classificadores [Perera et al. 2022], logo impedindo que invasores infiram dados sobre o usuário. Nesse contexto, as principais técnicas de ofuscamento de tráfego consistem em preenchimento de pacotes, modelagem de tráfego e injeção de tráfego [Chen et al. 2021].

- **Preenchimento de Pacotes:** essa técnica consiste em alterar o tamanho dos pacotes adicionando bytes redundantes a fim de ofuscar estatísticas no tráfego [Skowron et al. 2020];
- **Modelagem de Tráfego:** visa fazer com que estatísticas presentes no tráfego não reflitam características reais do usuário [Chen et al. 2021];
- ***Injeção de Tráfego*:** consiste em adicionar pacotes redundantes na rede de forma que um invasor monitorando o tráfego não seja capaz de distinguir os pacotes reais e os pacotes fictícios [Santos et al. 2022].

## 3. Trabalhos Relacionados

O trabalho desenvolvido por [Pinheiro et al. 2021] propõe um método de ofuscamento de tráfego baseado em preenchimento, onde dados extras são adicionados no *payload* dos pacotes com o intuito de remover padrões presentes nos tamanhos dos *frames* enviados por

diferentes dispositivos IoT. Segundo o autor, soluções similares adicionam quantidades fixas de *bytes* nos pacotes, algo problemático em um cenário de redes IoT. A solução utiliza um controlador SDN (*Software Defined Network*) executado por um ponto de acesso sem fio que recebe o tráfego IoT, um mecanismo de preenchimento e uma API *Representational State Transfer* (REST). O método possui 4 níveis de preenchimento distintos 400, 500, 700, 900 e ALL. Os níveis mais elevados adicionam uma quantidade maior de dados. O mecanismo analisa o tamanho de cada pacote e baseado no nível atual eleva a quantidade de *bytes* homogenizando o tamanho dos pacotes. O autor aplicou diferentes algoritmos de aprendizado de máquina em capturas de redes IoT antes e após o método ser aplicado e constata que houve uma redução drástica na taxa de acerto dos modelos após o preenchimento ser aplicado. O autor utiliza valores empíricos para modelar o mecanismo de preenchimento o que pode ser problemático caso seja necessário aplicar o sistema em diferentes redes.

Os autores de [Santos et al. 2022] propõem um método de ofuscamento de rede chamado Mitra. O projeto usa inserção de tráfego fictício (*Dummy Traffic*) para mitigar ataques baseados na inspeção de pacotes na rede. A proposta usa o tráfego original para modelar os pacotes fictícios adicionados na rede. O método provê quatro níveis de geração de tráfego. Os pacotes fictícios são gerados com base no tráfego real dos dispositivos, onde o MAC e o IP de destino dos pacotes são preservados, enquanto dados como IP de origem são gerados aleatoriamente. O tamanho do *frame* gerado é idêntico ao último frame real enviado. Para cada dispositivo presente na rede uma quantidade específica de pacotes falsos é gerada. Essa quantidade varia conforme o nível de ofuscamento escolhido. A abordagem utilizou o método de ofuscamento na mesma captura e compararam o impacto causado nas métricas. A quantidade de pacotes enviados na rede pelo método é modelada de forma empírica, além de não variar conforme a intensidade de tráfego na rede.

[Hafeez et al. 2019] propõem um sistema de ofuscamento que utiliza tráfego fictício para mascarar estatísticas presentes na rede. Os pacotes gerados pelo método são enviados em períodos de inatividade dos dispositivos IoT prevenindo que invasores detectem padrões no tráfego, já que os pacotes fictícios diferem pouco dos reais. A solução utiliza algoritmos de aprendizado de máquina que obtiveram uma acurácia de 87% e 86% respectivamente quando utilizados para detectar os dispositivos na rede. Para evitar ataques baseados em estatísticas, o autor propõe enviar constantemente tráfego fictício na rede independente da atividade dos dispositivos, e quando os dispositivos IoT estiverem inativos, tráfego fictício representando atividades também é inserido. Para gerar os pacotes fictícios o autor usa dados do tráfego real e aplica métodos de interpolação para escolher o tamanho dos pacotes e o tempo de envio. O trabalho identificou que dispositivos com menor frequência de envio apresentaram uma melhor acurácia mesmo após o tráfego fictício ser aplicado. O autor atribui esse fato a baixa diversidade de dispositivos IoT. O projeto usa o próprio tráfego para modelar os pacotes fictícios mantendo similaridade com os pacotes reais. Apesar do tráfego real possuir uma maior prioridade, o autor não explora nenhum método para melhorar a escolha da quantidade de tráfego fictício gerado, o que pode acarretar atrasos na comunicação dos dispositivos IoT.

A Tabela 1 compara os trabalhos apresentados com o projeto proposto. Em geral, os trabalhos apresentados definem um conjunto de níveis de ofuscamento, onde níveis

que adicionam um maior *overhead* no tráfego reduzem a taxa de acerto dos modelos de classificação utilizados para identificar os dispositivos na rede. A abordagem de ofuscamento proposta neste trabalho, utiliza preenchimento para remover estatísticas presentes nos pacotes. Como contribuição, este trabalho utiliza o modelo de otimização baseado em [Vale et al. 2006] para encontrar o tamanho do preenchimento que reduza a quantidade extra de *bytes* inseridos na rede. A abordagem utiliza 4 níveis de ofuscamento, onde o tamanho dos pacotes após o preenchimento é definido pela heurística utilizada.

| Autor | Abordagem | Metodologia | Algoritmos de Classificação Utilizados | Métricas Analisadas |
|---|---|---|---|---|
| [Pinheiro et al. 2021] | Preenchimento de pacotes | Define 4 níveis de preenchimento, onde um controlador SDN escolhe o nível utilizado conforme a intensidade do tráfego. | KNN, RF, DT e SVM | Acurácia, Recall e F1-Score |
| [Santos et al. 2022] | Inserção de Tráfego Fictício | Define 4 níveis de ofuscamento, onde cada nível adiciona uma quantidade maior de pacotes fictícios na rede. | XGBoost, CART, Random Forest e Bagging | Acurácia, Recall, Precisão e F1-Score |
| [Hafeez et al. 2019] | Inserção de Tráfego Fictício | Constantemente envia pacotes fictícios na rede e, em períodos de inatividade, pacotes simulando o tráfego real dos dispositivos presentes na rede. | RF e KNN | Precisão, Recall e F1-Score |
| Proposto | Preenchimento de pacotes | Define 4 níveis de preenchimento, onde uma heurística é utilizada para definir o tamanho do preenchimento que reduza o overhead. | KNN, RF, DT e SVM | Acurácia, Recall e F1-Score |

**Tabela 1. Comparação dos trabalhos relacionados**

## 4. Heurística de Preenchimento

A heurística desenvolvida é baseada em [Vale et al. 2006], que apresenta uma abordagem de preenchimento de pacotes voltado à comunicação de redes móveis. No contexto das *Universal Mobile Telecommunication System* (UTMS), os blocos de informações são preenchidos para que seus tamanhos encaixem em um conjunto de *frames* de tamanhos especificados pelo canal de comunicação. Neste trabalho foi adotada a nomenclatura de *level* em substituição a *frame*. A heurística define um modelo de otimização para encontrar tamanhos para esses *levels* que minimizem a quantidade de *bits* extras adicionados nos pacotes.

Sendo $\Omega = \{x_1, x_2, \ldots, x_N\}$ o conjunto com os possíveis tamanhos dos pacotes enviados em um canal de comunicação e $\underline{X} = \{X_1, X_2, \ldots, X_M\}$ o vetor contendo os tamanhos dos *frames* utilizados nesse canal, onde $M < N$ e $X_1 < X_2 < \cdots < X_M$, um pacote de tamanho $x_k$ deve ser preenchido para acomodar um *frame* de tamanho $X_m$ se e somente se $X_{m-1} < x_k \leq X_m$. A heurística visa escolher os valores de $\underline{X}$ de forma que reduza o *overhead* gerado pelo preenchimento. O custo médio de aplicar o preenchimento dos pacotes é definido como:

$$D(\underline{X}) = \sum_{m=1}^{M} X_m \cdot [F(X_m) - F(X_{m-1})], \tag{1}$$

$$F(W) = \sum_{n|x_n \leq W} \alpha_n, \tag{2}$$

onde a função $F(W)$ calcula a soma das frequências relativas dos pacotes de tamanhos menores que $W$, logo o termo $X_m[F(X_m) - F(X_{m-1})]$ é o custo médio de aplicar o preenchimento nos pacotes de tamanho entre $X_m$ e $X_{m-1}$.

A frequência relativa $a_k$ do tamanho do pacote $x_k$ é definida em (3) como a razão entre a quantidade de envios desse pacote e a quantidade total de pacotes enviados.

$$a_k = \frac{\text{Quantidade de envios do pacote } x_k}{\text{Quantidade total de pacotes enviados}} \tag{3}$$

Portanto, para encontrar o tamanho dos *levels* que reduzam o custo do preenchimento, basta encontrar os valores de $\underline{X}$ que minimizem a Equação 1. A Equação 4 resume a heurística como um modelo de otimização que pode ser utilizado por *solvers* para encontrar os valores contidos em $\underline{X}$ que minimizem o *overhead*, dessa forma $D(\underline{X})$ pode ser entendida como a função objetivo da heurística.

$$\begin{aligned} \min_{X} \quad & D = \sum_{m=1}^{M} X_m \cdot [F(X_m) - F(X_{m-1})] \\ \text{s.t} \quad & X_1 < X_2 < \cdots < X_M \end{aligned} \tag{4}$$

Após definir os valores de $\underline{X}$, o tamanho do pacote de tamanho $k$ pode ser definido a partir do Algoritmo 1, que percorre os valores contidos em $\underline{X}$ a fim de encontrar o primeiro valor $X_i$ que satisfaça $X_{i-1} < k \leq X_i$.

**Algoritmo 1:** preenchimento (k).

**Require:** o Vetor contendo o tamanho dos *levels* ($\underline{X}$) e o tamanho do pacote enviado (k).
**Ensure:** $\underline{X}$ minimiza o custo médio $D(\underline{X})$
  $M \leftarrow$ Quantidade de *frames* em $\underline{X}$.
  **for** $i \leftarrow 1, 2, \ldots, M-1$ **do**
    **if** $X_i < k \leq X_{i+1}$ **then**
      **return** $X_{i+1}$
    **end if**
  **end for**

## 5. Metodologia

Essa sessão apresenta o conjunto de passos necessários para implementar e avaliar a abordagem proposta. Além disso, é definida a métrica utilizada para calcular o *trade-off* entre privacidade e desempenho aplicado pela heurística de preenchimento.

### 5.1. Obtenção do tráfego IoT

A primeira etapa foi a obtenção dos dados de capturas de tráfego gerado por dispositivos IoT presentes em casas inteligentes que foram utilizados para treinar os modelos de classificação e testar a eficácia do método de preenchimento. Os dados utilizados são *datasets* preprocessados e disponibilizados por [Pinheiro et al. 2021], que consistem em capturas de tráfego realizadas por [Sivanathan et al. 2019] em uma rede com 28 dispositivos IoT e não IoT como câmeras inteligentes, sensores de fumaça, sensores de pressão arterial, etc. Os meta-dados contém o tamanho dos pacotes, tempo de envio e endereço IP de origem e destino. Os endereços *Media Acess Control* (MAC) de origem são utilizados

para rotular os dispositivos, onde um valor numérico distinto é atribuído a cada endereço. Além do *dataset* utilizado, o autor disponibiliza a implementação utilizada para aplicar o método de preenchimento, treinar os modelos de aprendizado de máquina e calcular o desempenho dos modelos após o método de preenchimento ser aplicado.

Nesse contexto, é utilizada a abordagem desenvolvida por [Pinheiro et al. 2021] para treinar e avaliar os modelos de aprendizado de máquina. Desse modo, os resultados obtidos são comparados com os resultados apresentados por [Pinheiro et al. 2021] para avaliar o desempenho da heurística e a quantidade de *bytes* adicional.

### 5.2. Abordagem de Preenchimento

A heurística de preenchimento é baseada em [Vale et al. 2006] a qual depende da distribuição de probabilidade do tamanho dos pacotes. Inicialmente é necessário obter os valores de $\alpha$ e $\Omega$ para o tráfego analisado. A Unidade Máxima de Transmissão (MTU) se refere ao tamanho máximo de um pacote transmitido na rede, logo $\Omega$ é definido como todos os valores menores ou iguais que a MTU no contexto do projeto. A frequência relativa $a_k$ é obtida percorrendo os arquivos CSV e armazenando a quantidade de ocorrências dos pacotes de tamanho k. As Equações 5 e 6 resumem os vetores $\Omega$ e $\alpha_k$.

$$\Omega = \{1, 2, \ldots, MTU\} \tag{5}$$

$$\alpha_k = \frac{\text{Quantidade de pacotes de tamanho k}}{\text{Quantidade total de pacotes nos } \textit{Datasets}} \tag{6}$$

Uma vez que os valores de $\Omega$ e $\alpha$ foram obtidos, é necessário obter os valores do vetor $\underline{X}$ que minimizem a função objetivo $D(\underline{X})$. A heurística foi implementada utilizando a linguagem Python e a biblioteca Scipy, onde o método Powell foi usado para obter os valores de $\underline{X}$ que minimizem o *byte overhead* gerado pelo preenchimento. A partir dos valores $\underline{X}$ é possível obter o tamanho do pacote após o preenchimento a partir do Algoritmo 1.

### 5.3. Análise de desempenho

A análise de desempenho do mecanismo de preenchimento proposto é feita aplicando os algoritmos de classificação KNN e DT após a heurística de preenchimento ser aplicada [Pinheiro et al. 2021]. As métricas analisadas são acurácia, *byte overhead* e *trade-off*.

A acurácia representa a razão entre a quantidade de acertos e quantidade de amostras analisadas, onde valores próximos a 1 indicam que os modelos de aprendizado de máquina são capazes de identificar com uma boa taxa de acerto os dispositivos IoT presentes na rede analisando o tamanho dos pacotes. Logo, o objetivo do método de preenchimento é reduzir a acurácia para valores próximos a 0.

O custo de aplicar os métodos de ofuscação no tráfego é analisado comparando o tamanho original do tráfego com o tamanho após o preenchimento ser aplicado. O *byte overhead*, na Equação 7, é calculado analisando o tamanho dos pacotes antes e após o algoritmo de preenchimento ser aplicado.

$$\text{Byte Overhead} = \frac{\text{Tamanho após preenchimento}}{\text{Tamanho original}} \tag{7}$$

A Equação 8 resume a métrica utilizada neste trabalho para calcular o *trade-off* aplicado pela abordagem de preenchimento. A equação se aproxima de 1 conforme a acurácia e o *overhead* são reduzidos, indicando um bom balanceamento entre privacidade e desempenho. Em contrapartida, um resultado próximo a 0 indica que a privacidade fornecida ao usuário, medida pela acurácia, é pequena quando comparado com o *overhead* aplicado no tráfego. Dessa forma, mesmo que o método de preenchimento reduza o êxito de um invasor monitorando o tráfego, um alto *byte overhead* reduzirá o valor da métrica, da mesma forma que um baixo *byte overhead* e uma alta acurácia fará o resultado tender a 0.

$$\textit{trade-off} = \frac{1 - \text{Acurácia}}{\textit{Byte Overhead}} \tag{8}$$

Dois cenários são analisados: invasor externo e interno. No caso de invasor externo, os algoritmos de aprendizado de máquina são treinados antes do método de preenchimento ser aplicado. Dessa maneira, é simulado um invasor que possui acesso a dispositivos IoT e pretende utilizar os modelos gerados para identificar os dispositivos presentes na rede do usuário. No cenário de invasor interno, os modelos são treinados com o tráfego após o tamanho dos pacotes serem alterados pelo preenchimento. Este cenário simula um invasor que, além dos dispositivos IoT, possui acesso ao método de preenchimento utilizado, como um provedor de acesso à Internet que analisa a atividade do usuário em sua residência.

A Figura 1 resume a abordagem de preenchimento proposta. Inicialmente *datasets* contendo capturas de tráfego IoT são utilizados para obter a frequência relativa de um pacote em função do seu tamanho, onde esses dados são utilizados pela heurística para encontrar os valores de $\underline{X}$. Por fim, os valores de $\underline{X}$ são usados pelo algoritmo de preenchimento para alterar o tamanho dos pacotes, consequentemente afetando o desempenho de modelos de classificação [1] .

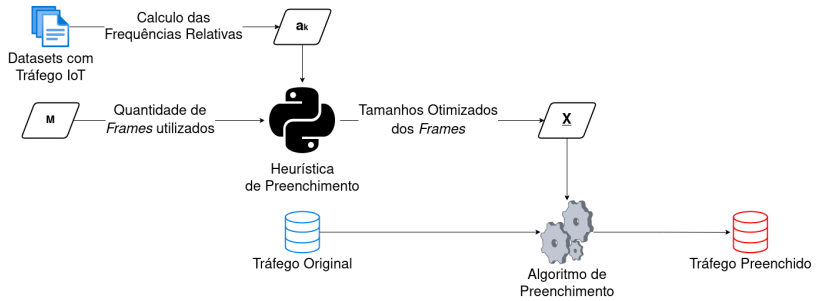



**Figura 1. Abordagem de preenchimento proposta.**

## 6. Resultados

Os resultados obtidos são analisados baseados nas métricas de acurácia, *byte-overhead* e *trade-off* dos modelos de classificação para os cenários de observador interno e externo. Além disso, baseado nas métricas, compara-se a heurística com os resultados de

[1]Implementação disponível em `https://github.com/franciscothiagoufc/Uma-Heuristica-para-Ofuscacao-de-Trafego-em-Internet-das-Coisas`

[Pinheiro et al. 2021], que propõe 4 níveis de preenchimento, *Level 100*, *Level 500*, *Level 700* e *Level 900*. A solução proposta utiliza 4 níveis de preenchimento *Level 66*, *Level 123*, *Level 124* e *Level 156*, onde o Algoritmo 1 é utilizado com 5, 4, 3 e 2 *levels* respectivamente. Os nomes dos níveis de preenchimento foram definidos utilizando o menor tamanho após o preenchimento de cada nível, com exceção do *Level 123*, pois esse teria o mesmo nome que o *Level 66*. A Tabela 2 indica o tamanho do preenchimento para cada *Level* proposto, onde os valores são obtidos a partir da frequência relativa de envio e a quantidade de *Levels* utilizados em cada nível de preenchimento.

| Nível de preenchimento | Tamanho original (k) | Tamanho após preenchimento |
|---|---|---|
| Level 66 | $1 < k \leq 66$ | 66 |
| | $66 < k \leq 123$ | 123 |
| | $123 < k \leq 235$ | 235 |
| | $235 < k \leq 543$ | 543 |
| | $543 < k \leq 1500$ | 1500 |
| Level 123 | $1 < k \leq 66$ | 66 |
| | $66 < k \leq 196$ | 196 |
| | $196 < k \leq 543$ | 543 |
| | $543 < k \leq 1500$ | 1500 |
| Level 124 | $1 < k \leq 124$ | 124 |
| | $124 < k \leq 541$ | 541 |
| | $541 < k \leq 1500$ | 1500 |
| Level 156 | $1 < k \leq 156$ | 156 |
| | $156 < k \leq 1500$ | 1500 |

**Tabela 2. Preenchimento para os 4 níveis propostos.**

### 6.1. *Byte Overhead*

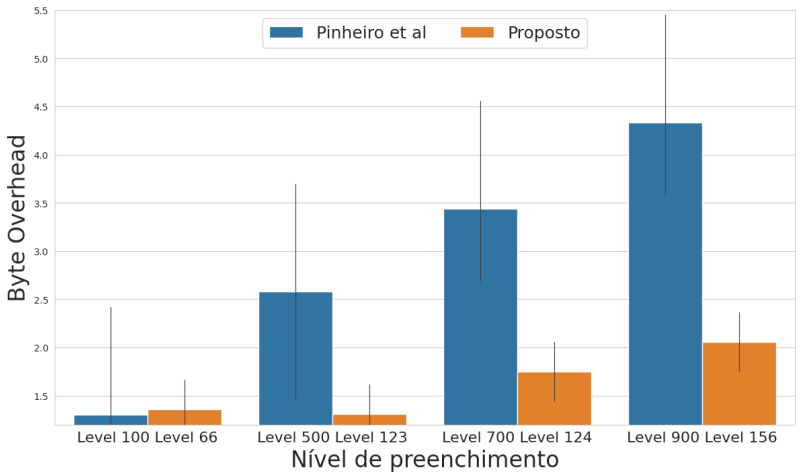
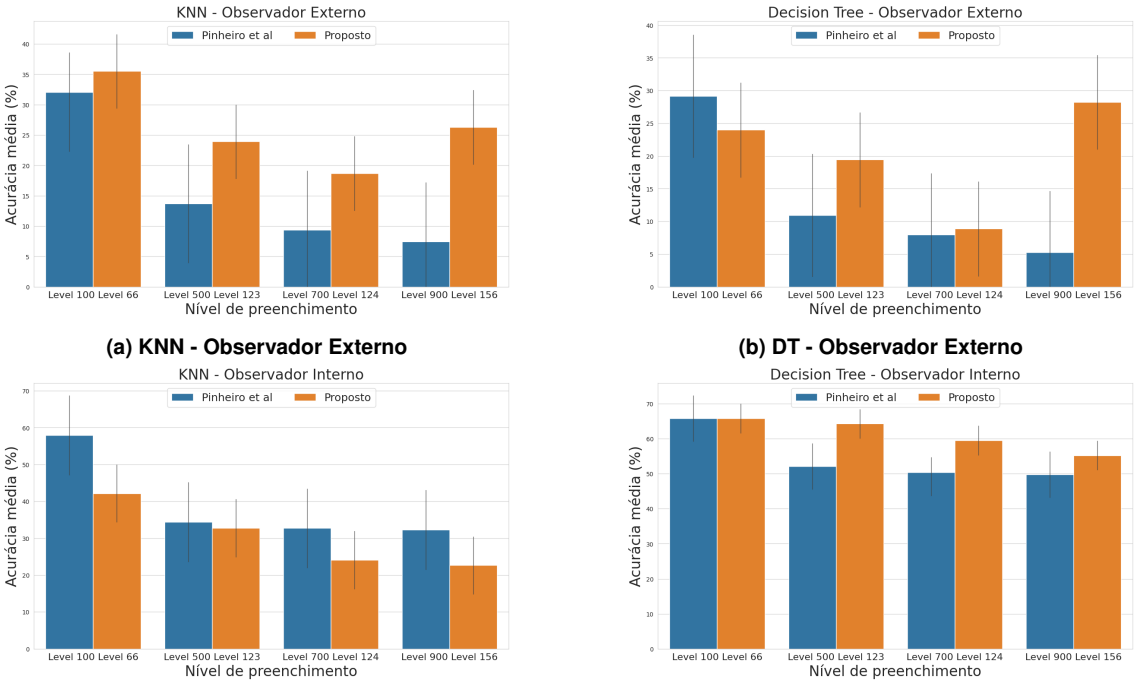



**Figura 2. *Byte overhead* dos métodos de preenchimento**

O *byte overhead* representa a razão entre o tamanho original do tráfego e o seu tamanho após a abordagem de preenchimento ser aplicada. A Figura 2 exibe a comparação do *byte overhead* gerado pelos níveis de preenchimento proposto por [Pinheiro et al. 2021] e pela heurística proposta. Observa-se que a abordagem de preenchimento proposta apresenta um custo de preenchimento menor que [Pinheiro et al. 2021], onde, no máximo, a quantidade de *bytes* após o preenchimento é 2,4 vezes maior que a quantidade original, o que reduz a chance do método de preenchimento resultar em atrasos e sobrecargas na rede.




**(a) KNN - Observador Externo** **(b) DT - Observador Externo**

**(c) KNN - Observador Interno** **(d) DT - Observador Interno**

**Figura 3. Acurácias Observador Externo e Interno**

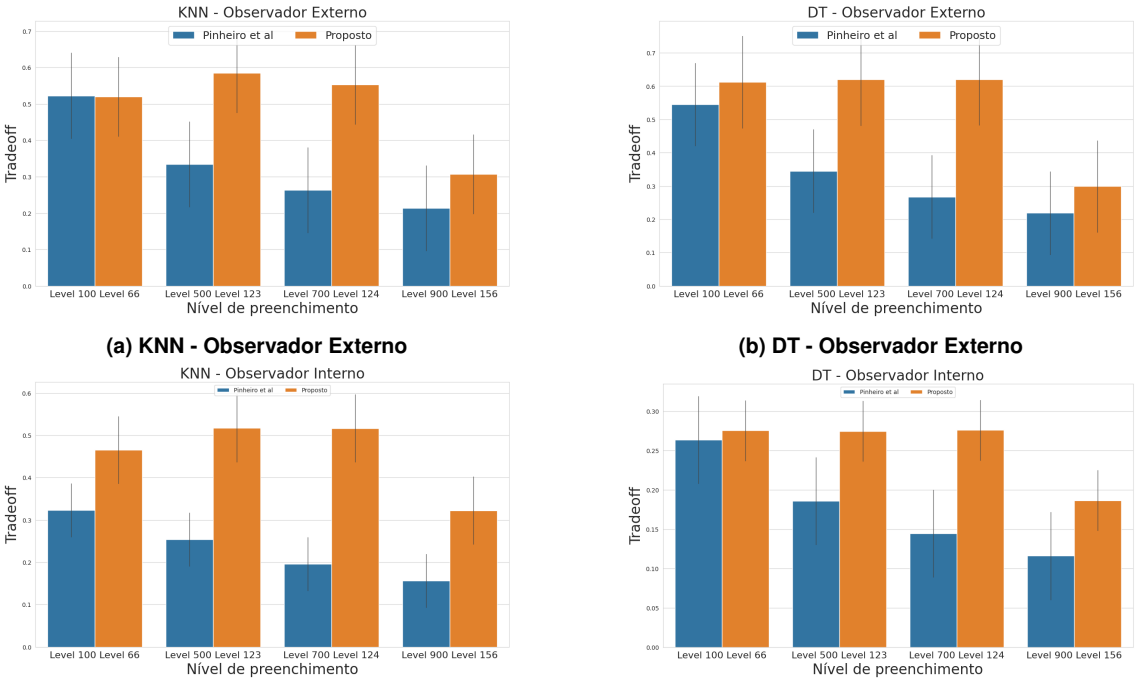
## 6.2. Identificação de Dispositivos IoT

As Figuras 3a e 3b comparam as acurácias obtidas pelos modelos de classificação após as abordagens de preenchimento serem aplicadas no cenário de um observador externo. Nos algoritmos KNN e DT, a acurácia diminui conforme o *overhead* aumenta, com exceção do nível de preenchimento *Level 156*, onde a acurácia apresentou um acréscimo mesmo sendo o nível de preenchimento que adiciona a maior quantidade de *bytes* extras. Em todos os casos, o método de preenchimento foi capaz de reduzir a acurácia dos modelos de classificação para valores abaixo de $50\%$.

As Figuras 3c e 3d ilustram as acurácias obtidas pelos modelos de classificação após aplicar as abordagens de ofuscamento em um cenário de um observador interno. Diferente do observador externo, a acurácia dos algoritmos tendem a decrescer conforme o tamanho do preenchimento aumenta em todos os níveis propostos. Todos os algoritmos utilizados obtiveram uma acurácia média inferior a $70\%$. Apesar de ser capaz de reduzir a acurácia dos modelos de classificação, a abordagem proposta neste trabalho, apresentou resultados piores que o trabalho comparado. Contudo, quando o *trade-off* entre a redução da acurácia e o *byte overhead* aplicado é levado em consideração, o método proposto apresenta melhores resultados que a abordagem comparada.

## 6.3. Análise do *Trade-off*

Essa seção analisa o *trade-off* entre privacidade, medida através da acurácia, e o desempenho, medido através do *byte overhead*, de ambas abordagens. Nesse contexto, a Equação 8 é utilizada para quantizar o *trade-off*, onde valores próximos a 1 indicam que o preenchimento conseguiu reduzir o êxito de um invasor monitorando o tráfego e minimizou a quantidade de *bytes* inseridos nos pacotes. Por outro lado, resultados próximos de zero




**(a) KNN - Observador Externo** **(b) DT - Observador Externo**

**(c) KNN - Observador Interno** **(d) DT - Observador Interno**

**Figura 4. *Trade-off* observador externo e interno.**

indicam que os modelos de classificação obtiveram uma alta acurácia ou o preenchimento adicionou um alto *overhead* no tráfego.

Reduzir a quantidade de *bytes* inseridos nos pacotes pode favorecer um invasor monitorando o tráfego de dispositivos IoT aumentando a acurácia dos modelos de classificação utilizados. Nesse contexto, analisar a acurácia e o *byte overhead* separadamente não reflete totalmente a eficácia da abordagem de preenchimento analisada.

As Figuras 4a e 4b ilustram a métrica apresentada na Equação 8 aplicada no cenário de um observador externo, enquanto as Figuras 4c e 4d ilustram os resultado em um cenário de um observador interno. Na maioria do níveis, com exceção do *Level 66* no KNN do observador externo, a abordagem de preenchimento desenvolvida apresenta um melhor *trade-off* em comparado com [Pinheiro et al. 2021]. A média dos *trade-offs* de cada nível de preenchimento é calculada e apresentada nas Tabelas 3a e 3b, onde a abordagem de preenchimento desenvolvida obteve melhores resultados quando o algoritmos KNN e DT são utilizados.

| Algoritmo | [Pinheiro et al. 2021] | Abordagem desenvolvida |
|---|---|---|
| KNN | 0.33 | 0.49 |
| DT | 0.34 | 0.53 |

**(a) Observador Externo.**

| Algoritmo | [Pinheiro et al. 2021] | Abordagem desenvolvida |
|---|---|---|
| KNN | 0.23 | 0.45 |
| DT | 0.17 | 0.25 |

**(b) Observador Interno.**

**Tabela 3. Média dos *trade-offs* das abordagens de preenchimento.**

Os resultados apresentados demonstram que abordagem desenvolvida conseguiu reduzir o desempenho de métodos de aprendizado de máquina utilizados para identificar os dispositivos presentes na rede. Ademais, quando é aplicada a métrica *trade-off* para quantificar entre privacidade e desempenho, a abordagem de preenchimento desenvolvida

apresenta melhores resultados em relação ao trabalho comparado. No cenário de um observador externo, a abordagem de preenchimento apresentou uma melhoria de $55\%$ quando o algoritmo DT é utilizado, enquanto, no cenário de um observador interno, foi possível obter uma melhoria de até $90\%$ quando o algoritmo KNN foi utilizado. Dessa forma, os resultados indicam que a heurística utilizada alcança um melhor equilíbrio entre desempenho e privacidade fornecida ao usuário.

## 7. Conclusão

O trabalho apresenta uma abordagem de ofuscamento de tráfego baseada em preenchimento de pacotes que utiliza uma heurística para balancear o *tradeof* entre privacidade e desempenho, portanto protegendo a privacidade do usuário contra invasores que utilizam análise de tráfego a fim de inferir informações sobre o usuário. Os resultados demonstram que o algoritmo de preenchimento utilizado apresenta uma redução no *byte overhead* quando comparado com outro métodos de preenchimento similar. A abordagem desenvolvida foi capaz de reduzir a acurácia dos modelos de classificação de tráfego, entretanto, apresentou valores maiores que o trabalho comparado. Contudo, quando o *trade-off* entre privacidade e desempenho é analisado, a abordagem desenvolvida apresentar resultados melhores que o trabalho comparado.

A solução apresentada depende da frequência relativa dos tamanhos dos pacotes enviados na rede para encontrar o tamanho do preenchimento que reduza o *overhead*. Contudo, redes IoT apresentam grande dinamicidade, logo as frequências de envio podem ser alteradas conforme novos dispositivos se conectam na rede, recebem atualizações, entre outros fatores. Nesse contexto, a abordagem apresentada pode não apresentar resultados otimizados em um cenário real.

Em trabalhos futuros, esse problema pode ser resolvido alterando a abordagem para reaplicar a heurística quando a frequência de envio é alterada. Dessa forma, o preenchimento aplicado nos pacotes é alterado à medida que as frequências relativas de envio são atualizadas. Por fim, espera-se que abordagem apresentada contribua no desenvolvimento de projetos no contexto de ofuscamento de tráfego que utilizam preenchimento de pacotes ou que alterem outras características dos envios realizados pelos dispositivos IoT.

## Referências


Chen, B., Liu, Y., Zhang, S., Chen, J., and Han, Z. (2021). A survey on smart home privacy data protection technology. In *2021 IEEE Sixth International Conference on Data Science in Cyberspace (DSC)*, pages 583–590.

Hafeez, I., Antikainen, M., and Tarkoma, S. (2019). Protecting iot-environments against traffic analysis attacks with traffic morphing. In *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 196–201. IEEE.

Liu, H. and Lang, B. (2019). Machine learning and deep learning methods for intrusion detection systems: A survey. *Applied Sciences*, 9(20).

Meng, T., Jing, X., Yan, Z., and Pedrycz, W. (2020). A survey on machine learning for data fusion. *Information Fusion*, 57:115–129.

Mohit, Ansari, S., and Kumar, A. (2021). Traffic privacy study on internet of things - smart home applications. In *2021 9th International Conference on Cyber and IT Service Management (CITSM)*, pages 1–6.

Perera, Y., Ahmed, N., Kanhere, S., Hu, W., and Jha, S. (2022). Iot traffic obfuscation: Will it guarantee the privacy of your smart home? In *ICC 2022 - IEEE International Conference on Communications*, pages 2954–2959.

Pinheiro, A. J., Araujo-Filho, P. F. D., Bezerra, J. D. M., and Campelo, D. R. (2021). Adaptive packet padding approach for smart home networks: A tradeoff between privacy and performance. *IEEE Internet of Things Journal*, 8:3930–3938.

Pinheiro, A. J., de M. Bezerra, J., Burgardt, C. A., and Campelo, D. R. (2019). Identifying iot devices and events based on packet length from encrypted traffic. *Computer Communications*, 144:8–17.

Poh, G. S., Gope, P., and Ning, J. (2021). Privhome: Privacy-preserving authenticated communication in smart home environment. *IEEE Transactions on Dependable and Secure Computing*, 18(3):1095–1107.

Rezaei, S. and Liu, X. (2019). Deep learning for encrypted traffic classification: An overview. *IEEE Communications Magazine*, 57(5):76–81.

Santos, B. V. d., Vergutz, A., Nogueira, M., and Macedo, R. T. (2022). Um método de ofuscaçao para proteger a privacidade no tráfego da rede iot. In *Anais do XL Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, pages 126–139. SBC.

Serror, M., Henze, M., Hack, S., Schuba, M., and Wehrle, K. (2018). Towards in-network security for smart homes. In *Proceedings of the 13th International Conference on Availability, Reliability and Security*, ARES 2018, New York, NY, USA. Association for Computing Machinery.

Shen, F., Zhang, S., Liu, Y., and Yang, Z. (2022). A survey of traffic obfuscation technology for smart home. In *2022 International Wireless Communications and Mobile Computing (IWCMC)*, pages 997–1002.

Sivanathan, A., Gharakheili, H. H., Loi, F., Radford, A., Wijenayake, C., Vishwanath, A., and Sivaraman, V. (2019). Classifying iot devices in smart environments using network traffic characteristics. *IEEE Transactions on Mobile Computing*, 18(8):1745–1759.

Skowron, M., Janicki, A., and Mazurczyk, W. (2020). Traffic fingerprinting attacks on internet of things using machine learning. *IEEE Access*, 8:20386–20400.

Tahaei, H., Afifi, F., Asemi, A., Zaki, F., and Anuar, N. B. (2020). The rise of traffic classification in iot networks: A survey. *Journal of Network and Computer Applications*, 154:102538.

Vale, E. R., B. Brandao, J. C., and Grivet, M. (2006). An algorithm for umts padding optimization. In *2006 IEEE 17th International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 1–5.

Xin, Y., Kong, L., Liu, Z., Chen, Y., Li, Y., Zhu, H., Gao, M., Hou, H., and Wang, C. (2018). Machine learning and deep learning methods for cybersecurity. *IEEE Access*, 6:35365–35381.