# Monitoramento e detecção de anomalias no tráfego de rede em smart homes

**Anderson B. Ribeiro**[1], **Rafael L. Gomes**[1]

[1]Centro de Ciência e Tecnologia (CCT) - Universidade Estadual do Ceará (UECE)
Fortaleza – CE – Brazil

anderson.bezerra@aluno.uece.br, rafaellgom@larces.uece.br

***Abstract.*** *Ensuring a smart home network security is challenging. These networks are composed of several devices connected to each other and to the internet. The devices use sensors to extract data from the environment, then transmit them to their actuators and respond to them. The problem is that each device has its peculiarities regarding the technology used, where vulnerabilities exist in each of these devices. Ensuring the security of each device individually is a complex task, which would generate a high computational cost. Within this context, the analysis of the traffic generated by the devices is presented as an alternative to monitor these devices and to identify possible anomalies. In this way, this work presents an application to identify anomalies in network traffic, considering the various existing devices.*

***Resumo.*** *Garantir a segurança de uma rede de smart home é algo desafiador. Essas redes são compostas por diversos dispositivos conectados entre si e com a Internet. Os dispositivos utilizam sensores para extrair dados do ambiente, depois transmitem para os seus atuadores e respondem a estes. O problema é que cada dispositivo tem suas particularidades quanto à tecnologia usada, onde vulnerabilidades existem em cada um desses dispositivos. Garantir a segurança de cada dispositivo individualmente é uma tarefa complexa, a qual geraria um alto custo computacional. Dentro deste contexto, a análise do tráfego gerado pelos dispositivos se apresenta como uma alternativa para monitoramento desses dispositivos e identificação de possíveis anomalias. Desta forma, este trabalho apresenta uma aplicação para identificação de anomalias no tráfego de rede, considerando os diversos dispositivos existentes.*

## 1. Introdução

A revolução da Internet levou a interconexão entre as pessoas em uma escala nunca antes vista. Com a crescente popularidade de dispositivos inteligentes, [Gubbi et al. 2013] preveem que a próxima grande revolução será a da interconexão entre objetos, criando um ambiente inteligente. O termo Internet das Coisas (do inglês, *Internet of Things* - IoT) representa os dispositivos a nossa volta que, mesmo sem percebermos, estão conectados à rede, analisando o ambiente, trocando informações [Gubbi et al. 2013].

O paradigma IoT vem sendo largamente estudado na última década, com novas aplicações surgindo a cada momento. Geralmente utilizam-se sensores para coletar informações e posteriormente realizar alguma ação com base na leitura. Hoje em dia já

existem aplicações IoT em hospitais, fábricas, transporte, e até mesmo na sua própria casa, criando o contexto de *smart home*.

O conceito de *smart home* vem sendo estudado desde o fim da década de 70, mas só com o avanço do tempo e da tecnologia tornou-se viável e popular [Jose and Malekian 2017]. Podemos definir uma *smart home* como casas equipadas com diversos dispositivos inteligentes, que utilizam sensores e atuadores para controlar seu uso, como sistemas de luzes que só ligam quando está escuro e tem alguém no cômodo, sistemas de refrigeração ou aquecimento que ligam quando o morador está chegando em casa, sistema de irrigação que liga quando a umidade da terra está baixa, etc. O uso dessa tecnologia, além de facilitar o dia-a-dia dos moradores, pode reduzir o consumo de água, energia, dentre outros.

*Smart homes* são um grande avanço, mas ainda há desafios a serem superados. O crescente número de usuários (com comportamentos imprevisíveis) e de novos dispositivos trouxe desafios para a automação residencial. A segurança desse tipo de dispositivo deve ser muito bem projetada, pois esses estão dentro de nossas casas e ainda são vulneráveis. Antigamente, tínhamos que nos preocupar somente com criminosos invadindo nossa casa fisicamente, mas hoje há também a preocupação de alguém explorar as vulnerabilidades da rede doméstica e nos invadir virtualmente, controlando dispositivos e as informações contidas neles [Jose and Malekian 2017].

Embora fosse ideal, garantir a segurança de cada dispositivo individualmente é uma tarefa complexa, a qual geraria um alto custo computacional. Dentro deste contexto, a análise do tráfego gerado pelos dispositivos se apresenta como um alternativa para monitoramento desses dispositivos e identificação de possíveis anomalias. O objetivo deste trabalho é o desenvolvimento de uma aplicação que analisa os pacotes que passam pelo *gateway* de uma *smart home*, a fim de identificar fluxos e possíveis dispositivos com comportamento anômalo.

O restante deste artigo está organizado da seguinte forma: a Seção 2 descreve o contexto de IoT, Smart Homes, aprendizado de máquina e detecção de anomalias, enquanto que os trabalhos relacionados existentes são apresentados na Seção 3; a Seção 4 introduz a aplicação proposta; a Seção 5 apresenta o resultado dos experimentos realizados; por fim, a Seção 6 conclui o artigo e cita os trabalhos futuros.

## 2. Fundamentação Teórica

### 2.1. IoT e Smart Homes

O termo Internet das Coisas foi utilizado pela primeira vez em 1999 pelo pesquisador britânico Kevin Ashton em um projeto que utilizava RFID (do inglês, *Radio-Frequency IDentification*) [Ashton et al. 2009]. Ela consiste em uma rede sem fio de dispositivos inteligentes conectada via sensores inteligentes e capazes de interagir sem muita intervenção humana [Li et al. 2015].

A essência da IoT está no conceito de os dispositivos se misturarem com a existência humana, fazer parte de nossa experiência. Está em existir uma integração entre nós humanos e as coisas à nossa volta, com os vários dispositivos se comunicando de forma inteligente uns com os outros para executar tarefas diárias. Cada dispositivo está conectado com cada dispositivo, se comunicando, transferindo informações, rece-

bendo informações e respondendo de forma inteligente, executando as ações adequadas [Mendez et al. 2017].

Com o avanço da tecnologia e a popularização dos dispositivos inteligentes, a ideia de se viver em uma *smart home* tornou-se mais viável. Segundo [Perumal et al. 2008], um ambiente *smart home* pode ser definido como uma entidade que ajusta seu funcionamento às necessidades dos moradores de acordo com as informações que coleta dos mesmos. O ambiente é altamente caracterizado pela heterogeneidade, com vários sistemas que precisam cooperar para realizar suas tarefas eficientemente.

A grande quantidade de sistemas heterogêneos, com diferentes padrões e protocolos de comunicação, e a grande quantidade de dados gerados torna o gerenciamento da rede difícil, inclusive no que se refere à segurança.

### 2.2. Computação de borda

No final da década de 90, foi criado o conceito de redes de entrega de conteúdo para acelerar a performance da rede. Essas redes utilizavam nós (*hosts*) na borda da rede, próximo aos usuários, como um tipo de cache, pré-carregando páginas web e economizando largura de banda. O conceito de computação de borda (do inglês, *Edge Computing*) é uma generalização das redes de entrega de conteúdo, mas utiliza os nós como cache dos servidores em nuvem. Outra diferença é que podemos utilizar esses nós para fazer processamento e executar algoritmos, por vezes mais rápido do que processar em nuvem e buscar o resultado [Satyanarayanan 2017].

Projetos como redes elétricas inteligentes (*Smart Grid*), sistemas de monitoramento de saúde inteligentes, sistemas de supervisão de indústrias de petróleo, e aplicações para detecção de ataques de rede utilizam dados voláteis, que podem mudar a qualquer momento e precisam de um monitoramento em tempo real para reduzir as falhas. Os dados capturados precisam ser rapidamente processados e analisados, para então tomar uma decisão. Nesses cenários a computação em borda é mais efetiva.

### 2.3. Aprendizado de máquina e estratégias de aprendizado

A ideia geral do aprendizado de máquina é que o computador aprenda a realizar uma determinada tarefa em um conjunto de dados (*dataset*) de treino. Depois ele deve executar a mesma tarefa em *datasets* diferentes, com dados que ainda não foram vistos [Louridas and Ebert 2016]. As tarefas em questão se referem à execução de algoritmos de reconhecimento de padrões, tomadas de decisão, classificação, entre outros. Ao aprender a executar esses algoritmos a máquina pode passar a repeti-los em conjuntos de dados indefinidamente maiores e prever seus comportamentos, sem a necessidade de auxílio humano.

Ainda segundo [Louridas and Ebert 2016], o aprendizado de máquina pode utilizar duas estratégias:

- Aprendizado supervisionado - Esta abordagem utiliza um *dataset* de treino que contém os dados de entrada e a saída esperada. Algumas das técnicas utilizadas por essa estratégia são: Máquinas de Vetores Suporte (*Support Vector Machines* - SVMs), regressão linear, redes Bayesianas, árvores de decisão, etc;

- Aprendizado não supervisionado - Esta utiliza um *dataset* de treino que contém dados, mas não informa a saída esperada. O computador deve achar a saída por si só. Algumas das técnicas utilizadas por essa estratégia são: clusterização, detecção de anomalias e boa parte dos algoritmos de redes neurais (a exceção é o *Multilayer perceptron*).

A aplicação desenvolvida neste trabalho utiliza um algoritmo de classificação para prever o comportamento de pacotes anômalos e identificá-los na rede.

### 2.4. Detecção de anomalias

É uma área de aprendizado de máquina que foca em encontrar um elemento que foge ao padrão previamente definido. Segundo [Goldstein and Uchida 2016], algoritmos de detecção de anomalias são utilizados em muitos domínios de aplicação e frequentemente melhoram os sistemas de detecção baseados em regras. Algoritmos para detecção de intrusos ou de fraudes são exemplos de aplicações que utilizam este tipo de algoritmo.

Assim como as estratégias de aprendizado de máquina, esses algoritmos utilizam um *dataset* de treino para definir um modelo que, combinado com um outro *dataset* de teste, irá gerar um resultado. [Goldstein and Uchida 2016] classificam essas estratégias de três formas:

- Detecção de anomalia supervisionada - O *dataset* de treino e o de teste já foram previamente rotulados. Esta classificação não é muito relevante, pois teríamos que assumir que todos os tipos de anomalias são conhecidas e que todos os rótulos estão corretos;
- Detecção de anomalia semi-supervisionada - Essa estratégia também usa *dataset* de treino e teste, porém somente os dados considerados normais são rotulados. Dessa forma tem-se um modelo de dados normais que serão comparados com os novos dados. Se o dado fugir do modelo, o computador o classifica como uma anomalia;
- Detecção de anomalia não supervisionada - O *dataset* de treino não foi rotulado. O algoritmo deve analisar os padrões e definir o que é normal ou não por si só.

A aplicação desenvolvida neste trabalho utiliza o aprendizado não supervisionado para definir o perfil de usuários da rede, pois o uso da rede que vai definir qual é o comportamento normal esperado, inviabilizando a rotulação prévia dos dados.

## 3. Trabalhos Relacionados

Em [Komninos et al. 2014], é realizado um estudo sobre os problemas, desafios e contramedidas de segurança em *smart grids* e *smart homes*. Neste estudo, os autores descrevem a arquitetura de ambos os ambientes e destacam os benefícios de sua interação, considerando a *smart home* como uma parte integral da *grid*. Em seguida, são apresentados algumas metas de segurança (confiabilidade, integridade, autenticidade, etc.), como garantir essas metas, que tipos de ataques podem afetar cada uma e o impacto que eles têm. Os autores propõem contramedidas para cada tipo específico de ataque citado e afirmam que, devido a heterogeneidade da rede, não há um método que aborda todos os casos. No entanto, a técnica utilizada neste trabalho se mostrará mais flexível em relação

a identificação de ataques, pois os ataques, de forma geral, podem ser identificados como anomalias.

Em [Jose and Malekian 2017], é realizado um experimento de automação residencial onde sensores detectam padrões de comportamento das pessoas na casa e comparam com os padrões já conhecidos dos moradores, para assim detectar invasores. Os autores expõem algumas vulnerabilidades das formas de comunicação utilizadas pelos sensores aos diversos tipos de ataque à rede, como o ZigBee, forma de comunicação utilizada no experimento que é vulnerável a ataques de repetição. O experimento realizado utiliza criptografia AES (do inglês, *Advanced Encryption Standard*) com uma chave de 128 bits e nunca compartilha a chave, garantindo que não seja possível a captura de informações pela rede, evitando assim o ataque mencionado. Embora a privacidade dos dados transmitidos pelos sensores esteja garantida, a segurança ainda pode ser comprometida de outras formas, como no caso de algum sensor apresentar um defeito e parar de analisar o ambiente. A aplicação desenvolvida neste trabalho identificaria esse tipo de problema ao constatar a redução na quantidade de dados passando pela rede.

Em [Raza et al. 2013], é apresentada uma nova solução para detecção de invasões em IoT chamada SVELTE. Seu endereçamento segue o padrão 6LoWPAN (*IPv6 over Low power Wireless Personal Area Networks*), uma versão comprimida do IPv6 (do inglês, *Internet Protocol version 6*), e o roteamento pelo protocolo RPL (IPv6 Routing Protocol for Low power and Lossy Networks). O sistema possui três módulos, um de mapeamento, um de detecção de invasão e um *mini-firewall*. O módulo de mapeamento reconstrói o roteamento feito pelo RPL e o estende com parâmetros adicionais de detecção. O módulo de detecção identifica, por meio de diversas técnicas, *spoofing*, informações alteradas, buracos de minhoca e ataques de repasse seletivo, podendo ser estendido para outros tipos de ataque. O *mini-firewall*, além de proteger contra as típicas ameaças externas, fornece proteção em tempo real das ameaças internas. Embora o SVELTE forneça proteção contra os tipos de ataques mais comuns, ainda restam muitas ameaças que conseguiriam penetrar as defesas do sistema. Outra desvantagem é que ele só funciona para um contexto específico (6LoWPAN e RPL), diferente do que será apresentado.

[Liu and Nielsen 2016] propõem um algoritmo de detecção de anomalias com aprendizado supervisionado e baseado em estatísticas do histórico do consumo de energia dos clientes e o uso de uma arquitetura lambda para aumentar a eficiência da atualização do modelo de detecção em tempo real. A arquitetura é implementada com tecnologias híbridas e avaliada em um ambiente fechado e com *datasets* reais. Embora a proposta tenha se mostrado muito eficiente, ela até então detecta apenas um tipo de anomalia. A proposta deste trabalho se mostra superior em relação a quantidade de anomalias que pode detectar, como será mostrado a seguir.

**Tabela 1. Trabalhos relacionados**

| Autor | Contexto | Sumário |
|---|---|---|
| [Komninos et al. 2014] | Smart Grids | Metas de segurança |
| [Jose and Malekian 2017] | Detecção de anomalias | Vulnerabilidade a ataques de repetição |
| [Raza et al. 2013] | IoT | Detecção de invasões |
| [Liu and Nielsen 2016] | Detecção de anomalias | Detecção em tempo real |

Este trabalho difere dos anteriormente citados por abstrair métodos de invasão ou defeitos em equipamentos, tratando-os de forma genérica como uma anomalia, e por analisar o fluxo de dados, em vez de cada pacote individualmente.

## 4. Proposta

Nesta seção, será apresentada a proposta do monitor de tráfego de redes *smart home*[1]. Na seção 4.1 será descrita a metodologia utilizada, em seguida, na seção 4.2, será explicada a proposta da aplicação. Na seção 4.3 serão descritos o ambiente e as principais bibliotecas utilizadas para os testes. Por fim, a seção 5 corresponde à descrição e resultado dos experimentos.

### 4.1. Metodologia

A metodologia adotada para o desenvolvimento da aplicação inclui os seguintes passos:

1. Criar um ambiente para executar os testes do *software* desenvolvido. O ambiente deve emular uma rede isolada para que a captura e organização dos pacotes seja simplificada, representando o contexto de *smart home*;
2. Monitorar uma determinada interface de rede e sua largura de banda, capturar os pacotes e organizá-los em uma tabela de fluxos;
3. Checar a tabela periodicamente para caso um tipo de pacote não seja novamente referenciado em um tempo predefinido, este será removido da tabela, já que trata-se de um caso não-recorrente;
4. Gerar um perfil da rede após um tempo de monitoramento ou quantidade de pacote pré-definidos;
5. Analisar o tráfego passante e comparar com o perfil definido. Caso os atributos do pacote fujam ao perfil ou a largura de banda exceder um determinado limite, considerar como uma anomalia.

### 4.2. Aplicação

O objetivo da aplicação é analisar os pacotes que passam pelo *gateway* da rede a fim de identificar fluxos ou dispositivos com comportamento anômalo. Para isso, executa-se o processo ilustrado na Figura 1.

Logo no início do processo o fluxo principal é dividido em dois. Seguindo o fluxo principal do processo executa-se a primeira etapa, que consiste em capturar os dados que serão usados para montar o perfil da rede. Nesta etapa é utilizado um *sniffer* no *gateway* da rede para capturar e copiar pacotes por um determinado tempo ou por quantidade de pacotes. Os dados copiados são guardados temporariamente em uma tabela, que é utilizada na próxima etapa.

Na segunda etapa, os dados capturados anteriormente são utilizados para montar o perfil da rede, treinando um classificador. No entanto, este presume que os dados seguem alguma distribuição, então antes de usar os dados é feito um pré-processamento para que sigam uma distribuição normal. Os atributos utilizados pelo classificador são:

- Endereço de IP (*Internet Protocol*) de origem;

[1]Monitor de tráfego de redes *smart home*. Disponível em: ¡https://github.com/Anderson0abr/AnomalyDetection¿. Acesso em: 16 de dezembro 2018

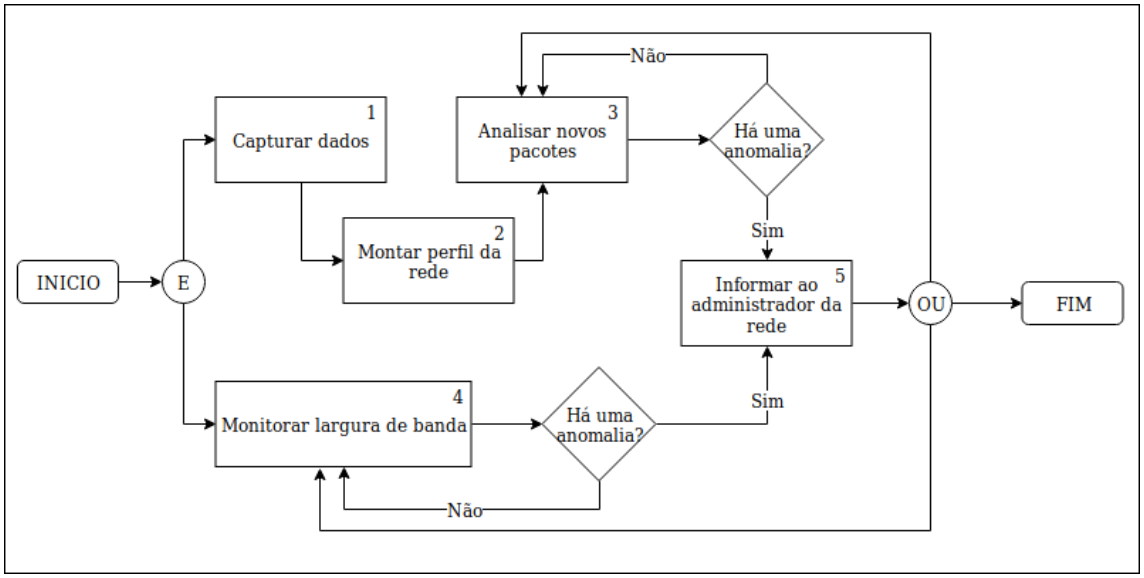


**Figura 1. Fluxograma do processo de monitoramento de fluxo de rede.**

- Endereço de IP de destino;
- Protocolo da camada de transporte;
- Porta de origem;
- Porta de destino;
- Tamanho do pacote.

Ao ser treinado, o classificador utiliza uma técnica de clusterização para agrupar todos os dados. A partir daí, inicia-se a terceira etapa, onde este identifica anomalias nos novos pacotes capturados medindo a distância de Mahalanobis entre seus dados e o *cluster*. A distância é calculada considerando a correlação entre os atributos, então quanto mais semelhante forem mais próximo do *cluster* estará. Caso essa distância não seja suficientemente pequena, o pacote será considerado como uma anomalia.

Segundo [Li et al. 2018], a distância de Mahalanobis é o método mais representativo entre os métodos de aprendizado de medida de distância (do inglês, *Distance Metric Learning* - DML), cuja ideia é computar a similaridade entre características de um problema de classificação específico para ajudar a melhorar a acurácia da classificação.

O fluxo secundário, e quarta etapa, consiste em uma *thread* de monitoramento que analisa constantemente o fluxo de dados e, após a fase de treino, calcula a média e desvio padrão do tamanho dos pacotes capturados. Em seguida, passa a verificar, em intervalos de tempo fixos, a taxa dos dados (tamanho total dos pacotes por segundo) que passaram pela interface de rede, o quanto esse valor variou desde a última medição e se a variação está dentro da tolerância definida pelo desvio padrão dos dados de treino. Se a tolerância for excedida, é considerado que houve uma anomalia no fluxo.

A quinta e última etapa pode ser chamada tanto pelo fluxo principal como pelo secundário. As duas etapas anteriores consistem em monitorar o fluxo a fim de encontrar uma anomalia, que após identificada deve ser tratada. A função desta etapa é alertar ao sistema, ou diretamente a um administrador caso não haja uma resposta automática programada, que há um problema e que uma providência deve ser tomada. Após executar sua função, a aplicação retorna o fluxo para as etapas de monitoramento ou é encerrada.

### 4.3. Ambiente

Para simular o funcionamento da aplicação no contexto de *smart home* foi utilizada uma rede emulada com Mininet, que é "um sistema para prototipagem rápida de grandes redes nos recursos restritos de um único laptop"[Lantz et al. 2010]. A rede emulada contém apenas um controlador (*root*), que é a própria máquina física, um hospedeiro (*host*), destino para onde será enviada uma série de pacotes, e um *switch*, que liga o controlador ao *host*. A interface entre o nó raiz do Mininet e o *switch* foi monitorada durante toda a execução da aplicação. A topologia da rede está ilustrada na Figura 2.

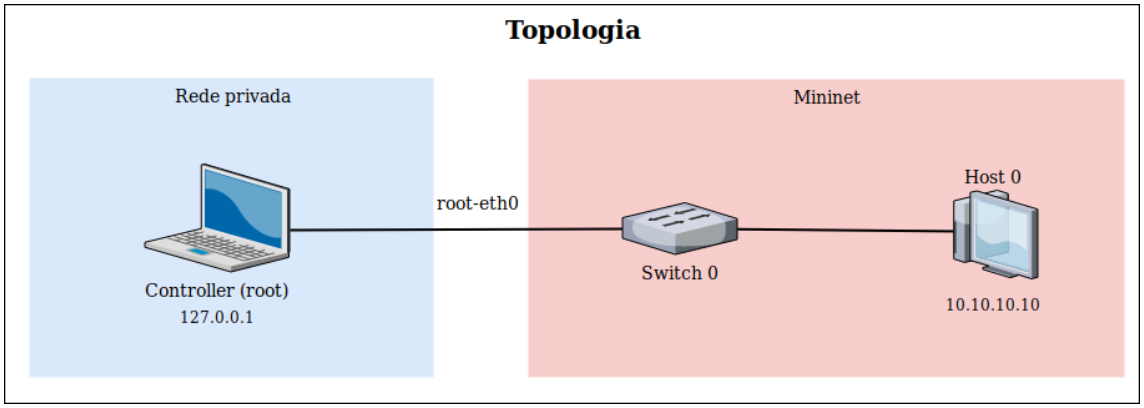


**Figura 2. Rede emulada com Mininet. A interface root-eth0 é monitorada na simulação.**

Para criar os pacotes, foi utilizada uma biblioteca de criação e manipulação de pacotes em Python, a *Scapy* [Biondi 2011]. Essa biblioteca nos permite criar pacotes com quaisquer protocolos e camadas que desejarmos e enviá-los à uma interface de rede. Com isso foi criada uma função que gera pacotes com IP de origem aleatória, seguindo uma distribuição uniforme, e com destino ao *host* criado. Também foram definidos os protocolos TCP (do inglês, *Transmission Control Protocol*) e UDP (do inglês, *User Datagram Protocol*) como possíveis protocolos da camada de transporte. Os pacotes TCP/IP são enviados da porta 20 para a porta 80 e tem tamanho 54 bytes, já os pacotes UDP/IP são enviados da porta 53 para a porta 53 e tem tamanho 42 bytes. O *sniffer* utilizado no *gateway* também faz parte desse pacote.

Ao capturar uma certa quantidade de pacotes, foi feito um pré-processamento de dados e montado o perfil da rede utilizando um pré-processador e um classificador da biblioteca de *machine learning*, também em Python, chamada *Scikit-learn* [Pedregosa et al. 2011].

Os dados capturados anteriormente foram pré-processados e usados para treinar um classificador que detecta anomalias nos dados posteriormente capturados. O classificador utiliza um robusto estimador de covariância, criado por [Rousseeuw 1984], para garantir a resistência à anomalias nos dados de treino e que as distâncias de Mahalanobis associadas reflitam o verdadeiro padrão de uso. Por fim a interface voltou a ser monitorada, agora utilizando o classificador para prever se cada novo pacote faz parte do perfil da rede ou não. Durante todo o processo, o fluxo de dados é constantemente monitorado a fim de detectar também uma alteração anormal na frequência de pacotes.

## 5. Experimentos

Os experimentos foram realizados em um notebook ASUS X555U Series, processador Intel Core i5-6200U CPU @ 2.30GHz x 4, 8 Gb de memória RAM. As métricas avaliadas foram a quantidade de anomalias de pacotes detectadas e a quantidade de erros de variação no fluxo de pacotes.

### 5.1. Experimento 1

O primeiro experimento simulou o comportamento normal, sem intervenções, em uma *smart home* com 20 dispositivos conectados. Durante o experimento cada dispositivo enviou pacotes a uma taxa de 1 pacote por segundo à interface de rede monitorada durante 1 hora.

Os primeiros 15 minutos (25% do tempo total) foram usados para montar o perfil da rede, armazenando os dados capturados para treinar o classificador e calcular média e desvio padrão do fluxo de pacotes. O perfil foi montado com 4140 pacotes capturados e a taxa média do fluxo foi de aproximadamente 220.70 bytes por segundo, com desvio padrão de 46.93 bytes.

Os 45 minutos restantes (75% do tempo total) foram usados na detecção dos dois tipos de anomalias, de pacote e de fluxo. Ao fim do experimento, o classificador identificou 1060 dos 11386 pacotes como anomalia (9.30% do total), e, das 45 vezes em que o fluxo foi checado, apenas 11 fugiram do desvio padrão da amostra (24.44% do total). Os resultados do experimento estão ilustrados na Figura 3.

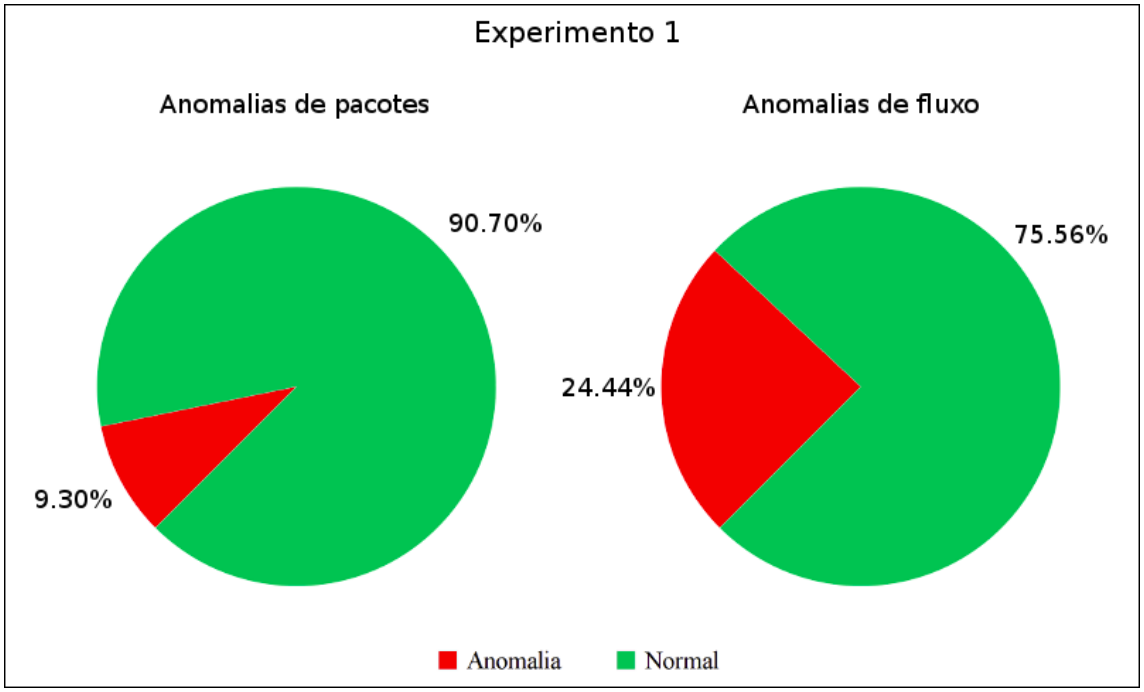


**Figura 3. Resultados do experimento 1 com porcentagens de anomalias detectadas**

### 5.2. Experimento 2

O segundo experimento simulou uma rede *smart home*, onde os dos 20 dispositivos, 2 apresentaram defeito e passam a enviar pacotes repetidos. Assim como no primeiro expe-

rimento, durante a fase de treino cada dispositivo enviou pacotes a uma taxa de 1 pacote por segundo à interface. Na fase de detecção os dispositivos defeituosos enviaram cada pacote 5 vezes. A interface de rede foi monitorada durante 1 hora.

Neste experimento o perfil foi montado com 4041 pacotes capturados e a taxa média do fluxo foi de aproximadamente 215.40 bytes por segundo, com desvio padrão de 46.83 bytes.

Durante a fase de detecção foram identificados 1137 dos 11237 pacotes como anomalia (10.11% do total), e, das 45 vezes em que o fluxo foi checado, 22 fugiram do desvio padrão da amostra (48.88% do total). Os resultados do experimento estão ilustrados na Figura 4.

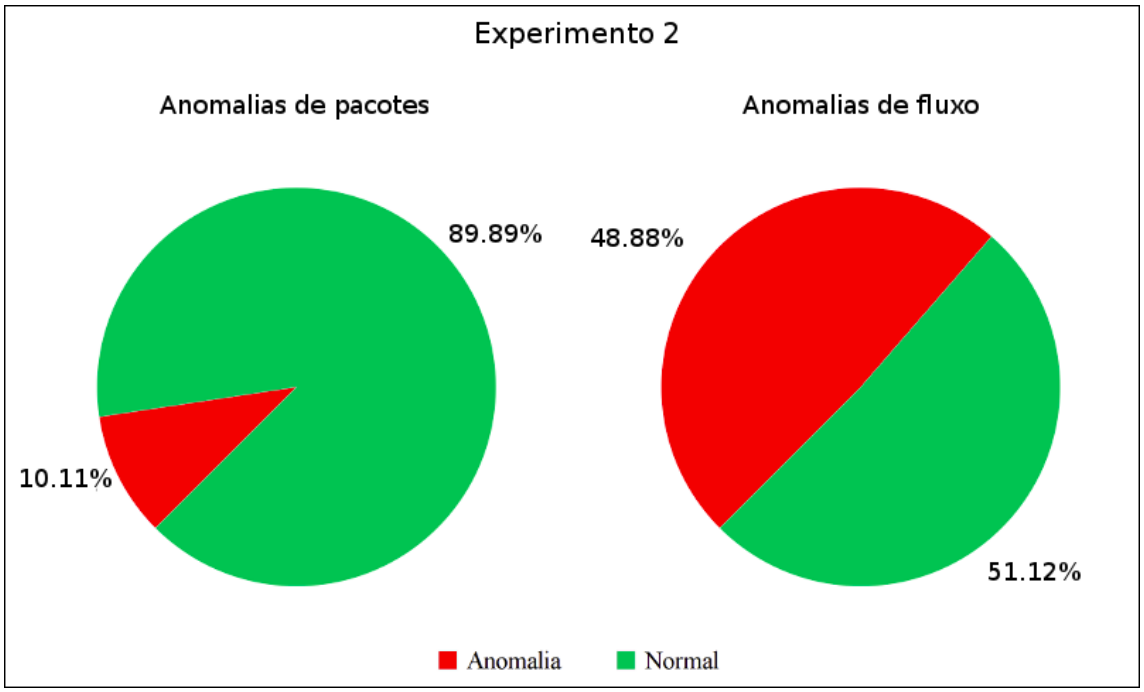


**Figura 4. Resultados do experimento 2 com porcentagens de anomalias detectadas**

## 6. Conclusões

Os experimentos realizados identificaram aproximadamente 10% dos pacotes analisados como anomalias, que é o valor padrão esperado pelo classificador utilizado. Em relação a variação na taxa de dados, o segundo experimento mostra a eficiência do método na identificação da anomalia no caso de um aumento na taxa de dados passantes, como em casos de dispositivos defeituosos enviando pacotes extras.

Como trabalhos futuros, pode-se primeiramente aumentar o número de protocolos reconhecidos. Essa tarefa exigirá um considerável esforço, devido ao número de protocolos existentes e seus diferentes atributos.

Outra melhoria pode ser implementar respostas às anomalias, já que atualmente somente é emitido um alerta no caso de uma ser identificada. Assim, a implementação de uma funcionalidade de correção garantirá uma melhoria significativa no software.

Por fim, em vez de usar dados gerados genericamente seguindo uma distribuição, pode-se realizar experimentos em *datasets* reais e observar como a aplicação se comporta.

## Referências


Ashton, K. et al. (2009). That 'internet of things' thing. *RFID journal*, 22(7):97–114.

Biondi, P. (2011). Scapy. Disponível em: `https://scapy.net/`. Acesso em: 05 jul. 2018.

Goldstein, M. and Uchida, S. (2016). A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. *PLOS ONE*, 11(4):1–31.

Gubbi, J., Buyya, R., Marusic, S., and Palaniswami, M. (2013). Internet of things (iot): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, 29(7):1645 – 1660.

Jose, A. C. and Malekian, R. (2017). Improving smart home security: Integrating logical sensing into smart home. *IEEE Sensors Journal*, 17(13):4269–4286.

Komninos, N., Philippou, E., and Pitsillides, A. (2014). Survey in smart grid and smart home security: Issues, challenges and countermeasures. *IEEE Communications Surveys Tutorials*, 16(4):1933–1954.

Lantz, B., Heller, B., and McKeown, N. (2010). A network in a laptop: Rapid prototyping for software-defined networks. In *Proceedings...*, pages 1–6, Monterey. ACM SIGCOMM Workshop on Hot Topics in Networks, ACM.

Li, L. et al. (2018). A mahalanobis metric learning-based polynomial kernel for classification of hyperspectral images. *Neural Computing and Applications*, 29(4):1103–1113.

Li, S., Xu, L. D., and Zhao, S. (2015). The internet of things: a survey. *Information Systems Frontiers*, 17(2):243–259.

Liu, X. and Nielsen, P. S. (2016). Regression-based online anomaly detection for smart grid data. *CoRR*, abs/1606.05781.

Louridas, P. and Ebert, C. (2016). Machine learning. *IEEE Software*, 33(5):110–115.

Mendez, D. M., Papapanagiotou, I., and Yang, B. (2017). Internet of things: Survey on security and privacy. *CoRR*, abs/1707.01879.

Pedregosa, F. et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Perumal, T. et al. (2008). Interoperability among heterogeneous systems in smart home environment. In *Proceedings...*, pages 177–186, Bali. 2008 IEEE International Conference on Signal Image Technology and Internet Based Systems, IEEE.

Raza, S., Wallgren, L., and Voigt, T. (2013). Svelte: Real-time intrusion detection in the internet of things. *Ad Hoc Networks*, 11(8):2661 – 2674.

Rousseeuw, P. J. (1984). Least median of squares regression. *Journal of the American Statistical Association*, 79(388):871–880.

Satyanarayanan, M. (2017). The emergence of edge computing. *Computer*, 50(1):30–39.