Pós-Graduação em Ciência da Computação

Antônio Janael Pinheiro

Preenchimento de Pacotes Adaptável em Redes de Casas Inteligentes: *Um Compromisso Entre Aprimoramento da Privacidade e Overhead de Comunicação*

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
http://cin.ufpe.br/~posgraduacao

Recife
2020

Antônio Janael Pinheiro

Preenchimento de Pacotes Adaptável em Redes de Casas Inteligentes: *Um Compromisso Entre Aprimoramento da Privacidade e Overhead de Comunicação*

Tese de Doutorado apresentada ao Programa de Pós-graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, como requisito parcial para obtenção do título de Doutor em Ciência da Computação.

Área de Concentração: Redes de Computadores e Sistemas Distribuídos
Orientador: Divanilson Rodrigo de Sousa Campelo

Recife
2020

Catalogação na fonte
Bibliotecária Monick Raquel Silvestre da S. Portes, CRB4-1217

P654p Pinheiro, Antônio Janael
Preenchimento de pacotes adaptável em redes de casas inteligentes: um compromisso entre aprimoramento da privacidade e *overhead* de comunicação / Antônio Janael Pinheiro. – 2020.
120 f.: il., fig., tab.

Orientador: Divanilson Rodrigo de Sousa Campelo.
Tese (Doutorado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2020.
Inclui referências e apêndices.

1. Redes de computadores. 2. Internet das coisas. I. Campelo, Divanilson Rodrigo de Sousa (orientador). II. Título.

004.6 CDD (23. ed.) UFPE - CCEN 2020 - 77

Antônio Janael Pinheiro

Preenchimento de Pacotes Adaptável em Redes de Casas Inteligentes: Um Compromisso Entre Aprimoramento da Privacidade e Overhead de Comunicação

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Doutor em Ciência da Computação.

Aprovado em: 02/03/2020.

_____

Orientador: Prof. Dr. Divanilson Rodrigo de Sousa Campelo

BANCA EXAMINADORA

_____

Prof. Dr. José Augusto Suruagy Monteiro
Centro de Informática /UFPE

_____

Prof. Dr. Kiev Santos da Gama
Centro de Informática /UFPE

_____

Prof. Carlos André Guimarães Ferraz
Centro de Informática / UFPE

_____

Prof. Dr. Moisés Renato Nunes Ribeiro
Departamento de Engenharia Elétrica / UFES

_____

Prof. Dr. Augusto José Venâncio Neto
Departamento de Informática e Matemática Aplicada / UFRN

Eu dedico este trabalho aos meus pais Ildevan Pinheiro e Jocilêda Pinheiro e às minhas irmãs Roberta M. Pinheiro e Vitória Pinheiro.

AGRADECIMENTOS

Agradeço à minha família o apoio e a compreensão de não me terem fisicamente próximo. Aos meus pais, Ildevan e Jocilêda, e irmãs, Vitória e Roberta, que me apoiaram, aconselharam e incentivaram em todos os momentos da minha vida. Ao meu orientador Divanilson Campelo, a inspiração, compreensão e incentivo constante. Aos colegas Caio Burgardt e Paulo Freitas, as contribuições no desenvolvimento de pesquisas, que resultaram em publicações científicas, conjuntas durante o Doutorado. Ao colega e amigo Jeandro Bezerra, o incentivo e a colaboração em trabalhos que também tornaram-se publicações. Aos meus amigos, as palavras de apoio. Aos professores do Centro de Informática da Universidade Federal de Pernambuco, a formação de uma base sólida para o desenvolvimento deste trabalho.

RESUMO

Dispositivos da *Internet* das Coisas (do inglês, *Internet of Things*, IoT), capazes de coletar informações sensíveis sobre seus usuários, introduzem inúmeras ameaças à privacidade dos proprietários de casas inteligentes (do inglês, *smart homes*). Diversas ameaças à privacidade provêm da análise de atributos do tráfego criptografado gerado por dispositivos IoT presentes em residências. Pesquisas anteriores apontaram que atributos do tráfego, como tamanho do pacote, combinados com técnicas para aprendizado de máquina, viabilizam a inferência de informações privadas dos usuários IoT, como presença em uma residência, distúrbios no sono, doenças e atividades íntimas. Uma das técnicas mais eficientes na mitigação dessas ameaças à privacidade consiste na ofuscação do tráfego, como o preenchimento (do inglês, *padding*), que expressa-se por intermédio da inserção de *bytes* extras nos pacotes a fim de ocultar os seus tamanhos originais. Geralmente, os mecanismos de preenchimento existentes selecionam o número de *bytes* inseridos nos pacotes de forma estática, que gera um *overhead* na comunicação de rede elevado ou um aprimoramento da privacidade irrisório. Esses mecanismos estáticos são particularmente inadequados para redes em que os padrões de tráfego sejam significativamente dinâmicos, como casas inteligentes. Assim, com a proliferação de equipamentos IoT presentes em residências, aparenta ser um momento propício para a condução de pesquisas sobre a viabilidade de adaptar o número de *bytes* em resposta às oscilações nas condições de uma rede doméstica. Esta tese investiga a influência do número de *bytes* no *trade-off* privacidade-*overhead*, assim como a praticabilidade de ajustar essa quantidade em resposta às variações nas condições de uma rede. Uma extensa análise do tráfego gerado por dispositivos IoT, tipicamente, presentes em residências, foi conduzida a fim de caracterizar esse tipo de equipamento quanto ao tamanho do pacote. Como resultado desta análise e de uma profunda revisão da literatura, esta tese estabelece uma solução para preenchimento de pacote com a finalidade de averiguar a viabilidade da adaptação do número de *bytes* em resposta aos padrões de tráfego proveniente de uma rede doméstica. Um conjunto de experimentos foram empregados para mensurar o aprimoramento na privacidade e o impacto no desempenho da comunicação proporcionado por esta solução de preenchimento. A solução de preenchimento é capaz de superar os mecanismos de preenchimento existentes no aprimoramento da privacidade, enquanto introduz um *overhead* significativamente menor. A solução proposta mostrou-se adequada para balancear o *trade-off* em resposta às mudanças nas condições de uma rede através da adaptação no número de *bytes* inseridos nos pacotes.

Palavras-chaves: Aprimoramento da Privacidade. Internet das Coisas. Preenchimento de Pacotes. Casas Inteligentes.

ABSTRACT

IoT devices, capable of collecting sensitive information about their users, introduce numerous threats to the privacy of smart homeowners. Several privacy threats come from analyzing the attributes of encrypted traffic generated by IoT devices present in homes. Previous research has shown that traffic attributes, such as packet length, combined with machine learning techniques, make it possible to infer private information from IoT users, such as presence in a home, sleep disturbances, illnesses, and intimate activities. One of the most efficient techniques for mitigating these privacy threats is traffic obfuscation techniques, such as padding, which is expressed through the insertion of extra bytes in packets in order to hide its original length. Generally, the existing padding mechanisms select a static number of bytes to be inserted into the packets, which generates a high network communication overhead or a negligible privacy improvement. These static mechanisms are particularly unsuitable for networks where traffic patterns are significantly dynamic, such as smart homes. Thus, with the proliferation of IoT equipment present in homes, it appears to be a favorable time for researching the feasibility of adapting the number of bytes in response to fluctuations in the conditions of a home network. This thesis investigates the influence of the number of bytes on the privacy-overhead trade-off, as well as the feasibility of adjusting this amount in response to variations in the conditions of a network. An extensive analysis of the traffic generated by IoT devices, typically present in homes, was conducted in order to characterize this type of equipment in terms of packet length. As a result of this analysis and a thorough review of the literature, this thesis establishes a solution for packet padding in order to ascertain the feasibility of adapting the number of bytes in response to traffic patterns coming from a home network. A set of experiments were used to measure the improvement in privacy and the impact on the communication performance provided by this padding solution. The padding solution can overcome the existing padding mechanisms in improving privacy while introducing significantly less overhead. The proposed solution proved to be adequate to balance the trade-off in response to changes in the conditions of a network by adapting the number of bytes inserted in the packets.

Keywords: Privacy Improvement. Internet of Things. Packet Padding. Smart Homes.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

6LowPAN	6 Low Personal Area Network
ANOVA	ANalysis Of VAriance
API	Application Programming Interface
BLE	Bluetooth Low Energy
BuFLO	Buffered Fixed-Length Obfuscation
CCPA	California Consumer Privacy Act
COPPA	Children's Online Privacy Protection Act
CPF	Cadastro de Pessoas Físicas
CPU	Central Processing Unit
CSV	Comma-Separated Values
DDoS	Distributed Denial of Service
DDR	Double Data Rate
DNS	Domain Name System
DT	Decision Tree
EDF	Empirical Distribution Function
ForCES	Forwarding and Control Element Separation
FTP	File Transfer Protocol
GAN	Generative Adversarial Net
GDPR	General Data Protection Regulation
HD	Hard Drive
HTTP	Hypertext Transfer Protocol
IEEE	Institute of Electrical and Electronic Engineers
IoT	Internet of Things
IP	Internet Protocol
IPv6	Internet Protocol version 6
ISP	Internet Service Provider
JSON	JavaScript Object Notation
k-NN	k-Nearest Neighbors
KS	Kolmogorov-Smirnov
LAN	Local Area Network

LGPD	Lei Geral de Proteção de Dados Pessoais
LTS	Long Term Support
MAC	Media Access Control
Mbps	Megabit per second
ML	Machine Learning
MLP	Multilayer Perceptron
MTU	Maximum Transmission Unit
NETCONF	Network Configuration Protocol
NFV	Network Function Virtualization
ONF	Open Networking Fundation
P4	Programming Protocol-independent Packet Processors
PCAP	Packet Capture
PMCMRplus	Pairwise Multiple Comparisons of Mean Rank
RAM	Random Access Memory
REST	REpresentational State Transfer
RF	Random Forest
SDN	Software-Defined Networking
SNMP	Simple Network Management Protocol
SSH	Secure Shell
SVM	Support Vector Machine
TCP	Transmission Control Protocol
Tor	The Onion Router
UDP	User Datagram Protocol
URL	Uniform Resource Locator
VM	Virtual Machine
VoIP	Voice over Internet Protocol
VPN	Virtual Private Network
WAN	Wide Area Network
WLAN	Wireless Local Area Network
WSGI	Web Server Gateway Interface
WSRT	Wilcoxon Signed-Rank Test
WTF-AD	Website Traffic Fingerprinting Protection with Adaptive Defense
XML	eXtensible Markup Language

SUMÁRIO

1 INTRODUÇÃO

Este capítulo apresenta uma introdução à privacidade para a *Internet* das Coisas (do inglês, *Internet of Things (IoT)*), destacando as ameaças produzidas pela classificação de metadados do tráfego de rede e as respectivas contramedidas, como preenchimento de pacotes (do inglês, *packet padding*). Nesta tese, o termo pacote é empregado como uma referência à unidade de dados, independente da camada de rede do modelo *Transmission Control Protocol (TCP)/Internet Protocol (IP)*. Também são apresentadas as motivações para a condução desta pesquisa, os objetivos e as contribuições do presente trabalho ao estado da arte. Por fim, é apresentada a estrutura geral desta tese.

1.1 CONTEXTO

A IoT compreende uma variedade de dispositivos com capacidades para sensoriamento, atuação no ambiente e comunicação em rede. Esses dispositivos estão presentes em diversos domínios, como casas e cidades inteligentes, veículos, indústria e hospitais. Diferentes domínios são compostos por dispositivos com requisitos distintos quanto à segurança, confidencialidade das informações coletadas e desempenho da comunicação. Por exemplo, dispositivos hospitalares coletam informações biométricas que contribuem para determinar o estado de saúde dos indivíduos, enquanto em residências são capturados dados sobre os hábitos domésticos, assim como o estado de saúde.

A casa é um dos domínios IoT mais relevantes, devido à sua proximidade com os indivíduos e a proporcionar melhorias na qualidade de vida de seus usuários ao automatizar tarefas domésticas (MARIKYAN; PAPAGIANNIDIS; ALAMANOS, 2019). Os dispositivos mais comuns em casas inteligentes (do inglês, *smart homes*) são assistentes pessoais (ex.: *Amazon Echo*, *Apple HomePod* e *Google Home*), câmeras de segurança, lâmpadas e tomadas inteligentes (Nguyen-An et al., 2020). Existe uma discussão em relação à categorização de *smartphones* como "*things*". Alguns autores defendem que *smartphones* são componentes da IoT, enquanto outros argumentam que esses equipamentos não pertencem a essa categoria de dispositivos. Esta tese assume que *smartphones* não compõem a IoT (SIVANATHAN et al., 2018).

Os dispositivos IoT de casas inteligentes (do inglês, *Smart Home IoT*, SH-IoT) estão mais suscetíveis a ataques cibernéticos que os seus pares em outros domínios (SERROR et al., 2018), visto que os seus usuários nem sempre possuem o conhecimento técnico necessário ou disposição para mantê-los seguros e atualizados (SERROR et al., 2018). Tipicamente, dispositivos SH-IoT possuem diversas falhas de segurança, como vulnerabilidades conhecidas, senhas padrões (*admin*/*admin*) e a impossibilidade de atualizar os seus *firmwares*.

1.2 MOTIVAÇÃO E JUSTIFICATIVA

A reivindicação por privacidade pode ser observada em diversas das sociedades, desde primitivas até as modernas (WESTIN, 1968). Privacidade é um conceito complexo com diferentes significados em distintas áreas do conhecimento humano. Neste trabalho, é adotada a definição concebida pelo professor Alan Westin, um pioneiro neste ramo de pesquisa: "privacidade é a reivindicação de indivíduos, grupos ou instituições para determinar por si mesmos quando, como e em que medida as informações sobre eles são comunicadas a outras pessoas" (WESTIN, 1968, p. 24).

Diferente de outros locais, como indústria, cidades e hospitais, em que os indivíduos estão cientes da possibilidade de estarem sob vigilância, a residência é percebida como um local seguro contra violações à intimidade individual (WESTIN, 1968). Em anos recentes, com o advento das casas inteligentes, o papel de uma residência na preservação da privacidade de um indivíduo tornou-se profundamente ameaçado (APTHORPE et al., 2017), visto que as atividades privadas dos residentes podem ser expostas para terceiros não autorizados.

A inserção de dispositivos IoT em uma residência constitui uma relevante fonte de ameaças à privacidade dos residentes, como a captura e o compartilhamento de informações sensíveis, e a ausência de controle de acesso aos dados capturados (HE et al., 2018). Diversas soluções foram apresentadas para lidar com essas ameaças, como mecanismos para controle de acesso (BELTRAN; SKARMETA, 2018), *frameworks* para definir quais informações podem ser capturadas e criptografia para proteção de dados (HE et al., 2018). Contudo, uma relevante fonte de ameaças à privacidade permanece pouco explorada: classificação de metadados do tráfego IoT (LIU; ZHANG; FANG, 2018), como o tamanho do pacote, o intervalo entre *frames* e a taxa de transmissão dos dados.

Padrões no tamanho do pacote característicos a determinadas atividades permitem inferir as mesmas a partir dos dispositivos SH-IoT, como a presença de indivíduos em uma residência, equipamentos ligados ou desligados, distúrbios no sono, problemas de saúde, atividades sexuais, práticas extraconjugais e preferências no entretenimento (APTHORPE et al., 2017; LIU; ZHANG; FANG, 2018). Em (PINHEIRO et al., 2019), foi demonstrado que é possível identificar comandos de voz para um assistente pessoal virtual (*Amazon Echo Dot*), interação entre este equipamento e um interruptor elétrico, movimentos no campo de visão de uma câmera de segurança e áudio próximo a este dispositivo. As informações mencionadas anteriormente podem ser usadas para inferir atividades domésticas privadas. Por exemplo, a partir da taxa de transmissão de um monitor de sono, um observador, como o *Internet Service Provider (ISP)*, é capaz de inferir quando uma pessoa deitou-se para dormir (APTHORPE et al., 2017).

A exposição dos dispositivos presentes em uma residência representa uma ameaça à privacidade individual, pois a partir do modelo de um equipamento é possível inferir as ações do mesmo ao interagir com os seus usuários (YAO et al., 2019). Esta tese assume que

observadores (indivíduos, agências estatais, corporações, o ISP, etc.) capturam o tráfego originado em uma casa inteligente após os dados atravessarem o roteador doméstico localizado na borda da rede. Caso uma solução para aprimorar a privacidade dos usuários de dispositivos IoT seja fornecida por terceiros, os fornecedores também representam uma ameaça. Essa coleta de dados é conduzida com o objetivo de inferir informações privadas dos proprietários de casas inteligentes, que podem ser usadas em chantagem e extorsão, o que interfere na autonomia individual, um dos aspectos fundamentais da privacidade (WESTIN, 1968). Se exposta sem a devida autorização dos indivíduos, informações inferidas a partir dos dispositivos IoT podem resultar em constrangimento público, humilhação e até demissão (Poh; Gope; Ning, 2019). Portanto, é essencial evitar que informações pessoais privadas sejam expostas a partir dos dispositivos SH-IoT.

A reivindicação por privacidade varia entre usuários de casas inteligentes. Inúmeros indivíduos demonstram preocupações em relação às informações capturadas por dispositivos SH-IoT (ZHENG et al., 2018; ZENG; MARE; ROESNER, 2017; KRAEMER, 2018). Alguns indivíduos desejam que determinadas atividades domésticas não sejam monitoradas, como caminhar de trajes íntimos, alimentação, preparação de alimentos e assistir a televisão (CHOE et al., 2011). Por outro lado, vários usuários de casas inteligentes admitem sacrificar a sua privacidade desde que tal renúncia evite a degradação no desempenho das aplicações IoT (TABASSUM; KOSINSKI; LIPFORD, 2019; SHIN; PARK; LEE, 2018).

Acatando a definição de privacidade adotada neste trabalho, a demanda dos indivíduos sobre a sua privacidade é soberana. Assim, um usuário de uma casa inteligente pode sacrificar a sua privacidade para evitar um possível prejuízo no desempenho das aplicações IoT. Uma solução eficiente deve ser capaz realizar o balanceamento entre privacidade e desempenho respeitando as preferências dos seus usuários, que podem variar ao longo do tempo.

1.3 PROBLEMÁTICA

O tráfego de rede expõe inúmeros metadados, como tamanho do pacote, intervalo entre *frames*, direção e duração do fluxo, além de estatísticas, como média, mediana, moda e desvio padrão. Pesquisas anteriores apontam que existem até 853 metadados (MAITI et al., 2017). A partir da análise de padrões presentes no tamanho do pacote IoT criptografado e técnicas para aprendizado de máquina, como o *Random Forest*, é possível identificar os dispositivos e as atividades cotidianas de seus usuários, como distúrbios no sono e atividades sexuais (LIU; ZHANG; FANG, 2018). Tipicamente, técnicas para aprendizado de máquina são utilizadas em conjunto com metadados para viabilizar a classificação do tráfego de rede criptografado. A classificação de tráfego gera inúmeras ameaças à privacidade, que motivaram o desenvolvimento de soluções para mitigar tais riscos, como ofuscar os valores dos metadados (LIU; ZHANG; FANG, 2018).

Cada metadado demanda uma técnica distinta para ocultá-lo. Ofuscar todos os metadados existentes é praticamente inviável, pois demandaria muito tempo e recursos computacionais, o que pode prejudicar o desempenho das aplicações IoT e a experiência dos usuários (LIU; ZHANG; FANG, 2018). Por isso, é fundamental ocultar os metadados que introduzem as ameaças mais significativas à privacidade. Os metadados mais utilizados em classificação de tráfego são o tamanho do pacote e o intervalo entre *frames*. Intervalos de tempo são sensíveis às condições de rede, como o congestionamento. Por isso, esta pesquisa lida somente com o tamanho, pois é mais adequado que o intervalo para a classificação do tráfego (HAJJAR; KHALIFE; DíAZ-VERDEJO, 2015). É oportuno pontuar que, ocultar um ou mais metadados não garante a completa preservação da privacidade, visto que outros atributos e estatísticas podem ser explorados na obtenção de informações sobre os indivíduos.

Apesar de contribuir para o aprimoramento da privacidade, a ofuscação apresenta algumas limitações, como *overhead* elevado e consumo de recursos computacionais para modificar os metadados. Essas restrições devem ser ponderadas na escolha da técnica de ofuscação usada na IoT (APTHORPE et al., 2019). Para ocultar o tamanho do pacote, existem três técnicas principais: preenchimento, fragmentação e *dummy packets*, que têm como vantagem aprimorar a privacidade, mas cada uma das alternativas citadas apresenta desvantagens particulares.

A técnica *dummy packets* é usada para gerar pacotes quando o tráfego originado nos dispositivos está ausente da rede (FEGHHI; LEITH, 2019; APTHORPE et al., 2019). A fragmentação e o preenchimento são oportunos em cenários opostos: o primeiro para fragmentar pacotes grandes (ex.: acima de 1000 *bytes*), enquanto o segundo é adequado para incrementar tamanhos pequenos (ex.: abaixo de 100 *bytes*) (IACOVAZZI; BAIOCCHI, 2014). Dessa forma, a escolha por fragmentação ou preenchimento depende da distribuição do tamanho do pacote que deve ser ofuscado.

Os pacotes extras gerados por soluções de *dummy packets* aumentam a quantidade de dados na rede, que pode prejudicar o desempenho das aplicações IoT (CHADDAD et al., 2018; APTHORPE et al., 2019). Dado que esses dispositivos permanecem longos períodos de tempo inativos (REN et al., 2019; SIVANATHAN et al., 2018), faz-se necessário gerar grandes volumes de *dummy packets* para preencher essas lacunas temporais. Para reduzir o impacto no desempenho da comunicação foram apresentados mecanismos para ajustar a quantidade de pacotes de acordo com as oscilações no volume de dados (APTHORPE et al., 2019). Quando o volume de tráfego aumenta, a quantidade de *dummy packets* é reduzida.

A fragmentação consiste em dividir unidades de dados, como *frames* e pacotes, em frações menores. Por exemplo, fragmentar um pacote com tamanho de 840 *bytes* em três unidades de 280 *bytes*. A principal desvantagem da fragmentação consiste em elevar a quantidade de pacotes a serem processados pelos dispositivos intermediários, como *switches* e roteadores. Espera-se que o *Internet Protocol version 6 (IPv6)* e o *6 Low*

Personal Area Network (6LowPAN) sejam adotados no endereçamento dos dispositivos IoT. Esses conjuntos de protocolos exigem que a fragmentação seja realizada na origem, ou seja, dispositivo IoT (DEERING; HINDEN, 2017; KUSHALNAGAR; MONTENEGRO; SCHUMACHER, 2007). Uma vulnerabilidade no 6LowPAN relacionada à fragmentação torna os dispositivos suscetíveis a ataques cibernéticos, que pode resultar na interrupção do tráfego legítimo e esgotamento dos recursos computacionais do equipamento alvo (ARSHAD et al., 2018).

O preenchimento consiste em incluir *bytes* extras nos pacotes/*frames* a fim de alterar o tamanho dos mesmos. Como técnica para aprimorar a privacidade, o preenchimento tem como propósito uniformizar o tamanho do pacote gerado por diferentes aplicações de rede. A principal limitação desta técnica reside em sua razão de existir: incrementar o tamanho do pacote. Ao elevar o tamanho, o preenchimento aumenta o volume de tráfego na rede, que pode prejudicar o desempenho da comunicação. A combinação de fragmentação e preenchimento apresenta mais desvantagens que vantagens (IACOVAZZI; BAIOCCHI, 2014). Fundamentalmente, ao utilizar preenchimento ou outra técnica de ofuscação, submete-se ao *trade-off* entre privacidade e *overhead*. Para obter maior privacidade faz-se necessário admitir algum grau de degradação no desempenho da comunicação devido ao *overhead* gerado. Por isso, o balanceamento entre privacidade e *overhead* é um aspecto fundamental de soluções para preenchimento de pacotes (CIFTCIOGLU et al., 2018).

Inúmeros trabalhos apontam que o tráfego IoT é composto majoritariamente por pacotes com tamanho próximo a 100 *bytes* (SIVANATHAN et al., 2018; KUSHALNAGAR; MONTENEGRO; SCHUMACHER, 2007). Nesta pesquisa, foi conduzida uma extensa análise do tráfego IoT disponibilizado pelos autores de (SIVANATHAN et al., 2018; REN et al., 2019), que compreende dados provenientes das principais categorias de equipamentos domésticos mencionadas anteriormente. Do tráfego analisado, foi identificado que 32,77% dos pacotes têm tamanhos inferiores a 100 *bytes*. Visto que uma fração significativa dos tamanhos gerados por dispositivos IoT são pequenos (próximos a 100 *bytes*), o preenchimento é a técnica de ofuscação mais adequada para aviltar o desempenho de mecanismos para aprendizado de máquina. Como os dados analisados são provenientes de três ambientes experimentais, os resultados dessa análise sugerem que os tamanhos podem variar significativamente entre um ambiente e outro.

O tráfego de uma rede doméstica é altamente dinâmico, sendo afetado por diversos fatores, como quantidade e tipo de dispositivos presentes em uma residência, número de residentes, parcela de indivíduos interagindo com os dispositivos e períodos do dia (REN et al., 2019). Além disso, a inserção e remoção de dispositivos IoT em residências é contínua, o que também influencia nas variações no volume de tráfego de uma rede doméstica. Teoricamente, um mesmo dispositivo pode produzir diferentes padrões de tráfego para usuários distintos, dependendo das interações com os indivíduos.

Em períodos nos quais a residência está vazia, o tráfego é reduzido para basicamente dados de *background* transmitidos entre dispositivo e seu fabricante (Hafeez; Antikainen;

Tarkoma, 2019), além de atualizações de *software* (SIVANATHAN et al., 2018). Monitorando o tráfego da residência é possível saber como esses dados se comportam. Soluções estáticas de preenchimento dificilmente serão adequadas para casas inteligentes, pois as mesmas introduzem um elevado *overhead* na comunicação ou aprimoram a privacidade de forma irrisória. As motivações expostas na seção anterior e a problemática descrita na presente seção motivaram a investigação do balanceamento do *trade-off* entre a privacidade e o *overhead* de acordo com as oscilações no tráfego de uma rede doméstica.

1.4 BALANCEAMENTO DO *TRADE-OFF* ENTRE A PRIVACIDADE E O *OVERHEAD*

Empregar preenchimento de pacotes para aprimorar a privacidade vem sendo explorado há algum tempo, inclusive para proteger os usuários de casas inteligentes contra a análise de tráfego (XIONG; SARWATE; MANDAYAM, 2018; UDDIN; NADEEM; NUKAVARAPU, 2019; PINHEIRO; BEZERRA; CAMPELO, 2018). A influência da quantidade de *bytes* no *trade-off* privacidade-*overhead* foi analisada superficialmente nos contextos de páginas *web* e aplicações de rede. Como mencionado anteriormente, existem técnicas para ofuscação de tráfego baseadas em *adaptive padding* que incorporam o *status* de uma rede na tomada de decisão sobre a modificação dos dados. Apesar de sua aparente similaridade com esta pesquisa, essas técnicas diferem do presente trabalho em um ponto crucial, as mesmas estão dispensadas de lidar com o *trade-off* privacidade-*overhead*. Esse tipo de técnica é capaz de elevar o nível de privacidade, ao mesmo tempo que reduz o *overhead*. Por outro lado, o preenchimento de pacotes impõe a escolha por privacidade ou desempenho da comunicação. Por isso, a presente pesquisa investiga como priorizar o aprimoramento na privacidade ou a redução no *overhead*, visto que alcançar ambos ao mesmo tempo é potencialmente impraticável. Até onde sabemos, esta tese consiste no primeiro trabalho a investigar o preenchimento de pacote adaptável às oscilações na utilização de uma rede doméstica.

A principal inovação deste trabalho em relação ao estado da arte sobre preenchimento de pacotes consiste em balancear o *trade-off* entre privacidade e *overhead* de acordo com o nível de utilização de uma rede. Considerando a privacidade individual, esse *trade-off* constitui um aspecto desejável do preenchimento de pacotes, visto que o mesmo possibilita a liberdade de escolha, conceito fundamental para alcançar a privacidade (WESTIN, 1968). A proposta desta tese de balancear o *trade-off* a partir das oscilações no tráfego independe do domínio de rede. Entretanto, considerando as crescentes ameaças à privacidade geradas pela presença de SH-IoT, esta pesquisa optou por investigar a viabilidade da proposta desta tese em redes domésticas.

Apesar dos esforços em apontar a influência do número de *bytes* no *trade-off* mencionado previamente, os resultados alcançados em pesquisas anteriores permanecem inconclusivos, visto que os mesmos foram obtidos através da comparação entre estratégias significativa-

mente distintas, como linear[1] e *Maximum Transmission Unit (MTU)*[2]. Considerando a influência do número de *bytes* no *trade-off* privacidade-*overhead*, este trabalho apresenta uma solução de preenchimento que busca balancear entre privacidade e *overhead* de acordo com o nível de utilização de uma rede doméstica.

O desenvolvimento desta pesquisa requer: 1) a análise do tráfego gerado por dispositivos SH-IoT; 2) identificar características do tráfego que justifiquem o uso de preenchimento; 3) investigar a interferência da quantidade de *bytes* no *trade-off* privacidade-*overhead*; 4) analisar o ajuste no número de *bytes* inseridos nos pacotes em resposta às variações na utilização da rede doméstica.

Para demonstrar a viabilidade de equilibrar o *trade-off* privacidade-*overhead* com base no *status* de uma rede, foi concebida uma solução de preenchimento composta pelos seguintes componentes: 1) estratégia de preenchimento que especifica o número de *bytes* inseridos nos pacotes; 2) mecanismo de preenchimento, algoritmo implementado nos dispositivos na borda da rede, como roteadores e *gateways*, responsável por modificar os tamanhos de acordo com a especificação da estratégia de preenchimento; 3) uma aplicação de rede, entidade incumbida de coletar as informações sobre a situação da rede e gerenciar o mecanismo de preenchimento; 4) uma *Application Programming Interface (API) REpresentational State Transfer (REST)*, exposta pelo mecanismo de preenchimento, que viabiliza a comunicação com a aplicação.

Um desafio relevante em soluções de preenchimento consiste na remoção dos *bytes* inseridos nos pacotes realizada pelo destino do tráfego. A solução apresentada nesta tese sugere que os *bytes* extras sejam inseridos entre o *payload* da camada de rede e o final do *frame* da camada de enlace. Dessa forma, o destino do tráfego não precisa ser modificado para remover os *bytes* extras, que não são transferidos para as aplicações (JANKOWSKI; MAZURCZYK; SZCZYPIORSKI, 2013).

As informações sobre a situação da rede são usadas pela aplicação para instruir o mecanismo de preenchimento. Essa aplicação utiliza a interface REST para enviar as instruções ao mecanismo. O mecanismo de preenchimento emprega a estratégia apresentada nesta tese para determinar a quantidade de *bytes* a serem inseridos no pacote. Ao ser implementado na borda da rede, o mecanismo apresentado é capaz de alterar todo o tráfego originado em uma residência (APTHORPE et al., 2019). Devido à sua capacidade de alterar atributos (nesse caso, o tamanho do pacote) do tráfego de rede, esse mecanismo pode ser categorizado como um *middlebox*[3] (CARPENTER; BRIM, 2002). A solução de preenchimento apresentada não lida com outros elementos classificados como *middlebox*, como *firewall* e *gateway*. Esse mecanismo de preenchimento busca modificar somente o tamanho do

[1] O tamanho é elevado para o seu múltiplo de 128 mais próximo

[2] Todos os pacotes passam a ter o tamanho igual a 1500 *bytes*.

[3] Classe de elementos, implementados em *hardware* ou *software*, intermediários entre a origem e o destino dos dados, que processa o tráfego que o atravessa, exceto *switches* e roteadores. Por exemplo, *firewall*, *gateway* e *proxy*.

pacote gerado por dispositivos IoT. Esse intento pode ser alcançado com o auxilio de um mecanismo capaz de distinguir entre os dispositivos IoT e demais equipamentos conectados a uma rede doméstica (PINHEIRO et al., 2019).

1.5 HIPÓTESE E QUESTÕES DE PESQUISA

Hipótese desta pesquisa: o *trade-off* privacidade-*overhead* pode ser *dinamicamente* balanceado ao incorporar informações sobre a utilização de uma rede. Questões de pesquisa desta tese:

- Q1: como adaptar a quantidade de *bytes* inseridos nos pacotes contribui para balancear o *trade-off* entre privacidade e *overhead* em resposta às variações na utilização da rede?
- Q2: de que forma os padrões no tráfego originado em casas inteligentes influenciam nas decisões sobre a quantidade de *bytes* inseridos nos pacotes por mecanismos de preenchimento?
- Q3: é viável introduzir uma solução de preenchimento sem alterar os dispositivos SH-IoT?

1.6 OBJETIVOS GERAL E ESPECÍFICOS

O objetivo geral desta tese reside em apresentar uma solução de preenchimento adaptável capaz de balancear o *trade-off* privacidade-*overhead* através do ajuste no número de *bytes* a serem inseridos em pacotes de uma rede doméstica IoT, dirigido por sua taxa de utilização. Além do objetivo geral, esta tese tem alguns objetivos específicos, apresentados a seguir:

- Analisar o tráfego SH-IoT com relação ao tamanho do pacote, gerado por dispositivos provenientes das principais categorias de equipamentos presentes em residências;
- Estabelecer os requisitos para uma estratégia de preenchimento projetada para ofuscar o tamanho do pacote gerado por dispositivos SH-IoT;
- Descrever uma estratégia capaz de atender às preferências dos usuários IoT sobre o *trade-off*.

1.7 PRINCIPAIS CONTRIBUIÇÕES

As principais contribuições desta tese são apresentadas a seguir:

- Apresentação de uma solução de preenchimento capaz de balancear o *trade-off* entre privacidade e *overhead* de acordo com a utilização de uma rede doméstica;

- Exposição de que a quantidade de *bytes* necessária para inserção nos pacotes tem relação com a distribuição do tamanho gerado por dispositivos conectados a uma rede doméstica.
- Apresentação de uma estratégia que respeita as preferências dos usuários sobre o *trade-off* privacidade-*overhead*, permitindo priorizar a privacidade ou *overhead*, além de viabilizar alterações nessas preferências ao longo do tempo;
- Avaliação da influência do número de *bytes* no *trade-off* entre a privacidade e o *overhead*, considerando o preenchimento de pacote como a única técnica de ofuscação adotada para preservar a privacidade;
- Viabilização do gerenciamento do *overhead* introduzido pelo preenchimento de pacote, permitindo que o número de *bytes* possa ser decrementado a fim de evitar exaurir os recursos de uma rede.

1.8 METODOLOGIA

Esta tese foi desenvolvida seguindo uma abordagem metodológica baseada nas sugestões apresentadas em (EASTERBROOK et al., 2008). As etapas do processo metodológico adotado nesta tese são descritas a seguir:

- Investigação do problema: a IoT abrange diversos domínios, constituídos por dispositivos com diferentes requisitos de desempenho e privacidade. Por isso, é fundamental definir um domínio a ser explorado no desenvolvimento da solução. Nesta pesquisa, optou-se por casas inteligentes, um domínio IoT com vasto potencial de ameaças à privacidade individual. Assim como domínios IoT, metadados do tráfego de rede também são diversos. O tamanho do pacote foi selecionado para ser ocultado, devido à sua relevância neste contexto de classificação do tráfego, como justificado na Introdução;
- Definição do modelo de ameaça: esta tese assume que um observador é capaz de capturar o tráfego gerado em uma residência, após os dados atravessarem o roteador de borda, e empregar modelos para aprendizado de máquina na identificação dos dispositivos e eventos IoT a partir do tamanho do pacote. Técnicas e algoritmos para aprendizado de máquina representam um grande desafio à privacidade, visto que estes mecanismos são empregados na inferência de informações privadas. Assim, esta tese considera o uso desses algoritmos por observadores de rede a ameaça que deve ser mitigada;
- Definição do modelo de funcionamento da solução de preenchimento: elaboração de um modelo de interação entre a aplicação de rede e o mecanismo que modifica os tamanhos;

- Elaboração de uma estratégia de preenchimento: projetar uma estratégia que permita ao usuário priorizar a privacidade ou desempenho da comunicação. Essa estratégia também precisa viabilizar o ajuste no número de *bytes* inseridos nos pacotes;
- Especificação de uma interface para comunicação entre *middlebox* e a aplicação: definir o modelo de funcionamento de uma interface para transmissão de comandos entre esses componentes da solução descrita neste capítulo;
- Implementação de um protótipo: desenvolvimento de um protótipo de preenchimento adaptativo que utilize a solução como referência;
- Avaliação: planejamento e execução de um conjunto de experimentos para avaliar o protótipo. Esses experimentos foram projetados para responder as questões de pesquisa;
- Analisar e apresentar os resultados: analisar e extrair *insights* dos resultados que contribuam para responder as questões de pesquisa formuladas. É essencial divulgar os resultados da avaliação através de artigos publicados em conferências e periódicos prestigiados, pois esta etapa é fundamental na pesquisa científica.

1.9 ESTRUTURA DA TESE

A estrutura desta tese é descrita a seguir:

- **Capítulo 1**: introduz o contexto, motivações, problemática, contribuições, objetivos e questões de pesquisa;
- **Capítulo 2**: apresenta os conceitos teóricos necessários à compreensão desta tese, além das principais ferramentas utilizadas em sua implementação e avaliação, e procedimentos para análise dos resultados obtidos nos experimentos;
- **Capítulo 3**: descreve as justificativas para as escolhas realizadas na concepção da solução baseada na proposta desta tese. O protótipo implementado com base nesta solução também é apresentado neste capítulo;
- **Capítulo 4**: apresenta os procedimentos adotados no projeto e execução dos experimentos empregados na avaliação do protótipo;
- **Capítulo 5**: apresenta a discussão e a análise dos resultados obtidos nos experimentos;

- **Capítulo 6**: apresenta a revisão da literatura e as contribuições ao estado da arte sobre ofuscação de tráfego e preenchimento de pacotes;

- **Capítulo 7**: apresenta as conclusões para esta pesquisa. Além disso, este capítulo discute as contribuições do presente trabalho ao estado da arte e os trabalhos futuros.

2 REFERENCIAL TEÓRICO

Este capítulo descreve os principais conceitos abordados nesta tese. Primeiro foi descrito o tema principal desta pesquisa, privacidade individual, em seguida são abordados os temas *Internet* das Coisas, aprendizado de máquina, classificação de tráfego, preenchimento de pacotes e redes definidas por *software*.

2.1 PRIVACIDADE INDIVIDUAL

A reivindicação dos indivíduos por privacidade existe desde os primórdios da humanidade (WESTIN, 1968). Diferentemente da percepção difundida em anos recentes, a demanda por privacidade não surgiu nos últimos anos (ou décadas) e consiste em um tema muito mais amplo e complexo que proteção de dados. É possível que mesmo dados protegidos sejam aproveitados para violar a privacidade dos indivíduos. Esse cenário pode ocorrer quando dados pessoais são coletados sem autorização, ainda que a proteção das informações coletadas seja garantida.

A privacidade exerce uma influência fundamental em diversos aspectos da vida de um indivíduo, como liberdade pessoal, desenvolvimento da sua personalidade e autonomia (WESTIN, 1968). Um indivíduo somente logrará de privacidade quando dispuser de liberdade para determinar com quem e quais das suas informações pessoais são compartilhadas (WESTIN, 1968). Aspectos da privacidade, como isolamento, anonimização e círculos de intimidade, são primordiais para o desenvolvimento da personalidade individual (WESTIN, 1968). A anonimização proporciona a liberdade para os usuários expressarem-se de forma espontânea. A autonomia assegura que um indivíduo está livre do controle de outrem (WESTIN, 1968).

A reivindicação dos indivíduos por privacidade induziu o desenvolvimento de legislações específicas sobre este tema. A primeira publicação jurídica sobre a preservação da privacidade foi elaborada por Samuel Warren e Louis Brandeis (WARREN; BRANDEIS, 1890), na qual os autores discutem a evolução das legislações em direção à proteção de aspectos imateriais dos indivíduos, como sentimentos e intelecto. Também são discutidas ameaças à privacidade geradas por fotografias, como a captura de imagens sem o consentimento das pessoas retratadas nas mesmas. Normas e regulações efetivas sobre proteção de dados pessoais foram aprovadas somente em anos recentes. Por exemplo, *General Data Protection Regulation (GDPR)*, Lei Geral de Proteção de Dados Pessoais (LGPD), *California Consumer Privacy Act (CCPA)* e *Children's Online Privacy Protection Act (COPPA)*.

As informações, coletadas e transmitidas por meios eletrônicos, podem identificar um indivíduo de três maneiras: 1) constituir uma relação de autoria; 2) ser descritiva; 3) representar um identificador único. Primeiro, um indivíduo pode produzir a informação,

como na comunicação por meios eletrônicos e interação com dispositivos IoT. Segundo, os dados podem descrever um estado permanente ou transitório do indivíduo, como sua data de nascimento ou condição de saúde. Terceiro, uma informação pode viabilizar a identificação de uma pessoa, a exemplo do Cadastro de Pessoas Físicas (CPF) ou número do cartão de crédito (KANG, 1998).

Quando inexiste vínculo entre uma informação e um indivíduo, a mesma não refere-se a dados pessoais e, de acordo com a descrição acima, inviabiliza a violação à privacidade (KANG, 1998). A redação da LGPD, no momento de escrita desta tese, estabelece que informações desassociadas de pessoas naturais não constituem uma possível ameaça à privacidade. Por exemplo, dados ofuscados ou criptografados são desvinculadas de indivíduos específicos. Logo, essas informações não representam um risco à privacidade dos seus titulares.

A coleta de informações pessoais pode ocorrer de diversas formas, como o usuário preencher um formulário, interação com conteúdos em redes sociais, buscas na Internet e captura de dados por meio de sensores. Este último meio de coleta de dados é um dos mais preocupantes em termos de privacidade, pois o usuário sequer precisa estar ciente sobre a captura de suas informações. Inúmeras informações podem ser coletadas através de sensores, como dados biométricos, temperatura, presença em um ambiente e registros médicos. Geralmente, esses sensores equipam dispositivos que compõem a IoT.

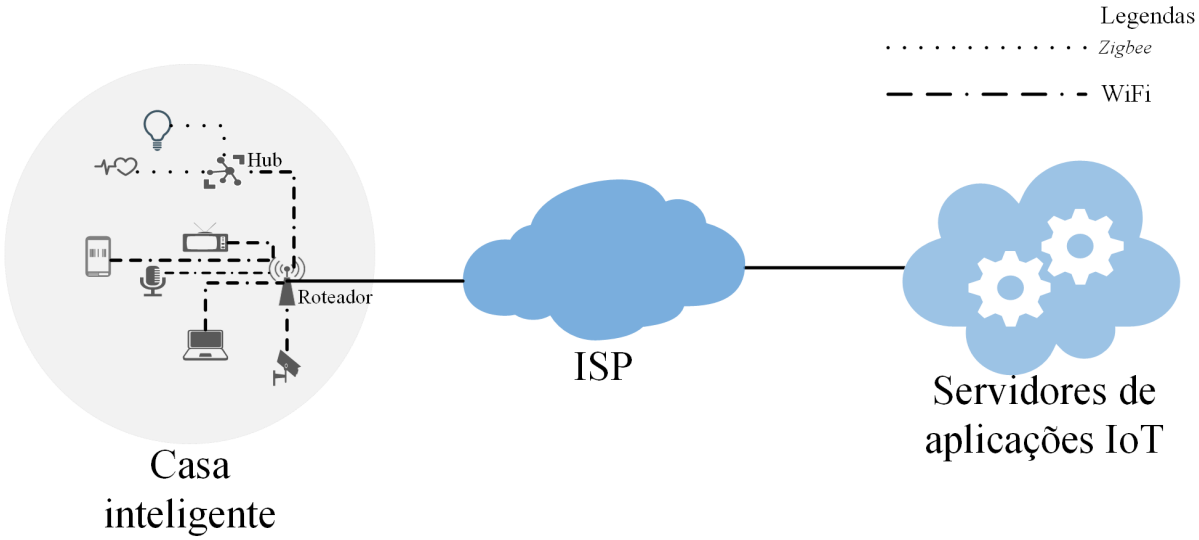
2.2 DISPOSITIVOS SH-IOT

A IoT tem como premissa básica o conceito da onipresença de objetos, equipados com atuadores e sensores, capazes de cooperarem entre si para alcançarem objetivos comuns (ATZORI; IERA; MORABITO, 2010; Poh; Gope; Ning, 2019). A IoT é uma arquitetura de informação baseada na Internet que descomplica o intercâmbio de bens e serviços (WEBER; WEBER, 2010). A IoT tem como propósito suplantar a lacuna entre objetos físicos e suas representações em sistemas de informação. Os sensores desempenham um papel fundamental para superar essa lacuna. Mudanças ambientais são monitoradas por sensores, enquanto os atuadores permite que os objetos lidem com essas variações (WEBER; WEBER, 2010).

A IoT exerce uma influência significativa no cotidiano de seus usuários, pois esses objetos estão presentes em vários domínios do dia-a-dia, como veículos, indústria, saúde, casas e cidades (Poh; Gope; Ning, 2019). A residência é um dos principais domínios da IoT, devido à conveniência proporcionada aos residentes, assegurada pela automação de tarefas domésticas (Poh; Gope; Ning, 2019). Apesar da variedade de dispositivos e sistemas, a IoT é composta essencialmente por equipamentos com poder computacional restrito, equipados com sensores, atuadores e capacidade de transmissão de dados em rede.

Tipicamente, os dispositivos SH-IoT transmitem dados para seus fabricantes e terceiros através do roteador doméstico, que realiza a interligação desses equipamentos com a

Figura 1 – Arquitetura típica de uma casa inteligente.



Fonte: elaborada pelo autor.

Internet (Poh; Gope; Ning, 2019). Geralmente, dispositivos SH-IoT desfrutam de uma mobilidade reduzida e mantêm-se conectados ao mesmo roteador por longos períodos. Alguns dispositivos IoT conectam-se diretamente ao roteador, por meio de enlaces sem fio ou cabeados, enquanto outros requerem a conexão com um *hub*. Este *hub* é responsável pela conversão de protocolos como *ZigBee* e *Bluetooth Low Energy (BLE)* em padrões Wi-Fi, compreendidos pelo roteador doméstico (Yang et al., 2019). A Figura 1 ilustra a arquitetura típica de uma casa inteligente. As principais entidades desse domínio IoT são descritas a seguir (Poh; Gope; Ning, 2019):

- Dispositivos IoT: coletam dados de seus usuários e automatizam afazeres domésticos;
- *Hub* IoT: interpreta protocolos particulares a equipamentos IoT em padrões Wi-Fi, compreendidos pelo roteador doméstico. Este dispositivo atua como um *proxy* entre equipamentos IoT e o roteador;
- Roteador: transmite os dados coletados pelos dispositivos aos provedores de serviços e vice-versa;
- ISP: aprovisiona a conexão entre os dispositivos presentes em uma residência e os serviços acessíveis através da Internet;
- Provedor de aplicações IoT: fornece os serviços aos usuários dos dispositivos, como processar requisições de um assistente pessoal, armazenar o vídeo de uma câmera e garantir acesso remoto.

O termo casa inteligente foi empregado nesta tese como uma referência a residências em que dispositivos IoT estão presentes, como assistentes pessoais virtuais, câmeras, detectores

de movimento e lâmpadas. Este trabalho ignora o nível de automação da residência e a quantidade de equipamentos, pois cada dispositivo é capaz de expor atividades domésticas dos seus usuários.

2.3 APRENDIZADO DE MÁQUINA

Aprendizado de máquina (do inglês, *Machine Learning (ML)*) é uma área da inteligência artificial incumbida de projetar sistemas capazes de aprender a partir dos dados. A ML contribui para construção de modelos matemáticos capazes de conduzir inferências a partir de amostras dos dados analisados (ALPAYDIN, 2014). A ML auxilia o desenvolvimento de técnicas para visão computacional, reconhecimento de linguagem natural, robótica, dentre outros.

As duas principais categorias do aprendizado de máquina são supervisionado (classificação e regressão) e não supervisionado (clusterização). Na abordagem supervisionada, os dados analisados são divididos em conjuntos de treinamento e teste (geralmente, na proporção de 75% e 25%, respectivamente). Na fase de treinamento, o conjunto de dados contém rótulos que indicam a classe a qual cada amostra pertence. Na fase de teste, os algoritmos buscam determinar os rótulos (que estão ausentes) das amostras (DUDA; HART; STORK, 2012). Em aprendizado não supervisionado, algoritmos criam *clusters*/grupos a partir de padrões presentes nos dados reconhecidos automaticamente.

Treinar um algoritmo de aprendizado de máquina consiste em alimentar o mesmo com dados que contêm atributos e rótulos para que este determine a relação entre os atributos e os rótulos. Esses atributos representam características das classes simbolizadas pelos rótulos. Por exemplo, os atributos podem ser estatísticas do tamanho gerado por dispositivos (classes) como câmeras, tomadas e lâmpadas representados por rótulos, como 0, 1 e 2 (esses algoritmos exigem valores numéricos como rótulos). Esses algoritmos são testados através da inserção de novos dados com os mesmos atributos fornecidos no treinamento (com valores distintos) mas sem os rótulos. O objetivo desta fase consiste em avaliar a capacidade de um modelo identificar nos dados de teste as classes observadas durante o treinamento.

Existem diversas formas de treinar um algoritmo, como considerar os k vizinhos mais próximos a cada amostra de treinamento, modelos baseados em árvores de decisão e técnicas que combinam múltiplos algoritmos, como *k-Nearest Neighbors (k-NN)*, *Decision Tree (DT)* e *Random Forest (RF)*, respectivamente. Os algoritmos para aprendizado supervisionado utilizados neste trabalho, selecionados devido à sua capacidade de alcançar um desempenho considerável a partir de um número diminuto de atributos do tráfego e demandar um consumo moderado dos recursos computacionais, são descritos brevemente a seguir:

- *k-Nearest Neighbors*: computa a distância euclidiana de cada instância de teste para

os k vizinhos mais próximos presentes no conjunto de treinamento (DUDA; HART; STORK, 2012);

- *Decision Tree*: concebe um modelo matemático com base em estrutura de árvore, em que cada nó representa uma verificação dos dados, cada ramificação constitui o resultado e as folhas são os rótulos contidos nas amostras dos dados analisados (ALPAYDIN, 2014);
- *Random Forest*: modelo constituído por um conjunto de classificadores em estrutura de árvore (BREIMAN, 2001).

Existem inúmeras métricas para quantificar o desempenho de um algoritmo para aprendizado supervisionado (classificador). A escolha dessas métricas depende do contexto em que a classificação é aplicada (ALPAYDIN, 2014). Na classificação do tráfego de rede, as métricas mais utilizadas são acurácia, *recall* e F1-*score* (Rezaei; Liu, 2019). Como a acurácia pode ser uma métrica tendenciosa, o *recall* e a F1-*score* contribuem para evitar possíveis vieses nos resultados. Essas métricas são descritas a seguir:

- Acurácia: porcentagem de amostras corretamente classificadas do total de instâncias no conjunto de teste analisadas;
- *Recall*: porcentagem de amostras dos dados de teste identificadas corretamente;
- F1-*score*: média harmônica da precisão[1] e *recall*.

Enquanto um modelo supervisionado excessivamente complexo pode apresentar um desempenho excepcional na fase de treinamento, é improvável que esses resultados permaneçam durante os testes, processo conhecido como *overfitting* (ALPAYDIN, 2014). Por isso, é fundamental empregar técnicas de avaliação que evitem o *overfitting*. A *cross-validation* é uma técnica que contribui para reduzir a probabilidade de ocorrer *overfitting* (ALPAYDIN, 2014). Com a k^2*-fold cross-validation*, o conjunto de dados é dividido aleatoriamente em k partes, em que todas contêm aproximadamente a mesma quantidade de amostras. Para constituir cada par de treino/teste, uma das k partes é selecionada como conjunto de avaliação e as demais são combinadas para formar os dados de treinamento. Esse procedimento é repetido k vezes (ALPAYDIN, 2014).

2.4 CLASSIFICAÇÃO DO TRÁFEGO DE REDE COM APRENDIZADO DE MÁQUINA

A classificação do tráfego consiste em uma atividade importante ao gerenciamento de uma rede, visto que possibilita o desenvolvimento de uma vasta gama de soluções. A classificação

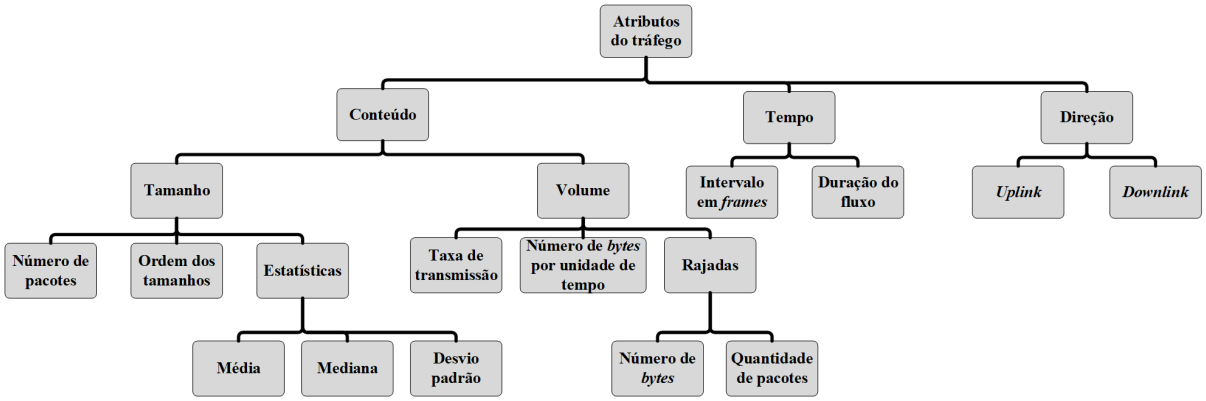
[1] porcentagem de instâncias que realmente pertencem à classe X dentre todas aquelas classificadas como X.

[2] Geralmente, 10 é o valor de k (ALPAYDIN, 2014).

do tráfego é aplicável a diversos cenários, como detecção de intrusão, identificar ataques para negação de serviço e automatizar a realocação dos recursos de uma rede (Rezaei; Liu, 2019). Inicialmente, a classificação do tráfego de rede foi baseada em portas "*well-known*" – da camada de transporte – e na inspeção da carga útil dos pacotes. Entretanto, o surgimento de aplicações que empregam portas dinâmicas e a disseminação de tráfego criptografado tornou esse modelo de classificação obsoleto. A comunidade de pesquisa em redes direcionou seus esforços para técnicas de classificação independentes de portas ou da carga útil dos pacotes (Rezaei; Liu, 2019). Dessa forma, emergiram inúmeros mecanismos capazes de superar a criptografia e alterações nas portas usadas por serviços conhecidos (Rezaei; Liu, 2019; Aceto et al., 2019).

A crescente utilização de criptografia para proteger o conteúdo dos dados prejudica a classificação do tráfego (Rezaei; Liu, 2019). Porém, metadados do tráfego, como tamanho do pacote, intervalo entre *frames*, taxa de transmissão, direção (*uplink* e *downlink*) e duração do fluxo de dados, são usados em conjunto com técnicas para aprendizado de máquina na classificação do tráfego (Rezaei; Liu, 2019; UDDIN; NADEEM; NUKAVARAPU, 2019). Metadados do tráfego não são ocultados por criptografia (Rezaei; Liu, 2019). A Figura 2 apresenta alguns dos principais atributos do tráfego usados por mecanismos de classificação projetados com base em aprendizado de máquina.

Figura 2 – Alguns dos principais atributos do tráfego de rede, assim como as estatísticas computadas a partir dos mesmos.



Fonte: elaborada pelo autor.

Geralmente, para caracterizar cada classe (dispositivos, aplicações, *websites*), são selecionados um conjunto de atributos do tráfego de rede, extraídos a partir dos metadados, como estatísticas média, desvio padrão, total de *bytes*, mínimo e máximo para o tamanho do pacote. Os atributos escolhidos dependem do cenário analisado, como identificar páginas *web*, aplicações de rede, aplicativos instalados em *smartphones* e dispositivos presentes em uma rede doméstica (Aceto et al., 2019; SIVANATHAN et al., 2018). A quantidade de atributos selecionados varia entre contextos e mecanismos, mas o ideal é minimizar esse valor (DUDA; HART; STORK, 2012). Na classificação do tráfego, os atributos são extraídos e

aproveitados para treinar e testar algoritmos para aprendizado de máquina supervisionado (Rezaei; Liu, 2019).

2.5 OFUSCAÇÃO DE TRÁFEGO ATRAVÉS DE PREENCHIMENTO DE PACOTES

Existem inúmeras soluções para preservar a privacidade individual em tráfego de rede, sendo a criptografia a mais popular (MAZURCZYK et al., 2016). Enquanto a criptografia oculta o conteúdo dos dados transmitidos em uma rede, atributos do tráfego podem revelar informações sensíveis sobre a comunicação em curso (MAZURCZYK et al., 2016). Por exemplo, trabalhos anteriores mostram que é possível identificar páginas *web* (LIBERATORE; LEVINE, 2006), *streaming* de vídeos (SCHUSTER; SHMATIKOV; TROMER, 2017) e aplicativos em *smartphones* (TAYLOR et al., 2018), através da análise do tamanho do pacote. Para mitigar essas ameaças à privacidade individual, inúmeros mecanismos usam o preenchimento para modificar o tamanho.

O preenchimento consiste em adicionar *bytes* extras ao pacote para eliminar padrões expostos por este atributo. Geralmente, são adicionados somente *bits* 0 a todos os pacotes do tráfego (JANKOWSKI; MAZURCZYK; SZCZYPIORSKI, 2013). A quantidade de *bytes* inseridos depende da estratégia utilizada. As principais estratégias de preenchimento existentes são descritas a seguir:

- Linear: o tamanho é elevado para o seu múltiplo de 128 mais próximo;
- Exponencial: incrementa o pacote até a potência de dois mais próxima;
- Ratos/Elefantes: os tamanhos são incrementados para 100 ou 1500 *bytes* de acordo com um limiar. Valores inferiores a 100 são incrementados para 100 *bytes*, enquanto os tamanhos superiores são elevados para 1500 *bytes*;
- *Random*: seja M a MTU e L o tamanho original, o número de *bytes* é selecionado aleatoriamente entre 1 e $M - L$;
- MTU: todos os pacotes passam a ter o tamanho igual à MTU;
- *Random* 255: em todos os pacotes é inserida uma quantidade selecionada aleatoriamente entre 1 e 255 *bytes*.

Cada uma dessas estratégias tem suas vantagens e desvantagens. Além disso, todas são fonte de *overhead* na comunicação de rede, o que representa o custo do preenchimento. Contudo, o preenchimento tem a vantagem de contribuir para a proteção da privacidade, representando uma solução relevante quando a criptografia é insuficiente. Originalmente, o preenchimento de pacotes foi empregado para garantir o tamanho mínimo das unidades de dados de alguns protocolos de rede, como *Ethernet* e IP. Posteriormente, esse procedimento foi empregado para alterar o tamanho do pacote como forma de eliminar padrões existentes

neste atributo. O preenchimento pode ser realizado nas camadas de aplicação, transporte, rede e enlace. O preenchimento na camada de enlace permite a modificação do tamanho de pacotes em que *payload* está criptografado.

O preenchimento fim-a-fim, em que os *bytes* são inseridos na origem e removidos no destino, é um procedimento complexo, pois requer a coordenação de entidades distribuídas entre diferentes localizações para remover os dados extras em destinações variadas. Esse procedimento pode ser realizado através de soluções como *Virtual Private Network (VPN)* e protocolos específicos para a transmissão de pacotes modificados por mecanismos de preenchimento, como o protocolo *obfs* (SHAHBAR; ZINCIR-HEYWOOD, 2015). Inserir *bytes* extras nos pacotes é um procedimento relativamente simples; a real complexidade reside em determinar o número de *bytes* a serem inseridos (RESCORLA, 2018). Essa complexidade está relacionada majoritariamente com o fato de que um excesso de *bytes* produz um *overhead* exacerbado, enquanto um número diminuto de *bytes* apresenta um aprimoramento da privacidade irrisório (UDDIN; NADEEM; NUKAVARAPU, 2019).

A quantidade de *bytes* inseridos nos pacotes é definida por uma estratégia de preenchimento. As estratégias mencionadas acima modificam os tamanhos de forma arbitrária e ignoram as características dos pacotes que devem ofuscar. As estratégias mais eficientes em aprimorar a privacidade são MTU e *random* (MAZURCZYK et al., 2016), enquanto as que geram menos *overhead* são linear, exponencial e *random* 255. A solução MTU produz o maior nível de privacidade, mas introduz um *overhead* excessivo, que interfere no desempenho da comunicação das aplicações de rede. Esse *overhead* é decorrente da elevação dos tamanhos para o máximo possível (MTU), que aumenta o tráfego na rede de forma demasiada.

Tipicamente, a estratégia *random* gera menos *overhead* que a MTU, mas permite que os pacotes sejam incrementados até a MTU. As estratégias linear, exponencial e *random* 255 produzem menos *overhead* que a MTU e *random* porque elevam pouco o tamanho do pacote, o que se traduz em um aprimoramento na privacidade pífio (DYER et al., 2012). Outra importante limitação das estratégias existentes consiste em sua natureza estática, que inviabiliza a reação aos eventos e oscilações no tráfego de uma rede. Essas estratégias modificam o tamanho continuamente da mesma forma, independente da situação da rede subjacente. Uma estratégia de preenchimento eficiente deve viabilizar a adaptação da quantidade de *bytes* em resposta às variações no volume de tráfego de uma rede (UDDIN; NADEEM; NUKAVARAPU, 2019).

Diferentemente do que argumentam os autores que apontam a estratégia MTU como a mais eficiente, incrementar o tamanho do pacote para o valor máximo permitido é desnecessário, mesmo quando o objetivo consiste em maximizar o aprimoramento na privacidade (SIBY et al., 2019). Como argumentado pelos autores de (SIBY et al., 2019), é possível obter um aprimoramento na privacidade comparável à MTU, desde que a estratégia de preenchimento seja concebida considerando características do tamanho que

se pretende ofuscar.

Os autores dos trabalhos à disposição na literatura argumentam que o preenchimento é eficiente por eliminar os padrões existentes no tamanho do pacote. Contudo, essa argumentação está parcialmente precisa. Visto que mecanismos para aprendizado de máquina buscam generalizar para dados não observados no conjunto de treinamento, somente eliminar os padrões é insuficiente, sobretudo quando os valores ofuscados são mantidos próximos aos tamanhos originais (DYER et al., 2012). Além de eliminar os padrões existentes, é fundamental evitar produzir novos padrões que possam ser explorados por observadores (CIFTCIOGLU et al., 2018). Por isso, modificar os tamanhos através de uma estratégia simples, como incrementar o tamanho em 100 *bytes*, mostra-se pouco frutífero, visto que pode gerar novos padrões. Selecionar os tamanhos aleatoriamente possibilita a criação de novos padrões. Empregar valores constantes aparenta ser a alternativa mais apropriada para ocultar padrões no tamanho (SIBY et al., 2019). Entretanto, é necessário distanciar os valores ofuscados dos tamanhos originais para reduzir o desempenho dos modelos para aprendizado de máquina. Também constitui uma decisão propícia padronizar o tamanho do pacote gerado por diferentes aplicações.

Soluções para preenchimento de pacotes têm como propósito preservar a privacidade dos indivíduos contra análise de tráfego conduzida por observadores de rede. Esses observadores podem constituir-se de diversas formas, como governos autoritários, agências governamentais de estados democráticos, corporações e indivíduos. Os seus objetivos com o monitoramento do tráfego de terceiros também pode divergir bastante, como bloquear comunicações específicas, identificar padrões de consumo e obter informações sigilosas para extorsão. Por isso, é essencial definir o modelo de ameaça que uma solução de preenchimento busca mitigar. Esse modelo é constituído pelas capacidades e objetivos de um adversário (DIXON; RISTENPART; SHRIMPTON, 2016).

2.6 REDES DEFINIDAS POR *SOFTWARE*

A principal característica das redes definidas por *software* consiste na implementação dos planos de controle e dados em dispositivos distintos. O plano de controle é responsável por gerenciar a rede, enquanto o plano de dados administra o encaminhamento do tráfego (KREUTZ et al., 2015). Na arquitetura SDN, o controle da rede é realizado por um *software* logicamente centralizado chamado de controlador. As funcionalidades de uma rede são implementadas em aplicações, como encaminhamento do tráfego, balanceamento de carga, descoberta de topologia e gerenciamento de políticas, que executam no topo do controlador SDN. Os elementos de encaminhamento, como *switches* e roteadores, são programáveis e têm seus comportamentos definidos pelo controlador.

Na arquitetura SDN, os planos de controle e dados comunicam-se por meio de interfaces, *southbound* para instruções entre controlador e *switches* e *northbound* para interação entre aplicações e controlador. O *OpenFlow*, um protocolo *open source* especificado pela *Open Networking Fundation (ONF)*, é uma interface *southbound* muito popular para gerenciar os elementos no plano de dados de uma rede SDN. O *OpenFlow* habilita *switches Ethernet* a serem remotamente configurados sem expor detalhes de sua implementação. Dessa forma, os fabricantes de *switches* podem permitir que *softwares* de terceiros gerenciem seus dispositivos. Além do *OpenFlow*, existem outras tecnologias SDN para o gerenciamento dos elementos de encaminhamento, como o *Forwarding and Control Element Separation (ForCES)*, *Simple Network Management Protocol (SNMP)* e *Network Configuration Protocol (NETCONF)* (HALEPLIDIS et al., 2015).

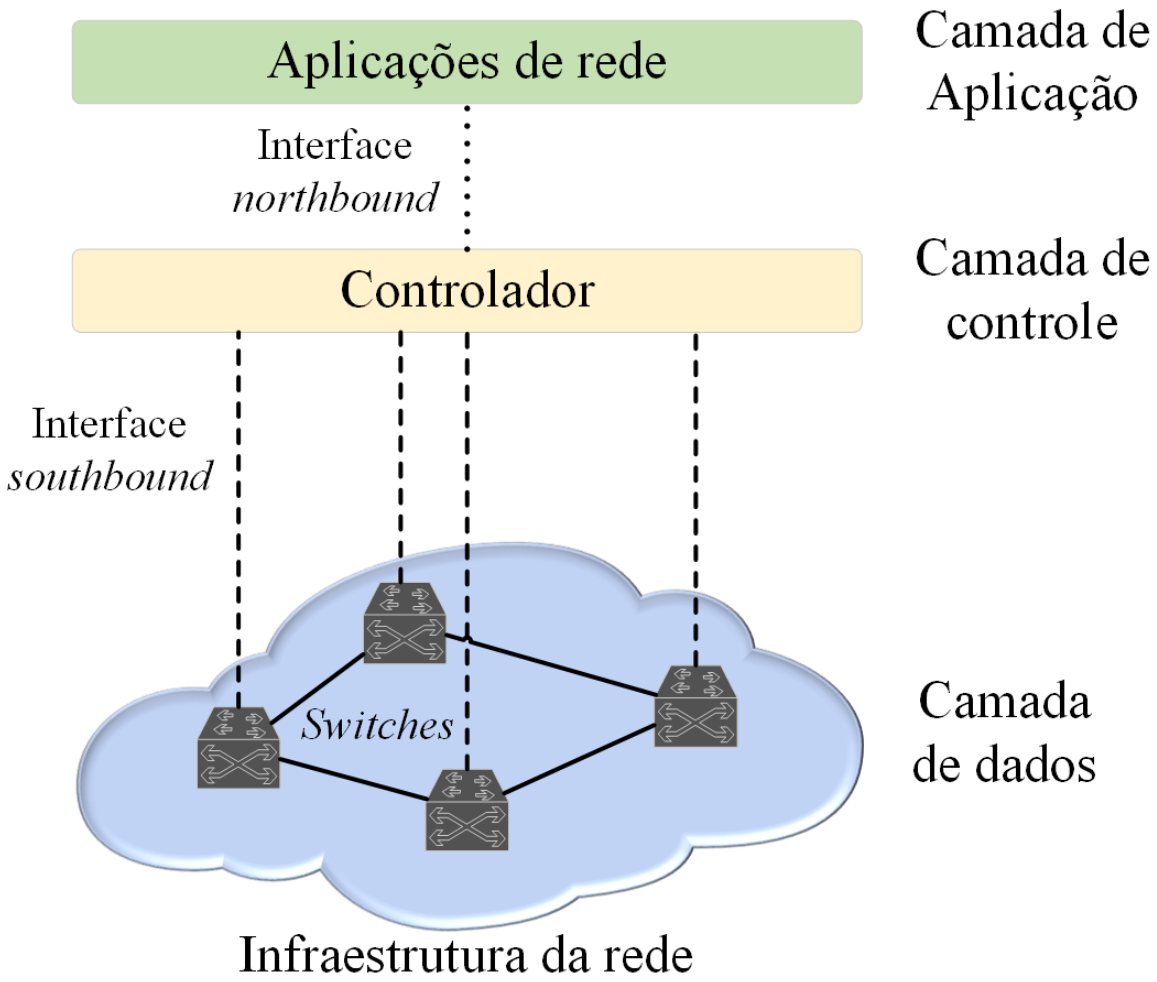
Como um protocolo de baixo nível, utilizado para manipular a memória dos *switches*, o *OpenFlow* não apresenta características favoráveis ao desenvolvimento de aplicações de alto nível. O controlador deve abstrair a complexidade apresentada pelo *OpenFlow*, apresentando às aplicações um modelo mais simples da rede. Por isso, o controlador implementa a API *northbound* de alto nível, utilizada pelos desenvolvedores na criação de aplicações para gerenciar a rede. Por exemplo, a API *northbound* fornece suporte às aplicações para engenharia de tráfego, qualidade de serviço e segurança. Essa interface pode ser implementada através do modelo arquitetural REST (Li et al., 2019c; HALEPLIDIS et al., 2015) e do formato de texto *JavaScript Object Notation (JSON)* (BRAY, 2014). Na figura 3 são ilustradas as três camadas da arquitetura SDN, controle, dados e aplicações, além das interfaces *northbound* e *southbound*.

REST é um padrão *web* orientado a recursos que utiliza o protocolo *Hypertext Transfer Protocol (HTTP)* para transmissão de dados. Em REST, cada componente de um sistema é representado como um recurso, identificado por uma *Uniform Resource Locator (URL)* e acessado através de métodos HTTP. Abaixo são descritos alguns dos métodos HTTP usados em serviços REST:

- *GET*: solicita um recurso ao servidor;
- *POST*: envia dados ao servidor;
- *PUT*: cria ou atualiza um recurso no servidor;
- *DELETE*: deleta um recurso.

Os dados transmitidos por meio do HTTP podem ser representados através de diferentes formatos, como texto puro, *eXtensible Markup Language (XML)*, JSON e *Google protocol buffer*s. O JSON é um dos formatos mais utilizados em conjunto com APIs REST, inclusive em interfaces *northbound* implementadas em controladores SDN (Li et al., 2019c). O JSON é um formato para transmissão de dados independente de linguagem de programação. O modelo para representação de dados adotado pelo JSON consiste em estruturas de dados

Figura 3 – Visão geral da arquitetura SDN, composta pelas chamadas de dados, controle e aplicação, encarregadas do encaminhamento dos dados, administração de uma rede e implementação dos protocolos, respectivamente.



Fonte: adaptado de: (KREUTZ et al., 2015)

universalmente reconhecidas, aceitas por *softwares* escritos em diferentes linguagens. Em JSON, os dados são representados como pares chave/valor, separados por dois pontos. A chave é uma *string* e o campo valor pode armazenar *string*, inteiro, variável *boolean*, *null*, objeto ou *array* (BRAY, 2014).

2.7 CONSIDERAÇÕES FINAIS

Neste capítulo, são descritos os principais conceitos abordados nesta tese. Foram apresentadas breves descrições sobre privacidade individual, Internet das Coisas, aprendizado de máquina, classificação do tráfego de rede, preenchimento de pacotes e redes definidas por *software*.

3 PREENCHIMENTO ADAPTATIVO PARA BALANCEAR O *TRADE-OFF* PRIVACIDADE-*OVERHEAD*

Neste capítulo, é descrita uma solução de preenchimento elaborada para demonstrar a praticabilidade de equilibrar o *trade-off* privacidade-*overhead* com base no *status* de uma rede. No restante desta tese, o termo proposta refere-se ao processo de adaptar o número de *bytes* em resposta às variações no *status* de uma rede, o que possibilita balancear o *trade-off* supracitado. Por sua vez, o termo solução diz respeito à técnica, descrita neste capítulo, elaborada para averiguar a praticabilidade de balancear o *trade-off* mencionado anteriormente. A próxima seção apresenta a visão geral e funcionamento desta solução, incluindo os relacionamentos entre os seus componentes.

3.1 MOTIVAÇÕES E VISÃO GERAL DA SOLUÇÃO DE PREENCHIMENTO

Devido ao *trade-off* mencionado anteriormente, os mecanismos de preenchimento existentes elevam demasiadamente ou insuficientemente o tamanho do pacote, independentemente da condição da rede subjacente (UDDIN; NADEEM; NUKAVARAPU, 2019). Em essência, os inúmeros mecanismos propostos buscam obter o máximo de privacidade com o mínimo de impacto no desempenho da comunicação. Contudo, devido à sua natureza estática, as soluções de preenchimento existentes produzem dois efeitos profundamente negativos: 1) geram um *overhead* excessivo que degrada o desempenho da comunicação e a experiência dos usuários; 2) falham em aprimorar a privacidade ao empenharem-se em evitar o *overhead*.

O *trade-off* mencionado acima consiste em abdicar do desempenho da comunicação para preservar a privacidade e vice-versa. O objetivo da solução apresentada neste capítulo não é eliminar esse *trade-off* – o que é potencialmente impossível – mas balancear entre privacidade e desempenho da comunicação ao adaptar o número de *bytes* em resposta às oscilações na utilização de uma rede. Esta tese assume que, sempre que o volume de tráfego for insuficiente para exaurir a capacidade da infraestrutura, é viável manter o número de *bytes* no valor máximo admitido, proporcionando o maior aprimoramento na privacidade enquanto interfere no desempenho das aplicações de forma negligenciável. Por outro lado, quando o volume de tráfego exceder a capacidade da rede, a quantidade de *bytes* pode ser reduzida para evitar impactar negativamente o desempenho da comunicação.

Neste ponto, aparenta ser oportuno pontuar que, quando o volume de tráfego atinge um nível próximo à capacidade da rede, faz-se necessário reduzir o número de *bytes* inseridos nos pacotes, como forma de evitar degradação no desempenho. Isto posto, é necessário abdicar de algum grau na privacidade a fim de manter o desempenho, desde que consentido pelos usuários da solução. A princípio, esse atenuamento no número de *bytes* pode aparentar uma exposição da privacidade, mas essa situação é aceitável desde que o usuário da solução projetada esteja de acordo com esse cenário. Considerando a

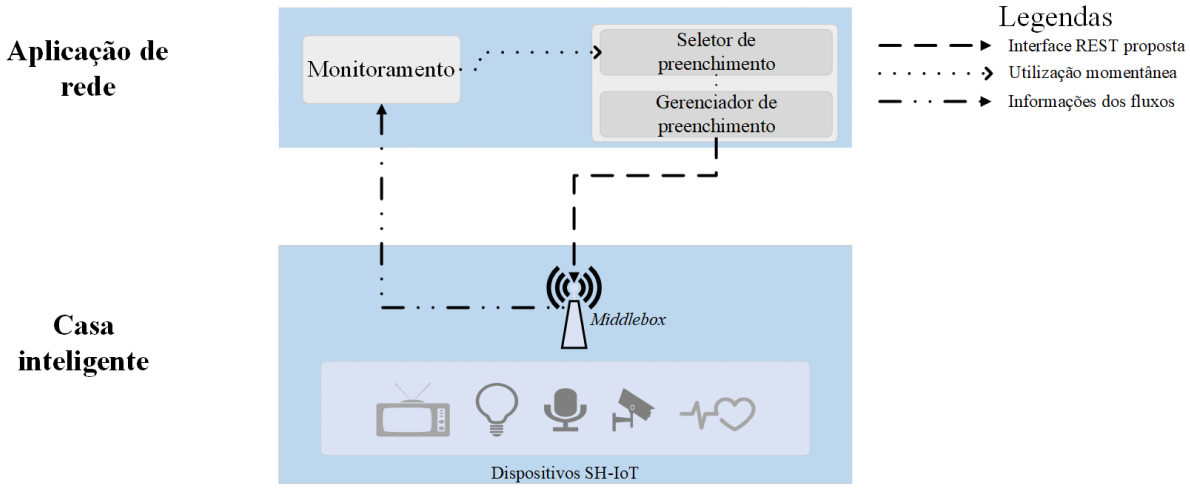
definição apresentada em (WESTIN, 1968), um indivíduo pode abdicar de algum nível de privacidade.

Os mecanismos existentes falharam em identificar um aspecto fundamental: o impacto gerado pela elevação da quantidade de tráfego depende da condição momentânea da rede. Assim, é possível aumentar a privacidade sem penalizar o desempenho, desde que exista capacidade ociosa na rede que acomode o tráfego adicional. O incremento no volume de tráfego produzido por mecanismos de preenchimento é essencialmente indesejável. Este trabalho reconhece esse fato, apenas é sugerido que essa elevação é aceitável e que, a mesma por si só não constitui um impedimento ao emprego de uma solução de preenchimento.

A Figura 4 apresenta a solução de preenchimento concebida. Nesta solução, uma aplicação de rede gerencia o mecanismo de preenchimento, instanciado no *middlebox*, através de uma interface REST proposta neste trabalho. Essa aplicação incorpora a utilização da rede para definir o número de *bytes* inseridos nos pacotes. O mecanismo acessa os pacotes gerados pelos dispositivos IoT para obter o seu tamanho, que é modificado com base nas instruções recebidas da aplicação. A solução proposta foi projetada para atender aos seguintes requisitos:

- Acomodar crescentes quantidades de dispositivos SH-IoT. Inúmeras pesquisas apontam para tendências no aumento da presença de dispositivos IoT em residências. Portanto, soluções para esse domínio IoT devem ter a capacidade de acomodar o número crescente de residências conectadas;
- Atender às demandas dos indivíduos por priorizar o aprimoramento na privacidade ou o desempenho da comunicação;
- Reduzir o *overhead* gerado pela solução.

Figura 4 – Visão geral da solução de preenchimento adaptativo.



Fonte: elaborada pelo autor.

É improvável que um típico usuário de dispositivos SH-IoT nutra interesse ou disponha dos conhecimentos técnicos necessários para implementar e manter uma solução de privacidade (BOUSSARD et al., 2018), como preenchimento de pacote. Um usuário IoT pode delegar o gerenciamento deste tipo de solução a terceiros. Dado que o indivíduo delegue a administração da solução a terceiros, é fundamental que o mesmo disponha de algum grau de confiança com o provedor do serviço. O ISP encontra-se em uma posição adequada para prover soluções de privacidade aos usuários de dispositivos SH-IoT (LEE; PAPPAS; PERRIG, 2019), como preenchimento de pacote. Contudo, o ISP não é uma entidade confiável e o mesmo como o fornecedor de uma solução de privacidade não é o cenário ideal. O próprio ISP é uma fonte de ameaças à privacidade dos proprietários de casas inteligentes, como a análise de tráfego para propaganda direcionada. Por isso, esta tese também considera um modelo de ameaça em que o ISP é o adversário.

Como o ISP controla a velocidade da conexão à Internet de uma residência, o mesmo encontra-se na melhor posição para ajustar o número de *bytes*. Em algumas regiões do mundo são impostas restrições quanto à quantidade de dados que um usuário de casa inteligente pode gerar em determinados períodos (ex.: mensalmente). Assim, a quantidade de dados extras pode ser restringida de acordo com o limite do plano de dados do usuário (APTHORPE et al., 2017). Para fornecer a solução de preenchimento, o ISP precisa obter a situação do tráfego de rede, que pode ser angariada através do monitoramento da infraestrutura. O monitoramento de rede é amplamente usado em procedimentos relacionados ao gerenciamento e segurança. Uma potencial solução pode ser monitorar a situação da rede através de uma entidade logicamente centralizada. Além disso, é fundamental que essa solução seja capaz de adaptar-se em resposta às variações no volume de tráfego de uma rede. Nas próximas seções são discutidas as motivações para o desenvolvimento dos componentes da solução de preenchimento descrita neste capítulo.

3.2 ESTRATÉGIA DE PREENCHIMENTO

O preenchimento de pacotes é tipicamente empregado para garantir que *frames Ethernet* e pacotes IP atendam a restrição quanto ao tamanho mínimo (JANKOWSKI; MAZURCZYK; SZCZYPIORSKI, 2013). Entretanto, este procedimento também é aplicado na ocultação de padrões no tamanho do pacote, que podem viabilizar a inferência de informações dos indivíduos, que representa uma fonte de ameaças à privacidade. Geralmente, um mecanismo de preenchimento busca alcançar o máximo de privacidade com o mínimo de *overhead*. Contudo, tipicamente, as soluções existentes abdicam da privacidade ou desempenho, sem conseguir conciliar esses objetivos conflitantes. Dadas as limitações das soluções existentes, faz-se necessário desenvolver uma estratégia capaz de conciliar privacidade e *overhead*. É importante que esta estratégia contemple características do tamanho do pacote gerado por dispositivos IoT, como valores mais comuns.

Esta tese propõe uma estratégia capaz de priorizar a privacidade ou *overhead* em

conformidade com as demandas dos usuários. Dessa forma, os indivíduos que priorizam a sua privacidade têm suas preferências acatadas, assim como os potenciais usuários da solução de preenchimento que preferem preservar o desempenho da comunicação. Para tanto, é imprescindível que a estratégia de preenchimento seja capaz de ajustar a quantidade de *bytes* inseridos nos pacotes. Essa estratégia busca atender aos seguintes requisitos:

- Possibilitar a priorização da privacidade ou desempenho das aplicações (*trade-off* privacidade-*overhead*);
- Viabilizar o ajuste na quantidade de *bytes* inseridos nos pacotes;
- Apresentar um desempenho no aprimoramento da privacidade comparável às estratégias MTU e *random*;
- Introduzir um *overhead* próximo às estratégias linear, exponencial e *random* 255.

A estratégia que compõe a solução de preenchimento é composta por quatro níveis (nomeados a partir do tamanho mínimo imposto em cada nível, 100, 500, 700 e 900). Em cada nível é inserido um número distinto de *bytes*. Visto que o tamanho de um *frame Ethernet* está limitado pelos valores de 64 e 1500 *bytes* (JANKOWSKI; MAZURCZYK; SZCZYPIORSKI, 2013), o uso de mais que quatro níveis pode acarretar em diferenças no *trade-off* privacidade-*overhead* demasiadamente sutis entre níveis adjacentes, ao mesmo tempo, menos que quatro podem tornar as mudanças entre níveis abruptas.

Os dois níveis inferiores priorizam a minimização do *overhead*. Mas, ainda assim, reduzem o desempenho de mecanismos para aprendizado de máquina, mesmo que o aprimoramento na privacidade seja diminuto comparado aos níveis superiores. Nos níveis inferiores, os tamanhos ofuscados são mantidos próximos aos seus valores originais, o que resulta em um reduzido *overhead*. Ao tornar os tamanhos dos pacotes constantes, os padrões característicos aos dispositivos são eliminados.

Os dois níveis superiores da estratégia de preenchimento buscam eficiência no aprimoramento da privacidade, ao custo de incorrer em um *overhead* mais elevado que os inferiores. O nível máximo obtém o melhor aprimoramento na privacidade. Apesar deste nível inserir o maior número de *bytes* permitido pela estratégia, espera-se que o mesmo inflija menos *overhead* que a estratégia MTU, que incrementa os pacotes para o seu tamanho máximo. Assim, mesmo no pior cenário, a estratégia deve introduzir menos *overhead* que o seu par mais eficiente disponível na literatura.

As características do tamanho do pacote variam de acordo com a composição de uma rede (os dispositivos presentes). Portanto, os valores mais apropriados para os níveis da estratégia podem variar entre diferentes residências, devido à diversidade de dispositivos SH-IoT. Entretanto, as contribuições desta tese permanecem válidas mesmo que os valores para os níveis da estratégia venham a diferir dos propostos neste trabalho, desde que permitam a adaptação do número de *bytes*.

Tabela 1 – Proporção para intervalos do tamanho do pacote proveniente do tráfego analisado.

Intervalo (bytes)	referência (SIVANATHAN et al., 2018) (%)	referência (REN et al., 2019) *testbed* US (%)	referência (REN et al., 2019) *testbed* UK (%)
<= 100	66,50	44,22	32,77
<= 200	66,84	52,56	42,42
<= 300	76,13	71,58	64,52
<= 400	86,23	75,07	67,96
<= 500	86,23	80,08	69,56
<= 1000	90,19	84,27	74,09
<= 1400	92,20	95,50	96,92

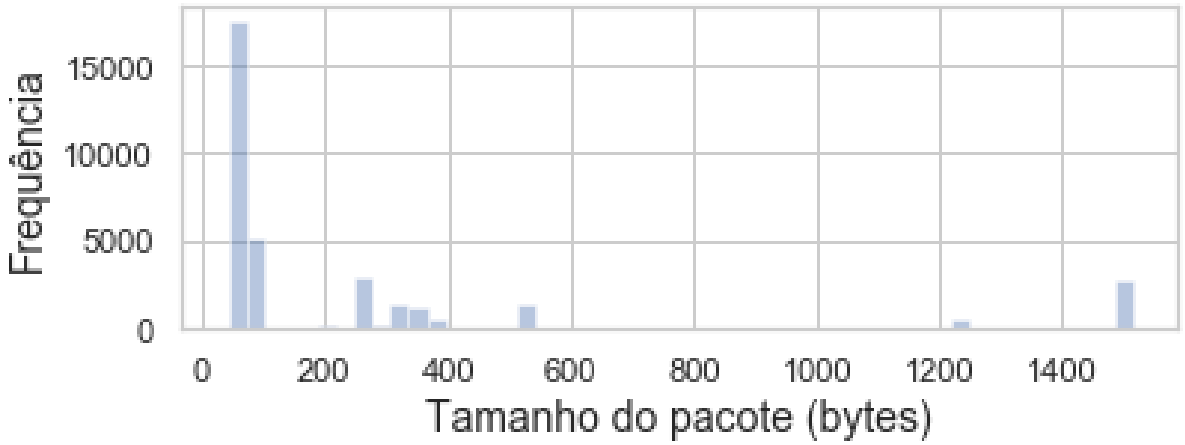
A partir da análise do tráfego SH-IoT disponibilizado pelos autores de (SIVANATHAN et al., 2018; REN et al., 2019), foram identificados os intervalos em que os tamanhos mais comuns estão concentrados. A Tabela 1 apresenta a proporção para alguns intervalos de tamanhos presentes no tráfego analisado. Por exemplo, a maioria dos tamanhos estão concentrados abaixo de 300 *bytes*, em que representam no mínimo 64,52%. A Figura 5 apresenta as distribuições de probabilidade para o tamanho do pacote presente nos dados disponibilizados pelos autores de (SIVANATHAN et al., 2018; REN et al., 2019). Os autores de (REN et al., 2019) disponibilizaram tráfego IoT proveniente de dois *testbeds*, um configurado nos Estados Unidos da América (US) e outro no Reino Unido (UK). Nos dados analisados, o tamanho máximo observado foi de 1514 *bytes*, valor que representa uma diminuta parcela do tráfego. O apêndice A apresenta a média, mediana, moda e terceiro quartil para o tamanho do pacote de cada um dos 102 dispositivos IoT analisados neste trabalho. Com base nessas observações sobre o tamanho do pacote gerado por dispositivos SH-IoT, uma estratégia de preenchimento foi desenvolvida.

Com base na análise do tráfego SH-IoT disponibilizado em (SIVANATHAN et al., 2018; REN et al., 2019), esta tese propõe que no nível mais inferior da solução, os tamanhos abaixo de 100 *bytes* sejam incrementados para este valor. Tamanhos entre 100 e 200 *bytes* são elevados para 200. Por fim, valores entre 200 e 300 *bytes* são incrementados para esse último. Tamanhos entre 300 e 1400 *bytes* são expandidos aleatoriamente, mas com restrições quanto ao valor máximo. Tamanhos menores que 1000 serão elevados para, no máximo, esse valor. Tamanhos entre 1001 e 1400 serão incrementados até 1400 *bytes*. Valores superiores a 1400 serão elevados precisamente para 1500 *bytes*.

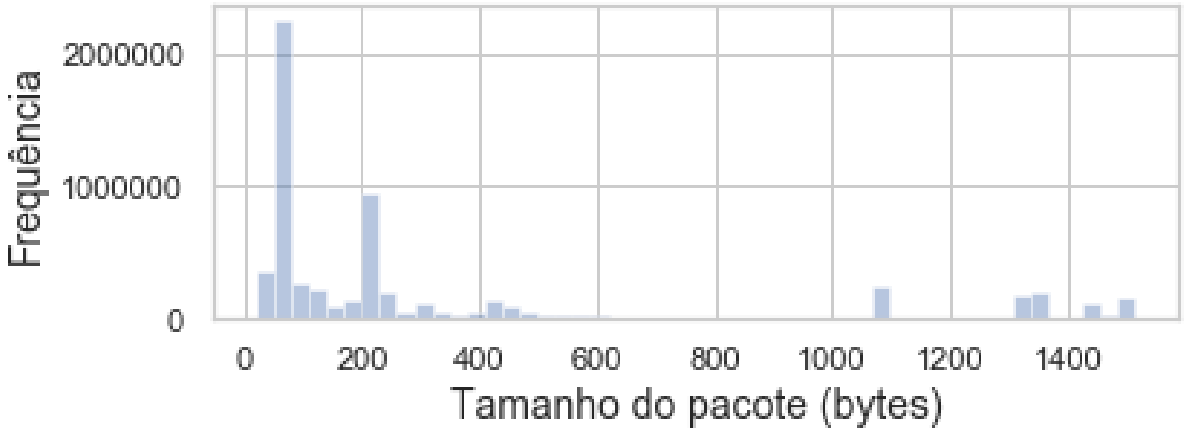
Valores maiores devem elevar a privacidade por distanciar os tamanhos ofuscados de suas contrapartes originais. Por exemplo, incrementar os tamanhos para 500 *bytes* afasta os valores ofuscados dos tamanhos originais inferiores a 300 *bytes*, a maioria. Para o segundo nível da estratégia, esta tese propõe incrementar os tamanhos para, no mínimo, 500 *bytes*. Valores entre 500 e 1000 *bytes* são incrementados aleatoriamente, restritos ao máximo de 1000 *bytes*. Novamente, tamanhos superiores a 1400 são elevados para 1500 *bytes*.

No terceiro nível da estratégia, esta tese sugere aumentar o tamanho mínimo para 700 *bytes*. Valores entre 700 e 1000 *bytes* são selecionados aleatoriamente. Novamente, tamanhos entre 1000 e 1400 *bytes* são incrementados aleatoriamente, como descrito anteriormente.

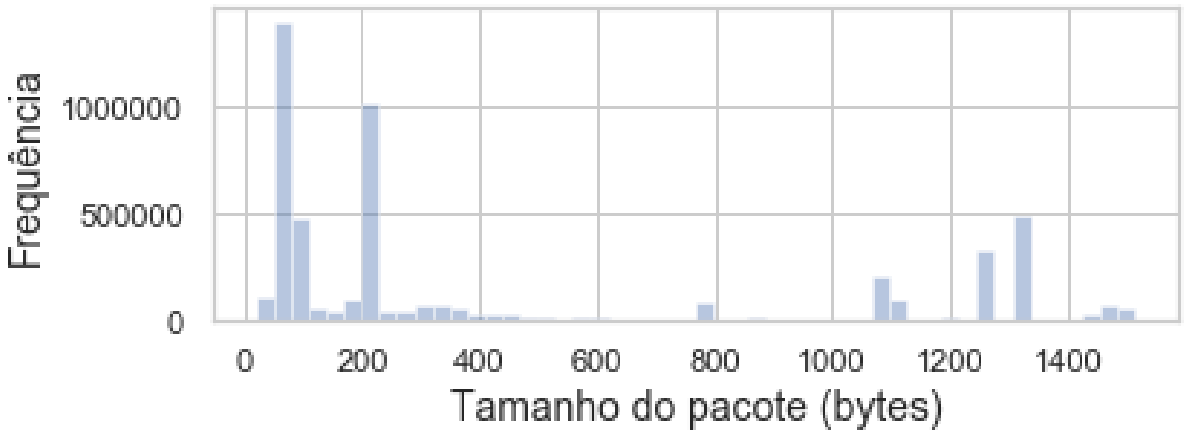
Figura 5 – Histograma para o tamanho do pacote do tráfego disponibilizado pelos autores de (SIVANATHAN et al., 2018; REN et al., 2019)



(a) Histograma do tamanho do pacote presente no tráfego disponibilizado pelos autores de (SIVANATHAN et al., 2018).



(b) Distribuição de probabilidade para o tamanho presente no tráfego disponibilizado pelos autores de (REN et al., 2019), capturado no *testbed* US.



(c) Histograma do tamanho do pacote encontrado no tráfego disponibilizado pelos autores de (REN et al., 2019), capturado no *testbed* UK.

Valores superiores a 1400 são elevados para 1500 *bytes*. No quarto e último nível, esta tese propõe incrementar os tamanhos para, no mínimo, 900 *bytes*. Tamanhos entre 900 e 1400 *bytes* são selecionados aleatoriamente. Valores superiores a 1400 são incrementados para 1500 *bytes*. A Tabela 2 apresenta as regras para modificação do tamanho em cada nível da estratégia.

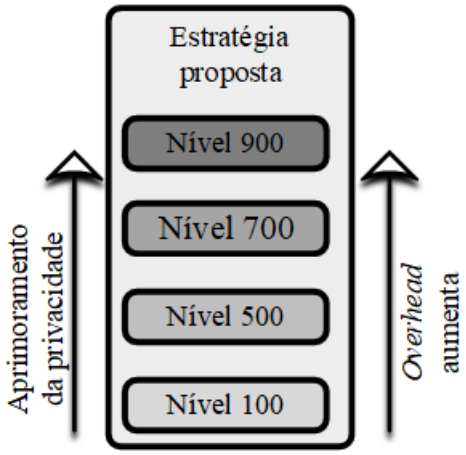
Tabela 2 – As regras que regem a modificação do tamanho do pacote em cada nível da estratégia.

Nível	Tamanho original	Tamanho ofuscado
100	<= 100	100
	<= 200	200
	<= 300	300
	> 300 < 999	aleatório entre 301 e 1000
500	<= 500	500
	> 500 e < 999	aleatório entre 501 e 1000
700	<= 700	700
	> 700 e < 999	aleatório entre 701 e 1000
900	<= 900	900
	> 900 e < 999	aleatório entre 901 e 1000
>= 999	>= 999 e <= 1399	aleatório entre 999 e 1400
	>= 1400 e < 1500	1500

Considerando o *trade-off* entre privacidade e *overhead*, o nível mais apropriado da estratégia depende do volume de tráfego na rede. Quando a rede estiver sobrecarregada, será preferível selecionar o nível mais inferior, pois produz o menor *overhead* introduzido pela estratégia. Por outro lado, quando a rede estiver ociosa, será adequado selecionar os níveis superiores, 700 e 900, que geram o *overhead* mais elevado, porém priorizam o aprimoramento da privacidade. A Figura 6 ilustra o *trade-off* entre privacidade e *overhead* presente na estratégia apresentada nesta tese. Percorrendo do nível mais inferior (nível 100) desta estratégia para o superior (nível 900), obtém-se o maior nível de privacidade e *overhead*.

Como mencionado anteriormente, a reivindicação dos indivíduos por privacidade deve ser soberana. Se o usuário priorizar o aprimoramento na privacidade, a solução de preenchimento deverá modificar os tamanhos com base nos níveis superiores (700 e 900), mesmo que incorra em degradação no desempenho das aplicações IoT. As preferências dos usuários podem ser obtidas através de aplicativos para *smartphones*, que estão além do escopo desta pesquisa. O consentimento dos usuários da solução foi obtido de forma implícita.

Figura 6 – *Trade-off* entre privacidade e *overhead* introduzido pela estratégia de preenchimento.



Fonte: elaborada pelo autor.

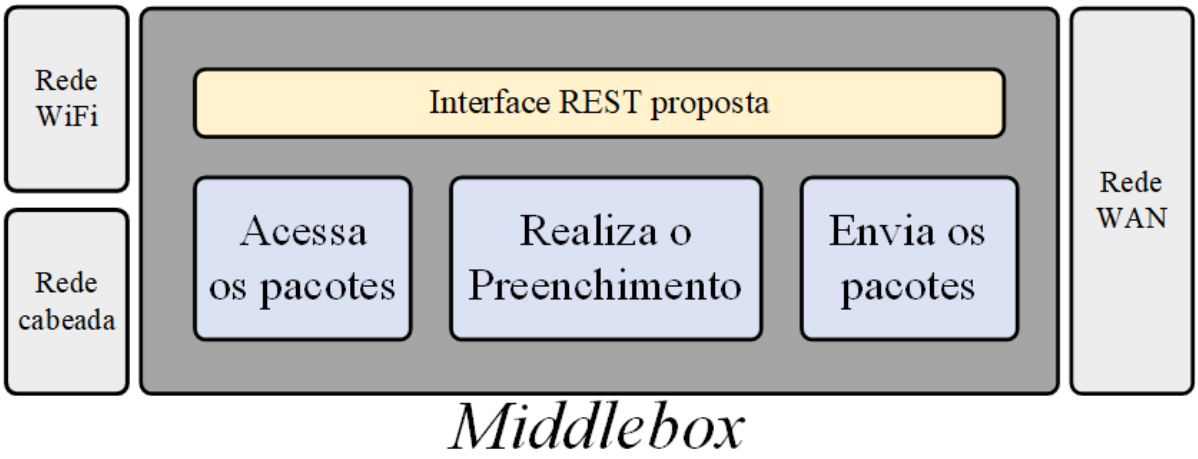
3.3 MECANISMO DE PREENCHIMENTO

Um mecanismo de preenchimento é a entidade responsável por modificar o tamanho dos pacotes com base em uma estratégia. Este mecanismo poderia ser implementado diretamente nos dispositivos IoT, mas encontraria inúmeras limitações técnicas, como capacidades de processamento e consumo de bateria. Além disso, dispositivos comerciais domésticos são proprietários e modificações estão restritas aos seus fabricantes (REN et al., 2019). Assim, a adoção da solução está sujeita à disposição dos fabricantes em incluírem a mesma em seus dispositivos, que é completamente irrealista, visto que a maioria dos fornecedores negligencia a segurança de seus produtos e usuários (SERROR et al., 2018). Também é inviável implementar legalmente soluções em dispositivos comerciais sem a expressa autorização de seus fabricantes.

Idealmente, o mecanismo seria implementado diretamente nos dispositivos IoT. Assim, o tráfego seria protegido desde a sua origem. Pelas razões expostas anteriormente, é muito improvável que seja possível implementar a solução projetada em dispositivos IoT de prateleira. Também seria ideal o próprio usuário administrar a solução, sem depender de terceiros, como o ISP. Contudo, concessões precisam ser feitas.

As restrições expostas acima impõem a busca por alternativas, como implementar a solução nos dispositivos de rede (Li et al., 2019a), atuando como um *middlebox*. O mecanismo de preenchimento é implementado como um *middlebox* em dispositivos como roteador doméstico e *gateway* IoT (APTHORPE et al., 2019; MIETTINEN et al., 2017; SERROR et al., 2018; XU et al., 2018). Implementar soluções de segurança e privacidade como *middleboxes* permite transpassar as restrições impostas pelos fornecedores de dispositivos IoT. É imperativo que a solução de preenchimento seja tão simples quanto possível, como forma de reduzir os requisitos sobre capacidade de processamento. Por isso, esta tese propõe uma estratégia de preenchimento que requer somente operações aritméticas simples.

Figura 7 – Visão geral do mecanismo de preenchimento.



Fonte: elaborada pelo autor.

O mecanismo é propositadamente simples, visto que o real valor do preenchimento reside na estratégia. Devido à sua localização na borda da rede doméstica, o *middlebox* é capaz de alterar os pacotes que entram e saem da residência. Contudo, somente os tamanhos que saem da residência são modificados. A segurança da rede local está além do escopo desta pesquisa. O *middlebox* recebe o tráfego IoT na interface *Local Area Network (LAN)* ou *Wireless Local Area Network (WLAN)*, modifica os tamanhos ao adicionar *bytes* entre o *payload* da camada de rede e o final do *frame* da camada de enlace e envia os dados alterados para a Internet através da WAN. A Figura 7 apresenta uma visão geral do mecanismo de preenchimento.

Se o tráfego for transmitido descriptografado, um observador poderá identificar a existência de *bytes* extras nos pacotes. Para evitar esse cenário problemático, os pacotes modificados devem ser criptografados antes de serem enviados aos seus destinos. A configuração de uma VPN também pode contribuir para ocultar a presença de *bytes* extras nos pacotes (Hafeez; Antikainen; Tarkoma, 2019; APTHORPE et al., 2019). Uma VPN pode ser fornecida pelo ISP aos seus clientes (LEE; PAPPAS; PERRIG, 2019), ou como uma solução integrada ao roteador doméstico por seu fabricante (APTHORPE et al., 2019).

O mecanismo é implantado no equipamento que interliga a rede doméstica à infraestrutura do ISP. Dessa forma, mesmo que uma residência disponha de múltiplos roteadores para servir aos dispositivos IoT, todo o tráfego originado na residência será ofuscado. Não há impedimento para implantação do mecanismo em múltiplos roteadores na mesma residência, configuração que pode contribuir para a redução em sua carga de trabalho. Geralmente, o preenchimento é implementado pelos roteadores para garantir o tamanho mínimo de um *frame Ethernet*.

Devido à sua localização na borda da rede residencial, o *middlebox* pode processar tráfego IoT e não IoT –*smartphones*, *tablets*, *laptops*, *desktops* (SIVANATHAN et al., 2017; SIVANATHAN et al., 2018). Por isso, é possível que a solução idealizada contribua para

aprimorar a privacidade em outros contextos além da SH-IoT, como páginas e aplicações *web*, aplicativos presentes em dispositivos móveis e chamadas VoIP. Contudo, é improvável que esse cenário se concretize, visto que diferentes aplicações têm comportamentos distintos (UDDIN; NADEEM; NUKAVARAPU, 2019), como pesquisas *web*, *streaming* de vídeo, transferência de arquivos, etc. A estratégia de preenchimento busca modificar essencialmente tamanhos particulares a dispositivos IoT, que possivelmente não compartilham das mesmas características com outras aplicações, como as citadas anteriormente. Seria de grande valor se a solução fosse apropriada para outras aplicações, pois ampliaria a proteção da privacidade dos usuários domésticos.

3.4 APLICAÇÃO DE REDE

O processo decisório sobre a atuação do mecanismo de preenchimento ocorre em uma aplicação de rede. Por exemplo, determinar a quantidade de *bytes* inseridos nos pacotes e quando esses valores devem ser alterados. O método usado para realizar essas decisões pode variar consideravelmente de um ambiente para outro, como regras baseadas em limiares, sistemas automáticos e mecanismos baseados em aprendizado de máquina. Essa aplicação é dividida em dois componentes: 1) seletor de preenchimento, determina o nível da estratégia a ser usado com base na utilização momentânea da rede; 2) gerenciador de preenchimento, realiza a configuração do *middlebox*.

A aplicação emprega a interface REST proposta para gerenciar o mecanismo de preenchimento. Toda a comunicação entre essas duas entidades ocorre de forma unidirecional, da aplicação para o *middlebox*. A aplicação instrui o *middlebox* a partir de informações coletadas da infraestrutura de rede. Uma grande variedade de dados podem ser obtidos de uma rede, mas esta tese está interessada essencialmente no nível de utilização do enlace monitorado. A coleta dessa informação pode ocorrer de diversas formas. Esta tese assume que propor novos métodos para monitoramento da rede é desnecessário para este trabalho. As informações atualmente coletadas sobre o tráfego de uma rede são suficientes para gerenciar o mecanismo de preenchimento.

A separação entre a entidade responsável por tomar as decisões e a que garante o cumprimento das mesmas contribui para a evolução independente de ambas. Por exemplo, o *middlebox* pode ser implantado em diferentes dispositivos de rede, como roteadores domésticos, dispositivos IoT e soluções *Network Function Virtualization (NFV)*, enquanto o processo decisório pode ocorrer de diversas formas, como as mencionadas anteriormente nesta seção. Dessa forma, é possível adaptar esses componentes da solução de preenchimento para diferentes ambientes de casas inteligentes.

3.5 INTERFACE REST

Uma API REST foi projetada especificamente para atender aos requisitos desta pesquisa. A API é encarregada de transmitir comandos entre a aplicação de rede e o *middlebox*. Esses comandos instruem o *middlebox* a respeito do número de *bytes* recomendado pela aplicação. Essa aplicação monitora continuamente a rede, e quando é possível aumentar a quantidade de *bytes* sem que o volume de dados total exceda a capacidade do enlace, o *middlebox* é instruído a elevar o nível de preenchimento. A aplicação informa ao *middlebox* que o mesmo deve incrementar ou decrementar o nível da estratégia de preenchimento. Por exemplo, deslocar do nível 300 para 500 e vice-versa.

Visto que dispositivos na borda da rede doméstica são, geralmente, computacionalmente restritos e a situação da rede tende a mudar rapidamente — que pode ocasionar chamadas frequentes à interface –, é essencial que a API consuma poucos recursos e imponha uma reduzida elevação no tráfego produzido pela transmissão dos comandos. Para atender a esses requisitos, esta tese propõe uma interface composta somente pelos comandos essenciais ao seu funcionamento. Assim, o excesso de dados na comunicação entre a aplicação e o *middlebox é evitado*. A interface possui somente um comando: *set_level*. O comando *set_level* permite que a aplicação instrua o *middlebox* a elevar ou reduzir o nível da estratégia de preenchimento. Esse comando transmite o nível que o *middlebox* deverá utilizar. A estrutura do comando *set_level* é apresentada a seguir.

Comando *set_level*:

```
1 {"level":"700"}
```

```
1 {"level":"100"}
```

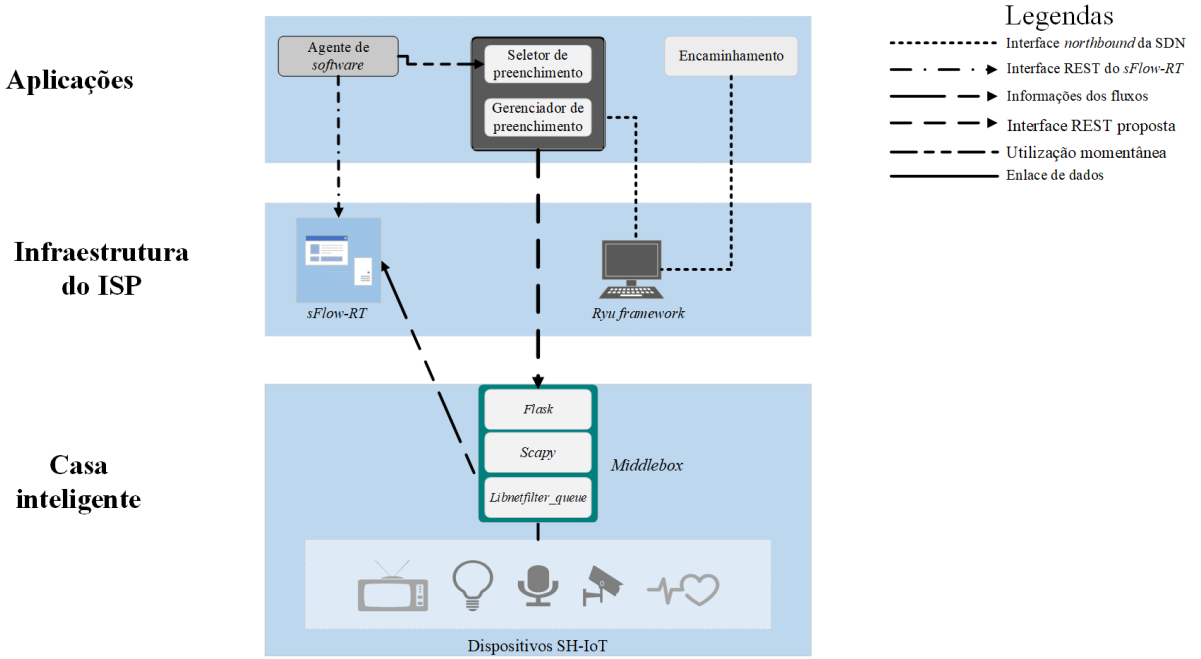
3.6 PROTÓTIPO

Esta seção apresenta um protótipo implementado a partir da descrição da solução proposta. Este protótipo viabilizou a avaliação da solução apresenta nesta tese. Este protótipo foi desenvolvido com base na arquitetura de redes definidas por *software*. As motivações para essa escolha são apresentada na próxima seção. A Figura 8 ilustra uma visão geral do protótipo.

3.6.1 Arquitetura de redes definidas por *software*

Como discutido na Seção 3.1, o ISP pode beneficiar-se de uma entidade logicamente centralizada no gerenciamento da solução de preenchimento. As redes definidas por *software* (*Software-Defined Networking (SDN)*) têm o controle logicamente centralizado implementado em um *software* chamado de controlador. Esse *software* tem uma visão global sobre a rede que gerencia, que o permite monitorar o tráfego e administrar a infraestrutura.

Figura 8 – Visão geral do protótipo e posicionamento das ferramentas usadas na implementação do mesmo.



Fonte: elaborada pelo autor.

Características presentes em SDN, como controle centralizado e programabilidade dos dispositivos de rede, são empregadas por mecanismos para diversos domínios, como *data center*, *wireless*, *Wide Area Network (WAN)* e computação em nuvem. Em anos recentes, emergiram inúmeras soluções de privacidade e segurança para residências que usam SDN para gerenciar mecanismos implementados nos dispositivos na borda da rede doméstica, como *gateways* e roteadores (BOUSSARD et al., 2018; Shirali-Shahreza; Ganjali, 2018; Thangavelu et al., 2019). O controlador SDN gerencia esses mecanismos a partir do ISP, o que dispensa o usuário doméstico do encargo de lidar com esta configuração (BOUSSARD et al., 2018). Ademais, é esperada uma gradual adoção da arquitetura SDN por ISPs (BOUSSARD et al., 2018; POULARAKIS et al., 2019). SDN é uma alternativa promissora gerenciar o *middlebox* responsável por modificar o tamanho do pacote.

No protótipo, uma aplicação no topo do controlador utiliza informações sobre a utilização momentânea da rede para instruir o *middlebox* para alterar a quantidade de *bytes* inseridos nos pacotes. Uma das principais vantagens da arquitetura SDN consiste em sua capacidade de adaptar-se às variações nas condições de tráfego de uma rede. Essa característica viabiliza ajustes na quantidade de *bytes* em conformidade com as condições da rede.

Assim como a arquitetura SDN, a solução de preenchimento desacopla a tomada de decisões da aplicação das mesmas, implementadas em entidades distintas. Uma aplicação no controlador SDN determina quantos *bytes* deverão ser inseridos nos pacotes, enquanto o algoritmo de preenchimento garante a modificação dos tamanhos. A arquitetura SDN

poderia ser melhor explorada na solução apresentada. Por exemplo, balanceamento de carga com base nos ajustes do número de *bytes*, aplicação de políticas de rede para aplicar o preenchimento e a classificação de equipamentos SH-IoT (Sivanathan; Gharakheili; Sivaraman, 2020).

O *Ryu framework* (Li et al., 2019c) foi selecionado como controlador SDN. Esse é um dos controladores mais corriqueiros na literatura sobre SDN (Li et al., 2019c). Nenhuma modificação neste *software* ou *switches* foi necessária. A interface para comunicação com o mecanismo de preenchimento foi implementada diretamente em uma aplicação executando no topo do controlador. Dessa forma, foram evitadas alterações no *Ryu framework*, que poderiam incorrer em erros de implementação potencialmente prejudiciais ao gerenciamento de uma rede. O *Ryu* é escrito em *Python* e foi executado pelo interpretador padrão desta linguagem. A aplicação executada no topo do controlador é passada como argumento para o *script Python* que inicia a operação do *Ryu*.

A aplicação de rede, um dos componentes da solução proposta, foi implementada como uma aplicação no topo do controlador SDN. No protótipo, essa aplicação é responsável por gerenciar o *middlebox*, enquanto outra aplicação SDN administra os *switches* da rede, como configurar as regras de encaminhamento dos dados. Na aplicação SDN, a interface para comunicação com o mecanismo de preenchimento foi implementada com o auxílio de uma biblioteca nativa do *Ryu* baseada em *Web Server Gateway Interface (WSGI)* (Li et al., 2019c). A biblioteca *json* foi utilizada para formatar os comandos transmitidos pela aplicação ao mecanismo de preenchimento. A biblioteca *requests* foi empregada na transmissão dos comandos com o método POST do HTTP.

3.6.2 Monitoramento da rede

O monitoramento da infraestrutura e do tráfego é uma tarefa inerente ao gerenciamento de rede (BOUSSARD et al., 2018). A arquitetura SDN permite a adoção de uma grande variedade de opções para monitoramento e possibilita ajustar a configuração de uma rede às oscilações no tráfego (BOUSSARD et al., 2018). Assim, a arquitetura SDN possibilita adaptar a quantidade de *bytes* inseridos nos pacotes a partir das variações na utilização de uma rede.

No monitoramento da rede, o protótipo emprega o protocolo *sFlow*. Na versão atual do protótipo, o *sFlow* proporciona a coleta de informações desnecessárias para gerenciar o *middlebox*, como dados a nível de fluxos. Contudo, esse protocolo poderá ser valioso em versões vindouras do protótipo. O protocolo *sFlow* é implementado através de agentes de *software* instanciados nos dispositivos de rede a serem monitorados (UJJAN et al., 2019). Esses agentes transmitem as informações coletadas do dispositivo monitorado para um *software* chamado coletor *sFlow*, que computa estatísticas para diversas métricas, como utilização do enlace, número de *bytes* transmitidos e porcentagem de pacotes descartados (UJJAN et al., 2019).

Tipicamente, as estatísticas computadas pelo coletor *sFlow* podem ser obtidas através de uma API REST. O controlador SDN é capaz de utilizar essa API para obter estatísticas do coletor *sFlow*, mas essa interação tem potencial de elevar o volume de tráfego na rede. Esta propõe um agente de *software* para obter as estatísticas através de chamadas a esta API REST disponibilizada pelo coletor, sendo executado na mesma máquina que o coletor. Este agente de *software* transmite a utilização momentânea do enlace monitorado, obtida do coletor, para uma aplicação em execução no topo do controlador SDN. Essa abordagem proporciona alguns benefícios, como independência de coletor específico e redução na quantidade de tráfego transmitido entre a aplicação SDN e coletor *sFlow*. A biblioteca nativa do *Ryu* baseada em WSGI contribuiu para a implementação de uma interface REST para comunicação entre a aplicação SDN e o agente de *software* que obtém informações do coletor.

O *software sFlow-RT(Real Time)* (UJJAN et al., 2019) foi selecionado como coletor *sFlow*. O agente de *software* que monitora o *sFlow-RT* foi implementado na linguagem *Python*. Esse agente requisita ao *sFlow-RT* a utilização momentânea do enlace monitorado a cada 60 segundos, que permite uma maior agilidade no ajuste do número de *bytes*. As requisições ao *sFlow-RT* são conduzidas através da interface REST disponibilizada por este *software*. As bibliotecas *json* e *requests* foram usadas para formatar e transmitir as informações sobre a utilização do enlace ao controlador, respectivamente. Como agentes do protocolo *sFlow*, que monitoram os *switches* e transmitem as informações coletadas para o *sFlow-RT*, foram aproveitados os mecanismos nativos do *software Open vSwitch*. O comando *ovs-vsctl* foi executado no interpretador de comandos *bash* para ativar o agente *sFlow* no *switch* virtual criado com o *Open vSwitch*. Durante os experimentos, somente um *switch* que representa um equipamento na rede do ISP foi monitorado com o *sFlow*.

Um *threshold* foi definido no *sFlow-RT*, que é acionado sempre que o nível de utilização do enlace monitorado supera 95%. O agente de *software* consulta o *sFlow-RT* a cada 60 segundos sobre o acionamento desse *threshold*. Quando esse *threshold* é superado, o agente de *software* obtém o nível de utilização atual do enlace e envia essa informação para a aplicação em execução no controlador. Essa aplicação utiliza as informações recebidas para gerenciar o nível da estratégia de preenchimento.

Neste trabalho, o *sFlow-RT* monitora a utilização do enlace posterior à modificação do tamanho, sendo influenciada pelo número de *bytes* inseridos nos pacotes. Por isso, o nível de utilização depende do número de *bytes* acrescentados aos pacotes. Por exemplo, o tráfego orgânico (produzido pelos dispositivos IoT) pode representar 10% da capacidade do enlace, enquanto os dados ofuscados podem constituir 14% ou 25%, a depender do nível de preenchimento. Considerando que o *sFlow-RT* obtém a utilização induzida pelo tráfego ofuscado, é essencial inferir o uso orgânico de um enlace.

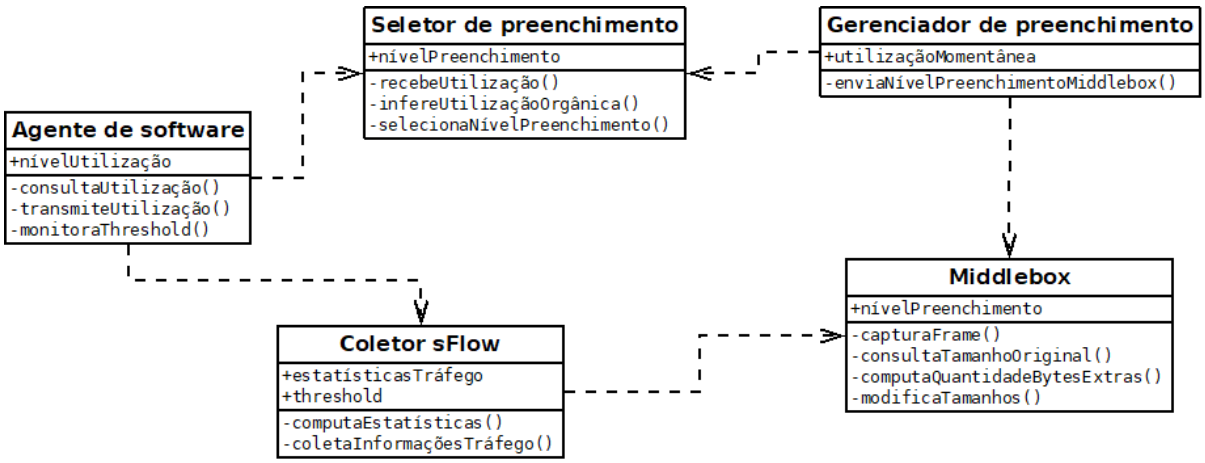
Ao receber o nível de utilização do enlace transmitido pelo agente de *software*, a aplicação no controlador infere o uso orgânico, com base no nível atual da estratégia

Tabela 3 – *Byte overhead* introduzido por cada um dos níveis da solução de preenchimento.

Nível da estratégia	*byte overhead* (%)
100	130
500	258
700	344
900	433

de preenchimento. O processo de inferência incorpora o *byte overhead* (relação entre o tamanho médio ofuscado e os valores originais multiplicada por 100 (DYER et al., 2012)) produzido por cada nível de preenchimento, valores obtidos através dos experimentos descritos na Seção 4.3.2. O *byte overhead* para cada nível de preenchimento adotado no protótipo é apresentado na Tabela 3. Para obter a utilização orgânica, a aplicação no controlador computa a razão entre o uso momentâneo do enlace e o *byte overhead* introduzido pelo nível de preenchimento atual. O valor obtido desta divisão determina o nível de preenchimento selecionado, com base nas regras apresentadas na Figura 10. Vale destacar que, o principal objetivo desta tese consiste em apresentar uma solução de preenchimento adaptável que ajusta o número de *bytes*, não necessariamente como essas quantidades são selecionadas. A Figura 9 ilustra um diagrama de classes do mecanismo de preenchimento e demais componentes do protótipo.

Figura 9 – Diagrama de classes e relacionamentos dos componentes do protótipo.

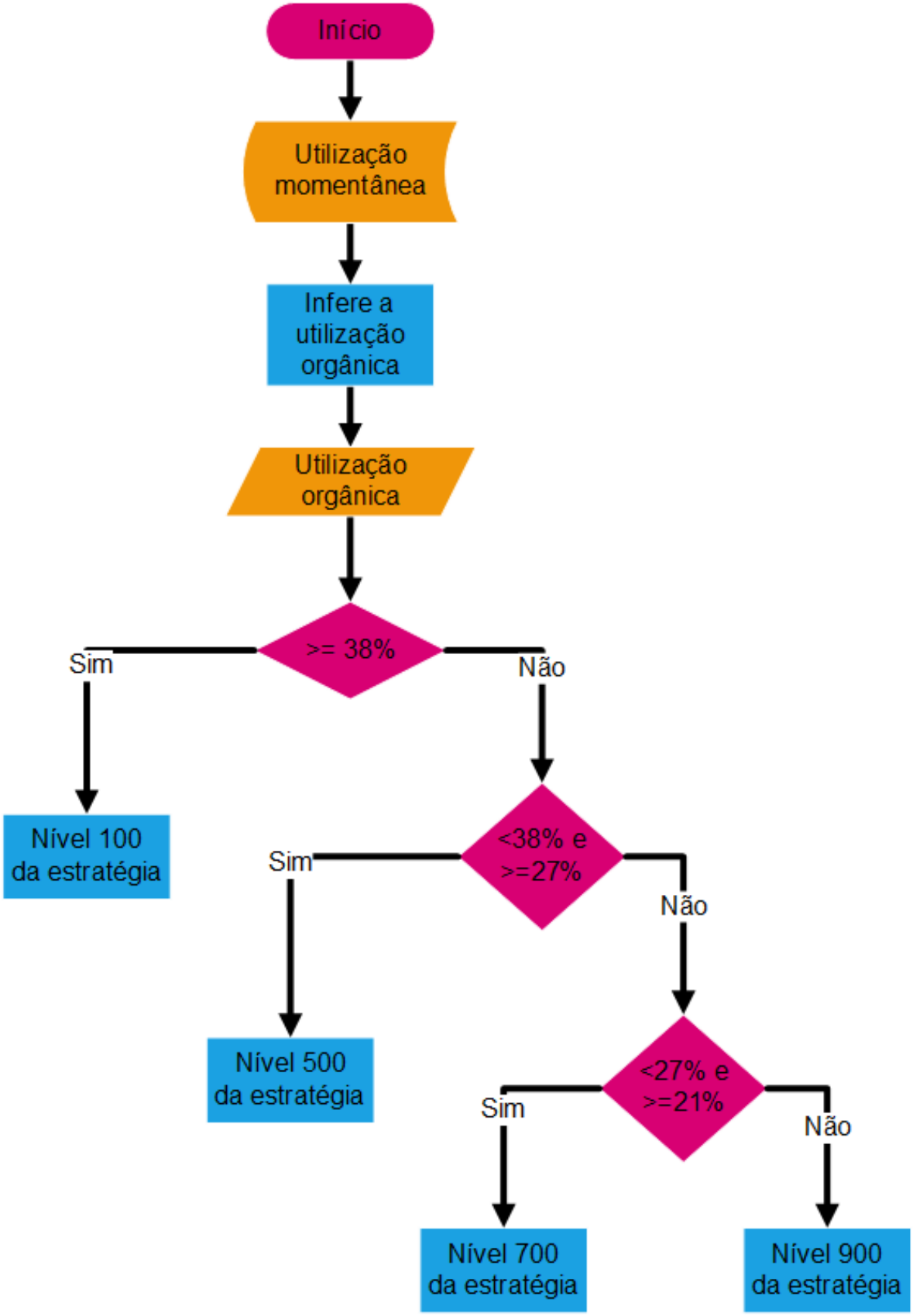


Fonte: elaborada pelo autor.

3.6.3 Interface REST

As limitações no *OpenFlow* motivaram o desenvolvimento de substitutos, como a linguagem *Programming Protocol-independent Packet Processors (P4)* e sua interface P4 *runtime* (BOSSHART et al., 2014). O *OpenFlow* e a P4 *runtime* requerem modificações em suas estruturas para transmitir informações que não estão previstas em suas especificações. Por isso, ao adotar essas interfaces no protótipo seria necessário modificá-las para transportar

Figura 10 – Fluxograma para decisão sobre o número de *bytes* inseridos nos pacotes.



Fonte: elaborada pelo autor.

os comandos da aplicação SDN para o *middlebox*. Essas restrições motivaram o desenvolvimento de interfaces especificamente para comunicação entre a aplicação no controlador e *middleboxes* (BOUSSARD et al., 2018; UDDIN; NADEEM; NUKAVARAPU, 2019). Considerando as limitações expostas anteriormente, uma API REST foi projetada especificamente para atender aos requisitos desta pesquisa.

O micro *framework Flask* (GRINBERG, 2018) foi selecionado para implementação da interface REST, que permite a comunicação entre mecanismo de preenchimento e aplicação SDN. Neste contexto, micro significa que o *Flask* dispõe somente dos atributos mínimos necessários ao desenvolvimento de uma API REST. Essa característica do *Flask* contribui para manter a interface leve.

Na formatação dos comandos transmitidos entre aplicação SDN e mecanismo de preenchimento, foi usado o formato JSON. O JSON foi selecionado devido à sua legibilidade e leveza, que contribui para minimizar a quantidade de tráfego produzido pela interface. O formato JSON e o *software Flask* são amplamente adotados na implementação de API REST (Li et al., 2019c).

3.6.4 Mecanismo de preenchimento

O mecanismo de preenchimento foi implementado na linguagem *Python*, com o auxílio das bibliotecas *libnetfilter_queue* e *Scapy*. A *libnetfilter_queue* foi usada para acessar os pacotes que atravessam o *middlebox*. Essa biblioteca permite a interação entre uma aplicação (o mecanismo de preenchimento) no espaço de usuário e o *netfilter*, *framework* responsável por gerenciar o tráfego de rede no *kernel Linux*[1]. Com a *libnetfilter_queue*, o mecanismo de preenchimento é capaz de acessar uma fila em que são armazenados os pacotes que atravessam o *middlebox*.

O *software iptables* foi configurado para redirecionar todos os pacotes que atravessam o *middlebox* para a fila mencionada anteriormente. O comprimento dessa fila foi definido empiricamente em 8192 posições. Assim, a capacidade desta fila está restrita a 8192 pacotes ao mesmo tempo. Comprimentos menores induziram a um decréscimo no desempenho do mecanismo de preenchimento. Esse decréscimo ocorreu devido à limitação na vazão do protótipo, que possibilitou a exaustão da fila e o consequente descarte de pacotes. Por outro, valores maiores para o comprimento da fila apresentaram uma contribuição negligenciável à execução do processo de preenchimento.

Após receber um pacote, o mecanismo de preenchimento precisa obter o seu tamanho para decidir quanto incrementá-lo. Por isso, é necessária uma ferramenta que possibilite a manipulação do tráfego de rede através da linguagem *Python*. A biblioteca *Scapy* permite o gerenciamento do tráfego de rede em *Python*. No mecanismo de preenchimento, a biblioteca

[1] O protótipo foi implementado em uma máquina virtual com sistema operacional *Ubuntu Linux*. A configuração completa de *hardware* e *software* desta máquina virtual é descrita no Capítulo 4.

Scapy foi empregada na obtenção do tamanho do pacote e inserção dos *bytes* extras. Esses *bytes* são posicionados após o final do *payload* do protocolo da camada de rede.

No protótipo, o número de *bytes* inseridos nos pacotes foi gerenciado através de um parâmetro denominado m (tamanho *m*ínimo). O valor desse parâmetro foi ajustado a partir das instruções recebidas da aplicação. O mecanismo de preenchimento operou através de duas *threads*, uma para comunicação com a aplicação por intermédio da interface REST e a outra para alterar o tamanho do pacote. O pacote resultante da modificação do seu tamanho é enviado para o *kernel Linux*, com auxílio da *libnetfilter_queue*. No *kernel*, o pacote modificado é transmitido para o próximo salto no caminho dos dados até o seu destino. Os *bytes* extras são adicionados na origem e removidos no destino pelo *driver* da placa de rede. No destinatário dos dados, esses *bytes* são descartados ainda na camada de enlace, evitando que os mesmos sejam repassados para as camadas superiores.

Como discutido na seção 3.3, é recomendável incorporar alguma forma de evitar que observadores percebam a presença de *bytes* extras nos pacotes, como uma VPN. Entretanto, vale destacar que, esta VPN não compõe a solução descrita neste capítulo. Logo, a implementação de uma VPN incorporada ao protótipo descrito nesta seção é desnecessária.

3.7 LIMITAÇÕES

Esta seção descreve as principais limitações da solução descrita neste capítulo, identificadas ao longo desta pesquisa.

- Limitações de abrangência: 1) existem inúmeras outras possibilidades de violar a privacidade dos indivíduos em suas residências através da análise do tráfego gerado por dispositivos IoT. É pouco provável que exista um mecanismo capaz de preservar todos os aspectos relacionados à privacidade dos usuários de dispositivos IoT. A solução proposta nesta tese lida somente com as ameaças geradas pela análise do tamanho do pacote, como a identificação dos dispositivos SH-IoT; 2) a segurança dos dispositivos estão além do escopo desta pesquisa; 3) observadores fisicamente próximos aos dispositivos presentes em residências, como vizinhos, podem inferir informações privadas dos usuários;

- Dependência de uma entidade externa, como o ISP. Em um cenário ideal, os usuários de dispositivos IoT administrariam a solução apresentada, o que eliminaria a dependência de terceiros. Contudo, é pouco provável que um típico usuário de equipamentos IoT disponha de conhecimento ou interesse em gerenciar uma solução de preenchimento. Uma alternativa para eliminar essa limitação reside na automatização da configuração e manutenção da solução, o que evitaria a dependência de uma entidade não confiável como o ISP;

- Ataques do tipo *Distributed Denial of Service* (Distributed Denial of Service (DDoS)) volumétrico podem induzir a redução no nível de privacidade. Dispositivos IoT comprometidos podem constituir *botnets* com o intuito de inviabilizar o funcionamento de um serviço através de ataques DDoS. Ao incrementar artificialmente o volume de tráfego gerado por dispositivos IoT, ataques DDoS podem induzir a solução de preenchimento a reduzir o número de *bytes* inseridos nos pacotes para minimizar o impacto no desempenho da comunicação, como esperado. Dessa forma, o nível de privacidade pode ser reduzido devido a dispositivos comprometidos por *malwares*. Apesar de neste momento não lidar com esse tipo de ataque, o protótipo da solução proposta pode ser estendido para gerenciar o preenchimento na presença de um DDoS. O protótipo implementa o protocolo *sFlow*, que é útil na detecção de ataques DDoS (UJJAN et al., 2019). Parece desejável reduzir a quantidade de *bytes* inseridos nos pacotes durante um DDoS, que naturalmente incrementa o volume de tráfego em uma rede.

3.8 CONSIDERAÇÕES FINAIS

Neste capítulo, foram apresentadas as motivações para o desenvolvimento da solução de preenchimento que concretiza a proposta desta tese de balancear o *trade-off* privacidade-*overhead*. Também foi descrita uma visão geral desta solução e o seus principais componentes, assim como os mesmos se relacionam entre si. Em paralelo, as motivações para as escolhas tomadas no projeto destes componentes foram apresentadas. Por fim, foi detalhado um protótipo implementado a partir da descrição da solução de preenchimento.

4 AVALIAÇÃO DO PROTÓTIPO DA SOLUÇÃO DE PREENCHIMENTO PROJETADA

Este capítulo apresenta a metodologia, projeto dos experimentos, ferramentas e ambiente experimental adotados na avaliação do protótipo da solução descrita no Capítulo 3. Esses experimentos foram projetados especificamente para responder às questões de pesquisa apresentadas no Capítulo 1.

4.1 METODOLOGIA DA AVALIAÇÃO

A avaliação tem como objetivo averiguar a capacidade da solução proposta de aprimorar a privacidade enquanto adapta o número de *bytes* em resposta as oscilações na utilização de uma rede. Por isso, foram realizados experimentos para mensurar o aprimoramento da privacidade e o *overhead* introduzido pela solução.

Quanto ao método de pesquisa, existem diversas possibilidades, em que a alternativa mais adequada varia entre pesquisas (EASTERBROOK et al., 2008). Por exemplo, experimento controlado e *survey*. Neste trabalho, o experimento controlado foi adotado como método de pesquisa. Em um experimento controlado, variáveis independentes são manipuladas a fim de observar o impacto que essas mudanças causam na(s) variável(eis) dependente(s) (EASTERBROOK et al., 2008). O impacto sobre variáveis dependentes é observado com a finalidade de testar as hipóteses levantadas durante esta pesquisa (EASTERBROOK et al., 2008). As características desta pesquisa que justificam a escolha do experimento controlado são apresentadas a seguir:

- Controle sobre variáveis;
- Possibilidade de aleatorização;
- Replicação de baixo custo.

A avaliação desta pesquisa está diretamente relacionada aos experimentos em laboratório, estabelecidos de acordo com o método de experimento controlado, pois há controle sobre as variáveis, a aleatorização é factível e a replicação dos experimentos é viável a um baixo custo. Neste experimentos, há controle sobre as seguintes variáveis: número de *bytes* e volume de tráfego. Na avaliação do protótipo, foram variados os valores dessas variáveis para investigar os seus efeitos no desempenho da solução de preenchimento, considerando o aprimoramento na privacidade, o *overhead* e o impacto no desempenho da comunicação.

Existem diversas alternativas para avaliar o desempenho de um sistema, como medição, simulação e modelagem analítica, cada uma dotada de vantagens e desvantagens (JAIN, 1990). Tipicamente, a medição proporciona os resultados mais confiáveis, mas constitui a técnica de avaliação mais custosa. Por outro lado, simulação e modelagem analítica impõem

custos menores que a medição, porém, são menos precisas que esta última (JAIN, 1990). Assim, nesta pesquisa, foi utilizada a medição como técnica para avaliar o desempenho do protótipo, visto que essa opção é considerada a que produz os resultados mais precisos (JAIN, 1990). A avaliação da solução proposta exigiu o projeto dos experimentos controlados, descrito nas próximas seções.

4.2 PROJETO DOS EXPERIMENTOS CONDUZIDOS NESTA PESQUISA

O experimento controlado utilizado nesta pesquisa foi projetado com base nos procedimentos descritos pelos autores de (PFLEEGER, 1995). Segundo os autores de (PFLEEGER, 1995), o planejamento de um experimento consiste das seguintes etapas:

- Concepção;
- Projeto;
- Preparação;
- Execução;
- Análise;
- Disseminação e tomada de decisão.

4.2.1 Concepção e projeto

Na fase de concepção, foram definidos os objetivos dos experimentos conduzidos nesta pesquisa e as métricas selecionadas para avaliar o desempenho do protótipo. Os objetivos da avaliação deste protótipo são divididos em dois grupos: quantificar o aprimoramento na privacidade e mensurar o impacto (causado pelo *overhead*) no desempenho da comunicação. As métricas para avaliar o desempenho do protótipo, agrupadas por objetivo, são descritas a seguir:

- Aprimoramento na privacidade:
  - Acurácia: porcentagem de amostras do tráfego classificadas corretamente do total de instâncias avaliadas;
  - *Recall*: porcentagem de amostras identificadas corretamente;
  - F1-*score*: média harmônica da precisão[1] e o *recall*.
- Impacto no desempenho da comunicação:

[1] Porcentagem de instâncias que realmente pertencem à classe X dentre todas aquelas classificadas como X.

- *Jitter*: variação do atraso na transmissão dos pacotes em uma rede. Um *jitter* muito elevado pode acarretar na degradação do desempenho de aplicações, especialmente aquelas que requerem chamada de voz e transmissão de vídeo, como a transferência dos dados gerados por uma câmera de segurança;
- *Goodput*: quantidade de dados úteis recebida pelo destino do tráfego por unidade de tempo (geralmente, um segundo). Quanto mais elevada for a *goodput*, maior é a quantidade de dados recebida pelo destino;
- Perda de pacotes: ocorre quando um ou mais pacotes são descartados antes de alcançarem o seu destino. É de suma importância que a perda de pacotes seja minimizada. Os pacotes descartados são retransmitidos pela origem dos dados, o que pode pesar sobre a elevação no volume de tráfego, causando impacto negativo ao *jitter* e à *goodput*.

Na fase de projeto dos experimentos, são definidas as hipóteses nula e alternativa adotadas na avaliação do protótipo. Esta tese propõe uma solução para adaptar o número de *bytes* inseridos no pacote em resposta às variações no volume de tráfego de uma rede doméstica. Portanto, a hipótese nula (representada por H_0) levantada nesta pesquisa afirma que a solução é capaz de aprimorar a privacidade enquanto adapta o nível de preenchimento com base na utilização de um enlace. Logo, a hipótese alternativa (simbolizada por H_1), o complemento da H_0, declara que a proposta é incapaz de aprimorar a privacidade enquanto considera a utilização de uma rede. Essas hipóteses são testadas por meio dos experimentos descritos neste capítulo.

4.2.2 Preparação e execução

Na fase de preparação dos experimentos, foram definidas as ferramentas usadas na avaliação do protótipo. Foram utilizadas as versões estáveis mais recentes dessas ferramentas, no momento de escrita desta tese. Uma breve descrição dessas ferramentas, assim como as motivações que determinaram a seleção das mesmas, são apresentadas a seguir:

- *Mininet* 2.2.2: o *Mininet* é um *software* para emulação de redes definidas por *software*. Esse emulador foi selecionado devido à sua ampla adoção na construção de experimentos com redes SDN. Infelizmente, o *Mininet* apresenta impedimentos à construção de redes em larga escala. Por isso, foi impraticável realizar experimentos em escalas maiores que as empregadas nesta pesquisa. Existem alternativas para usar o *Mininet* em uma escala maior, mas contribuem infimamente aos experimentos conduzidos neste trabalho;
- *Open vSwitch* 2.9.5: *software* usado pelo *Mininet* na criação de *switches* virtuais de uma rede SDN;

- *Ryu framework* 3.4: controlador SDN implementado na linguagem *Python* e amplamente utilizado na academia e indústria (Li et al., 2019c). Esse *software* foi selecionado devido às constantes atualizações em relação às versões do *OpenFlow* promovidas por seus mantenedores;

- *Python* 3.6: linguagem de programação de propósito geral. O protótipo da solução foi implementado nesta linguagem devido à sua sintaxe clara, que contribui para a redução no tempo de prototipagem;

- *PyPy* 7.3.0: interpretador otimizado para acelerar a execução de código escrito em *Python*. Esse *software* foi utilizado para impulsionar o desempenho do protótipo no processamento dos pacotes;

- *iPerf* 2.0.12: *software* capaz de produzir tráfego de rede. O *iPerf* atua com base no modelo cliente-servidor, em que o primeiro transmite o tráfego para o segundo. O servidor *iPerf* avalia várias métricas, como *goodput*, *jitter* e perda de pacotes;

- *Wireshark* 2.2.5 e *Tcpdump* 4.9.2: *softwares* capazes de capturar e examinar o tráfego de uma rede. Estes *softwares* foram usados para analisar o tráfego gerado pelos dispositivos IoT;

- R 3.4.4: linguagem de programação propícia à análise estatística. Os testes estatísticos conduzidos na análise dos resultados foram realizados neste *software*;

- *Pandas* 0.24.2: biblioteca da linguagem *Python* para análise de dados. Essa biblioteca foi empregada na análise do tráfego gerado pelos dispositivos IoT;

- *Scikit-learn* 0.20.3: suíte de algoritmos para aprendizado de máquina implementados na linguagem *Python*. Os mecanismos usados para avaliar o aprimoramento na privacidade foram implementados com o auxílio desta suíte;

- *k-Nearest Neighbors* (k-NN): calcula a distância euclidiana de cada instância de teste para os k vizinhos mais próximos presentes no conjunto de treinamento (DUDA; HART; STORK, 2012). O k-NN dispensa suposições sobre a normalidade dos dados analisados, aspecto raramente encontrado no mundo real (DUDA; HART; STORK, 2012);

- *Decision Tree*: algoritmo que constrói modelos para aprendizado de máquina com base em estruturas de árvore. Neste algoritmo, cada nó representa uma verificação, cada ramificação o resultado, por fim, as folhas retratam os rótulos que indicam os dispositivos IoT (ALPAYDIN, 2014). Esse algoritmo viabiliza a construção de modelos simples, mas que alcançam resultados robustos (ALPAYDIN, 2014);

- *Random Forest*: algoritmo que emprega múltiplos classificadores baseados em estruturas de árvore na identificação dos dispositivos (BREIMAN, 2001). Esse algoritmo é resistente ao *overfitting* que contribui para a obtenção de um desempenho consistente.

Na fase de execução, os experimentos são realizados de acordo com a especificação apresentada na seção 4.3. Nesta fase, foram obtidos os resultados a serem analisados com base nos procedimentos descritos na próxima seção.

4.2.3 Análise dos resultados

Na análise dos resultados, foram computadas as medidas de tendência central média, mediana e moda, que contribuem para mensurar o desempenho da solução de preenchimento. Outras estatísticas, como desvio padrão e variância, foram desconsideradas na análise dos resultados, pois as medições coletadas não seguem uma distribuição de probabilidade Normal/Gaussiana (MONTGOMERY; RUNGER, 2010). Além de computar as estatísticas mencionadas, foram realizados testes de hipóteses, procedimentos estatísticos que contribuem para as conclusões a respeito dos resultados.

Testes de hipóteses contribuem para quantificar a incerteza sobre os resultados (MONTGOMERY; RUNGER, 2010). Além de amplamente usados na literatura, testes de hipóteses empregam intervalos de confiança na análise dos dados, um método popular para comparação de alternativas. Uma hipótese estatística é uma afirmação sobre os parâmetros, como média e mediana, de uma ou mais populações (MONTGOMERY; RUNGER, 2010). A hipótese nula, denotada por H_0, é usada para verificar a hipótese formulada sobre o problema investigado (por exemplo, a solução de preenchimento baseada na proposta desta pesquisa contribui para o aprimoramento da privacidade) e pode ser rejeitada ou não com base nos resultados obtidos na avaliação. Quando H_0 é rejeitada, a hipótese aceita é o seu complemento, chamada de hipótese alternativa, denotada por H_1.

As hipóteses nula e alternativa são rejeitadas ou não com base no valor do p-*value*, computado por testes estatísticos. O p-*value* corresponde à probabilidade de rejeitar a hipótese nula quando a mesma é verdadeira (MONTGOMERY; RUNGER, 2010). Em todos os testes de hipóteses conduzidos, foi adotado um nível de confiança de 95% que, induz a um nível de significância igual a 5%, representado pela letra grega α. Nesses testes, a hipótese nula é rejeitada quando o p-*value* é menor que o α (neste trabalho, 5% ou 0,05) (MONTGOMERY; RUNGER, 2010).

Inúmeros testes estatísticos estão disponíveis para avaliação das hipóteses ao computarem o *p-value*. O teste estatístico mais adequado depende da distribuição dos dados analisados, em que existem duas alternativas: 1) paramétrica, os dados seguem uma distribuição Normal, também conhecida como gaussiana; 2) não paramétrica, os valores examinados não são distribuídos de acordo com a distribuição Normal. O tipo da distribuição de probabilidade dos dados pode ser determinado por um teste de aderência.

Nesta tese, foram utilizados o teste de *Komolgorov-Smirnov* para identificar o tipo de distribuição dos dados analisados, paramétrica ou não paramétrica.

O teste de aderência de *Kolmogorov-Smirnov (KS)* baseia-se na diferença máxima entre as distribuições hipotética e empírica de uma amostra, dada por $T = sup_x|F^*(x) - F_n(x)|$. Nesta fórmula, $F^*(x)$ é a função de distribuição hipotética[2] e, $F_n(x)$ é a função de distribuição empírica (do inglês, *Empirical Distribution Function (EDF)*) estimada da amostra analisada (RAZALI; WAH, 2011).

Como foram analisados os resultados produzidos por múltiplas fontes, a solução de preenchimento e os mecanismos apresentados em pesquisas anteriores, foi selecionado o teste estatístico de *Friedman*, procedimento capaz de determinar a existência de diferença entre mais que duas alternativas. O teste estatístico de *Friedman*[3] (FRIEDMAN, 1937) é um equivalente não paramétrico ao procedimento *ANalysis Of VAriance (ANOVA)*. Neste teste, os resultados dos procedimentos analisados são ordenados do maior para o menor, sendo um posto (posição relativa em uma sequência) atribuído a cada valor. Quando a hipótese nula é rejeitada (há diferença estatística entre algum dos procedimentos), pode-se aplicar testes para verificar o procedimento com o melhor resultado, como o *Wilcoxon Signed-Rank Test (WSRT)*.

O procedimento WSRT considera que X_1 é uma amostra aleatória proveniente de uma distribuição contínua e simétrica com média (e mediana) μ. O teste calcula a diferença $X_i - \mu_0$, para $i = 1, 2, ..., n$. Ranqueia a diferença absoluta $|X_i - \mu_0|$, para $i = 1, 2, ..., n$ em ordem crescente, e posteriormente atribui aos postos os sinais (positivo ou negativo) de suas diferenças correspondentes. Nesse teste, postos são posições $1, 2, 3, ..., n$ que os valores da variável aleatória X ocupam quando colocados em ordem crescente. Dado que W^+ é a soma dos postos positivos e W^- a soma dos postos negativos, calcula-se W (o menor valor entre o W^+ e W^-) com base na seguinte equação: $W = min(W^+, W^-)$. W^*_α é o valor obtido da tabela de valores críticos do teste *Wilcoxon Signed-Rank Test* (por exemplo, para uma amostra com tamanho 15, $\alpha = 0,05$ e H_0:$\mu = \mu_0$, $W^*_\alpha = 25$). Esta tabela com os valores para W^*_α pode ser encontrada em (MONTGOMERY; RUNGER, 2010). O *Wilcoxon Signed-Rank Test* é adequado para amostras pareadas, quando dois procedimentos são aplicados ao mesmo conjunto de dados. Por exemplo, dois medicamentos são testados com as mesmas cobaias.

Os testes estatísticos mencionados acima foram aplicados para avaliar os seguintes cenários: 1) aprimoramento na privacidade proporcionado pelos mecanismos de preenchimento; 2) *byte overhead* introduzido pelas soluções analisadas. Toda a análise estatística foi conduzida com o *software* R 3.4.4 para *Ubuntu Linux* 18.04 *Long Term Support (LTS)*. O teste de *Friedman* foi implementado com o auxílio da biblioteca PMCMRplus (do inglês, *Pairwise Multiple Comparisons of Mean Rank (PMCMRplus)*).

[2] Nesta tese, a distribuição Normal

[3] Nomeado em homenagem ao seu formulador, Milton Friedman, nobel em economia.

4.2.4 Disseminação e tomada de decisão

Alguns aspectos são importantes para disseminação de uma pesquisa científica, como replicabilidade e publicação dos resultados. A replicabilidade é um requisito da pesquisa científica, que demanda uma nítida e completa documentação dos procedimentos adotados na avaliação (PFLEEGER, 1995). Esses procedimentos são descritos detalhadamente na próxima seção. Todos os aspectos pertinentes à avaliação desta pesquisa, como variáveis, objetivos, configuração do ambiente experimental e ferramentas utilizadas, são descritos neste capítulo. Dessa forma, outros pesquisadores poderão reproduzir os experimentos realizados nesta pesquisa.

Outro aspecto fundamental de uma pesquisa científica consiste na divulgação dos resultados. Além desta tese, os resultados obtidos nesta pesquisa foram divulgados através de artigos científicos, publicados em periódicos e conferências prestigiosas. Para a conveniência do leitor, uma lista com a relação desses artigos está disponível no apêndice 7.2 ao final desta tese. A relevância do tema identificação de dispositivos e eventos IoT a partir do tamanho do pacote foi validada no artigo (PINHEIRO et al., 2019). A importância do preenchimento de pacotes para preservar a privacidade de usuários dos dispositivos SH-IoT foi corroborada em (PINHEIRO; BEZERRA; CAMPELO, 2018). O mérito de adaptar o número de *bytes* em resposta às oscilações na utilização de uma rede é discutido em um artigo submetido ao *Institute of Electrical and Electronic Engineers (IEEE) Internet of Things Journal*.

4.3 AMBIENTE EXPERIMENTAL CONFIGURADO PARA OS EXPERIMENTOS

Esta seção descreve o ambiente experimental concebido para avaliação do protótipo. Todos os experimentos foram executados em duas máquinas virtuais (do inglês, *Virtual Machine (VM)*), cada uma configurada com a seguinte composição de *hardware* e *software*: *Central Processing Unit (CPU) Intel® Xeon®* 2 GHz 64-bit, *Random Access Memory (RAM) Double Data Rate (DDR)* com 8 GB, *Hard Drive (HD)* de 10 GB e o sistema operacional *Ubuntu Linux* 18.04 LTS. Uma máquina virtual proporciona algumas vantagens para esta pesquisa, como flexibilidade de custos, fácil replicação (a instância de uma VM pode ser facilmente replicada) e ambiente controlado. O *sFlow-RT*, assim como o agente de *software* implementado para monitorá-lo, foi implantado em uma VM, enquanto as demais ferramentas usadas na avaliação foram executadas na outra VM. Os experimentos conduzidos para avaliar o desempenho da solução são descritos detalhadamente nas próximas seções.

4.3.1 Aprimoramento da privacidade

Nesta seção, são descritos os experimentos conduzidos para mensurar o aprimoramento na privacidade proporcionado pela solução. Esses experimentos contribuem para a obtenção

de indícios sobre a influência do número de *bytes* inseridos nos pacotes no desempenho da solução de preenchimento. Também foram avaliadas as principais estratégias de preenchimento existentes, com o objetivo de comparar os seus desempenhos com os resultados da estratégia projetada.

Como esta tese busca preservar a privacidade dos indivíduos contra as ameaças impostas por observadores de rede, foi necessário compreender o modelo de ameaça que a solução pretende mitigar. O procedimento para avaliar a solução depende do adversário do qual o usuário IoT deve ser protegido. Por isso, foram conduzidos experimentos diferentes para cada adversário considerado nesta tese.

4.3.1.1 Modelos de adversário

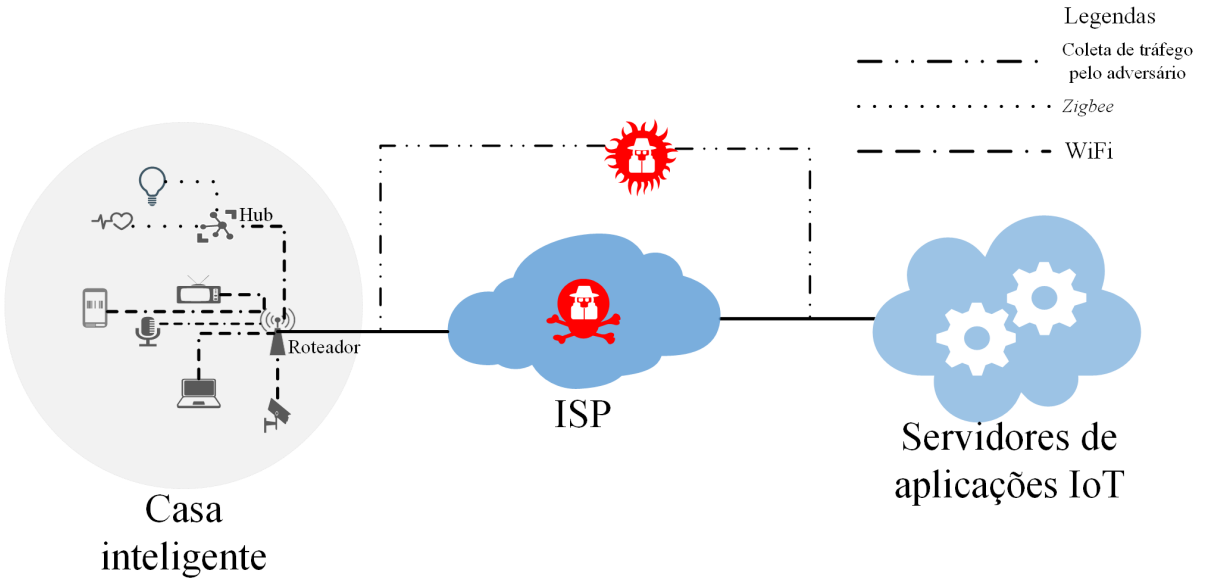
Dois modelos de adversários foram considerados: 1) um observador externo monitora as atividades ocorridas no interior de uma residência a partir do tráfego de rede gerado pelos dispositivos IoT. Esta tese assume que o adversário externo captura o tráfego gerado pelos dispositivos IoT de suas vítimas externamente aos domicílios (UDDIN; NADEEM; NUKAVARAPU, 2019), após o roteador doméstico, em que o mecanismo de preenchimento é implementado; 2) o ISP pode valer-se de sua posição para espreitar os seus usuários. Como o usuário delega a administração da solução de preenchimento ao ISP, é essencial mensurar quanto o mesmo precisa confiar no provedor da solução. Este trabalho assume que os adversários atuam de forma passiva, coletando informações dos indivíduos a partir do tráfego gerado por dispositivos IoT, sem investigar ativamente[4] a rede de suas vítimas (Hafeez; Antikainen; Tarkoma, 2019).

Os cenários descritos acima são deveras plausíveis, visto que, tipicamente, dispositivos IoT são financeiramente acessíveis, que possibilita a constituição de *testbeds* por observadores com a finalidade de coletar dados dos equipamentos e usá-los no treinamento de algoritmos para aprendizado de máquina supervisionado (UDDIN; NADEEM; NUKAVARAPU, 2019; APTHORPE et al., 2019). Esses algoritmos treinados com dados IoT são, posteriormente, usados pelos observadores para inferir informações privadas dos indivíduos (Hafeez; Antikainen; Tarkoma, 2019). Como os procedimentos empregados na inferência de informações privadas nos dois modelos de ameaça são distintos, os experimentos para avaliar o aprimoramento na privacidade também diferem mutuamente. A Figura 11 ilustra a localização em que os observadores podem coletar o tráfego originado em uma casa inteligente. A avaliação da solução na mitigação de ameaças produzidas por observadores externos e pelo ISP é descrita a seguir.

Considerando que técnicas para aprendizado de máquina são usadas por observadores para inferir informações de suas vítimas, esse tipo de mecanismo é comumente empregado na avaliação do desempenho de soluções para preenchimento de pacote (UDDIN; NADEEM; NUKAVARAPU, 2019). Por isso, algoritmos para aprendizado de máquina foram usados

[4] Um adversário ativo é aquele que comunica-se diretamente com as suas vítimas.

Figura 11 – Modelos de ameaças considerados nesta tese e pontos em que os observadores podem capturar o tráfego.



Fonte: elaborada pelo autor.

para mensurar o aprimoramento na privacidade acarretado pela solução. Neste contexto, a privacidade é aprimorada ao reduzir o desempenho desses algoritmos, como decrementar a acurácia, o que sugere um decréscimo na quantidade de informações que um observador consegue inferir de suas vítimas. Esta avaliação mostra-se necessária devido ao objetivo de reduzir o desempenho desse tipo de mecanismo.

Para mensurar o desempenho da solução na mitigação de ameaças introduzidas por observadores externos, os algoritmos treinados com tráfego gerado por dispositivos IoT são testados[5] descritos na Seção 2.3 com os dados ofuscados (tráfego cujo tamanho foi modificado pelo mecanismo de preenchimento). Esta tese assume que, um observador externo está ciente sobre a possibilidade do tráfego capturado ser modificado por um mecanismo de preenchimento, mas desconhece o procedimento adotado na alteração dos tamanhos (CIFTCIOGLU et al., 2018).

Para cada *dataset*, os seguintes procedimentos foram seguidos para preparar os dados para treino e teste dos algoritmos: 1) o tráfego gerado por cada dispositivo é separado com base no endereço *Media Access Control (MAC)*, então um rótulo é selecionado para representar unicamente esse equipamento, como 0, 1, 2, 3 e 4; 2) os dados gerados por cada dispositivo são agrupados em janelas de um segundo (informação obtida do *timestamp* dos *frames*); 3) para cada janela de tempo, foram computadas as estatísticas média, desvio padrão e número de *bytes*. Depois o rótulo correspondente ao dispositivo que gerou o tráfego foi associado a essas estatísticas. Ao final deste processo foram obtidas algumas milhares de amostras do tráfego gerado por dispositivos IoT, representadas pelas três

[5] Os procedimentos para treino e teste dos algoritmos para aprendizado supervisionado foram

estatísticas mencionadas anteriormente. Esse procedimento foi repetido para os *datasets* originais e com preenchimento.

Os três classificadores são treinados e testados com as estatísticas computadas a partir dos dados disponibilizados pelos autores de (SIVANATHAN et al., 2018; REN et al., 2019) e ofuscados pelo mecanismo de preenchimento, respectivamente. Este mesmo procedimento é aplicado em (UDDIN; NADEEM; NUKAVARAPU, 2019; PINHEIRO; BEZERRA; CAMPELO, 2018). Este experimento simula o cenário em que um observador treina classificadores com tráfego IoT e os utiliza para inferir os dispositivos presentes na residência de suas vítimas (APTHORPE et al., 2019). Este experimento contribui para avaliar a capacidade da solução em degradar o desempenho de algoritmos para aprendizado de máquina.

Como o provedor da solução pode aplicar o mecanismo de preenchimento nos dados de treinamento, que pode contribuir para incrementar o desempenho dos algoritmos nos dados ofuscados, foram conduzidos experimentos que simulam esse cenário. Assim, foi empregada uma 10-*fold cross-validation* para mensurar o desempenho do mecanismo de classificação com os dados ofuscados, como sugerido pelos autores de (DYER et al., 2012). Dessa forma, os algoritmos de aprendizado supervisionado são treinados e testados com os dados ofuscados.

Além de avaliar a solução elaborada, é importante averiguar como o seu desempenho compara-se ao das estratégias de preenchimento mais relevantes encontradas na literatura. Para viabilizar essa comparação, as principais estratégias de preenchimento apresentadas em trabalhos relacionados foram implementadas em *scripts Python*, que modificam somente o tamanho do pacote presente em arquivos CSV. Esses arquivos CSV foram criados a partir dos conjuntos de dados que contêm tráfego IoT.

4.3.1.2 Comparação com as principais estratégias de preenchimento

Como mencionado anteriormente, diversas estratégias de preenchimento foram apresentadas em trabalhos anteriores. As principais estratégias existentes são: *random*, *random* 255, linear, exponencial, MTU e Ratos/Elefantes. A estratégia *random* foi implementada pelo mecanismo apresentado em nosso trabalho anterior (PINHEIRO; BEZERRA; CAMPELO, 2018), assim como pelo *software obfsproxy* (SHAHBAR; ZINCIR-HEYWOOD, 2015). A *random* 255 é empregada por protocolos de segurança populares, como TLS (DIERKS; RESCORLA, 2008), além dos autores de (DYER et al., 2012). Linear, exponencial, MTU são analisadas em (LIBERATORE; LEVINE, 2006; DYER et al., 2012). A estratégia linear é analisada novamente em (SIBY et al., 2019). A MTU também foi implementada pelos mecanismos apresentados em (UDDIN; NADEEM; NUKAVARAPU, 2019; APTHORPE et al., 2019). Por fim, a estratégia Ratos/Elefantes é empregada pelo mecanismo proposto em (IACOVAZZI; BAIOCCHI, 2015; IACOVAZZI; BAIOCCHI, 2014).

Os autores de (XIONG; SARWATE; MANDAYAM, 2018) mapeiam o tamanho original x aleatoriamente para o valor ofuscado $x_1 \in X_1 = \{d_1, d_2, ..., d_{|X_1|}\}$. X_1 é o conjunto de todos

os tamanhos que podem ser selecionados, assumindo $d_1 < d_2 < d_3 ... < d_{|X_1|}$ (sem perda de generalidade). Logo, o mecanismo apresentado em (XIONG; SARWATE; MANDAYAM, 2018) seleciona os tamanhos aleatoriamente, procedimento semelhante ao executado pela estratégia *random*. O mecanismo proposto em (UDDIN; NADEEM; NUKAVARAPU, 2019) simula o comportamento da estratégia MTU, como admitido por seus autores. No melhor cenário, o procedimento proposto em (UDDIN; NADEEM; NUKAVARAPU, 2019) é equivalente à MTU.

Neste ponto, é necessário destacar que as diferentes estratégias de preenchimento existentes têm objetivos distintos, induzidos pelo *trade-off* privacidade-*overhead*. As estratégias linear, exponencial e *random* 255 produzem um aprimoramento na privacidade pífio, mas minimizam o *overhead* na comunicação. Por outro lado, a MTU e a *random* apresentam uma significativa melhoria na privacidade, enquanto introduzem um *overhead* elevado. A estratégia Ratos/Elefantes, ao utilizar dois níveis de preenchimento, intenciona equilibrar a privacidade e o *overhead* (IACOVAZZI; BAIOCCHI, 2015). Comparar diretamente estratégias que priorizam a privacidade com aquelas que minimizam o *overhead* seria parcial.

Essa mesma conjectura surge na comparação entre o nível mais elevado da solução concebida nesta pesquisa e estratégias que minimizam o *overhead*, como linear, exponencial e *random* 255. Por isso, foram comparados o desempenho das estratégias linear, exponencial e *random* 255 com o nível 100 da solução. O nível 500 é comparado com a estratégia Ratos/Elefantes. Por fim, foram comparadas as estratégias *random* e MTU com os níveis 700 e 900 da solução. Estas últimas estratégias aprimoram a privacidade ao mesmo tempo que produzem um elevado *overhead*, que pode interferir no desempenho da comunicação. No restante desta tese, o termo $Preenchimento(m = x)$ foi empregado como referência aos níveis da solução descrita no Capítulo 3. O m consiste no parâmetro que representa o tamanho mínimo para o qual os pacotes são alterados, enquanto o x representa o nível de preenchimento (100, 500, 700 e 900). Por exemplo, $preenchimento(m = 900)$ significa que os tamanhos são modificados para, no mínimo, 900 *bytes*., como descrito na Seção 3.2.

4.3.1.3 Implementação dos algoritmos de aprendizado de máquina

Tipicamente, os mecanismos para classificação de tráfego IoT a partir de metadados apresentados nos trabalhos relacionados estão indisponíveis publicamente. Essa restrição nos motivou a desenvolver um mecanismo para identificação de dispositivos e eventos IoT com base no tamanho do pacote (PINHEIRO et al., 2019). No desenvolvimento deste mecanismo, este trabalho assume que, devido à sua natureza de propósito específico, um dispositivo IoT gera pacotes com tamanhos característicos. Por exemplo, pacotes com 491, 304 e 172 *bytes* do *Amazon Echo Dot*, 258 *bytes* da *Amcrest security camera* (modelo IP2M-841B) e 149 *bytes* do TP-*Link smart plug* (modelo HS100). Assim, é possível identificar os dispositivos IoT a partir do tamanho do pacote, com o auxílio de técnicas

para aprendizado de máquina.

Para identificar dispositivos IoT e comportamentos dos seus usuários, o mecanismo de classificação desenvolvido utiliza somente três estatísticas do tamanho do pacote. Essas estatísticas são computadas a partir do tráfego de rede agrupado em janelas de um segundo, intervalo que agiliza o processo de classificação dos dispositivos. As seguintes estatísticas do tamanho foram empregadas: média, desvio padrão e número de *bytes*. A seleção dessas estatísticas foi fundamentada em uma extensa análise do tamanho do pacote gerado por dispositivos IoT tipicamente presentes em residências (PINHEIRO et al., 2019).

A biblioteca *pandas* foi empregada na inspeção dos dados contidos nos arquivos CSV. A biblioteca *Scikit-learn* foi utilizada na implementação dos três classificadores empregados na avaliação do mecanismo de classificação e aprimoramento na privacidade proporcionado pela solução: *k-Nearest Neighbors* (k-NN[6]), *Random Forest* e *Decision Tree*. Esses algoritmos são amplamente empregados na classificação do tráfego de rede, devido à sua capacidade de alcançar um desempenho significativo, enquanto evitam a necessidade de usar modelos mais complexos e computacionalmente custosos, como redes neurais (SIVANATHAN et al., 2018). Algoritmos baseados em redes neurais recorrentes e convolucionais poderiam ser aplicados nos dados analisados, porém são desnecessários visto que as técnicas implementadas obtiveram um desempenho interessante (DACREMA; CREMONESI; JANNACH, 2019), apresentado e analisado no Capítulo 5.

Múltiplos classificadores foram implementados para investigar possíveis tendências na identificação de dispositivos IoT. Ademais, múltiplos classificadores contribuem para minimizar a probabilidade de ocorrer obliquidade nos resultados (ALPAYDIN, 2014). Os classificadores selecionados remetem a diferentes categorias, como vizinhos mais próximos, baseados em estrutura de árvores e *ensemble* (*Random Forest*). A eficiência desse tipo de técnica motiva a desenvolver a solução para preenchimento de pacotes apresentada nesta tese.

O tráfego IoT, armazenado em arquivos *Packet Capture (PCAP)*, foi convertido em arquivos de texto no formato CSV com o *software Wireshark*. A solução de preenchimento foi implementada através de *scripts Python*. Esses *scripts* inspecionam os arquivos CSV que contêm o tráfego IoT e modificam o tamanho do pacote, em conformidade com o nível da estratégia adotado momentaneamente. Esses *scripts* produzem novos arquivos CSV contendo os dados originais, exceto o tamanho do pacote, que é modificado. Dessa forma, garantiu-se que somente o tamanho é alterado. Este procedimento é baseado em trabalhos relacionados (LIBERATORE; LEVINE, 2006; DYER et al., 2012).

[6] Tipicamente, 5 é o valor para a variável k (número de vizinhos mais próximos) (DUDA; HART; STORK, 2012).

4.3.1.4 Descrição dos conjuntos de dados usados

Na avaliação do mecanismo para identificação de dispositivos a partir do tamanho do pacote, foram usados os conjuntos de dados disponibilizados pelos autores de (SIVANATHAN et al., 2018; REN et al., 2019). Esses dados foram os únicos disponíveis publicamente, e encontrados, que atenderam aos requisitos desta pesquisa, como dados validados em conferência ou periódico de prestígio internacional, tráfego bruto (para evitar possíveis inconsistências nos resultados ocasionados por dados previamente tratados) e proveniente das principais categorias de dispositivos SH-IoT. Também foram encontrados outros *datasets* com dados IoT que não atendem aos requisitos mencionados anteriormente, como medições de temperatura e tráfego proveniente de equipamentos comprometidos por *malwares*.

Os *datasets* disponibilizados pelos autores de (SIVANATHAN et al., 2018) são compostos por 20 arquivos PCAP, em que cada um corresponde a 24 horas de captura do tráfego gerado por 21 dispositivos IoT. Esta base de dados foi utilizada em alguns trabalhos (SIVANATHAN et al., 2018; SANTOS et al., 2018; PINHEIRO et al., 2019). Os dados disponibilizados pelos autores de (REN et al., 2019) foram coletados em dois *testbeds*, um estabelecido na *Northeastern University* nos Estados Unidos da América e outro no *Imperial College* baseado no Reino Unido, compostos por 46 e 35 dispositivos IoT, respectivamente. Portanto, os autores de (REN et al., 2019) disponibilizaram tráfego gerado por 81 dispositivos IoT, capturados ao longo do mês de Abril de 2019. Esses *datasets* contendo tráfego de rede devem ser convertidos em arquivos *Comma-Separated Values (CSV)*, que são posteriormente analisados por técnicas para aprendizado de máquina.

Nos dados disponibilizados em (SIVANATHAN et al., 2018), o mecanismo de classificação foi aplicado em cada um dos 20 arquivos CSV que, nos permitiu repetir este experimento 20 vezes e obter múltiplas medições, usadas em testes de hipóteses. Como o tráfego disponibilizado em (REN et al., 2019) foi compartilhado como um único *dataset*, o experimento com o mecanismo de classificação foi repetido por 20 vezes no mesmo conjunto de dados, mesmo procedimento praticado nos dados disponibilizados por (SIVANATHAN et al., 2018). O mecanismo de classificação foi avaliado com o auxílio de uma 10-*fold cross-validation*, procedimento bastante comum na literatura em virtude de sua robustez (DEMŠAR, 2006). Estes experimentos foram baseados nos procedimentos descritos pelos autores de (DEMŠAR, 2006), devido às suas relevantes contribuições para a literatura sobre avaliação de modelos para aprendizado de máquina.

4.3.2 *Byte overhead* e impacto no desempenho da comunicação

Foi avaliado o impacto no desempenho da comunicação provocado pelos quatro níveis da solução, assim como pelas estratégias existentes. Posteriormente, foi avaliado um cenário em que a solução adapta o número de *bytes* em resposta às variações na utilização de uma

rede, como proposto nesta tese. Esses experimentos indicam quanto a quantidade de *bytes* impacta no desempenho da comunicação. Este conjunto de experimentos, aliado à avaliação do aprimoramento na privacidade, descritos anteriormente, fornece evidências sobre a relevância de ajustar o número de *bytes* para balancear o *trade-off* privacidade-*overhead*.

O impacto no desempenho da comunicação ocorre devido à elevação na quantidade de *bytes* transmitidos pela rede (UDDIN; NADEEM; NUKAVARAPU, 2019). Por isso, foi averiguado o *byte overhead* para cada estratégia avaliada. O *byte overhead* foi computado a partir dos arquivos CSV produzidos pelos *scripts* descritos na Seção 4.3.1. A partir desses arquivos, foi extraído somente o tamanho do pacote, que foi salvo em um novo arquivo CSV, um para cada estratégia avaliada. Desses arquivos CSV, foram computadas as estatísticas média, mediana e moda do tamanho do pacote.

Esta tese assume que os *bytes* extras acarretam em degradação no desempenho da comunicação somente quando o volume de tráfego excede a capacidade da rede. Portanto, é plausível que, quando a utilização orgânica estiver suficientemente baixa, estratégias que geram elevado número de *bytes* produzam um diminuto impacto no desempenho da comunicação. Esse cenário deve ocorrer devido aos dados extras consumirem apenas a capacidade ociosa. Assim, este trabalho considera viável empregar estratégias fonte de um elevado *overhead*, desde que exista capacidade ociosa suficiente para acomodar os *bytes* extras. Por outro lado, é apropriado usar estratégias fonte de baixo *overhead* quando o enlace estiver próximo à exaustão.

Para avaliar as premissas apresentadas no parágrafo anterior, foram conduzidos experimentos com todas as estratégias de preenchimento, em vários níveis de utilização de um enlace de dados. Por exemplo, para examinar a influência da MTU no desempenho da comunicação, esta estratégia foi avaliada em um enlace com um nível de utilização variando de 10% a 100%. Seguindo este mesmo raciocínio, as estratégias linear, exponencial e *random* 255 foram avaliadas em uma rede com os níveis de utilização descritos anteriormente. Estes experimentos exigem uma ferramenta que permita variar o volume de tráfego transmitido em uma rede.

O *iPerf* foi usado para gerar tráfego de rede, com variações no volume de dados, iniciando em uma baixa utilização (ex.: 10% da capacidade do enlace) até a completa saturação (100% de utilização). O *iPerf* mensurou as métricas *goodput*, *jitter* e perda de pacotes. Cogitou-se estimar o tempo necessário para transmitir os dados entre o cliente e o servidor *iPerf*, representado pela métrica atraso. Entretanto, experimentos iniciais mostraram que o atraso é severamente afetado quando a rede está saturada, independente do mecanismo de preenchimento avaliado. Por isso, atraso não foi avaliado, pois é provável que essa métrica contribuiria de forma marginal para a avaliação da solução.

Como mencionado anteriormente, os *bytes* extras são automaticamente removidos no destinatário do tráfego, sem repassá-los para as aplicações. Por isso, a instância do *iPerf* que atua como servidor recebe somente o conteúdo gerado pela origem dos dados, sem os

bytes inseridos pelo mecanismo de preenchimento. Dessa forma, esses *bytes* não interferem no funcionamento da aplicação em execução, como o *iPerf* neste experimento. O volume de dados recebidos pelo servidor *iPerf* não é ampliado pelo preenchimento que, interfere no desempenho da comunicação somente ao exaurir a capacidade da rede.

Na avaliação do protótipo, a capacidade do enlace não é fundamentalmente relevante. A solução tem como propósito adaptar o número de *bytes* em resposta às variações no volume de tráfego, em que o impacto de sua elevação depende da capacidade do enlace. Por isso, nesta avaliação, variar o volume de tráfego é mais importante que a capacidade do enlace, preferencialmente produzindo dados para estressar a infraestrutura da rede. Como discutido anteriormente, os padrões no tráfego de rede tendem a variar entre residências. Contudo, o volume de tráfego selecionado para avaliar a solução de preenchimento deve ser representativo, considerando uma residência com dispositivos SH-IoT.

Os autores de (REN et al., 2019) não indicam o volume de dados gerado pelos dispositivos IoT analisados. Por outro lado, os autores de (SIVANATHAN et al., 2018) apontam que, o pico no tráfego IoT analisado foi de 1 *Megabit per second (Mbps)*. Por isso, a solução de preenchimento foi avaliada com tráfego gerado em até 10Mbps, 10 vezes o pico nos dados IoT analisados pelos autores de (SIVANATHAN et al., 2018). Para garantir a exaustão do enlace de dados empregado nos experimentos, a sua largura de banda foi delimitada em 10Mbps. Avaliar a solução de preenchimento com um volume de tráfego mais elevado, como 100Mbps e 1000Mbps, seria interessante. Contudo, a proposta de adaptar o número de *bytes* em resposta ao *status* de uma rede independe do volume de tráfego momentâneo. Por isso, os 10Mbps são suficientes para avaliar a proposta desta pesquisa.

O *iPerf* gerou o tráfego que representa os dados produzidos por dispositivos IoT, iniciando em 1Mbps (10% da capacidade do enlace) até atingir 10Mbps (100% da capacidade), em que a cada 360 segundos, o volume de dados foi incrementado em 1Mbps. O *iPerf* foi configurado para produzir pacotes com 214 *bytes*, tamanho médio computado a partir do tráfego disponibilizado em (SIVANATHAN et al., 2018). Assim, a solução de preenchimento foi avaliada com um tamanho representativo do tráfego gerado por dispositivos SH-IoT.

Cada nível da solução de preenchimento, assim como as estratégias existentes, foi avaliado em um experimento com duração de 60 minutos. A solução de preenchimento também foi avaliada durante 60 minutos. O *iPerf* mensurou as métricas avaliadas em intervalos de um segundo. Portanto, os experimentos foram repetidos 3600 vezes para cada mecanismo analisado, que produziram o mesmo número de amostras para cada métrica avaliada. Visto que são seis estratégias existentes, quatro níveis de preenchimento e a solução elaborada nesta tese, esses experimentos para avaliar o impacto no desempenho da comunicação tiveram uma duração total de 660 minutos. Devido à baixa variabilidade nos resultados destes experimentos, essa duração foi suficiente para avaliar a solução de preenchimento.

Na avaliação, a solução apresentada foi iniciada no nível 900, que proporciona um maior

aprimoramento da privacidade, enquanto introduz um *overhead* elevado. Essa configuração contribui para avaliar a capacidade da solução de reduzir o nível de preenchimento à medida que o volume de tráfego aumenta. Assim, caso as premissas adotadas nesta pesquisa mostrem-se corretas, conforme a utilização do enlace aproxima-se de 100%, o número de *bytes* será reduzido a fim de evitar uma sobrecarga na rede doméstica. Essa redução poderá ser perceptível ao constatar que, a utilização da rede permanece inferior à capacidade do enlace.

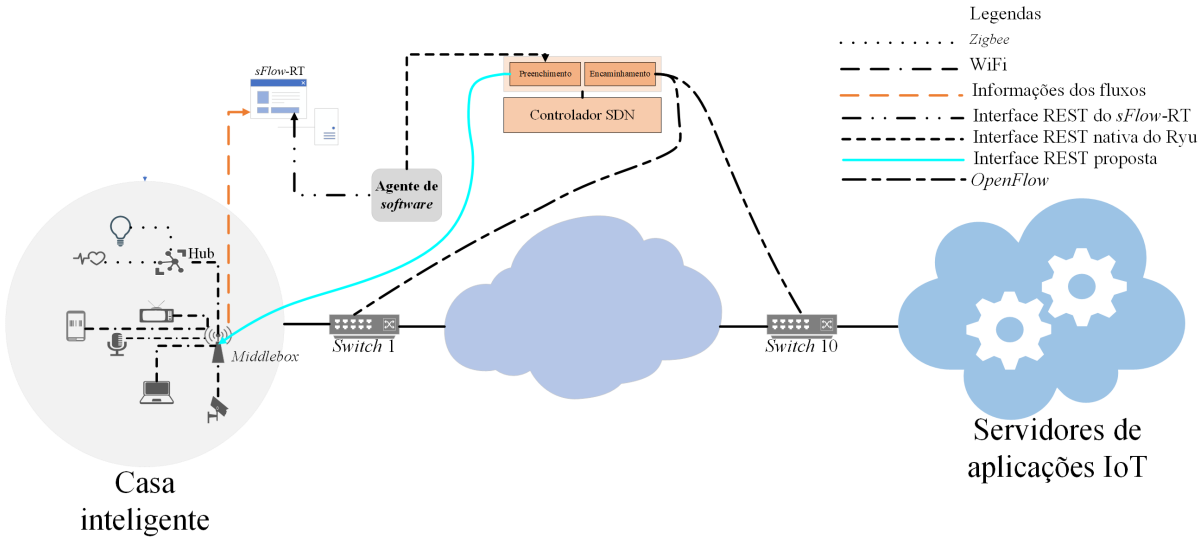
O emulador *Mininet* foi adotado na construção de uma rede SDN que emula uma infraestrutura para gerenciar o tráfego proveniente de uma casa inteligente (XU et al., 2016). A topologia de rede usada na avaliação do impacto no desempenho da comunicação é ilustrada na Figura 12. Os *hosts* emulados pelo *Mininet* representam a residência e os servidores IoT, ilustrados na Figura 12, em que o cliente e o servidor *iPerf* são implementados, respectivamente. O servidor *iPerf* gera arquivos de *logs* que contêm os valores das métricas avaliadas. A partir destes arquivos, as medições para as métricas examinadas foram extraídas e armazenadas em arquivos CSV, posteriormente analisados com o *software* R. Todo o tráfego transmitido entre cliente e servidor *iPerf* atravessa o mecanismo de preenchimento, posicionado no *middlebox* apresentado na Figura 12. A fim de garantir que os experimentos não fossem afetados pelo consumo dos recursos computacionais das VMs, o uso de CPU e memória foi monitorado com o *htop*, um utilitário do sistema operacional *Linux*.

Um mecanismo capaz de alterar atributos do tráfego de rede, que o atravessa, pode interferir negativamente no desempenho da comunicação. Por isso, os experimentos descritos nesta seção contribuem para elucidar a capacidade de processamento do tráfego do mecanismo de preenchimento. Esse mecanismo foi implementado de acordo com a solução descrita no Capítulo 3 e as estratégias existentes. Dessa forma, todos os procedimentos analisados encontram-se sob as mesmas condições experimentais.

Na topologia ilustrada na Figura 12, o mecanismo de preenchimento foi posicionado no *middlebox*, que encaminha o tráfego transmitido entre cliente e servidor *iPerf*, representados pelos equipamentos IoT e servidores de aplicações, respectivamente. Os *switches* 1 e 10, o *sFlow*-RT, o agente de *software* e o controlador estão localizados na infraestrutura do ISP. A nuvem esboçada na Figura 12 representa a infraestrutura do ISP que conecta a residência aos servidores de serviços IoT.

O *software Open vSwitch* foi usado pelo *Mininet* na criação de 10 *switches* (valor selecionado arbitrariamente) que, emulam essa nuvem. Esses *switches* foram posicionados em sequência para garantir que o tráfego gerado pelo *iPerf* atravesse todos os comutadores. Cada *switch*, no caminho entre origem e destino dos dados, gera alguma interferência no desempenho ao processar o tráfego, ainda que ínfimo. Por isso, considerando que estes experimentos avaliam o impacto no desempenho da comunicação introduzido pela solução de preenchimento, um número maior de *switches* poderia interferir nesta avaliação,

Figura 12 – Topologia empregada nos experimentos para avaliar o impacto da solução de preenchimento no desempenho da comunicação.



Fonte: elaborada pelo autor.

impactando negativamente na exatidão dos resultados.

O agente *sFlow* que, monitora a utilização da rede doméstica, foi implementado no *switch* 1 (ilustrado na Figura 12), que representa um equipamento da infraestrutura do ISP. Dessa forma, o roteador doméstico ficou dispensado do encargo de implementar o protocolo *sFlow*. O agente *sFlow* coleta as informações do tráfego ofuscado, após atravessar o mecanismo de preenchimento, em que os pacotes têm seus tamanhos modificados, e as transmite para o coletor *sFlow*. Para averiguar que os tamanhos foram modificados, o *software Tcpdump* foi executado no *switch* 1 e, capturou todo o tráfego que atravessa esse comutador e o salvou em arquivos PCAP. O *Wireshark* foi empregado na análise desses arquivos. Dessa forma, foi possível atestar que os tamanhos foram modificados pelo mecanismo de preenchimento.

4.4 CONSIDERAÇÕES FINAIS

Este capítulo apresentou em detalhes as etapas seguidas no projeto dos experimentos empregados na avaliação do protótipo da solução proposta. Foram descritas as ferramentas, metodologia, métricas e variáveis do experimento controlado adotadas nesta avaliação. Este capítulo também descreve detalhadamente o ambiente experimental usado na avaliação, incluindo os modelos de ameaça à privacidade, procedimentos para mensurar o desempenho da solução de preenchimento e as técnicas estatísticas aplicadas durante a análise dos resultados.

5 RESULTADOS DA AVALIAÇÃO

Neste capítulo, são apresentados os resultados obtidos nos experimentos descritos no capítulo anterior. Visto que foram conduzidos uma série de experimentos para avaliar o desempenho da solução descrita no Capítulo 3, os resultados são apresentados de acordo com o teste realizado para obtê-los. Esses resultados foram obtidos em quatro cenários diferentes, descritos no Capítulo 4: 1) observador externo; 2) observador interno; 3) *byte overhead*; 4) impacto no desempenho da comunicação. Também são apresentadas a discussão sobre os resultados dos testes de hipóteses descritos na seção 4.2.3. Por fim, na Seção 5.5, as questões de pesquisa apresentadas no Capítulo 1 são respondidas.

5.1 APRIMORAMENTO DA PRIVACIDADE

Como mencionado no Capítulo 4, um mecanismo para identificação de dispositivos IoT a partir do tamanho dos pacotes foi aplicado ao tráfego de rede disponibilizado pelos autores de (SIVANATHAN et al., 2018; REN et al., 2019). As Tabelas 4 e 5 apresentam os resultados obtidos na identificação de dispositivos IoT, considerando um cenário em que os tamanhos não são ofuscados por soluções para preenchimento de pacotes. A Tabela 4 apresenta os resultados para identificação dos 21 dispositivos IoT que produziram o tráfego disponibilizado por (SIVANATHAN et al., 2018), enquanto a Tabela 5 mostra o desempenho do mecanismo de classificação nos dados produzidos por 81 equipamentos, disponibilizados pelos autores de (REN et al., 2019). Na Tabela 5, estão inclusos os resultados para os *testbeds* US e UK. Dessa forma, foi garantida uma comparação entre múltiplos dispositivos do mesmo modelo, operados de formas distintas por usuários diferentes. Os resultados mostrados nas Tabelas 4 e 5 apresentam indícios sobre a capacidade de um observador inferir informações privadas dos indivíduos a partir da identificação dos dispositivos IoT.

Tabela 4 – Desempenho dos classificadores na identificação de 21 dispositivos IoT a partir do tráfego disponibilizado pelos autores de (SIVANATHAN et al., 2018). Estes classificadores foram avaliados com uma 10-*fold cross-validation*.

Métrica	k-NN	DT	RF
Acurácia média (%)	94	96	96
Recall médio (%)	94	96	96
F1-*score* médio (%)	94	96	96

Os resultados apresentados nas Tabelas 4 e 5 mostram que o mecanismo de classificação atingiu até 96% de acurácia na identificação de 21 dispositivos IoT. No mínimo, o mecanismo de classificação atingiu 88% de acurácia na distinção de 35 equipamentos IoT, considerando o desempenho do *Random Forest*, o mais acurado dentre os algoritmos analisados. Esses

Tabela 5 – Desempenho dos classificadores na identificação de 81 dispositivos IoT disponibilizados por (REN et al., 2019), dos quais 46 estiveram presentes no *testbed* configurado na *Northeastern University* (US) e 35 no *Imperial College* (UK).

Métrica	*Testbed*	k-NN	DT	RF
Acurácia média (%)	US	85	88	89
	UK	82	87	88
Recall médio (%)	US	85	88	89
	UK	82	87	88
F1-*score* médio (%)	US	85	88	89
	UK	82	87	88

resultados justificam as preocupações dos indivíduos sobre violações à privacidade a partir da análise dos dados capturados e transmitidos por dispositivos IoT.

Pode-se observar uma redução no desempenho do mecanismo para identificação de dispositivos IoT da Tabela 5 em relação à Tabela 4. Essa contração é compreensível, visto que, ao contrário dos dados disponibilizados em (SIVANATHAN et al., 2018), o tráfego disponibilizado por (REN et al., 2019) é proveniente de múltiplos dispositivos da mesma categoria, como sete *smart hubs*, onze câmeras de segurança, duas *smart* TVs e seis *smart speakers* (*Google Home Mini*, três modelos do *Amazon Echo* e dois modelos do *Harman Kardon*, *Invoke* e *Allure*). Teoricamente, os dispositivos pertencentes a uma categoria lidam com dados e eventos equivalentes, o que os induz a produzir tráfego com padrões semelhantes. Assim, é provável que o mecanismo de classificação tenha encontrado dificuldade para distinguir entre dispositivos da mesma categoria. Uma lista contendo todos os dispositivos analisados nesta tese está disponível nas tabelas apresentadas no Apêndice A.

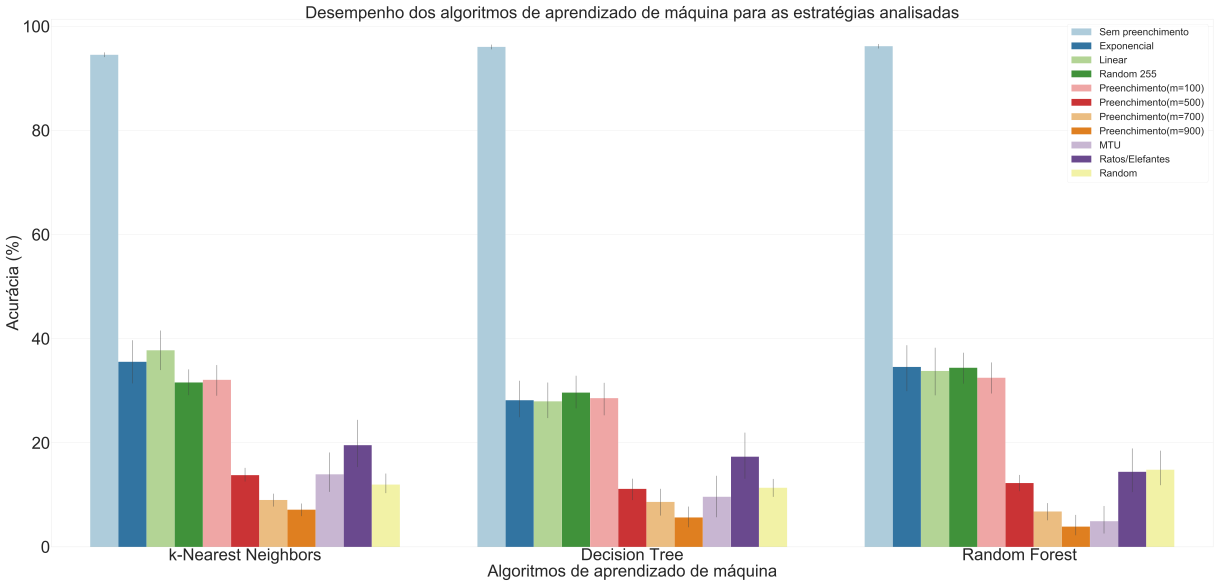
Considerando esse cenário com múltiplos dispositivos da mesma categoria, o mecanismo de classificação atingiu resultados interessantes. Esses resultados apresentam indícios do desempenho que um observador pode obter na identificação dos dispositivos IoT presentes na residência de suas vítimas. A identificação dos dispositivos IoT representa uma etapa importante no processo de inferência das informações privadas dos seus usuários (REN et al., 2019). Após a identificação de um dispositivo, é possível inferir eventos ocorridos nos equipamentos IoT. Assim, impossibilitar a identificação dos dispositivos dificulta a inferência de informações pessoais privadas.

5.1.1 Observador externo

Esta seção apresenta os resultados para o aprimoramento na privacidade em relação a um observador externo a uma residência. Como o propósito da estratégia de preenchimento proposta é inibir a identificação dos dispositivos IoT, quanto menor o desempenho dos algoritmos analisados, melhor. Os mesmos resultados foram obtidos para as métricas

acurácia, *recall* e F1-*score*. As Figuras 13, 14 e 15 apresentam os resultados para esse cenário em cada um dos três conjuntos de dados analisados. Os resultados mostrados nestas figuras sugerem que, o número de *bytes* inseridos nos pacotes influencia o nível de aprimoramento na privacidade, como discutido nesta tese. Nos *datasets* disponibilizados pelos autores de (SIVANATHAN et al., 2018) e (REN et al., 2019) (dados capturados no *testbed* US), em que os resultados são apresentados nas Figuras 13 e 14, respectivamente, a solução de preenchimento foi capaz de proporcionar uma melhoria na privacidade superior àquela assegurada pela estratégia MTU.

Figura 13 – Desempenho dos algoritmos de aprendizado de máquina analisados no aprimoramento da privacidade nos dados disponibilizados por (SIVANATHAN et al., 2018).



O nível *preenchimento*($m = 900$) foi capaz de reduzir a acurácia do *Random Forest* na identificação de dispositivos IoT de 96% para 3,92% (tráfego disponibilizado pelos autores de (SIVANATHAN et al., 2018)) e de 89% para 2,09% (considerando os dados disponibilizados pelos autores de (REN et al., 2019) capturados no *testbed* US). Mesmo no pior cenário, dados disponibilizados pelos autores de (REN et al., 2019) capturados no *testbed* UK, a solução reduziu a acurácia do *Random Forest* de 88% para 10,12%, uma significativa degradação no desempenho. A partir dos resultados apresentados nas Figuras 13, 14 e 15, é possível observar que o desempenho das estratégias de preenchimento varia entre os *datasets* analisados. Esses resultados sugerem que a eficácia do preenchimento varia entre diferentes ambientes de rede doméstica, como esperado.

Ao alterar o tamanho dos pacotes gerados por diferentes dispositivos para os mesmos valores, a solução de preenchimento contribui para eclipsar as particularidades aos diferentes equipamentos IoT. Dessa forma, os algoritmos treinados com os dados gerados pelos dispositivos IoT, que contêm tamanhos característicos aos equipamentos analisados,

Figura 14 – Desempenho dos algoritmos de aprendizado de máquina no aprimoramento da privacidade a partir do tráfego disponibilizado pelos autores de (REN et al., 2019), capturado no *testbed* US.

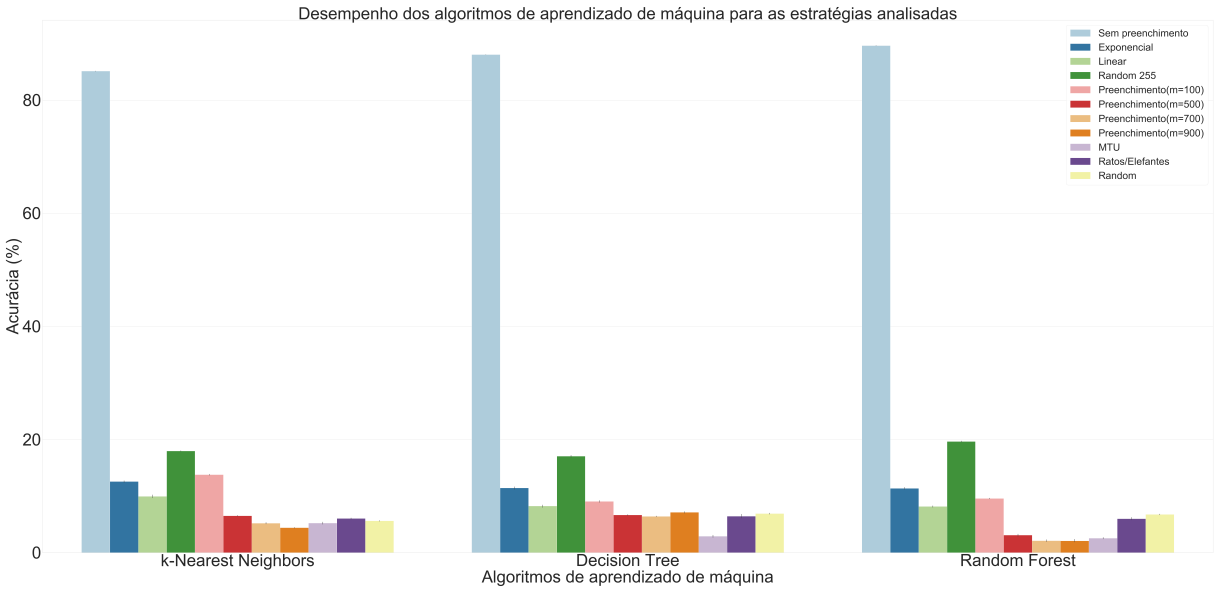
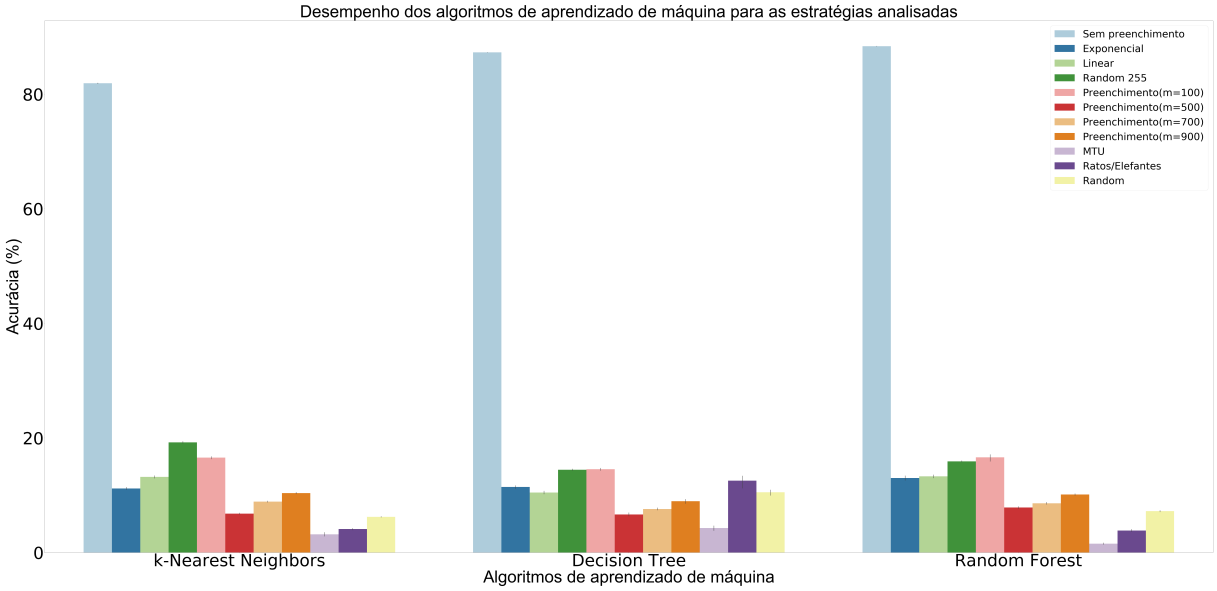


Figura 15 – Desempenho dos algoritmos de aprendizado de máquina no aprimoramento da privacidade a partir do tráfego disponibilizado pelos autores de (REN et al., 2019), capturado no *testbed* UK.



encontram dificuldade para diferenciar os dispositivos a partir do tráfego ofuscado pela solução.

Um dos fatores que pode contribuir para a degradação no desempenho dos algoritmos avaliados consiste na gradual uniformidade dos tamanhos de pacotes ofuscados do nível 100 a 900. Quanto mais elevado o nível da estratégia, mais uniformes são os tamanhos. Por exemplo, no nível 100 da estratégia, no mínimo, aproximadamente 64% dos pacotes têm seus tamanhos modificados para 300 *bytes*, enquanto no nível 900 por volta de 70% dos tamanhos são iguais a 900 *bytes*. Assim, quando o nível de preenchimento é incrementado, um classificador enfrenta um cenário mais desafiador para identificar os padrões aprendidos a partir do tráfego IoT original.

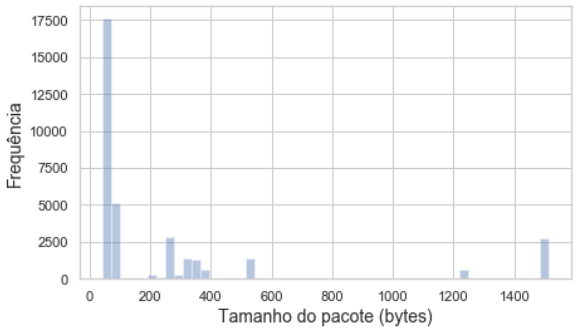
A sensibilidade dos algoritmos para aprendizado de máquina em relação ao preenchimento pode estar relacionada ao tamanho do pacote original. Como discutido anteriormente, o preenchimento é apropriado para tamanhos pequenos (próximos a 100 *bytes*), logo, valores superiores a 1000 *bytes* podem ser pouco sensíveis às modificações conduzidas. Por isso, os resultados para o *testbed* UK apontam para uma menor variabilidade entre os níveis da solução de preenchimento, quando comparado aos outros *datasets*.

Presumivelmente, o desempenho da solução de preenchimento foi superado pela estratégia MTU nos dados provenientes do *testbed* UK (Figura 15) porque nesse conjunto de dados, 26,96% dos tamanhos encontram-se acima de 900 *bytes* (essa proporção é de 16,48% e 9,80% para o *testbed* US e os dados disponibilizados por (SIVANATHAN et al., 2018), respectivamente). Assim, 26,96% dos pacotes capturados no *testbed* UK foram submetidos a uma diminuta alteração em seus tamanhos, mesmo no nível mais elevado da solução.

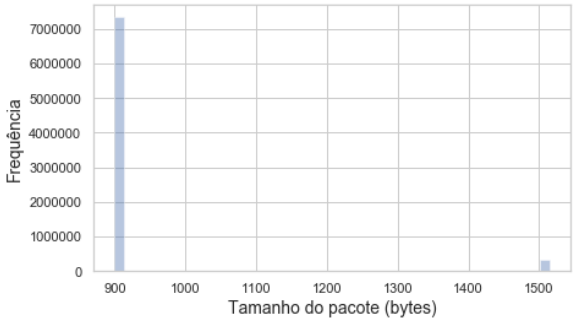
A Figura 16 apresenta uma comparação entre as distribuições do tamanho original e ofuscado no nível 900. As Figuras 16(a) e 16(b) ilustram a distribuição do tamanho do pacote original disponibilizado pelos autores de (SIVANATHAN et al., 2018) e sua contraparte ofuscada, respectivamente. A Figura 16(c) apresenta o histograma para o tamanho dos pacotes disponibilizados pelos autores de (REN et al., 2019) capturados no *testbed* US, enquanto a Figura 16(d) mostra os valores ofuscados a partir destes dados. As Figuras 16(e) e 16(f) apresentam a distribuição de probabilidade para o tamanho original e ofuscado proveniente do *testbed* UK, nesta ordem. A Figura 16 mostra que, em todos os *datasets* analisados, a maioria dos tamanhos são modificados para 900 *bytes*. A Figura 16(e) mostra que o tráfego capturado no *testbed* UK apresenta uma maior concentração de tamanhos acima dos 900 *bytes* que os outros *datasets*. Dessa forma, os dados provenientes do *testbed* UK estão menos suscetíveis ao preenchimento que os outros conjuntos de dados.

É factível que essa significativa degradação no desempenho dos algoritmos analisados seja decorrente da distribuição do tamanho do pacote que, concentra-se majoritariamente abaixo de 900 *bytes*. Dessa forma, elevar os tamanhos para 900 *bytes* garante que a maioria dos pacotes compartilhem o mesmo tamanho, argumento adotado para justificar a suposta eficiência da estratégia MTU (UDDIN; NADEEM; NUKAVARAPU, 2019), sem incrementar

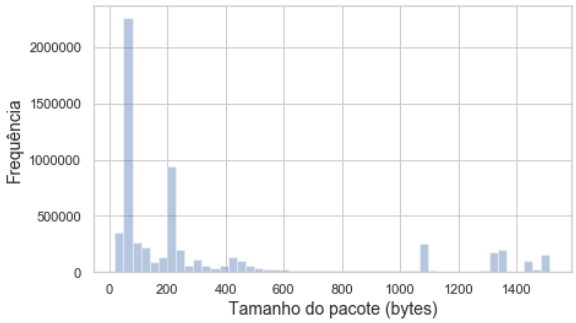
Figura 16 – Comparação entre a distribuição do tamanho do pacote original e ofuscado no nível 900.



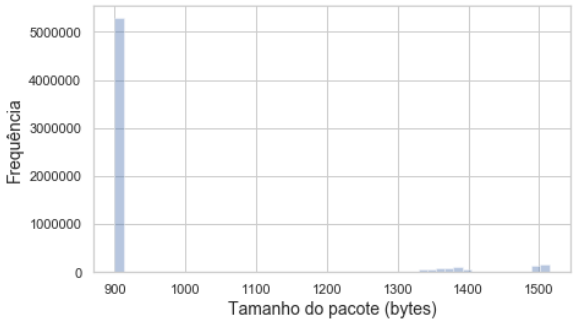
(a) Distribuição do tamanho original.



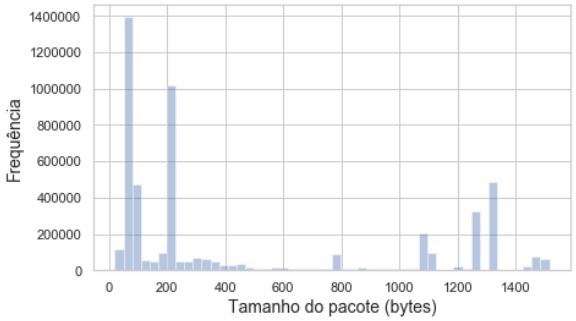
(b) Histograma do tamanho ofuscado.



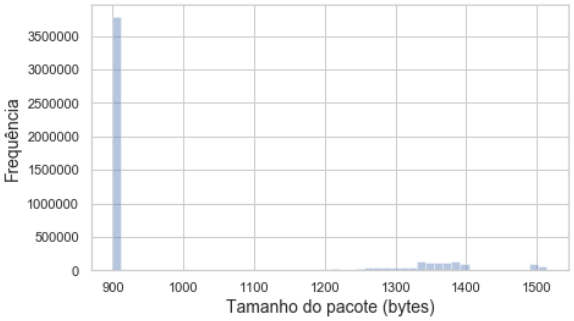
(c) Tamanho original.



(d) Tamanho ofuscado.



(e) Histograma do tamanho original.



(f) Distribuição do tamanho ofuscado.

os seus valores para o máximo permitido. Assim, viabilizando um aprimoramento na privacidade comparável à estratégia MTU a um custo significativamente menor. Neste ponto, é essencial destacar que o objetivo da proposta desta tese consiste em balancear o *trade-off* privacidade-*overhead* ao adaptar o número de *bytes* em resposta às variações na utilização de uma rede. Assim, mesmo que a solução de preenchimento baseada na proposta desta pesquisa seja superada pela MTU, não há prejuízo ao seu objetivo.

Quanto à escalabilidade da solução descrita no Capítulo 3, os resultados também mostram-se promissores. As Figuras 13 e 14 mostram que a solução de preenchimento reduziu o desempenho do *Random Forest* para 3,92% e 2,09%, quando ofuscou dados de 21 e 46 dispositivos IoT, respectivamente. Esses resultados sugerem que a elevação no número de equipamentos IoT foi incapaz de interferir negativamente no desempenho da solução. É provável que o desempenho de um mecanismo para ofuscar metadados do tráfego seja beneficiado pela elevação no número de dispositivos IoT protegidos (Hafeez; Antikainen; Tarkoma, 2019). Nos dados provenientes do *testbed* UK, o desempenho do *Random Forest* foi de 10,12%. É provável que os resultados obtidos nos dados originários do *testbed* UK (apresentados na Figura 15), composto por 35 dispositivos, sejam decorrentes do elevado tamanho dos pacotes contidos nesse conjunto de dados, não necessariamente relacionados à quantidade de dispositivos.

5.1.2 O provedor da solução é um adversário

Este cenário pressupõe que os fornecedores de soluções de preenchimento deduzem informações sobre os usuários de seus produtos. Os resultados da avaliação deste cenário são apresentados nas Figuras 17, 18 e 19. Como esperado, o provedor da solução de preenchimento possui uma vantagem na identificação de dispositivos em comparação com observadores externos. No entanto, o desempenho dos classificadores é reduzido significativamente, mesmo para o provedor da solução. Novamente, incrementar o número de *bytes* influencia a redução no desempenho dos classificadores na identificação de dispositivos IoT.

O desempenho da solução de preenchimento foi próximo ao da solução MTU. Esses resultados contribuem para mostrar ao usuário da solução o quanto o mesmo precisará confiar no provedor. O procedimento *random* apresentou o melhor desempenho quando comparado aos seus pares. Esses resultados sugerem que os valores selecionados aleatoriamente podem superar os tamanhos fixos para proteção contra o fornecedor de uma solução de preenchimento. Contudo, como discutido anteriormente, estratégias que selecionam o número de *bytes* aleatoriamente podem incorrer em elevado *overhead*. Logo, para ponderar o *trade-off* privacidade-*overhead*, é essencial determinar o *byte overhead* produzido por mecanismos de preenchimento.

Figura 17 – Desempenho das estratégias de preenchimento analisadas. Resultados obtidos a partir dos dados disponibilizados pelos autores de (REN et al., 2019), considerando o *testbed* US.

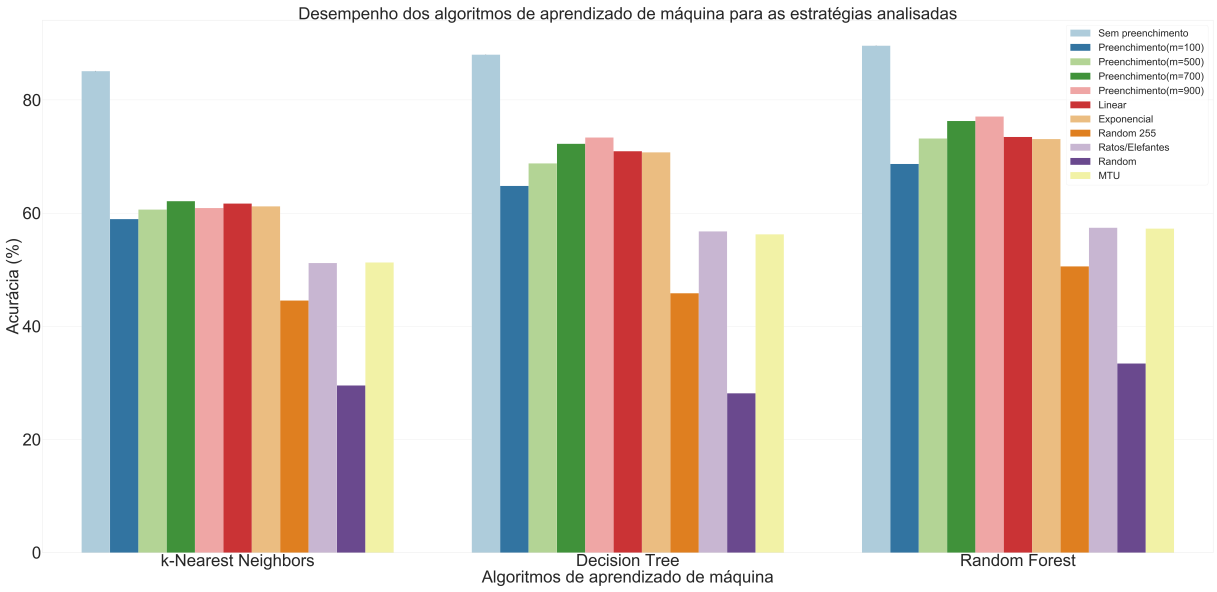


Figura 18 – Desempenho das estratégias de preenchimento analisadas. Resultados obtidos a partir dos dados disponibilizados pelos autores de (REN et al., 2019), considerando o *testbed* UK.

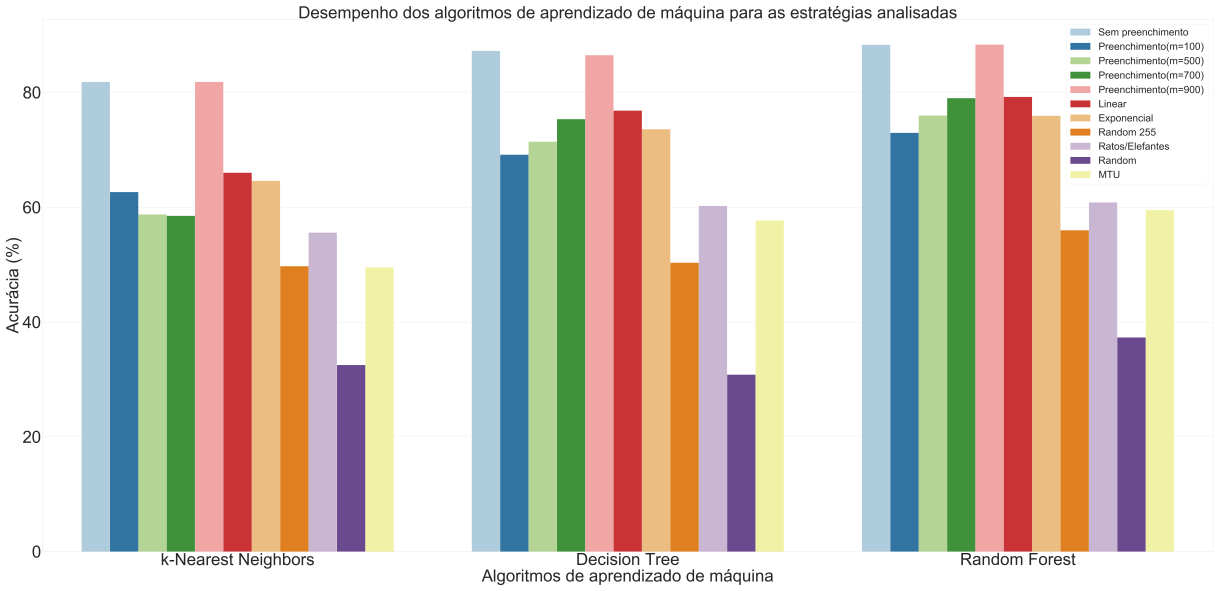
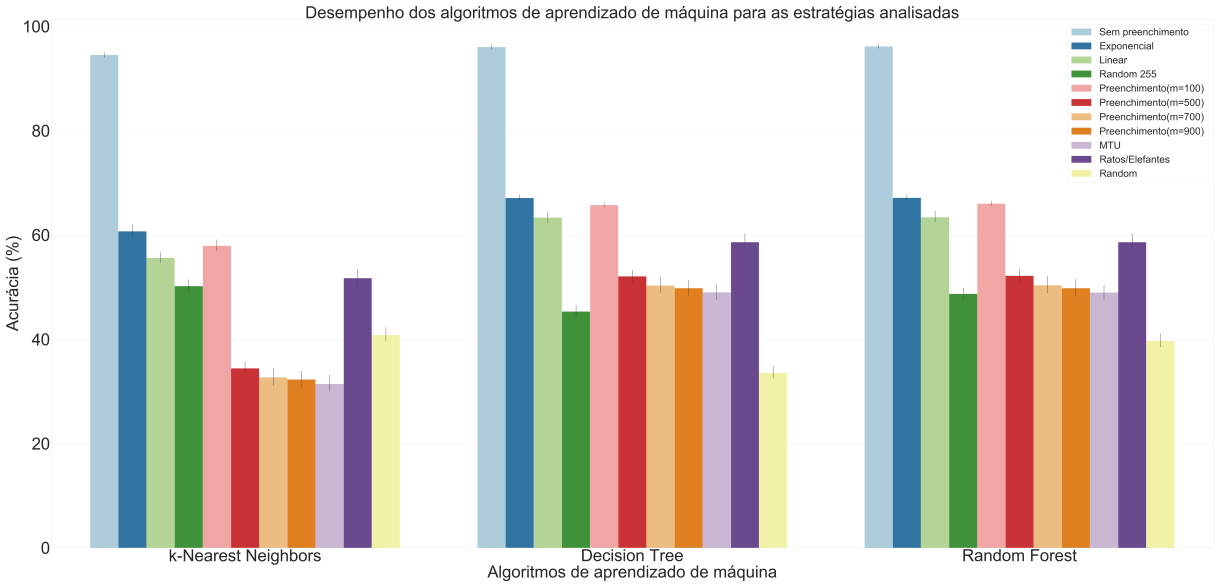


Figura 19 – Desempenho das estratégias de preenchimento analisadas. Resultados obtidos a partir dos dados disponibilizados pelos autores de (SIVANATHAN et al., 2018).



5.2 BYTE OVERHEAD

Como discutido anteriormente, o preenchimento de pacotes contribui para aprimorar a privacidade enquanto impõe o custo de produzir *overhead*. Portanto, é fundamental analisar o *overhead* gerado pela solução para considerar o *trade-off* com a privacidade. Além do *byte overhead*, foram computadas as estatísticas média, mediana e moda para o tamanho ofuscado. Essas medições contribuem para a comparação entre as diferentes estratégias de preenchimento analisadas. Os resultados para essas estatísticas são apresentados no apêndice B. As Figuras 20, 21 e 22 apresentam o *byte overhead* produzido por cada uma das estratégias analisadas e pelos níveis da solução.

Os resultados mostrados nas Figuras 20, 21 e 22 sugerem que, mesmo no nível mais alto da solução, o *byte overhead* produzido é significativamente menor que o induzido pela estratégia MTU. O nível 100 da solução de preenchimento introduziu o menor *overhead*, semelhante às estratégias linear, exponencial e *random* 255. Os níveis 500 e 700 da solução geram um *overhead* mais elevado que as estratégias linear, exponencial e *random* 255, ao mesmo tempo que mantém-se inferior a ratos/elefantes e *random*. Como esperado, a estratégia MTU produziu o maior *byte overhead*. O *byte overhead* está relacionado diretamente com o tamanho original dos pacotes gerados pelos dispositivos IoT. Desta forma, quanto maior o pacote, menor é o *overhead* e vice-versa. Por exemplo, os *traces* capturados no *testbed* UK apresentam um *byte overhead* menor porque os tamanhos originais são os maiores dentre os conjuntos de dados analisados.

A Figura 23 apresenta a relação entre a acurácia alcançada pelo *Random Forest* e o *byte overhead* médio produzido pelos mecanismos avaliados nos dados disponibilizados

Figura 20 – *Byte overhead* produzido pelas estratégias de preenchimento ao tamanho do pacote disponibilizado pelos autores de (SIVANATHAN et al., 2018).

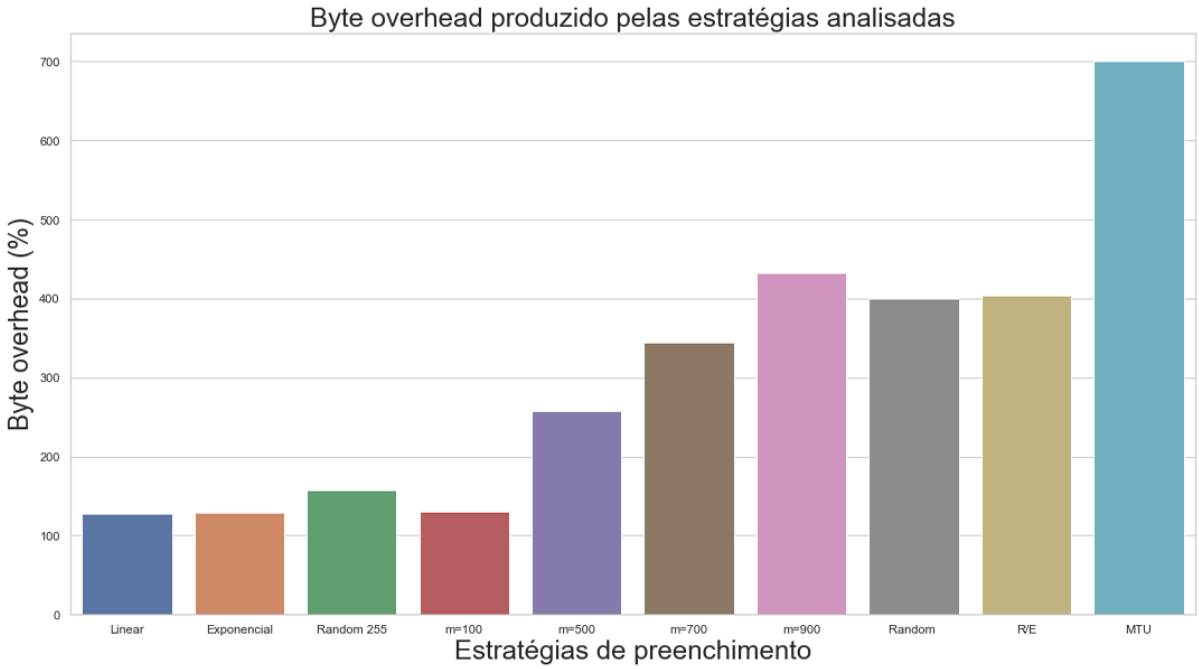
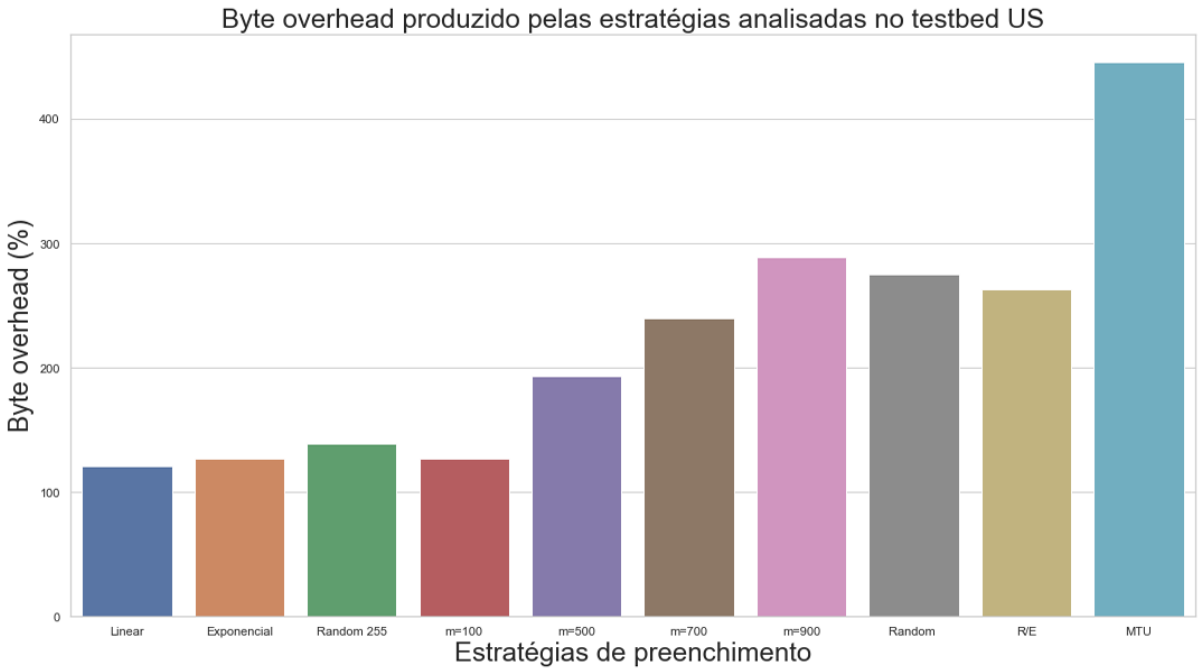


Figura 21 – *Byte overhead* induzido pelas estratégias de preenchimento ao tamanho do pacote capturado no *testbed* US disponibilizado em (REN et al., 2019).



pelos autores de (SIVANATHAN et al., 2018). Esses resultados indicam que, para alcançar um maior aprimoramento na privacidade é necessário admitir um *byte overhead* mais elevado. Essa mesma relação pode ser observada nos dados disponibilizados pelos autores

Figura 22 – *Byte overhead* introduzido pelas estratégias de preenchimento ao tamanho do pacote proveniente do *testbed* UK disponibilizado pelos autores de (REN et al., 2019).

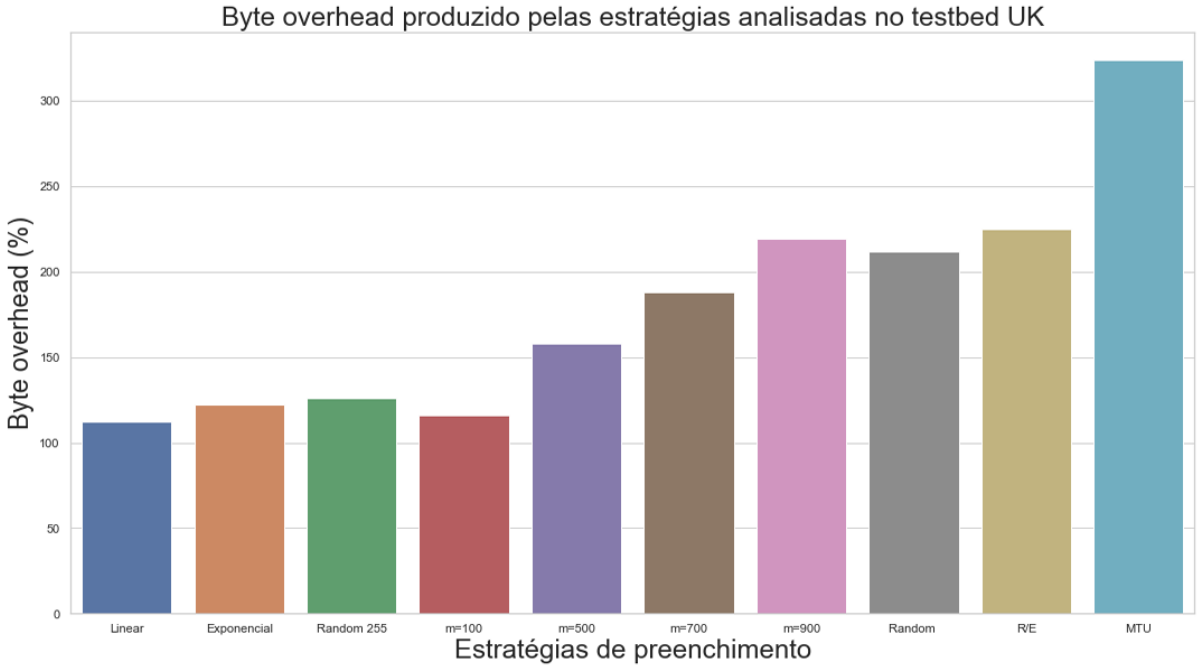
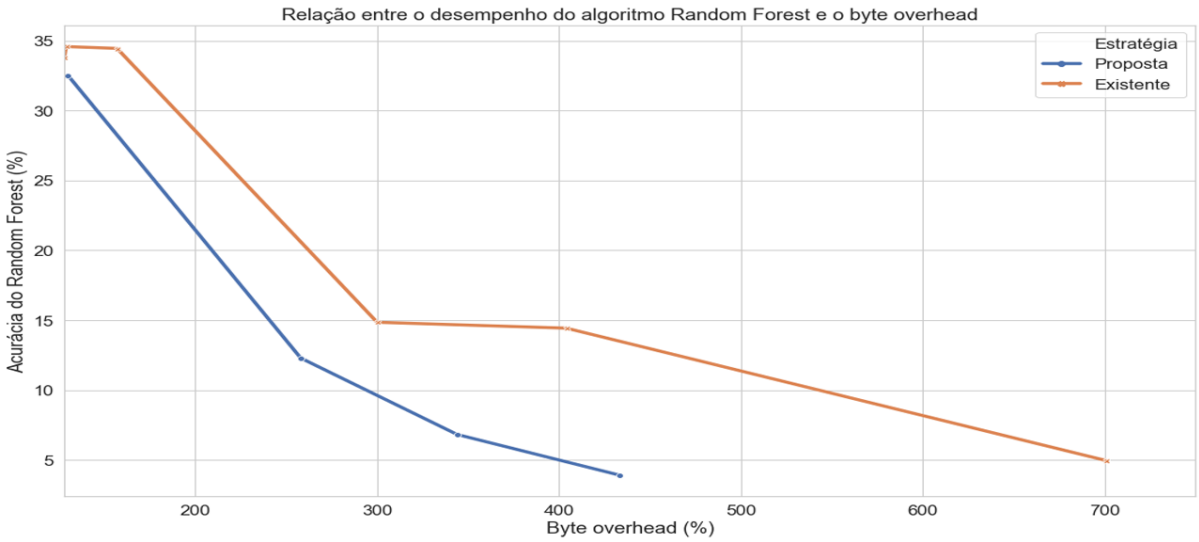


Figura 23 – *Trade-off* entre privacidade e *byte overhead* para os dados disponibilizados pelos autores de (SIVANATHAN et al., 2018).



(REN et al., 2019). Essa relação entre privacidade e *byte overhead*, torna a capacidade da solução em gerar menos *overhead* do que a MTU e, ao mesmo tempo, proporcionar uma melhoria comparável na privacidade, um aspecto interessante da solução apresentada nesta tese. Dado que os *bytes* extras aumentam o volume de tráfego, o nível de *byte overhead* deve influenciar o desempenho da comunicação, pois consome os recursos de uma rede.

5.3 DESEMPENHO DA COMUNICAÇÃO

Como discutido na Seção 4, um mecanismo de preenchimento, atuando como *middlebox*, interfere negativamente no desempenho da comunicação. Os resultados dos experimentos descritos na Seção 4.3.2 sugerem que o protótipo foi capaz de processar até aproximadamente 5Mbps, incorrendo em um impacto à *goodput* negligenciável. Logo, o protótipo foi capaz de gerenciar até 5 vezes o pico no tráfego produzido pelos dispositivos IoT analisados pelos autores de (SIVANATHAN et al., 2018). Por isso, é esperado que a *goodput* permaneça abaixo de 5Mbps, mesmo que o *iPerf* produza tráfego a uma velocidade superior a este valor.

Além da interferência do protótipo, a degradação no desempenho da comunicação é decorrente da elevação na utilização de uma rede. A Figura 24 ilustra a ampliação no volume de tráfego induzido por cada nível da solução, enquanto a Figura 25 apresenta os resultados para os mecanismos existentes. Como os resultados para a estratégia Ratos/Elefantes são similares aos da solução MTU, os mesmos foram omitidos da Figura 25 para evitar sobreposição entre essas soluções. O mesmo ocorreu com as estratégias linear e exponencial, em que a segunda foi omitida. Essas figuras indicam que o número de *bytes* inseridos nos pacotes influencia o volume de tráfego de uma rede, que interfere na utilização de um enlace. A partir de aproximadamente 5Mbps do tráfego gerado pelo *iPerf*, a elevação na utilização da rede cessa, pois o protótipo atingiu o seu limite. Nas Figuras 24 e 25, é possível observar que há uma relação entre o *byte overhead* mostrado na Figura 20 e o incremento no volume de dados em uma rede. Por exemplo, o nível 900 da solução incrementa o tamanho do pacote e a utilização da rede em aproximadamente 4 vezes.

As Figuras 24 e 25 mostram que, enquanto a utilização orgânica permanecer suficientemente baixa (ex.: 10%), mesmo as estratégias que produzem o *byte overhead* mais elevado são incapazes de exaurir a capacidade da rede (a relação superar 100%). Contudo, quando o volume de tráfego original é elevado, essas estratégias induzem o enlace à exaustão, o que deve degradar o desempenho das aplicações de rede. Como discutido na Seção 4.3.2, mesmo em seu nível máximo, através das estratégias MTU e *preenchimento*($m = 900$), o preenchimento é incapaz de produzir efeitos negativos significativos no desempenho da comunicação, desde que exista capacidade ociosa na rede suficiente para acomodar os *bytes* extras. Esses resultados sugerem que, é possível evitar o impacto no desempenho da comunicação, mesmo no nível máximo de preenchimento, desde que a rede opere abaixo da sua capacidade.

Figura 24 – Relação entre o volume de tráfego e a capacidade do enlace para os níveis da solução de preenchimento.

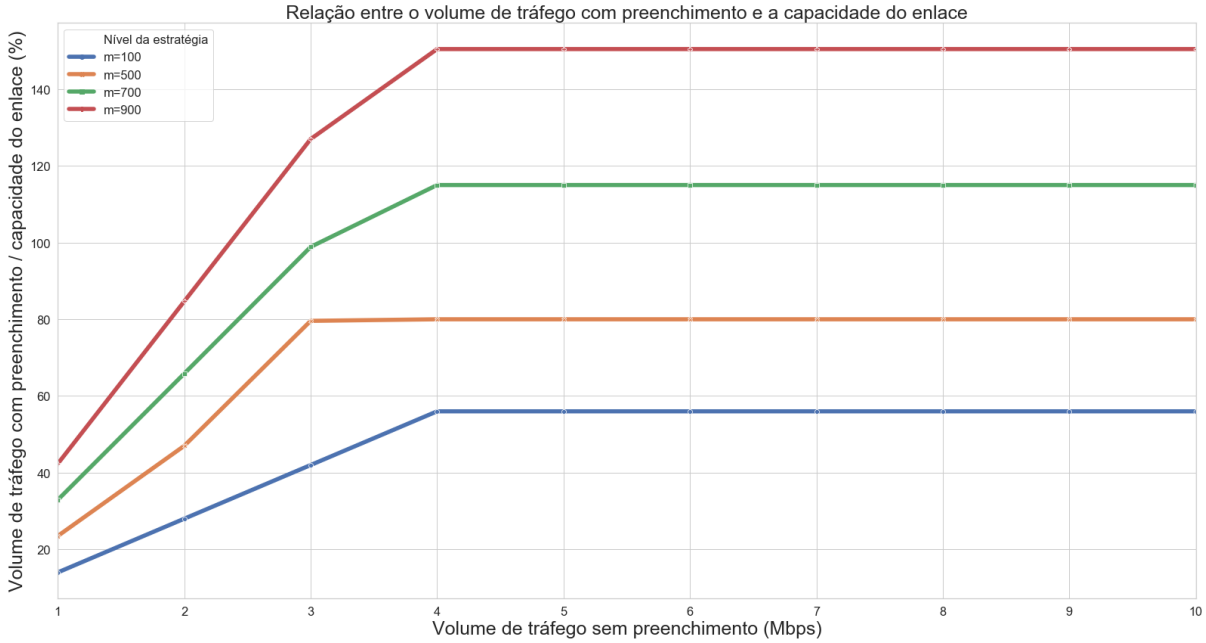
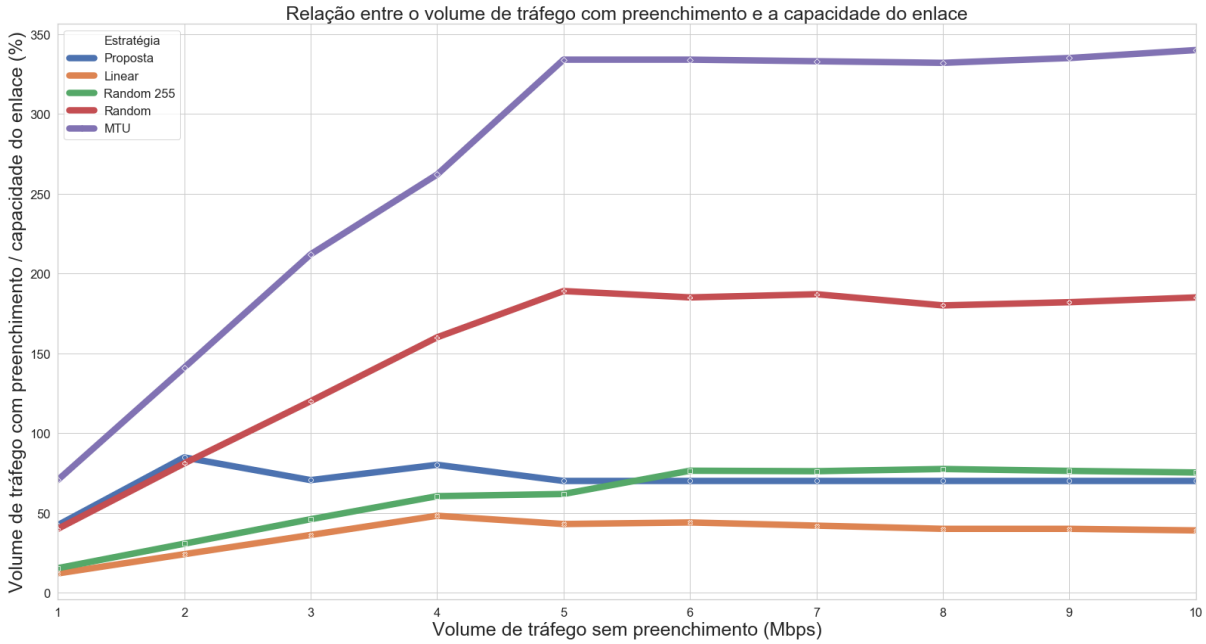


Figura 25 – Relação entre o volume de tráfego e a capacidade do enlace para os mecanismos de preenchimento analisados e para a proposta de adaptar o número de *bytes* em resposta às oscilações no volume de dados.



A Figura 26 apresenta o impacto no desempenho da comunicação produzido pela solução de preenchimento, considerando as métricas *goodput*, *jitter* e perda de pacotes. A partir dos resultados apresentados na Figura 26, pode-se inferir que, o tráfego extra tende a impactar no desempenho quando suficiente para exaurir a capacidade da rede.

Portanto, nesse cenário, faz-se necessário reduzir a quantidade de dados na rede para evitar sobrecarga da infraestrutura. O acréscimo no número de *bytes* produz uma redução na *goodput* da rede, quando o tráfego resultante da modificação do tamanho do pacote é suficiente para exceder a capacidade da rede. Esse impacto negativo no desempenho da comunicação também pode ser observado nas métricas *jitter* e perda de pacotes, que são elevadas de acordo com o aditamento no número de *bytes*. A ausência de uma tendência clara nos resultados do *jitter* está relacionada com a imprecisão do *Mininet* na medição do tempo (Wu et al., 2017).

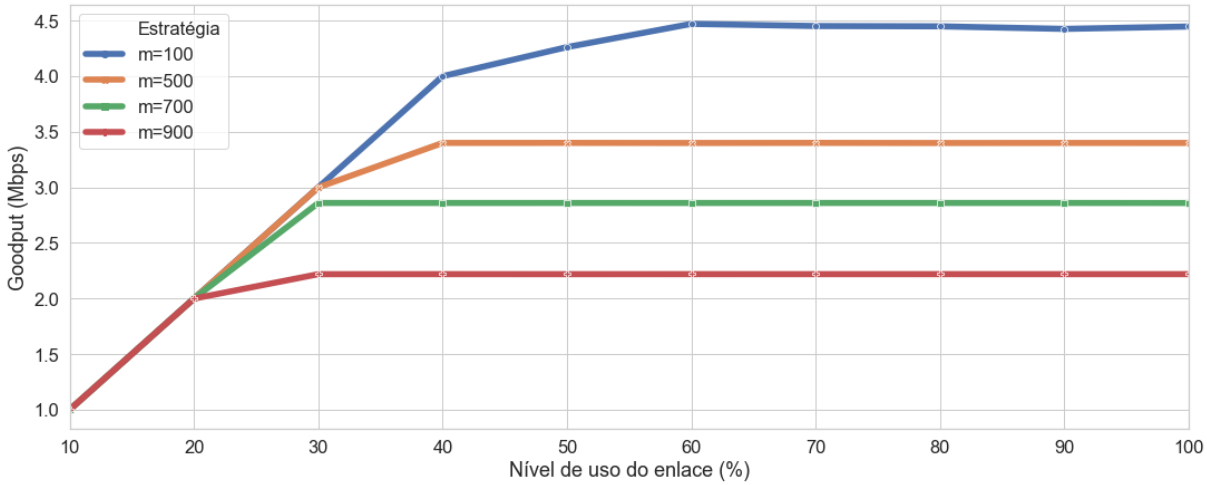
A Figura 27 ilustra o *overhead* na comunicação introduzido pela solução de preenchimento ao adaptar o número de *bytes* em resposta às oscilações na utilização da rede. Novamente, os resultados para as estratégias Ratos/Elefantes, linear e *Random* 255 foram omitidos para evitar sobreposições com as outras soluções existentes. A Figura 27 também apresenta a interferência gerada pelos principais mecanismos de preenchimento existentes. Os resultados apresentados nessa figura sugerem que, o *byte overhead* produzido pelas estratégias de preenchimento demonstrado na Figura 20 influencia no impacto às métricas apresentadas nas Figuras 26 e 27.

As estratégias que produzem menor *overhead*, como *preenchimento*($m = 100$), linear, exponencial e *random* 255, geram um impacto diminuto no desempenho da comunicação. Esses resultados podem ser decorrentes da elevação no tráfego produzido por essas estratégias ser insuficiente para exaurir o enlace. Por outro lado, as estratégias que produzem maior *byte overhead* também introduzem o mais elevado impacto no desempenho da comunicação, considerando as métricas *goodput*, *jitter* e perda de pacotes.

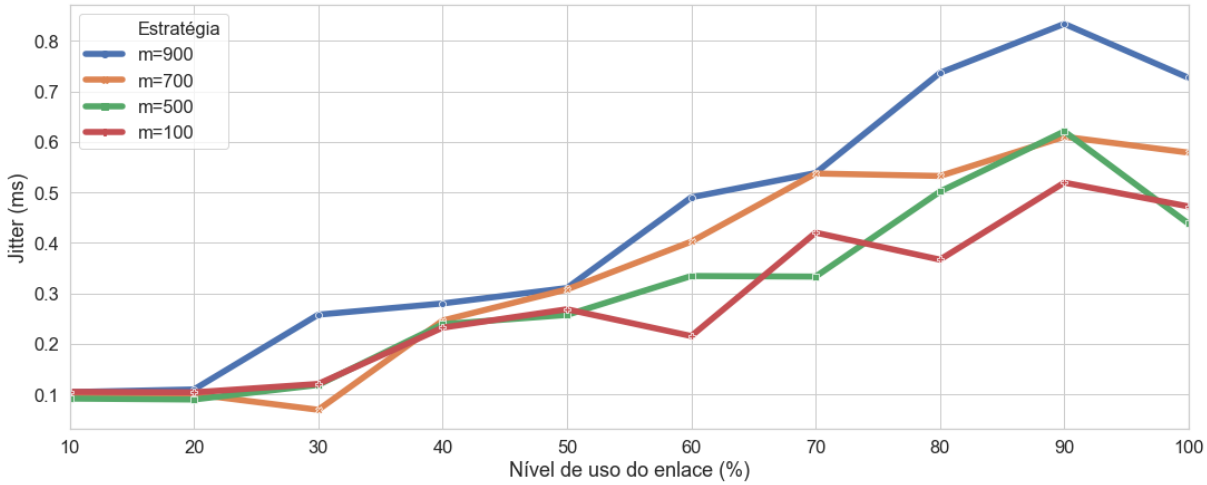
Os resultados mostrados nas Figuras 26 e 27 apresentam indícios que apontam para outra hipótese discutida na Seção 4.3.2: as estratégias que geram o menor *overhead* podem ser empregadas quando a rede estiver próxima à exaustão. Esse cenário é factível porque, ao incorrerem em um reduzido *overhead*, estratégias como *preenchimento*($m = 100$), linear, exponencial e *random* 255, produzem um impacto no desempenho da comunicação desprezível. Portanto, essas estratégias são mais adequadas quando a utilização da rede está próxima ao limite da capacidade da infraestrutura. Os resultados apresentados na

Ao adaptar o número de *bytes* em resposta às variações na utilização da rede, é possível manter o impacto no desempenho da comunicação no mínimo, mesmo quando o tráfego orgânico tende a exaurir os recursos da infraestrutura. Os resultados apresentados nesta seção não permitem rejeitar a hipótese de que, a solução proposta é capaz de aprimorar a privacidade ao considerar o nível de utilização de uma rede. Esses resultados também sugerem que a solução foi capaz de cumprir o seu propósito de adaptar o número de *bytes*. Contudo, é importante mensurar a incerteza sobre o desempenho da solução. Por isso, conduzimos testes de hipóteses que nos auxiliam na tomada de decisão sobre os resultados obtidos.

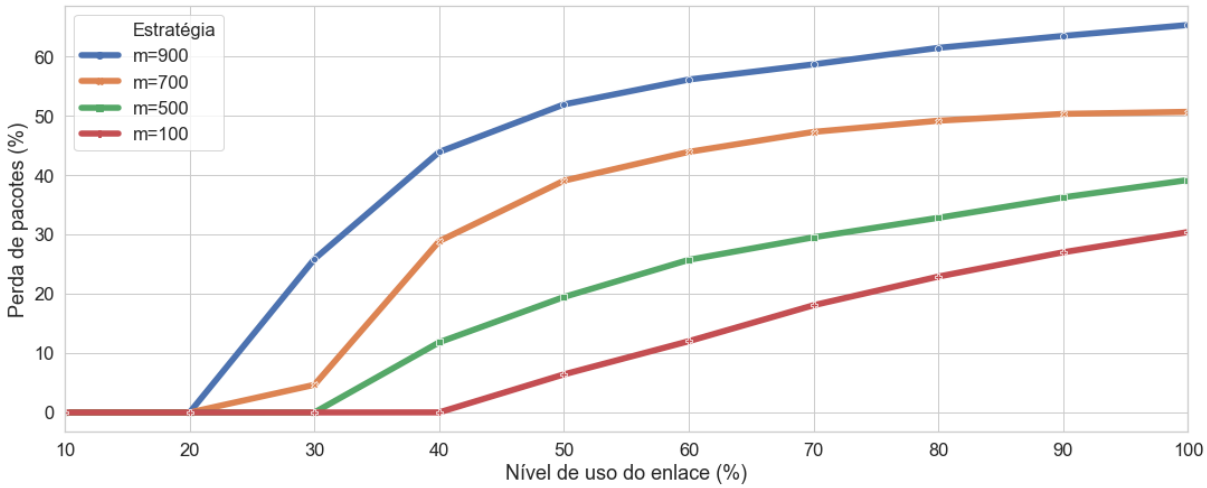
Figura 26 – Impacto dos diferentes níveis de preenchimento no desempenho da comunicação, considerando as métricas *goodput*, *jitter* e perda de pacotes.



(a) Resultados para a *goodput*.

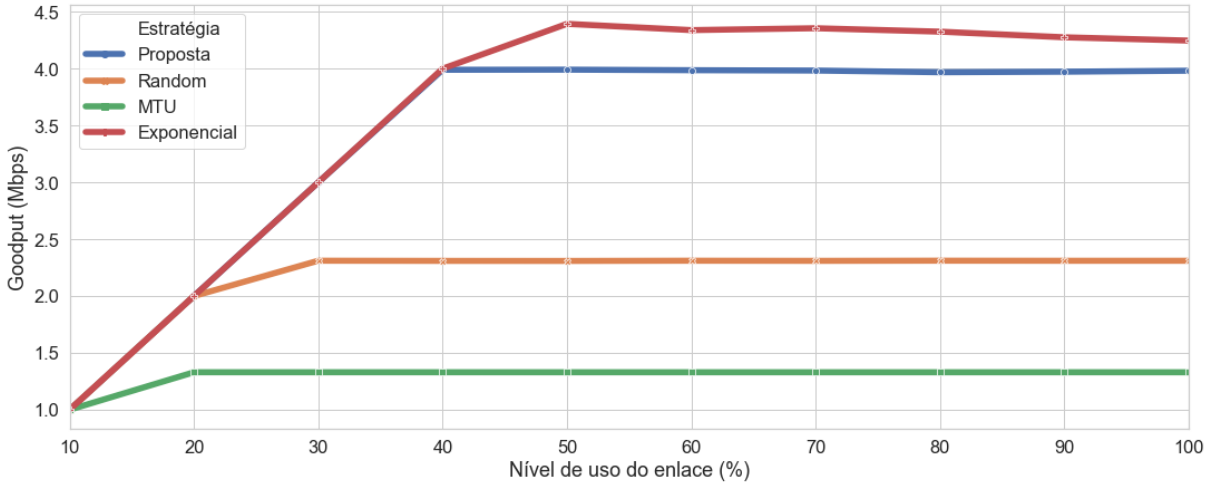


(b) *Jitter* obtido nos experimentos.

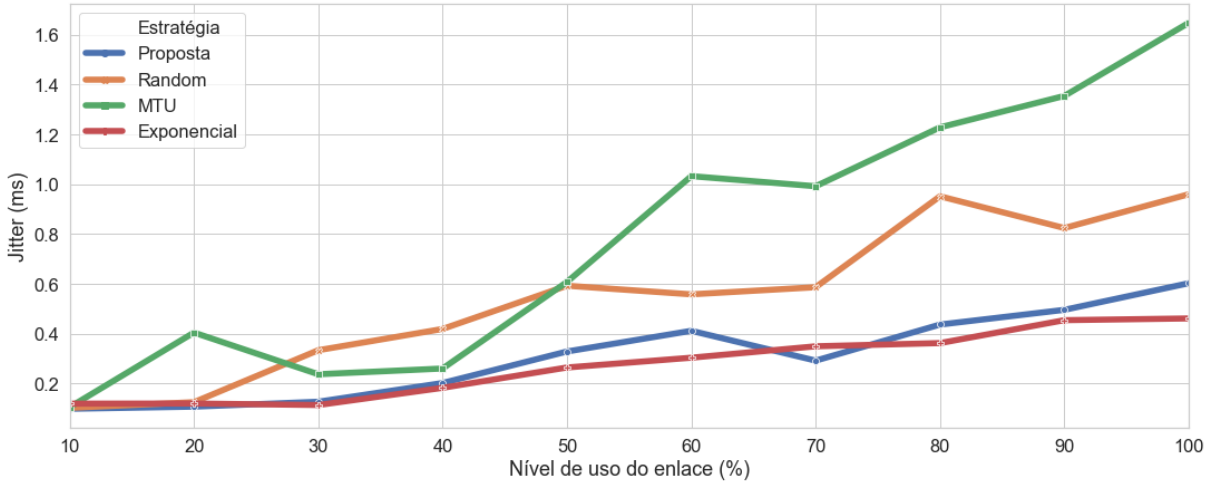


(c) Perda de pacotes acarretada pela solução.

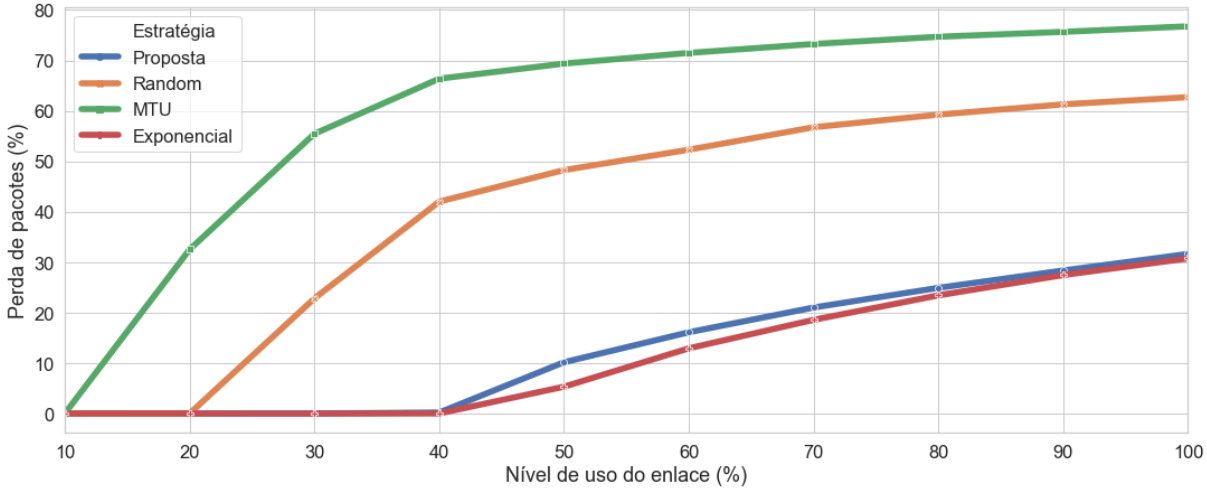
Figura 27 – Interferência no desempenho da comunicação introduzida pelos mecanismos de preenchimento analisados e pela proposta de balancear o *trade-off* privacidade-*overhead*.



(a) Resultados alcançados para a *goodput* do tráfego.



(b) *Jitter* mensurado nos experimentos.



(c) Perda de pacotes introduzida pelos mecanismos analisados.

5.4 TESTES DE HIPÓTESES

Os testes de hipótese conduzidos neste trabalho têm como objetivo responder às seguintes questões: 1) qual estratégia apresentou os melhores resultados no aprimoramento da privacidade; 2) qual mecanismo introduziu um maior *overhead*. Os testes de hipóteses foram aplicados somente quando os resultados são nitidamente próximos.

Como o *Random Forest* apresentou o melhor desempenho na identificação dos dispositivos IoT, que representa uma maior ameaça à privacidade em relação aos outros algoritmos analisados, foram utilizados somente os resultados produzidos por este algoritmo. Visto que as métricas acurácia, *recall* e F1-*score* apresentaram resultados idênticos, faz-se desnecessário analisar os dados de todas individualmente. Por isso, os testes de hipóteses foram aplicados somente nos resultados para a acurácia obtida com o *Random Forest*.

O teste de *Friedman* foi usado para determinar a existência de diferença no desempenho das estratégias avaliadas, em ambos os cenários de observador externo e interno. Este teste também foi aplicado na análise dos resultados para o *byte overhead*. As hipóteses para o teste de *Friedman* são apresentadas abaixo.

H_0 : não há diferença significativa no desempenho dos mecanismos avaliados.

H_1 : existe uma diferença significativa no desempenho dos mecanismos avaliados.

Em todos os dados analisados, o teste de *Friedman* resultou em um p-*value* de $2,2^{-16}$. Como esse p-*value* é inferior a 0,05 (o nível de significância, α), a hipótese nula de que os mecanismos apresentaram um desempenho estatisticamente não diferente deve ser descartada. Logo, algumas estratégias apresentaram um desempenho superior às demais. Por isso, o teste *Wilcoxon Signed-Rank Test* (WSRT) foi aplicado para avaliar os seguintes pares de estratégias: 1) *preenchimento*($m = 100$), linear, exponencial e *random* 255; 2) *preenchimento*($m = 900$), MTU, Ratos/Elefantes e *random*; 3) os quatro níveis (100, 500, 700 e 900) de preenchimento. As hipóteses para este teste são apresentadas a seguir.

H_0 : a estratégia A supera o desempenho de B.

H_1 : a estratégia A é superada pelo desempenho de B.

Os resultados do teste WSRT apontam que, para os dados apresentados na Figura 13, o nível 100 da solução é superior a linear, exponencial e *random* 255. O nível *preenchimento*($m = 900$) supera o desempenho da MTU, Ratos/Elefantes e *random*. Por fim, há uma diferença significativa entre os níveis de preenchimento. Quanto aos resultados apresentados na Figura 19, o nível 900 é superior a Ratos/Elefantes, mas superado por MTU e *random*. O *preenchimento*($m = 100$) supera a estratégia exponencial, enquanto é suplantado pela linear e *random* 255. Novamente, há diferença entre os diferentes níveis

da solução. Os resultados para estes testes sugerem que, o número de *bytes* influencia o aprimoramento da privacidade.

Para os dados apresentados na Figura 14, o nível 100 da solução supera as estratégias exponencial e *random* 255, enquanto é suplantada pela linear. O nível 900 é superior à MTU, *random* e Ratos/Elefantes. Além disso, o *preenchimento*($m = 700$) também superou a MTU. No cenário com adversário externo ao *testbed* UK, apresentado na Figura 15, o nível *preenchimento*($m = 500$) é superior ao *preenchimento*($m = 100$), enquanto *preenchimento*($m = 900$) supera *preenchimento*($m = 100$). Logo, novamente, o número de *bytes* influencia o aprimoramento na privacidade. Por fim, em ambos os *testbeds* US e UK, os resultados para o cenário em que o provedor da solução de preenchimento é um adversário apresentam o mesmo desempenho em todas a repetições. Por isso, não foi possível conduzir testes de hipóteses nos resultados obtidos neste cenário.

Quanto ao *byte overhead*, nos dados disponibilizados pelos autores de (SIVANATHAN et al., 2018), foram comparados os seguintes conjuntos de estratégias: 1) *preenchimento*($m = 100$), linear e exponencial; 2) *preenchimento*($m = 900$), *random* e Ratos/Elefantes. As demais estratégias apresentaram um *byte overhead* suficientemente distinto para diferenciá-las sem o auxílio de testes de hipóteses. Logo, comparar tais estratégias constitui um excesso. Nas estratégias linear, exponencial e *preenchimento*($m = 100$), o teste de *Friedman* resultou em um p-*value* de $2,2^{-16}$. Por isso, a hipótese nula foi rejeitada, o que nos impeliu a aplicar o teste WSRT. Este teste apontou que o *preenchimento*($m = 100$) introduz um *overhead* menor que as estratégias linear e exponencial. Por outro lado, o WSRT apresenta indícios de que o nível 900 da solução introduz um *byte overhead* superior à *random* e Ratos/Elefantes.

Nos resultados do *byte overhead* produzidos a partir dos dados disponibilizados pelos autores de (REN et al., 2019), o teste de *Friedman* gerou um p-*value* de $2,2^{-16}$. Logo, a hipótese nula deve ser rejeitada. Por isso, o teste WSRT foi aplicado nesses resultados para o *byte overhead*. O teste de WSRT nos dados capturados no *testbed* em UK aponta que, o *preenchimento*($m = 100$) gera menos *overhead* que a estratégia exponencial e a *random* 255, enquanto produz mais *overhead* que a linear. Novamente, o *preenchimento*($m = 900$) gera um *byte overhead* superior à *random* e a Ratos/Elefantes. Nos dados coletados no *testbed* US, o *preenchimento*($m = 100$) introduz menos *overhead* que *random* 255 e linear, ao mesmo tempo que supera a exponencial. Por fim, o *preenchimento*($m = 900$) produz um *overhead* mais elevado que a Ratos/Elefantes.

Os testes de hipóteses conduzidos neste trabalho contribuíram para algumas conclusões a respeito dos resultados obtidos na avaliação do protótipo. Primeiro, o número de *bytes* influencia o nível de aprimoramento na privacidade. Segundo, exceto para o *testbed* UK, o *preenchimento*($m = 900$) é capaz de proporcionar um aprimoramento na privacidade comparável à MTU, mesmo produzindo um *overhead* significativamente menor. Por fim, o *byte overhead* e a melhoria na privacidade proporcionados pelo *preenchimento*($m = 100$)

são comparáveis aos seus pares linear, exponencial e *random* 255.

5.5 RESPOSTAS ÀS QUESTÕES DE PESQUISA

No Capítulo 1 desta tese, foram apresentadas as principais questões de pesquisa que guiaram a condução deste trabalho. Neste ponto, essas questões são revisitadas e respondidas com base nos resultados analisados neste capítulo. Para conveniência do leitor, as questões (destacadas em negrito) são reapresentadas, seguidas de suas respectivas respostas. Os resultados apresentados neste capítulo contribuem para responder à principal questão enfrentada por esta pesquisa, reescrita abaixo.

Como adaptar a quantidade de *bytes* inseridos nos pacotes contribui para balancear o *trade-off* entre privacidade e *overhead* em resposta às variações na utilização da rede? Tanto o aprimoramento da privacidade quanto o impacto no desempenho da comunicação são influenciados pelo número de *bytes* inseridos nos pacotes. Os resultados obtidos nos experimentos contribuem para mostrar que, o número de *bytes* pode ser adaptado em respostas às variações na utilização de uma rede doméstica.

De que forma os padrões no tráfego originado em casas inteligentes influenciam nas decisões sobre a quantidade de *bytes* inseridos nos pacotes por mecanismos de preenchimento? A análise do tráfego IoT disponibilizado pelos autores de (SIVANATHAN et al., 2018; REN et al., 2019) apresentam indícios de que, os padrões no tamanho do pacote gerado por dispositivos IoT podem diferir significativamente entre ambientes experimentais, que representam residências. Por isso, a quantidade de *bytes* inseridos nos pacotes pode ser adaptada para os padrões típicos de uma residência. Inclusive, os resultados para o aprimoramento da privacidade sugerem que é possível obter um desempenho muito interessante, a um custo, em termos de *overhead*, significativamente inferior ao proporcionado pelos mecanismos existentes. Esses resultados são possíveis devido às características do tráfego IoT, como tamanhos, tipicamente, próximos à 100 *bytes*.

Os resultados apresentados neste capítulo também contribuem para responder a seguinte questão: **é viável introduzir uma solução de preenchimento sem alterar os dispositivos SH-IoT?** Como discutido anteriormente, equipamentos comerciais podem ser modificados somente por seus fabricantes. Por isso, o mecanismo de preenchimento proposto foi implementado em um dispositivo de rede, que representa um roteador doméstico. Esse mecanismo teve desempenho como esperado, em que foi remotamente configurado por uma aplicação SDN. Portanto, é possível introduzir uma solução de preenchimento sem modificar os dispositivos SH-IoT.

5.6 CONSIDERAÇÕES FINAIS

Neste capítulo, são apresentados os resultados obtidos nos experimentos descritos no capítulo anterior. Também são expostas as análises conduzidas na averiguação dos resultados. Também foram aplicados testes de hipóteses nesses resultados para quantificar a incerteza sobre esses dados coletados na avaliação. Além disso, o impacto no desempenho da comunicação introduzido pela solução foi analisado. Por fim, este capítulo apresenta as respostas às questões de pesquisas.

6 TRABALHOS RELACIONADOS

Este capítulo descreve os principais trabalhos relacionados a esta tese. Foram analisados trabalhos oriundos de diferentes contextos conexos à ofuscação de tráfego como método de aprimoramento da privacidade para casas inteligentes. Uma vez que este domínio IoT permanece pouco explorado quanto a soluções de preenchimento e ofuscação de tráfego, foram examinadas pesquisas desenvolvidas em outros contextos, como páginas *web* e aplicações de rede. Por fim, com base na revisão da literatura conduzida nesta pesquisa, neste capítulo são apontadas as contribuições desta tese ao estado da arte sobre preenchimento de pacotes. A Figura 28 apresenta uma visão geral dos metadados e técnicas de ofuscação de tráfego empregadas nos trabalhos analisados neste capítulo.

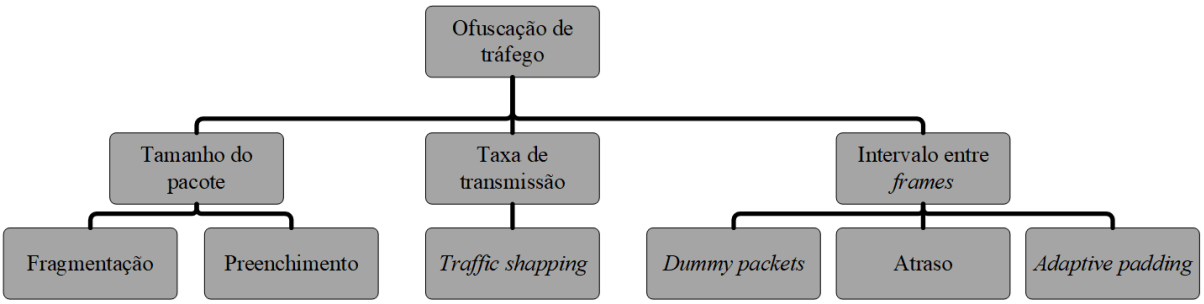


Figura 28 – Visão geral das técnicas de ofuscação de tráfego empregadas nos trabalhos analisados neste capítulo.
Fonte: elaborada pelo autor.

6.1 OFUSCAÇÃO DO TRÁFEGO DE REDE

Em decorrência das ameaças à privacidade geradas pela classificação de metadados do tráfego, surgiram inúmeros mecanismos para ofuscação. Os mecanismos *SkypeMorph* (MOGHADDAM et al., 2012) e *StegoTorus* (WEINBERG et al., 2012) têm como objetivo camuflar o tráfego produzido pelo *The Onion Router (Tor)* através de protocolos populares, como *Voice over Internet Protocol (VoIP)* e HTTP. Assim, usuários do Tor podem acessar a Internet de forma privada, mesmo em regimes que suprimem as liberdades individuais, como China e Irã. O *SkypeMorph* usa APIs e protocolos do *software Skype* para estabelecer chamadas VoIP destinadas a um servidor na rede *Tor*. O *SkypeMorph* utiliza um mecanismo adicional para modular o tráfego HTTP em áudio, transmitido para o servidor via chamada VoIP. O *StegoTorus* converte uma sequência ordenada de pacotes em "blocos" de dados com tamanhos variáveis (selecionados aleatoriamente) e os transmite em uma ordem distinta da original. Esses blocos são criptografados e podem ser camuflados como tráfego dos protocolos HTTP, *Skype* ou *Ventrilo*.

O *software Obfsproxy* (SHAHBAR; ZINCIR-HEYWOOD, 2015) modifica o tamanho do pacote e o intervalo entre *frames*, e criptografa o tráfego resultante destas alterações. Para ocultar esses metadados, o *Obfsproxy* implementa um protocolo adicional chamado de *obfs* e adota o modelo cliente-servidor. O cliente *Obfsproxy* encapsula os pacotes originais com o protocolo *obfs*; adiciona *bytes* extras selecionados aleatoriamente ao pacote obfs; criptografa os dados; por fim, o tráfego criptografado é transmitido para o servidor *Obfsproxy*. No lado do servidor, os dados são descriptografados e os *bytes* extras removidos. O mecanismo *ScrambleSuit* (SHAHBAR; ZINCIR-HEYWOOD, 2015) realiza o mesmo procedimento do *Obfsproxy*. Os mecanismos *Obfsproxy*, *ScrambleSuit*, *SkypeMorph* (MOGHADDAM et al., 2012) e *StegoTorus* (WEINBERG et al., 2012) compõem o projeto *Tor Pluggable Transports* (SHAHBAR; ZINCIR-HEYWOOD, 2015). Apesar de suas diferenças, esses mecanismos compartilham uma característica comum às soluções de ofuscação em geral: produzir elevado *overhead* na comunicação.

Ofuscar os metadados de forma estática produz efeitos profundamente negativos, como elevado *overhead*, que pode interferir no desempenho das aplicações de rede. Esses efeitos nocivos no desempenho da comunicação motivaram o desenvolvimento de soluções, baseadas em *dummy packets*, capazes de adaptar o número de pacotes extras em resposta à condição momentânea da rede. Geralmente, essas soluções são referenciadas como *adaptive padding*, técnica usada para ofuscar padrões expostos pelos intervalos entre *frames*. Neste contexto, *padding* refere-se a *link padding*, um sinônimo para *dummy packets*. Neste ponto, é importante esclarecer que, a única relação entre o preenchimento de pacotes e o *adaptive padding* consiste em sua finalidade de aprimorar a privacidade. O objetivo do *adaptive padding* não é balancear entre privacidade e *overhead* mas minimizar o impacto no desempenho da comunicação. *Adaptive padding* reduz o impacto no desempenho ao diminuir a quantidade de pacotes extras gerados, ao passo que o volume de tráfego produzido pelos dispositivos presentes em uma rede aumenta. Os principais trabalhos que apresentam soluções com base em *adaptive padding* são analisados nos próximos parágrafos.

Em (CAI; NITHYANAND; JOHNSON, 2014), os autores apontam as deficiências de soluções para ofuscação de tráfego que modificam metadados estaticamente, como gerar *overhead* excessivo. Devido a essas limitações, os seus autores apresentam um mecanismo capaz de ajustar o número de *dummy packets* a partir do volume de tráfego corrente. Os autores de (JUAREZ et al., 2016) apresentaram um mecanismo chamado de *Website Traffic Fingerprinting Protection with Adaptive Defense (WTF-AD)* que produz *dummy packets* somente quando a utilização do enlace está substancialmente baixa. Por outro lado, quando a utilização está elevada, em decorrência do tráfego gerado pelas aplicações de rede, o número de *dummy packets* é reduzido para evitar agravos no desempenho da comunicação. Quando o volume de tráfego aumenta organicamente, o número de *dummy packets* pode ser reduzido, uma vez que o intervalo entre *frames* decai naturalmente. Assim, o nível de privacidade é mantido enquanto o *overhead* produzido pelos *dummy packets* é reduzido.

Os autores de (LU et al., 2018; BHAT et al., 2019) apresentaram um mecanismo chamado de *DynaFlow*, que produz *dummy packets* e introduz atraso aos *frames* em resposta às variações no volume de tráfego. Os autores de (Al-Naami et al., 2019) propõem um mecanismo que emprega amostragem estatística para ocultar padrões expostos pelo tamanho e intervalos entre pacotes em *bursts* (sequências de *frames* em uma direção específica, *uplink* ou *downlink*). Em (Li et al., 2019b), os autores propõem uma solução chamada de *FlowGAN* que camufla o tráfego de uma aplicação para dificultar a sua identificação. Por exemplo, alterar os padrões no tráfego de um mensageiro para assemelhar-se a um jogo eletrônico. Segundo os autores de (Li et al., 2019b), a ideia central de seu mecanismo consiste em aprender automaticamente os atributos do tráfego que caracterizam uma aplicação. Os autores de (Li et al., 2019b) adotaram o modelo de redes generativas adversariais (do inglês, *Generative Adversarial Net (GAN)*) no desenvolvimento do *FlowGAN*.

Mecanismos de *dummy packets* são adequados para ocultar o intervalo entre *frames*. Para obscurecer o tamanho, a técnica de ofuscação mais apropriada consiste no preenchimento. Apesar de ocultar metadados distintos, intervalo entre *frames* e tamanho do pacote, mecanismos de *dummy packets* e preenchimento compartilham uma limitação relevante: a natureza estática. Como mencionado anteriormente, essa incapacidade de adaptar-se às variações no tráfego de uma rede produzem efeitos negativos e motivaram o desenvolvimento de soluções adaptativas.

Como discutido no Capítulo 1, mecanismos de *dummy packets* são potencialmente inadequados para casas inteligentes, em consequência ao provável número de pacotes extras necessários para ocultar o intervalo entre *frames* gerados por dispositivos IoT (APTHORPE et al., 2017). Embora soluções adaptativas possam reduzir o *overhead*, os mesmos estão dispensados de lidar com o *trade-off* privacidade-*overhead*, pois conseguem manter o nível de privacidade enquanto refreiam o *overhead* (APTHORPE et al., 2019). Como contribuição a estes trabalhos de *dummy packet* adaptativo, esta pesquisa analisa o *trade-off* privacidade-*overhead* em uma solução para preenchimento de pacote adaptativo, quando é improvável obter ambos, sendo necessário balanceá-los em respostas às variações em uma rede.

6.2 PREENCHIMENTO DE PACOTES

As ameaças à privacidade desencadeadas pela análise do tamanho do pacote motivaram o desenvolvimento de inúmeros mecanismos para preenchimento de pacotes. Os autores de (LIBERATORE; LEVINE, 2006) apresentaram um mecanismo de preenchimento para dificultar a identificação das páginas *web* acessadas pelos indivíduos. Esse mecanismo gera um *overhead* na vazão do tráfego superior a 150%. A solução apresentada em (LIBERATORE; LEVINE, 2006) foi analisada pelos autores de (DYER et al., 2012), assim como outros mecanismos (baseados nas estratégias descritas na Seção 2.5) para preenchimento de pacotes. Os autores de (DYER et al., 2012) argumentam que as soluções analisadas são ineficientes devido ao elevado *overhead* ou aprimoramento na privacidade pífio. Em

(DYER et al., 2012), foi proposto um mecanismo de preenchimento chamado de *Buffered Fixed-Length Obfuscation (BuFLO)*, que emprega intervalos e tamanhos fixos de 1500 *bytes*. Esse mecanismo produz um *overhead* na vazão do tráfego de até 400%. Os autores de (DYER et al., 2012) admitem que esse nível de *overhead* pode causar danos severos no desempenho da comunicação de rede.

Em (IACOVAZZI; BAIOCCHI, 2014; IACOVAZZI; BAIOCCHI, 2015), os autores investigam o *trade-off* entre proteger a privacidade dos indivíduos em uma rede através de preenchimento e o impacto no desempenho da comunicação, decorrente do *overhead*. Os autores de (IACOVAZZI; BAIOCCHI, 2014; IACOVAZZI; BAIOCCHI, 2015) apresentaram um mecanismo de preenchimento (baseado na estratégia Ratos/Elefantes descrita na Seção 2.5) para ocultar padrões expostos por aplicações de rede, como *Secure Shell (SSH)*, HTTP, VoIP e *File Transfer Protocol (FTP)*. Os autores de (IACOVAZZI; BAIOCCHI, 2014) avaliaram a quantidade de informações expostas por aplicações de rede aos observadores a partir do tamanho do pacote. Os resultados desta avaliação demonstram que o mecanismo proposto em (IACOVAZZI; BAIOCCHI, 2014) alcançou um aprimoramento na privacidade pífio.

Os autores de (LIU et al., 2014) apresentaram um mecanismo de preenchimento para preservar a privacidade dos usuários de páginas *web*. Em (LIU et al., 2014), os pacotes são agrupados e o número de *bytes* é selecionado a partir do tamanho dos integrantes de cada grupo. Todos os pacotes membros de um grupo compartilham o mesmo tamanho. Em (DYER; COULL; SHRIMPTON, 2015) autômatos probabilísticos são aplicados na ofuscação de padrões nos metadados do tráfego de rede. O mecanismo apresentado em (DYER; COULL; SHRIMPTON, 2015) é controlado por uma linguagem específica de domínio.

Os autores de (CIFTCIOGLU et al., 2018) apresentaram um mecanismo de preenchimento que modifica a distribuição de probabilidade do tamanho do pacote. Esse mecanismo oculta os tamanhos que podem expor os protocolos responsáveis pela transmissão dos dados, como TCP e *User Datagram Protocol (UDP)*. Os autores de (CHADDAD et al., 2018) apresentaram um mecanismo capaz de modificar a distribuição de probabilidade do tamanho do pacote gerado por aplicativos instalados em *smartphones*. Por exemplo, mascarar a distribuição do tamanho gerado pelo *Skype* como tráfego de um jogo. Essa solução tem como objetivo prevenir a identificação das aplicações instaladas em dispositivos móveis, como *smartphones* e *tablets*. Ao analisarem o *trade-off* entre privacidade e *overhead*, os autores de (CHADDAD et al., 2018) argumentam que os usuários de dispositivos móveis podem evadir-se de soluções que degradem o desempenho da comunicação e a sua experiência.

Em (SIBY et al., 2019), os autores analisam os principais mecanismos de preenchimento existentes para o protocolo *Domain Name System (DNS)*. Esses mecanismos modificam os tamanhos para os seus múltiplos mais próximos de 128, mesmo procedimento adotado pela estratégia linear. Os autores de (SIBY et al., 2019) apontam que os mecanismos analisados são ineficientes. Por isso, em (SIBY et al., 2019) é proposta a modificação dos tamanhos de requisições e respostas DNS para valores fixos de 825 *bytes*. O mecanismo proposto em

(SIBY et al., 2019) dificulta a identificação de páginas *web* a partir do tráfego DNS.

Os autores de (Fathi-Kazerooni; Kaymak; Rojas-Cessa, 2019) analisaram a eficiência do preenchimento de pacotes para ocultar a identidade de aplicações em execução em um computador pessoal, como o navegador *web Google Chrome*, o reprodutor de música *Spotify* e o mensageiro *WhatsApp*. Os autores de (Fathi-Kazerooni; Kaymak; Rojas-Cessa, 2019) modificaram estaticamente todos os tamanhos para 1500 *bytes*, como especificado pela estratégia MTU. Os resultados da análise conduzida em (Fathi-Kazerooni; Kaymak; Rojas-Cessa, 2019) sugerem que o preenchimento de pacotes contribui para degradar o desempenho de vários algoritmos para aprendizado de máquina, em que os mais impactados foram *Decision Tree*, *Support Vector Machine (SVM)*, *Bagged Trees* e *Multilayer Perceptron (MLP)*. Apesar de os autores de (Fathi-Kazerooni; Kaymak; Rojas-Cessa, 2019) não analisarem o *overhead* gerado por seu mecanismo de preenchimento, é provável que o mesmo introduza um custo elevado, considerando a adoção da estratégia MTU para selecionar o número de *bytes* inseridos nos pacotes.

Em anos recentes, surgiram inúmeros mecanismos de preenchimento, endereçados a diferentes contextos, como páginas *web*, aplicações e protocolos de rede. Tipicamente, soluções de preenchimento são projetadas para ocultar o tamanho em domínios nos quais esse metadado tem potencial para expor informações privadas dos indivíduos. Logo, visto que inúmeros trabalhos demonstraram a viabilidade de inferir informações privadas dos indivíduos a partir do tamanho do pacote gerado por dispositivos IoT (APTHORPE et al., 2017), é provável que o preenchimento contribua para aprimorar a privacidade dos usuários deste tipo de equipamento. A próxima seção apresenta algumas soluções de ofuscação desenvolvidas especificamente para IoT e casas inteligentes.

6.3 OFUSCAÇÃO DE DADOS PARA IOT E CASAS INTELIGENTES

Em (TAKBIRI et al., 2017), os autores analisam os efeitos da ofuscação de tráfego na ocultação de informações sensíveis presentes em séries temporais compostas por dados de localização dos usuários IoT. Em (MALEKZADEH et al., 2018; MALEKZADEH; CLEGG; HADDADI, 2018) são apresentados mecanismos para ocultar informações sensíveis presentes em séries temporais provenientes de sensores IoT. Esses mecanismos substituem seções das séries temporais que contêm dados sensíveis por partições com informações irrelevantes.

Os autores de (APTHORPE et al., 2017) apresentaram um mecanismo de *traffic shapping* que mantém a taxa de transmissão dos dispositivos IoT constante. O objetivo deste mecanismo consiste em ocultar informações privadas que podem ser inferidas a partir da taxa de transmissão, como dispositivos presentes em uma residência, movimentos no interior de uma casa e distúrbios no sono (APTHORPE; REISMAN; FEAMSTER, 2017). Os autores de (APTHORPE et al., 2019) apresentam mecanismo em que a quantidade de pacotes extras é ajustada de acordo com o volume de tráfego produzido por dispositivos IoT. Dessa forma, quando os equipamentos estão ociosos, o número de pacotes adicionais

pode ser reduzido. Esse mecanismo é baseado em *adaptive padding*, conceito discutido anteriormente. Como mencionado previamente, esse tipo de mecanismo é apropriado para ocultar o intervalo entre *frames*, não o tamanho do pacote. Ademais, esse tipo de mecanismo é dispensado de lidar com o *trade-off* privacidade-*overhead*.

Os autores de (XIONG; SARWATE; MANDAYAM, 2018), apresentaram um mecanismo de preenchimento para IoT baseado em *local differential privacy*. Em (XIONG; SARWATE; MANDAYAM, 2018), os autores argumentam que um único tamanho é suficiente para apontar a identidade de um dispositivo IoT. Por isso, em (XIONG; SARWATE; MANDAYAM, 2018), são ofuscados os tamanhos que apresentam o maior potencial para revelar informações sobre os usuários de dispositivos IoT. O mecanismo proposto em (XIONG; SARWATE; MANDAYAM, 2018) é eficiente apenas quando um observador analisa somente um pacote, um cenário improvável. Os próprios autores admitem que esse mecanismo deve ser ineficiente quando múltiplos pacotes forem analisados, o cenário mais realista. Além disso, a avaliação do desempenho no aprimoramento da privacidade foi bastante limitada. Sequer foram conduzidos experimentos para avaliar a capacidade do mecanismo em refrear técnicas para aprendizado de máquina.

Em (Kennedy et al., 2019), os autores analisam a viabilidade de identificar comandos de voz proferidos por usuários de *smart speakers* (ex.: *Amazon Echo* e *Goooogle Home*) a partir do tráfego criptografado. Os resultados dessa análise indicam que é possível inferir comandos de voz a partir de metadados do tráfego criptografado, como o tamanho do pacote. Por isso, os autores de (Kennedy et al., 2019) implementaram e avaliaram o mecanismo BuFLO, apresentado em (DYER et al., 2012), para ocultar os padrões no tamanho do pacote gerado por comandos de voz provenientes de *smart speakers*. Os resultados dessa avaliação sugerem que o preenchimento de pacotes realizado pelo BuFLO, que modificou os tamanhos para 1000 *bytes*, contribui para degradar o desempenho de algoritmos para aprendizado de máquina. Contudo, o *overhead* na comunicação gerado por esse mecanismo atinge 548%, o que pode afetar negativamente a experiência dos usuários de *smart speakers*.

Um mecanismo projetado para degradar o desempenho de algoritmos para aprendizado de máquina através da simulação do tráfego gerado por dispositivos SH-IoT é apresentado em (Hafeez; Antikainen; Tarkoma, 2019). Esse mecanismo produz tráfego em que o tamanho do pacote, o intervalo entre *frames* e a taxa de transmissão são moldados para aparentarem serem originados pelos dispositivos SH-IoT. Esses dados simulados contribuem para ofuscar estatísticas computadas a partir dos metadados mencionados anteriormente. Contudo, o mecanismo proposto em (Hafeez; Antikainen; Tarkoma, 2019) é incapaz de simular fielmente o comportamento do tráfego gerado por equipamentos IoT, que permite aos observadores diferenciar os dados originais dos simulados, como apontado por seus autores. Em (Hafeez; Antikainen; Tarkoma, 2019), *traffic shapping* também foi adotado para garantir que a taxa de transmissão do tráfego de *background* seja mantida constante.

Os autores de (UDDIN; NADEEM; NUKAVARAPU, 2019) propõem ajustar a probabilidade de modificar o tamanho de um pacote com base no volume de tráfego de uma rede. Assim, a probabilidade de um tamanho ser alterado é incrementada ao passo que o volume da rede decresce, e vice-versa. Dado que um pacote foi selecionado para ser modificado, seu tamanho é incrementado para 1500 *bytes*. Logo, o tamanho é mantido em 1500 *bytes*, somente a probabilidade de um pacote ser modificado é adaptável em resposta às oscilações no volume de tráfego. Os autores de (UDDIN; NADEEM; NUKAVARAPU, 2019) direcionam os seus esforços para aplicações móveis, não necessariamente dispositivos IoT. Em (UDDIN; NADEEM; NUKAVARAPU, 2019), o mecanismo proposto foi implementado em *smartphones* equipados com o sistema operacional móvel *Android*, que permite alterações em seu *software*, diferentemente de dispositivos IoT comerciais. Em (UDDIN; NADEEM; NUKAVARAPU, 2019), os autores argumentam que, residências são locais menos sensíveis quanto à privacidade que estabelecimentos comerciais, como restaurantes e cafeterias. Esses autores argumentam que, há menos riscos à privacidade dos usuários IoT quando estão em suas residências.

Tabela 6 – Relação entre a presente tese e os trabalhos relacionados.

Trabalho	Tamanho adaptativo	*Trade-off* privacidade-*overhead*	Casa inteligente
(MOGHADDAM et al., 2012)	✗	✓	✗
(WEINBERG et al., 2012)	✗	✓	✗
(SHAHBAR; ZINCIR-HEYWOOD, 2015)	✗	✓	✗
(CAI; NITHYANAND; JOHNSON, 2014)	✗	✓	✗
(JUAREZ et al., 2016)	✗	✓	✗
(LU et al., 2018)	✗	✓	✗
(BHAT et al., 2019)	✗	✓	✗
(Al-Naami et al., 2019)	✗	✓	✗
(Li et al., 2019b)	✗	✓	✗
(LIBERATORE; LEVINE, 2006)	✗	✓	✗
(DYER et al., 2012)	✗	✓	✗
(IACOVAZZI; BAIOCCHI, 2014)	✗	✓	✗
(IACOVAZZI; BAIOCCHI, 2015)	✗	✓	✗
(LIU et al., 2014)	✗	✓	✗
(DYER; COULL; SHRIMPTON, 2015)	✗	✓	✗
(CIFTCIOGLU et al., 2018)	✗	✓	✗
(CHADDAD et al., 2018)	✗	✓	✗
(SIBY et al., 2019)	✗	✓	✗
(MALEKZADEH; CLEGG; HADDADI, 2018)	✗	✗	✗
(MALEKZADEH et al., 2018)	✗	✗	✗
(APTHORPE et al., 2017)	✗	✓	✓
(APTHORPE et al., 2019)	✗	✓	✓
(XIONG; SARWATE; MANDAYAM, 2018)	✗	✓	✓
(UDDIN; NADEEM; NUKAVARAPU, 2019)	✗	✓	✗
(Fathi-Kazerooni; Kaymak; Rojas-Cessa, 2019)	✗	✓	✗
(Kennedy et al., 2019)	✗	✓	✓
(Hafeez; Antikainen; Tarkoma, 2019)	✗	✓	✓
Esta tese	✓	✓	✓

Em pesquisas anteriores, alguns autores investigaram soluções de *dummy packets* que

adaptam o número de pacotes extras de forma dinâmica em resposta às variações no volume de tráfego. Esses trabalhos sugerem que mecanismos de ofuscação estáticos são ineficientes. Contudo, esses trabalhos não necessitam balancear entre privacidade e *overhead*, o problema abordado nesta pesquisa. Os autores de (UDDIN; NADEEM; NUKAVARAPU, 2019) propõem adaptar a probabilidade de ofuscar um pacote, mas o tamanho permanece essencialmente constante. Por fim, a Tabela 6 apresenta a relação dos principais tópicos abordados nesta tese com os trabalhos relacionados analisados neste capítulo.

6.4 CONSIDERAÇÕES FINAIS

Este capítulo apresentou os principais trabalhos relacionados ao preenchimento de pacote para aprimorar a privacidade individual. Também foram destacadas as contribuições desta tese à literatura sobre preenchimento e ofuscação de tráfego para casas inteligentes. Para a conveniência do leitor desta tese, a relação entre os principais tópicos abordados nesta pesquisa e os trabalhos relacionados foi apresentada na tabela acima.

7 CONCLUSÃO

Neste capítulo, a solução de preenchimento apresentada nesta tese é brevemente revisitada. Ademais, são descritas as contribuições desta pesquisa para a literatura sobre preenchimento de pacotes para aprimorar a privacidade. Por fim, são discutidas extensões para a proposta de balancear o *trade-off* a serem exploradas em trabalhos futuros.

Esta tese propõe que o *trade-off* privacidade-*overhead* característico a técnicas para preenchimento de pacotes pode ser balanceado com base no *status* de uma rede. A fim de averiguar a praticabilidade desta proposta, uma solução para balancear esse *trade-off* foi concebida para o domínio IoT de casas inteligentes. Essa solução determina o número de *bytes* inseridos nos pacotes gerados pelos dispositivos IoT em resposta às variações no tráfego de uma residência.

O aprimoramento na privacidade proporcionado pela solução foi avaliado com o tráfego gerado por 102 dispositivos IoT, provenientes das principais categorias de equipamentos SH-IoT. Também foram conduzidos experimentos para mensurar o *byte overhead* e o impacto no desempenho da comunicação. Por fim, as principais estratégias de preenchimento existentes foram avaliadas e comparadas com a solução proposta.

Os resultados obtidos nos experimentos sugerem que a solução de preenchimento é capaz de minimizar o impacto no desempenho da comunicação ao adaptar o número de *bytes* inseridos nos pacotes. Os resultados também indicam que a solução é capaz de ajustar o *trade-off* entre privacidade e *overhead* em resposta as variações na utilização de uma rede. As principais contribuições desta tese e os trabalhos futuros são discutidos nas próximas seções.

7.1 DISCUSSÃO SOBRE AS CONTRIBUIÇÕES

As principais contribuições desta tese, apresentadas inicialmente no capítulo 1, são discutidas a seguir.

- Apresentação de uma solução de preenchimento baseada em SDN capaz de balancear o *trade-off* entre privacidade e *overhead* de acordo com a utilização de uma rede doméstica. Geralmente, os mecanismos de preenchimento existentes modificam os tamanhos continuamente da mesma maneira, desconsiderando a situação momentânea de uma rede. Esse tipo de solução é inapropriado para domínios em que o tráfego varia constantemente, como redes domésticas em que estão presentes dispositivos IoT. Esta tese apresenta uma solução de preenchimento capaz de preservar a privacidade ou o *overhead* em resposta às flutuações na utilização da rede doméstica;

- Exposição de que a quantidade de *bytes* necessária para inserção nos pacotes tem relação com a distribuição do tamanho gerado por dispositivos conectados a uma rede doméstica. A depender da distribuição do tamanho do pacote gerado por dispositivos IoT, é possível modificar a maioria ou uma parcela considerável dos *frames* para que compartilhem um valor comum a todos. Por exemplo, pacotes com tamanho inferior a 489 *bytes* podem ser modificados para esse valor, que representa 75% do total. Mesmo que alguns pacotes mantenham tamanhos distintos dos que foram modificados, é plausível que os mesmos sejam insuficientes para revelar informações privadas dos usuários de dispositivos IoT. Ao modificar uma parcela significativa dos tamanhos, as estatísticas computadas a partir desse atributo do tráfego podem diferir significativamente das medidas extraídas dos valores originais. Reduzir o número de *bytes* inseridos nos pacotes contribui para atenuar o *byte overhead* e o consequente impacto no desempenho da comunicação, que representam o principal custo para adoção do preenchimento de pacotes. Ao discutir e mostrar que é possível alcançar um aprimoramento na privacidade interessante, ao mesmo tempo em que é inserida uma quantidade de *bytes* menor comparada à estratégia existente considerada ótima, esta tese lança luz sobre a possibilidade de reduzir o custo do preenchimento, enquanto proporciona-se um aprimoramento na privacidade pertinente. Os resultados apresentados nesta tese abrem oportunidades de pesquisa para determinar o número ideal de *bytes* a serem inseridos nos pacotes;

- Apresentação de uma estratégia que respeita as preferências dos usuários sobre o *trade-off* privacidade-*overhead*, permitindo priorizar a privacidade ou *overhead*, além de viabilizar alterações nessas preferências ao longo do tempo. Em se tratando de privacidade, a liberdade de escolha é um aspecto fundamental. A estratégia proposta proporciona a liberdade para que o usuário da mesma determine suas preferências sobre preservar a privacidade ou o desempenho da comunicação. Como essas preferências são mutáveis e tendem a variar ao longo do tempo, a estratégia permite que tais predileções sejam reajustadas;

- Avaliação da influência do número de *bytes* no *trade-off* entre a privacidade e o *overhead*, considerando o preenchimento de pacote como a única técnica de ofuscação adotada para preservar a privacidade. Autores de trabalhos anteriores argumentam que o número de *bytes* influencia o *trade-off* entre privacidade e *overhead*. Contudo, esses autores avaliam estratégias que modificam os tamanhos de maneiras distintas, como linear e MTU, o que dificulta a avaliação do impacto do número de *bytes*. Visto que estas estratégias modificam os tamanhos de formas distintas, permanece inconclusivo o argumento de que os resultados do *trade-off* privacidade-*overhead* são decorrentes unicamente do número de *bytes*. Por outro lado, nesta tese, foi variado essencialmente o número de *bytes* inseridos nos pacotes. Assim, a influência

da quantidade de *bytes* no *trade-off* entre privacidade e *overhead* foi avaliada. Os resultados desta análise podem contribuir para o avanço em direção a soluções de preenchimento mais precisas na priorização da privacidade ou *overhead*;

- Viabilização do gerenciamento do *overhead* introduzido pelo preenchimento de pacote, permitindo que o número de *bytes* possa ser decrementado a fim de evitar exaurir os recursos de uma rede. Como previamente discutido neste trabalho, o *overhead* constitui o custo associado com o preenchimento de pacotes. Tipicamente, os mecanismos de preenchimento existentes estabelecem encargos permanentes, que independem da situação momentânea de uma rede. Admitindo que uma rede é um sistema dinâmico, em que a situação muda continuamente, soluções estáticas são provavelmente inadequadas. Logo, permitir que o custo (*overhead*) seja administrado com base na situação de uma rede pode representar uma oportunidade de evitar a exaustão dos recursos computacionais da infraestrutura. Essa possibilidade de gerenciar o *overhead* estabelece inúmeras oportunidades de pesquisa sobre como a administração deste custo será conduzida.

7.2 TRABALHOS GERADOS

Os trabalhos publicados e submetidos pelo autor desta tese são apresentados a seguir:

- Trabalhos gerados relacionados diretamente a esta tese:
  - **Pinheiro, A. J.**; Bezerra, J. M.; Burgardt, C. A. P.; Campelo, D. R. Identifying IoT Devices and Events Based on Packet Length From Encrypted Traffic. Computer Communications, [S.l.], v.144, p.8 – 17, 2019;
  - **Pinheiro, A. J.**; Bezerra, J. M.; Campelo, D. R. Packet Padding for Improving Privacy in Consumer IoT. In: IEEE Symposium On Computers And Communications (ISCC), 2018, Natal, Rio Grande do Norte, BR;
  - **Pinheiro, A. J.**; Burgardt, C. A. P.; Campelo, D. R. Preservando a Privacidade na Internet das Coisas com Pseudônimos Usando SDN. XVIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSeg), 2018, Natal, Rio Grande do Norte, BR;
  - **Pinheiro, A. J.**; Freitas Araújo-Filho, P.; Bezerra, J. M.; Campelo, D. R. Adaptive Packet Padding Approach for Smart Home Networks: A Trade-off between Privacy and Performance. IEEE Internet of Things Journal. (sob revisão).
- Trabalhos gerados sem relação direta com esta tese:

- Bezerra, J. M.; **Pinheiro, A. J.**; De Souza, Criston P.; Campelo, D. R. Performance Evaluation of Elephant Flow Predictors in Data Center Networking. Future Generation Computer Systems, v. 102, p. 952-964, 2019;
- Bezerra, J. M.; Pedro Olímpio; **Pinheiro, A. J.**; De Souza, Criston P.; Campelo, D. R. A Randomized Rounding Approach for Scheduling Elephant Flows in Data Center Networks. In: International Federation for Information Processing (IFIP) Networking Conference, 2020, Paris, França. (sob revisão);
- Bezerra, J. M.; **Pinheiro, A. J.**; Bonfim, M. S.; Monteiro, J. A. S.; Campelo, D. R. Engenharia de Tráfego em Redes Definidas por Software. 1ed.:, 2016, v., p.101-150. XXXIV Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC), 2016. (capítulo de livro).

7.3 TRABALHOS FUTUROS

Alguns dos trabalhos futuros que poderão ser explorados são discutidos nesta seção.

- Pesquisar técnicas para estabelecer o número de *bytes* a serem inseridos nos pacotes. Uma das contribuições desta tese consiste em apresentar indícios de que o número de *bytes* adotado por mecanismos de preenchimento está relacionado com a distribuição do tamanho do pacote originalmente gerado por dispositivos IoT. É provável que esse número de *bytes* varie entre os dispositivos SH-IoT. Essa constatação nos motiva a investigar mais profundamente técnicas para determinar esse número. Por isso, a determinação do valor para o qual os tamanhos serão alterados poderá ser explorada em um trabalho futuro. Como foi observado nesta pesquisa que esse valor pode estar relacionado à distribuição do tamanho, é possível que métricas para quantificar a informação contida em uma distribuição de probabilidade possam contribuir para determinar o número de *bytes*, como a *entropy*, *Kullback–Leibler Divergence* e *Mutual Information* (WAGNER; ECKHOFF, 2018);

- Investigar métodos e técnicas para gerenciar o *overhead* ao adaptar o número de *bytes* inseridos nos pacotes. Apesar de atender ao seu propósito, o mecanismo para ajustar o número de *bytes* pode ser aprimorado. Atualmente, o ajuste no número de *bytes* ocorre em modo reativo, em resposta as variações na utilização da rede. Pretendemos investigar em trabalhos futuros técnicas para ajustar a quantidade de *bytes* de forma preditiva, e personalizada para cada residência. Acreditamos que essa adaptabilidade no número de *bytes* é viável com o auxílio de mecanismos capazes de predizer as variações no volume de tráfego doméstico, gerado por dispositivos IoT. Com técnicas para aprendizado de máquina poderemos projetar um mecanismo capaz de aprender os padrões no volume do tráfego de cada residência gerenciada

pela solução de preenchimento. Dessa forma, será possível personalizar o ajuste no número de *bytes* para cada domicílio.

- Investigar a viabilidade de incorporar a proposta de balancear o *trade-off* a partir do *status* de uma rede em dispositivos móveis, como *smartphones* e *tablets*. Assim como dispositivos IoT, *smartphones* e *tablets* estão suscetíveis à exposição de informações dos seus usuários, através da análise do tamanho do pacote (CHADDAD et al., 2018). Portanto, para preservar a privacidade dos usuários de dispositivos móveis, o preenchimento de pacotes é uma solução apropriada. Novamente, existem desafios relacionados ao *overhead* produzido por mecanismos de preenchimento, que pode inviabilizar a adoção deste tipo de solução em dispositivos móveis (UDDIN; NADEEM; NUKAVARAPU, 2019). Logo, a proposta de balancear o *trade-off* entre privacidade e *overhead* pode contribuir para a adoção de mecanismos de preenchimento em dispositivos móveis. Diferentemente de equipamentos IoT presentes em residências, *smartphones* estão constantemente alternando entre redes, o que impõem desafios adicionais para soluções de preenchimento. Devido a essa mobilidade, é necessário que o preenchimento seja realizado nos próprios dispositivos;

- Investigar a possibilidade da solução aplicar diferentes níveis de preenchimento para dispositivos IoT distintos. Alguns equipamentos, como câmeras e sensores de movimento, podem representar uma maior ameaça à privacidade, visto que podem expor a presença/ausência dos indivíduos em suas residências. É factível que os usuários de dispositivos IoT nutram maiores preocupações sobre sua privacidade em relação a equipamentos específicos. Um usuário pode priorizar a ofuscação do tráfego de um determinado equipamento IoT. Por isso, pretendemos investigar em trabalhos futuros a viabilidade de aplicar diferentes níveis de preenchimento para dispositivos distintos;

A pesquisa que originou esta tese investigou a praticabilidade de balancear o *trade-off* privacidade-*overhead* em resposta as variações no tráfego de uma rede. Como consequência desta investigação, foram geradas algumas contribuições ao estado da arte sobre preenchimento de pacotes. Este trabalho abre oportunidades de pesquisa que podem ser exploradas nos trabalhos futuros mencionados acima.

REFERÊNCIAS

Aceto, G.; Ciuonzo, D.; Montieri, A.; Pescapé, A. Mobile encrypted traffic classification using deep learning: Experimental evaluation, lessons learned, and challenges. *IEEE Transactions on Network and Service Management*, v. 16, n. 2, p. 445–458, June 2019. ISSN 2373-7379.

Al-Naami, K.; El Ghamry, A.; Islam, M. S.; Khan, L.; Thuraisingham, B. M.; Hamlen, K. W.; Alrahmawy, M.; Rashad, M. Bimorphing: A bi-directional bursting defense against website fingerprinting attacks. *IEEE Transactions on Dependable and Secure Computing*, p. 1–1, 2019. ISSN 1545-5971.

ALPAYDIN, E. *Introduction to machine learning*. [S.l.]: MIT press, 2014.

APTHORPE, N.; HUANG, D. Y.; REISMAN, D.; NARAYANAN, A.; FEAMSTER, N. Keeping the smart home private with smart(er) IoT traffic shaping. *Proceedings on Privacy Enhancing Technologies*, Sciendo, Berlin, v. 2019, n. 3, p. 128 – 148, 2019. Disponível em: <https://content.sciendo.com/view/journals/popets/2019/3/article-p128.xml>.

APTHORPE, N.; REISMAN, D.; FEAMSTER, N. A smart home is no castle: Privacy vulnerabilities of encrypted IoT traffic. *CoRR*, abs/1705.06805, 2017. Disponível em: <http://arxiv.org/abs/1705.06805>.

APTHORPE, N.; REISMAN, D.; SUNDARESAN, S.; NARAYANAN, A.; FEAMSTER, N. Spying on the smart home: Privacy attacks and defenses on encrypted IoT traffic. *arXiv preprint arXiv:1708.05044*, 2017.

ARSHAD, J.; AZAD, M. A.; SALAH, K.; JIE, W.; IQBAL, R.; ALAZAB, M. A review of performance, energy and privacy of intrusion detection systems for iot. *CoRR*, 2018. Disponível em: <http://arxiv.org/abs/1812.09160>.

ATZORI, L.; IERA, A.; MORABITO, G. The Internet of things: A survey. *Computer networks*, Elsevier, v. 54, n. 15, p. 2787–2805, 2010.

BELTRAN, V.; SKARMETA, A. F. Overview of device access control in the IoT and its challenges. *IEEE Communications Magazine*, p. 1–7, 2018. ISSN 0163-6804.

BHAT, S.; LU, D.; KWON, A.; DEVADAS, S. Var-CNN and DynaFlow: Improved attacks and defenses for website fingerprinting. *CoRR*, 2019. Disponível em: <http://arxiv.org/abs/1802.10215>.

BOSSHART, P.; DALY, D.; GIBB, G.; IZZARD, M.; MCKEOWN, N.; REXFORD, J.; SCHLESINGER, C.; TALAYCO, D.; VAHDAT, A.; VARGHESE, G.; WALKER, D. P4: Programming Protocol-independent Packet Processors. *SIGCOMM Comput. Commun. Rev.*, ACM, New York, NY, USA, v. 44, n. 3, p. 87–95, jul. 2014. ISSN 0146-4833. Disponível em: <http://doi.acm.org/10.1145/2656877.2656890>.

BOUSSARD, M.; BUI, D. T.; DOUVILLE, R.; JUSTEN, P.; SAUZE, N. L.; PELOSO, P.; VANDEPUTTE, F.; VERDOT, V. Future spaces: Reinventing the home network for better security and automation in the IoT era. *Sensors*, v. 18, n. 9, 2018. ISSN 1424-8220. Disponível em: <http://www.mdpi.com/1424-8220/18/9/2986>.

BRAY, T. *The JavaScript Object Notation (JSON) Data Interchange Format*. IETF, 2014. RFC 7159 (Proposed Standard). (Request for Comments, 7159). Disponível em: <http://www.ietf.org/rfc/rfc7159.txt>.

BREIMAN, L. Random forests. *Machine learning*, Springer, v. 45, n. 1, p. 5–32, 2001.

CAI, X.; NITHYANAND, R.; JOHNSON, R. CS-BuFLO: A congestion sensitive website fingerprinting defense. In: *Proceedings of the 13th Workshop on Privacy in the Electronic Society*. New York, NY, USA: ACM, 2014. p. 121–130. ISBN 978-1-4503-3148-7. Disponível em: <http://doi.acm.org/10.1145/2665943.2665949>.

CARPENTER, B.; BRIM, S. *Middleboxes: Taxonomy and Issues*. IETF, 2002. RFC 3234 (Informational). (Request for Comments, 3234). Disponível em: <http://www.ietf.org/rfc/rfc3234.txt>.

CHADDAD, L.; CHEHAB, A.; ELHAJJ, I. H.; KAYSSI, A. App traffic mutation: Toward defending against mobile statistical traffic analysis. In: *IEEE Conference on Computer Communications Workshops*. [S.l.: s.n.], 2018. p. 27–32.

CHOE, E. K.; CONSOLVO, S.; JUNG, J.; HARRISON, B.; KIENTZ, J. A. Living in a glass house: A survey of private moments in the home. In: *Proceedings of the 13th International Conference on Ubiquitous Computing*. New York, NY, USA: ACM, 2011. p. 41–44. ISBN 978-1-4503-0630-0. Disponível em: <http://doi.acm.org/10.1145/2030112.2030118>.

CIFTCIOGLU, E.; HARDY, R.; CHAN, K.; SCOTT, L.; OLIVEIRA, D.; VERMA, G. Chaff allocation and performance for network traffic obfuscation. In: *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*. [S.l.: s.n.], 2018. p. 1565–1568. ISSN 2575-8411.

DACREMA, M. F.; CREMONESI, P.; JANNACH, D. Are we really making much progress? a worrying analysis of recent neural recommendation approaches. In: *Proceedings of the 13th ACM Conference on Recommender Systems*. New York, NY, USA: ACM, 2019. p. 101–109. ISBN 978-1-4503-6243-6. Disponível em: <http://doi.acm.org/10.1145/3298689.3347058>.

DEERING, S.; HINDEN, R. *Internet Protocol, version 6 (IPv6) specification*. [S.l.], 2017.

DEMŠAR, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, v. 7, n. Jan, p. 1–30, 2006.

DIERKS, T.; RESCORLA, E. RFC 5246: The transport layer security (TLS) protocol. *The Internet Engineering Task Force*, 2008.

DIXON, L.; RISTENPART, T.; SHRIMPTON, T. Network traffic obfuscation and automated Internet censorship. *IEEE Security Privacy*, v. 14, n. 6, p. 43–53, Nov 2016. ISSN 1540-7993.

DUDA, R. O.; HART, P. E.; STORK, D. G. *Pattern classification*. [S.l.]: John Wiley & Sons, 2012.

DYER, K. P.; COULL, S. E.; RISTENPART, T.; SHRIMPTON, T. Peek-a-boo, i still see you: Why efficient traffic analysis countermeasures fail. In: *2012 IEEE Symposium on Security and Privacy*. San Francisco, CA, USA: [s.n.], 2012. p. 332–346. ISSN 1081-6011.

DYER, K. P.; COULL, S. E.; SHRIMPTON, T. Marionette: A programmable network traffic obfuscation system. In: *USENIX Security Symposium*. Washington, D.C, USA: [s.n.], 2015. p. 367–382.

EASTERBROOK, S.; SINGER, J.; STOREY, M.-A.; DAMIAN, D. Selecting empirical methods for software engineering research. In: SHULL, F.; SINGER, J.; SJøBERG, D. (Ed.). *Guide to Advanced Empirical Software Engineering*. Springer London, 2008. p. 285–311. ISBN 978-1-84800-043-8. Disponível em: <http://dx.doi.org/10.1007/978-1-84800-044-5_11>.

Fathi-Kazerooni, S.; Kaymak, Y.; Rojas-Cessa, R. Identification of user application by an external eavesdropper using machine learning analysis on network traffic. In: *2019 IEEE International Conference on Communications Workshops (ICC Workshops)*. [S.l.: s.n.], 2019. p. 1–6. ISSN 2474-9133.

FEGHHI, S.; LEITH, D. J. An efficient web traffic defence against timing-analysis attacks. *IEEE Transactions on Information Forensics and Security*, v. 14, n. 2, p. 525–540, 2019. ISSN 1556-6013.

FRIEDMAN, M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, Taylor & Francis, v. 32, n. 200, p. 675–701, 1937. Disponível em: <https://www.tandfonline.com/doi/abs/10.1080/01621459.1937.10503522>.

GRINBERG, M. *Flask web development: developing web applications with python*. [S.l.]: "O'Reilly Media, Inc.", 2018.

Hafeez, I.; Antikainen, M.; Tarkoma, S. Protecting IoT-environments against traffic analysis attacks with traffic morphing. In: *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. [S.l.: s.n.], 2019. p. 196–201. ISSN null.

HAJJAR, A.; KHALIFE, J.; DíAZ-VERDEJO, J. Network traffic application identification based on message size analysis. *Journal of Network and Computer Applications*, v. 58, p. 130 – 143, 2015. ISSN 1084-8045. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1084804515002167>.

HALEPLIDIS, E.; PENTIKOUSIS, K.; DENAZIS, S.; SALIM, J. H.; MEYER, D.; KOUFOPAVLOU, O. Software-defined networking (SDN): Layers and architecture terminology. In: *RFC 7426*. [S.l.]: IRTF, 2015.

HE, W.; GOLLA, M.; PADHI, R.; OFEK, J.; DÜRMUTH, M.; FERNANDES, E.; UR, B. Rethinking access control and authentication for the home Internet of things (IoT). In: USENIX ASSOCIATION. *Proceedings of the 27th USENIX Conference on Security Symposium*. [S.l.], 2018. p. 255–272.

IACOVAZZI, A.; BAIOCCHI, A. Internet traffic privacy enhancement with masking: Optimization and tradeoffs. *IEEE Transactions on Parallel and Distributed Systems*, v. 25, n. 2, p. 353–362, Feb 2014. ISSN 1045-9219.

IACOVAZZI, A.; BAIOCCHI, A. Protecting traffic privacy for massive aggregated traffic. *Computer Networks*, v. 77, p. 1 – 17, 2015. ISSN 1389-1286. Disponível em: <http://www.sciencedirect.com/science/article/pii/S138912861400437X>.

JAIN, R. *The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling*. [S.l.]: John Wiley & Sons, 1990.

JANKOWSKI, B.; MAZURCZYK, W.; SZCZYPIORSKI, K. Padsteg: introducing inter-protocol steganography. *Telecommunication Systems*, Springer, v. 52, n. 2, p. 1101–1111, 2013.

JUAREZ, M.; IMANI, M.; PERRY, M.; DIAZ, C.; WRIGHT, M. Toward an efficient website fingerprinting defense. In: ASKOXYLAKIS, I.; IOANNIDIS, S.; KATSIKAS, S.; MEADOWS, C. (Ed.). *Computer Security – ESORICS 2016*. [S.l.]: Springer International Publishing, 2016. p. 27–46. ISBN 978-3-319-45744-4.

KANG, J. Information privacy in cyberspace transactions. *Stanford Law Review*, JSTOR, p. 1193–1294, 1998.

Kennedy, S.; Li, H.; Wang, C.; Liu, H.; Wang, B.; Sun, W. I can hear your alexa: Voice command fingerprinting on smart home speakers. In: *2019 IEEE Conference on Communications and Network Security (CNS)*. [S.l.: s.n.], 2019. p. 232–240.

KRAEMER, M. J. Preserving privacy in smart homes: A socio-cultural approach. In: *Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems*. New York, NY, USA: ACM, 2018. p. DC12:1–DC12:4. ISBN 978-1-4503-5621-3. Disponível em: <http://doi.acm.org/10.1145/3170427.3173018>.

KREUTZ, D.; RAMOS, F. M. V.; VERíSSIMO, P. E.; ROTHENBERG, C. E.; AZODOLMOLKY, S.; UHLIG, S. Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, v. 103, n. 1, p. 14–76, Jan 2015. ISSN 0018-9219.

KUSHALNAGAR, N.; MONTENEGRO, G.; SCHUMACHER, C. *IPv6 over low-power wireless personal area networks (6LoWPANs): overview, assumptions, problem statement, and goals*. [S.l.], 2007.

LEE, T.; PAPPAS, C.; PERRIG, A. Bootstrapping privacy services in today's Internet. *SIGCOMM Comput. Commun. Rev.*, ACM, New York, NY, USA, v. 48, n. 5, p. 21–30, jan. 2019. ISSN 0146-4833. Disponível em: <http://doi.acm.org/10.1145/3310165.3310169>.

Li, J.; Chen, R.; Su, J.; Huang, X.; Wang, X. Me-TLS: Middlebox-enhanced TLS for internet of things devices. *IEEE Internet of Things Journal*, p. 1–1, 2019.

Li, J.; Zhou, L.; Li, H.; Yan, L.; Zhu, H. Dynamic traffic feature camouflaging via generative adversarial networks. In: *2019 IEEE Conference on Communications and Network Security (CNS)*. [S.l.: s.n.], 2019. p. 268–276.

Li, L.; Oikonomou, G.; Beach, M.; Nejabati, R.; Simeonidou, D. An SDN agent-enabled rate adaptation framework for WLAN. In: *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*. [S.l.: s.n.], 2019. p. 1–6. ISSN 1550-3607.

LIBERATORE, M.; LEVINE, B. N. Inferring the source of encrypted http connections. In: *Proceedings of the 13th ACM Conference on Computer and Communications Security*. New York, NY, USA: ACM, 2006. p. 255–263. ISBN 1-59593-518-5. Disponível em: <http://doi.acm.org/10.1145/1180405.1180437>.

LIU, J.; ZHANG, C.; FANG, Y. EPIC: A differential privacy framework to defend smart homes against Internet traffic analysis. *IEEE Internet of Things Journal*, v. 5, n. 2, p. 1206–1217, April 2018.

LIU, W. M.; WANG, L.; CHENG, P.; REN, K.; ZHU, S.; DEBBABI, M. PPTP: Privacy-preserving traffic padding in web-based applications. *IEEE Transactions on Dependable and Secure Computing*, v. 11, n. 6, p. 538–552, Nov 2014. ISSN 1545-5971.

LU, D.; BHAT, S.; KWON, A.; DEVADAS, S. DynaFlow: An efficient website fingerprinting defense based on dynamically-adjusting flows. In: *Proceedings of the 2018 Workshop on Privacy in the Electronic Society*. New York, NY, USA: ACM, 2018. p. 109–113. ISBN 978-1-4503-5989-4. Disponível em: <http://doi.acm.org/10.1145/3267323.3268960>.

MAITI, R. R.; SIBY, S.; SRIDHARAN, R.; TIPPENHAUER, N. O. Link-layer device type classification on encrypted wireless traffic with COTS radios. In: FOLEY, S. N.; GOLLMANN, D.; SNEKKENES, E. (Ed.). *Computer Security – ESORICS 2017*. Cham: Springer International Publishing, 2017. p. 247–264. ISBN 978-3-319-66399-9.

MALEKZADEH, M.; CLEGG, R. G.; CAVALLARO, A.; HADDADI, H. Protecting sensory data against sensitive inferences. *arXiv preprint*, 2018.

MALEKZADEH, M.; CLEGG, R. G.; HADDADI, H. Replacement autoencoder: A privacy-preserving algorithm for sensory data analysis. *CoRR*, 2018. Disponível em: <http://arxiv.org/abs/1710.06564>.

MARIKYAN, D.; PAPAGIANNIDIS, S.; ALAMANOS, E. A systematic review of the smart home literature: A user perspective. *Technological Forecasting and Social Change*, v. 138, p. 139 – 154, 2019. ISSN 0040-1625. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0040162517315676>.

MAZURCZYK, W.; WENDZEL, S.; ZANDER, S.; HOUMANSADR, A.; SZCZYPIORSKI, K. *Information hiding in communication networks: fundamentals, mechanisms, applications, and countermeasures*. [S.l.]: John Wiley & Sons, 2016. v. 7.

MIETTINEN, M.; MARCHAL, S.; HAFEEZ, I.; ASOKAN, N.; SADEGHI, A. R.; TARKOMA, S. IoT sentinel: Automated device-type identification for security enforcement in IoT. In: *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. [S.l.: s.n.], 2017. p. 2177–2184. ISSN 1063-6927.

MOGHADDAM, H. M.; LI, B.; DERAKHSHANI, M.; GOLDBERG, I. SkypeMorph: Protocol obfuscation for Tor bridges. In: *Proceedings of the 2012 ACM Conference on Computer and Communications Security*. New York, NY, USA: ACM, 2012. p. 97–108. ISBN 978-1-4503-1651-4. Disponível em: <http://doi.acm.org/10.1145/2382196.2382210>.

MONTGOMERY, D. C.; RUNGER, G. C. *Applied statistics and probability for engineers*. [S.l.]: John Wiley & Sons, 2010.

Nguyen-An, H.; Silverston, T.; Yamazaki, T.; Miyoshi, T. Generating IoT traffic in smart home environment. In: *2020 IEEE 17th Annual Consumer Communications Networking Conference (CCNC)*. [S.l.: s.n.], 2020. p. 1–2.

PFLEEGER, S. Experimental design and analysis in software engineering. *Annals of Software Engineering*, Springer, v. 1, n. 1, p. 219–253, 1995. ISSN 1022-7091. Disponível em: <http://dx.doi.org/10.1007/BF02249052>.

PINHEIRO, A. J.; BEZERRA, J. de M.; BURGARDT, C. A.; CAMPELO, D. R. Identifying IoT devices and events based on packet length from encrypted traffic. *Computer Communications*, v. 144, p. 8 – 17, 2019. ISSN 0140-3664. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0140366419300052>.

PINHEIRO, A. J.; BEZERRA, J. M.; CAMPELO, D. Packet padding for improving privacy in consumer IoT. In: *2018 IEEE Symposium on Computers and Communications (ISCC)*. Natal, Rio Grande do Norte, BR: IEEE, 2018.

Poh, G. S.; Gope, P.; Ning, J. PrivHome: Privacy-preserving authenticated communication in smart home environment. *IEEE Transactions on Dependable and Secure Computing*, p. 1–1, 2019.

POULARAKIS, K.; IOSIFIDIS, G.; SMARAGDAKIS, G.; TASSIULAS, L. Optimizing gradual SDN upgrades in isp networks. *IEEE/ACM Trans. Netw.*, IEEE Press, Piscataway, NJ, USA, v. 27, n. 1, p. 288–301, fev. 2019. ISSN 1063-6692. Disponível em: <https://doi.org/10.1109/TNET.2018.2890248>.

RAZALI, N. M.; WAH, Y. B. Power comparisons of shapiro-wilk, kolmogorov-smirnov, lilliefors and anderson-darling tests. *Journal of Statistical Modeling and Analytics*, v. 2, n. 1, p. 21–33, 2011.

REN, J.; DUBOIS, D. J.; CHOFFNES, D.; MANDALARI, A. M.; KOLCUN, R.; HADDADI, H. Information exposure from consumer IoT devices: A multidimensional, network-informed measurement approach. In: *Proceedings of the Internet Measurement Conference*. New York, NY, USA: ACM, 2019. p. 267–279. ISBN 978-1-4503-6948-0. Disponível em: <http://doi.acm.org/10.1145/3355369.3355577>.

RESCORLA, E. *The transport layer security (TLS) protocol version 1.3*. [S.l.], 2018.

Rezaei, S.; Liu, X. Deep learning for encrypted traffic classification: An overview. *IEEE Communications Magazine*, v. 57, n. 5, p. 76–81, May 2019. ISSN 1558-1896.

SANTOS, M. R. P.; ANDRADE, R. M. C.; GOMES, D. G.; CALLADO, A. C. An efficient approach for device identification and traffic classification in IoT ecosystems. Natal, Rio Grande do Norte, BR, p. 1–6, 2018.

SCHUSTER, R.; SHMATIKOV, V.; TROMER, E. Beauty and the burst: Remote identification of encrypted video streams. In: *USENIX Security*. [S.l.: s.n.], 2017.

SERROR, M.; HENZE, M.; HACK, S.; SCHUBA, M.; WEHRLE, K. Towards in-network security for smart homes. In: *Proceedings of the 13th International Conference on Availability, Reliability and Security*. New York, NY, USA: ACM, 2018. p. 18:1–18:8. ISBN 978-1-4503-6448-5. Disponível em: <http://doi.acm.org/10.1145/3230833.3232802>.

SHAHBAR, K.; ZINCIR-HEYWOOD, A. N. Traffic flow analysis of Tor pluggable transports. In: *2015 11th International Conference on Network and Service Management*. [S.l.: s.n.], 2015. p. 178–181.

SHIN, J.; PARK, Y.; LEE, D. Who will be smart home users? an analysis of adoption and diffusion of smart homes. *Technological Forecasting and Social Change*, v. 134, p. 246 – 253, 2018. ISSN 0040-1625. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0040162518300696>.

Shirali-Shahreza, S.; Ganjali, Y. Protecting home user devices with an SDN-based firewall. *IEEE Transactions on Consumer Electronics*, v. 64, n. 1, p. 92–100, Feb 2018. ISSN 0098-3063.

SIBY, S.; JUAREZ, M.; DIAZ, C.; VALLINA-RODRIGUEZ, N.; TRONCOSO, C. Encrypted DNS–> privacy? a traffic analysis perspective. *arXiv preprint arXiv:1906.09682*, 2019.

SIVANATHAN, A.; GHARAKHEILI, H. H.; LOI, F.; RADFORD, A.; WIJENAYAKE, C.; VISHWANATH, A.; SIVARAMAN, V. Classifying IoT devices in smart environments using network traffic characteristics. *IEEE Transactions on Mobile Computing*, p. 1–1, 2018. ISSN 1536-1233.

Sivanathan, A.; Gharakheili, H. H.; Sivaraman, V. Managing IoT cyber-security using programmable telemetry and machine learning. *IEEE Transactions on Network and Service Management*, p. 1–1, 2020. ISSN 2373-7379.

SIVANATHAN, A.; SHERRATT, D.; GHARAKHEILI, H. H.; RADFORD, A.; WIJENAYAKE, C.; VISHWANATH, A.; SIVARAMAN, V. Characterizing and classifying IoT traffic in smart cities and campuses. In: *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. [S.l.: s.n.], 2017. p. 559–564.

TABASSUM, M.; KOSINSKI, T.; LIPFORD, H. R. I don't own the data": End user perceptions of smart home device data practices and risks. In: *Fifteenth Symposium on Usable Privacy and Security (SOUPS 2019)*. Santa Clara, CA: USENIX Association, 2019. Disponível em: <https://www.usenix.org/conference/soups2019/presentation/tabassum>.

TAKBIRI, N.; HOUMANSADR, A.; GOECKEL, D. L.; PISHRO-NIK, H. Matching anonymized and obfuscated time series to users' profiles. *CoRR*, 2017. Disponível em: <http://arxiv.org/abs/1710.00197>.

TAYLOR, V. F.; SPOLAOR, R.; CONTI, M.; MARTINOVIC, I. Robust smartphone app identification via encrypted network traffic analysis. *IEEE Transactions on Information Forensics and Security*, v. 13, n. 1, p. 63–78, Jan 2018. ISSN 1556-6013.

Thangavelu, V.; Divakaran, D. M.; Sairam, R.; Bhunia, S. S.; Gurusamy, M. DEFT: A distributed IoT fingerprinting technique. *IEEE Internet of Things Journal*, v. 6, n. 1, p. 940–952, Feb 2019. ISSN 2327-4662.

UDDIN, M.; NADEEM, T.; NUKAVARAPU, S. Extreme SDN framework for IoT and mobile applications flexible privacy at the edge. *International Conference on Pervasive Computing and Communications*, v. 200, p. 250, 2019.

UJJAN, R. M. A.; PERVEZ, Z.; DAHAL, K.; BASHIR, A. K.; MUMTAZ, R.; GONZáLEZ, J. Towards sFlow and adaptive polling sampling for deep learning based DDoS detection in SDN. *Future Generation Computer Systems*, 2019. ISSN 0167-739X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167739X19318333>.

WAGNER, I.; ECKHOFF, D. Technical privacy metrics: A systematic survey. *ACM Comput. Surv.*, ACM, New York, NY, USA, v. 51, n. 3, p. 57:1–57:38, jun. 2018. ISSN 0360-0300. Disponível em: <http://doi.acm.org/10.1145/3168389>.

WARREN, S. D.; BRANDEIS, L. D. The right to privacy. *Harvard law review*, JSTOR, p. 193–220, 1890.

WEBER, R. H.; WEBER, R. *Internet of Things*. [S.l.]: Springer, 2010.

WEINBERG, Z.; WANG, J.; YEGNESWARAN, V.; BRIESEMEISTER, L.; CHEUNG, S.; WANG, F.; BONEH, D. StegoTorus: A camouflage proxy for the Tor anonymity system. In: *Proceedings of the 2012 ACM Conference on Computer and Communications Security*. New York, NY, USA: ACM, 2012. p. 109–120. ISBN 978-1-4503-1651-4. Disponível em: <http://doi.acm.org/10.1145/2382196.2382211>.

WESTIN, A. F. Privacy and freedom. *Washington and Lee Law Review*, v. 25, n. 1, p. 166, 1968.

Wu, X.; Yang, Q.; Liu, X.; Jin, D.; Lee, C. W. A hardware-in-the-loop emulation testbed for high fidelity and reproducible network experiments. In: *2017 Winter Simulation Conference (WSC)*. [S.l.: s.n.], 2017. p. 408–418.

XIONG, S.; SARWATE, A. D.; MANDAYAM, N. B. Defending against packet-size side-channel attacks in IoT networks. In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [S.l.: s.n.], 2018. p. 2027–2031. ISSN 2379-190X.

XU, C.; REN, J.; ZHANG, D.; ZHANG, Y. Distilling at the edge: A local differential privacy obfuscation framework for IoT data analytics. *IEEE Communications Magazine*, v. 56, n. 8, p. 20–25, August 2018. ISSN 0163-6804.

XU, K.; WANG, X.; WEI, W.; SONG, H.; MAO, B. Toward software defined smart home. *IEEE Communications Magazine*, v. 54, n. 5, p. 116–122, May 2016. ISSN 0163-6804.

Yang, A.; Zhang, C.; Chen, Y.; Zhuansun, Y.; Liu, H. Security and privacy of smart home systems based on the internet of things and stereo matching algorithms. *IEEE Internet of Things Journal*, p. 1–1, 2019.

YAO, Y.; BASDEO, J. R.; KAUSHIK, S.; WANG, Y. Defending my castle: A co-design study of privacy mechanisms for smart homes. In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. New York, NY, USA: ACM, 2019. (CHI '19), p. 198:1–198:12. ISBN 978-1-4503-5970-2. Disponível em: <http://doi.acm.org/10.1145/3290605.3300428>.

ZENG, E.; MARE, S.; ROESNER, F. End user security & privacy concerns with smart homes. In: *Symposium on Usable Privacy and Security*. [S.l.: s.n.], 2017.

ZHENG, S.; APTHORPE, N.; CHETTY, M.; FEAMSTER, N. User perceptions of smart home IoT privacy. *Proc. ACM Hum.-Comput. Interact.*, ACM, New York, NY, USA, v. 2, p. 200:1–200:20, nov. 2018. ISSN 2573-0142. Disponível em: <http://doi.acm.org/10.1145/3274469>.

APÊNDICE A – ESTATÍSTICAS DO TAMANHO DO PACOTE

As Tabelas 7, 8, 9 e 10 apresentam a média, mediana, moda e terceiro quartil para o tamanho do pacote gerado pelos dispositivos IoT.

Tabela 7 – Estatísticas média, mediana, moda e terceiro quartil para o tamanho do pacote contido no tráfego disponibilizado em (SIVANATHAN et al., 2018).

Dispositivo	Média	Mediana	Moda	Terceiro quartil
Blipcare Blood Pressure meter	105,7	59	54	113
iHome	98,73	108	54	123
Smart Things hub	71,41	60	60	60
Samsung SmartCam	346,7	421	66	489
Amazon Echo	115,7	90	66	98
Withings Smart Baby Monitor	79,48	66	66	66
Belkin WeMo Switch	366,2	118	66	350
Belkin WeMo Motion Sensor	112,8	66	66	118
Withings Aura Smart Sleep Sensor	138,5	66	66	225
Triby Speaker	104,4	66	66	70
PIX-STAR Photo-Frame	111,9	78	66	110
HP Printer	134,9	86	86	86
Insteon Camera	104,4	90	102	102
Light Bulbs LiFX Smart Bulb	96,43	92	123	123
Dropcam	206,6	156	156	156
TP-Link Smart plug	107,8	66	172	172
Withings Smart scale	300,7	333	333	342
Netatmo weather station	171,3	113	350	350
NEST Protect smoke alarm	294,9	350	509	509
Netatmo Welcome	471,3	86	1510	1510
TP-Link Day Night Cloud câmera	532	160	1514	1270

A partir das Tabelas 8, 9 e 10, pode-se observar uma variação nas estatísticas para o mesmo dispositivo em ambientes experimentais distintos. Essas variações são decorrentes de interações diferentes dos usuários com os mesmos dispositivos em *testbeds* distintos. Os dados usados para computar as estatísticas apresentadas nas Tabelas 8 e 10 foram disponibilizados como um conjunto de ações desempenhadas por cada equipamento nos *testbeds*. Para cada ação, os autores repetiram os experimentos múltiplas vezes e armazenam os dados coletados em arquivos PCAP.

Tabela 8 – Estatísticas média, mediana, moda e terceiro quartil para o tamanho do pacote contido no tráfego capturado no *testbed* configurado na *Northeastern University* e disponibilizado pelos autores de (REN et al., 2019).

Dispositivo	Média	Mediana	Moda	Terceiro quartil
Smarter iKettle	233,43	187	171	251
Amcrest security camera	544,20	436	1358	512
Apple TV	354,32	119	60	583
Blink Camera	1027,69	1467	1467	1467
Blink Security Hub	147,09	135	135	135
Behmon Brewer	425,24	427	491	491
Flux Bulb	180,65	218	218	218
Amazon CloudCam	250,61	144	128	312
D-Link Motion Sensor	190,15	112	74	229
Samsung Dryer	305,79	150	74	487
Amazon Echo Dot	304,63	215	491	491
Amazon Echo Plus	199,57	146	172	191
Amazon Echo Spot	178,86	98	50	172
Amazon Fire TV	369,06	256	74	565
Samsung Fridge	325,16	176	74	551
Google Home Mini	756,48	397	1434	1434
Insteon Hub	205,04	60	60	379
Harman Kardon Invoke with Cortana	559,86	393	1494	873
Lefun Camera	623,23	220	1328	1328

Os dados gerados em cada atividade de um equipamento foram organizados em pastas específicas. Por isso, é possível determinar quais ações foram executadas por cada dispositivo. A partir desses dados, foi possível determinar que para um mesmo equipamento, diferentes ações foram realizadas em conjuntos de dados distintos. Por exemplo, no *testbed* UK, os autores interagem com a tomada da TP-*Link* através do *Google Home*, o que não ocorre no *testbed* US.

Tabela 9 – Estatísticas média, mediana, moda e terceiro quartil para o tamanho do pacote contido no tráfego capturado no *testbed* US e disponibilizado pelos autores de (REN et al., 2019).

Dispositivo	Média	Mediana	Moda	Terceiro quartil
LG *Smart* TV	367,15	341	438	438
Lightify Hub	107,05	86	74	124
Luohe Spy Camera	500,72	359	1074	1074
MagicHome Strip	73,15	62	58	68
Microseven Camera	256,70	242	242	242
GE Microwave	363	315	315	347
Nest Thermostat	486,28	223	74	619
Philips Bulb	121,79	122	74	138
Ring Doorbell	496,91	214	214	722
Roku TV	343,31	113	74	439
Samsung Smart TV	276,14	200	74	373
Sengled Hub	242,72	210	104	428
SmartThings Hub	171,11	116	113	168
Anova Sous Vide	129,68	74	58	135
Philips Hub	302,21	88	60	340
WeMo Plug	457,65	265	74	555
TP-Link Bulb	420,01	221	58	653
TP-Link Plug	383,95	257	74	639
Wansview Cam	159,38	60	60	210
Samsung Washer	318,47	150	74	504
Wink Hub	240,86	167	151	332
Xiaomi Cleaner	141,17	106	74	138
Xiaomi Hub	133,72	106	74	186
Xiaomi Ricecooker	141,29	113	74	170
Xiaomi Strip	118,47	106	74	138
Yi Camera	374,62	103	46	854
Zmodo Doorbell	1203,50	1434	1434	1434

Tabela 10 – Estatísticas média, mediana, moda e terceiro quartil para o tamanho do pacote contido no tráfego capturado no *testbed* configurado no *Imperial college* e disponibilizado em (REN et al., 2019).

Dispositivo	Média	Mediana	Moda	Terceiro quartil
MagicHome Strip	107,20	69	60	78
Harman Kardon Allure with Alexa	496,93	90	60	1088
Apple TV	452,44	66	66	1119
Blink Camera	817,63	1451	146	1467
Blink Security Hub	112,83	90	66	135
Bosiwo Camera Wired	883,68	1328	132	1328
WiMaker Spy Camera	218,85	126	260	260
Amazon Echo Dot	235,52	97	60	172
Amazon Echo Plus	257,145	90	60	172
Amazon Echo Spot	293,97	97	60	172
Amazon Fire TV	573,14	78	151	1514
Google Home Mini	525,03	156	66	1434
Google Home	515,47	156	66	1434
Honeywell Thermostat	302,54	139	54	651
Insteon Hub	138	62	60	254
Lightify Hub	150,58	108	108	236
Nest Thermostat	295,87	74	66	326
Netatmo Weather Station	194,92	78	78	344
Ring Doorbell	53299	214	214	1251
Roku TV	363,32	74	66	384
Samsung TV *Wired*	450,65	66	66	819
Sengled Hub	139,82	85	60	191
Smarter Coffee Machine	176,60	171	54	235
SmartThings Hub	145,54	66	60	151
Anova Sous Vide	64,06	63	54	68
Philips Hub	210,95	66	60	219
WeMo Plug	463,51	66	60	840
TP-Link Bulb	277,42	60	54	256
TP-Link Plug	244,82	74	66	305
Wansview Camera Wired	245,26	60	60	279
Xiaomi Camera	368,75	102	102	774
Xiaomi Cleaner	115,68	90	90	90
Xiaomi Hub	122,48	106	86	150
Yi Camera	362,98	171	60	496

APÊNDICE B – RESULTADOS COMPLEMENTARES

Estatísticas para o *byte overhead* induzido pelas soluções de preenchimento analisadas são apresentados nas Tabelas 11 e 12.

Tabela 11 – Estatísticas para o tamanho modificado pelas estratégias de preenchimento aplicadas aos dados disponibilizados em (REN et al., 2019).

Estratégia	*Testbed*	Média (*bytes*)	Mediana (*bytes*)	Moda (*bytes*)
Original	US	336	134	60
	UK	462	214	60
Linear	US	409	256	128
	UK	519	256	138
Exponencial	US	428	256	64
	UK	564	256	1500
Ratos/Elefantes	US	884	1500	1500
	UK	1041	1500	1500
Random 255	US	470	292	1500
	UK	585	331	1500
MTU	US	1500	1500	1500
	UK	1500	1500	1500
Random	US	924	973	1514
	UK	981	1081	1514
$Preenchimento(m = 100)$	US	428	200	100
	UK	540	300	100
$Preenchimento(m = 500)$	US	651	500	500
	UK	735	500	500
$Preenchimento(m = 700)$	US	809	700	700
	UK	872	700	700
$Preenchimento(m = 900)$	US	974	900	900
	UK	1014	900	900

Tabela 12 – Estatísticas para o tamanho modificado pelos mecanismos de preenchimento avaliados nos dados disponibilizados pelos autores de (SIVANATHAN et al., 2018).

Estratégia	Média (*bytes*)	Mediana (*bytes*)	Moda (*bytes*)
Original	214	118	156
Linear	275	128	128
Exponencial	277	128	128
Ratos/Elefantes	865	1500	1500
Random 255	337	264	1514
MTU	1501	1500	1500
Random	858	867	1514
Preenchimento($m = 100$)	279	200	100
Preenchimento($m = 500$)	554	500	500
Preenchimento($m = 700$)	738	700	700
Preenchimento($m = 900$)	928	900	900