Detecção de Botnets em Dispositivos IoT Utilizando Análise de Consultas DNS com One-class SVM*

Luciano Hanna El Adji[1], **Luís Felipe Ignácio Cunha**[1], **Raphael Carlos Santos Machado**[1,2]

[1] Instituto de Computação – Universidade Federal Fluminense (UFF)
Niterói – RJ – Brasil

[2]Clavis Segurança da Informação
Rio de Janeiro - RJ - Brasil

lucianohanna@id.uff.br, lfignacio@ic.uff.br, raphaelmachado@ic.uff.br

Abstract. *The growing proliferation of IoT devices has expanded the attack surface for botnets, which use Domain Generation Algorithms (DGA) for evasive communication, posing a challenge for detection. We propose an approach that analyzes the lexical characteristics of domains in DNS queries with One-class SVM for detecting malicious activity in IoT devices, allowing for monitoring at network concentration points without direct access to the devices. The results demonstrated high effectiveness, with an accuracy of 99.42%, precision of 99.99%, recall of 99.42%, and a false positive rate of 5.45%; identifying 17 of the 25 tested DGA families with 100% recall.*

Resumo. *A crescente proliferação de dispositivos IoT expandiu a superfície de ataque para botnets, que se utilizam de Algoritmos de Geração de Domínios (Domain Generation Algorithm - DGA) para uma comunicação evasiva, representando um desafio para a detecção. Propomos uma abordagem que analisa as características léxicas de domínios em consultas DNS com One-class SVM para detecção de atividade maliciosa em dispositivos IoT, permitindo o monitoramento em pontos de concentração da rede sem acesso direto aos dispositivos. Os resultados demonstraram alta eficácia, com acurácia de 99,42%, precisão de 99,99%, recall de 99,42% e taxa de falsos positivos de 5,45%; identificando com 100% de recall 17 das 25 famílias de DGA testadas.*

1. Introdução

A Internet das Coisas (Internet of Things - IoT) tem apresentado um crescimento expressivo nos últimos anos, com bilhões de dispositivos conectados à internet realizando funções diversas [Whitmore et al. 2015], desde automação residencial e monitoramento de saúde até aplicações industriais e de infraestrutura urbana. Esta expansão, contudo, traz consigo desafios significativos de segurança, uma vez que muitos dispositivos IoT são desenvolvidos com recursos computacionais limitados — geralmente por serem projetados com microcontroladores simples e foco em baixo custo, tamanho reduzido e economia de energia — e frequentemente apresentam vulnerabilidades devido a restrições de design ou falta de atualizações de segurança.

*Este trabalho foi financiado pelo CNPq, FAPERJ e Clavis Segurança da Informação - Projeto FINEP: PlatCiber: Plataforma Integrada de Inteligência para Monitoramento de Ameaças, Registro de Eventos e Resposta a Incidentes nas áreas de Segurança e Defesa Cibernética.

Neste cenário, as *botnets* – redes de dispositivos comprometidos controlados remotamente – têm se tornado uma ameaça crescente ao ecossistema IoT. Eventos como o ataque do *Mirai* em 2016 [Antonakakis et al. 2017], que afetou milhares de dispositivos IoT e causou indisponibilidade em serviços críticos da internet, demonstram a seriedade do problema. As *botnets* IoT podem ser utilizadas para diversos fins maliciosos, incluindo ataques de negação de serviço distribuído (Distributed Denial of Service - DDoS), mineração de criptomoedas, roubo de dados e propagação de malware [Angrishi 2017].

A detecção de *botnets* em ambientes IoT apresenta desafios particulares. Diferentemente de sistemas convencionais, os dispositivos IoT geralmente possuem capacidade limitada para executar soluções de segurança. Além disso, muitos destes dispositivos operam com *firmware* proprietário, dificultando a implementação de agentes de segurança diretamente nos equipamentos. Adicionalmente, dispositivos comprometidos frequentemente mantêm sua funcionalidade aparentemente normal, tornando a detecção ainda mais complexa.

Neste trabalho, propomos uma abordagem para detecção de *botnets* em dispositivos IoT baseada na análise de consultas DNS. O Sistema de Nomes de Domínio (Domain Name System - DNS) é um componente fundamental da infraestrutura da internet, traduzindo nomes de domínio em endereços IP. As comunicações entre dispositivos IoT e servidores de comando e controle (Command and Control - C&C) de *botnets* frequentemente dependem de resoluções DNS, criando padrões que podem ser identificados através de análise apropriada.

Um método comum utilizado por *botnets* para dificultar o bloqueio de suas comunicações C&C é o uso de Algoritmos de Geração de Domínios (Domain Generation Algorithms - DGA) [Plohmann et al. 2016]. Estes algoritmos são implementados tanto no código malicioso que infecta os dispositivos quanto nos sistemas controlados pelos operadores da *botnet*. Utilizando sementes compartilhadas, como data atual ou outras informações públicas disponíveis, o mesmo conjunto de domínios é gerado deterministicamente em ambos os lados. O dispositivo infectado tenta se conectar sequencialmente aos domínios gerados até encontrar um que foi efetivamente registrado pelo operador da *botnet*. Esta sincronização permite que os atacantes alternem facilmente entre diferentes domínios C&C sem necessidade de atualizar o código malicioso nos dispositivos comprometidos, tornando as técnicas de bloqueio baseadas em listas estáticas de domínios maliciosos ineficazes. A análise de padrões de *queries* DNS pode ser particularmente eficaz na detecção deste comportamento, uma vez que dispositivos infectados por *malware* que utiliza DGA tipicamente apresentam padrões distintivos de múltiplas consultas DNS para domínios que seguem características algorítmicas específicas.

Nossa proposta utiliza técnicas de aprendizado de máquina, especificamente *One-class SVM (Support Vector Machine)*, para modelar o comportamento normal das requisições DNS de dispositivos IoT e identificar desvios que possam indicar infecção por *botnet*. O *One-Class SVM* é um algoritmo de detecção de anomalias que aprende o padrão normal dos dados [Schölkopf et al. 2001]; neste contexto, é útil por detectar *botnets* mesmo sem exemplos prévios de ataques, o que é comum em ambientes IoT.

Diferentemente de métodos que exigem instrumentação direta dos dispositivos, nossa abordagem pode ser implementada em *gateways* de rede ou resolvedores de DNS

locais, permitindo monitoramento centralizado sem interferir na operação dos dispositivos.

Os resultados experimentais obtidos demonstram que a análise de consultas DNS com *One-class SVM* é uma abordagem promissora para detecção de *botnets* em dispositivos IoT. O modelo proposto alcançou acurácia de 99,42%, precisão de 99,99%, *recall* de 99,42%, *F1-score* de 99,71% e taxa de falsos positivos de 5,45%. Em particular, a abordagem obteve 100% de *recall* em 17 das 25 famílias de DGA testadas, demonstrando sua eficácia mesmo diante da diversidade de algoritmos de geração de domínios utilizados por diferentes *botnets*.

Ao oferecer uma solução prática, escalável e não intrusiva para a detecção de *botnets*, o estudo contribui significativamente para o fortalecimento da segurança cibernética, em conformidade com as diretrizes da Doutrina Militar de Defesa Cibernética do Brasil [Ministério da Defesa 2023], que reconhece a importância da proteção de infraestruturas críticas e da vigilância contínua sobre ameaças emergentes no ciberespaço.

2. Trabalhos Relacionados

A detecção de *botnets* em dispositivos IoT tem sido objeto de pesquisa intensiva nos últimos anos, com abordagens que variam desde análise de tráfego de rede até monitoramento de recursos do sistema. Nesta seção, revisamos os trabalhos mais relevantes para nossa proposta, com ênfase em técnicas baseadas em anomalias e no uso de informações DNS para detecção de atividades maliciosas.

Bezerra et al. [Bezerra et al. 2018] propuseram um sistema de detecção de *botnets* baseado em *host* para dispositivos IoT utilizando classificação *One-class SVM*. Seu método analisa o uso de recursos como CPU, memória, temperatura da CPU e diferença de potencial elétrico para identificar comportamentos anômalos. Embora eficaz, esta abordagem requer monitoramento direto dos dispositivos, o que pode não ser viável em muitos cenários devido a limitações de acesso. Nossa proposta difere ao focar na análise de consultas DNS, permitindo detecção em pontos de concentração da rede sem necessidade de acesso direto aos dispositivos.

Meidan et al. [Meidan et al. 2018] desenvolveram o *N-BaIoT*, um sistema de detecção de intrusão baseado em rede que utiliza *autoencoders* profundos para modelar o comportamento legítimo de dispositivos IoT a partir do tráfego de rede. Seus experimentos com *botnets Mirai* e *BashLite* mostraram excelentes resultados, porém o uso de *deep learning* impõe requisitos computacionais significativos, potencialmente inviáveis para *gateways* ou dispositivos com recursos limitados. Nossa proposta difere ao focar exclusivamente na análise de tráfego DNS, o que representa um subconjunto significativamente menor de dados a serem processados, e utiliza SVM de classe única, que apresenta menor complexidade computacional para treinamento e inferência.

Drichel et al. [Drichel et al. 2024] realizaram um estudo abrangente sobre a robustez de classificadores de DGA. Entretanto, sua abordagem difere fundamentalmente da nossa em seu escopo e objetivo: enquanto eles focam na classificação binária universal entre domínios legítimos e maliciosos independentemente da origem, nosso trabalho propõe uma detecção contextualizada que aprende o comportamento normal de consultas DNS específicas de dispositivos IoT. Nossa proposta classifica como anômala qualquer consulta que desvie do padrão estabelecido para um dispositivo específico, mesmo

que o domínio seja globalmente legítimo. Esta especificidade contextual é particularmente valiosa em ambientes IoT, onde os dispositivos normalmente apresentam padrões de comunicação restritos e previsíveis.

Hooshmand et al. [Hooshmand et al. 2024] propuseram um sistema de detecção de anomalias em redes utilizando *ensemble learning* com *XGBoost* e *explainable AI*, processando características gerais do tráfego de rede nos *datasets* NSL-KDD e UNSW-NB15. Seu modelo obteve 99,01% de acurácia e 99,01% de *recall* para UNSW-NB15; e 99,37% de acurácia e 99,37% de *recall* para NSL-KDD na classificação binária, resultados comparáveis aos 99,42% de acurácia e 99,42% de *recall* que nossa abordagem alcançou. No entanto, nossa proposta se destaca por analisar apenas consultas DNS, um subconjunto específico e significativamente menor do tráfego de rede, o que demanda menos recursos computacionais e viabiliza a implementação mesmo em ambientes com recursos limitados, característica essencial para o monitoramento de dispositivos IoT.

3. Abordagem

Nossa abordagem para detecção de *botnets* em dispositivos IoT baseia-se na análise de consultas DNS utilizando técnicas de aprendizado de máquina para detecção de anomalias. Como ilustrado na Figura 1, o método proposto pode ser implementado em pontos de concentração da rede, como *gateways* ou resolvedores de DNS local, sem necessidade de instrumentação nos dispositivos IoT, o que é particularmente vantajoso considerando as limitações de recursos destes equipamentos.

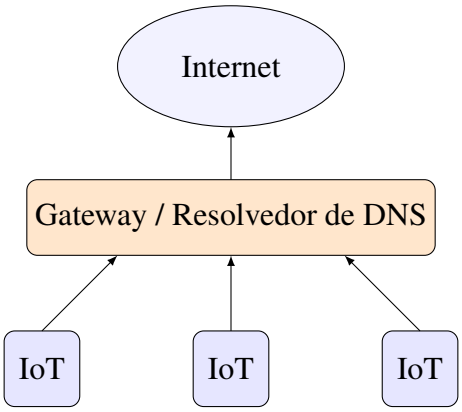

Figura 1. Diagrama conceitual ilustrando a implementação do sistema em pontos de concentração de rede (em laranja), sem instrumentação dos dispositivos IoT.

A arquitetura do sistema proposto, cujos componentes serão detalhados nas seções a seguir, consiste em três etapas principais:

1. Coleta e pré-processamento de dados DNS
2. Treinamento do modelo *One-class SVM* com dados benignos
3. Avaliação do modelo utilizando dados benignos e malignos

3.1. Extração de Características

Para cada consulta DNS, extraímos um conjunto de características léxicas do domínio que permitem diferenciar entre consultas normais e anômalas, sendo estas últimas frequentemente associadas a comunicações de *botnets*, particularmente aquelas que empregam

algoritmos de geração de domínios (DGAs). Nossa implementação permite configurar quais características são utilizadas, possibilitando a avaliação de diferentes combinações para otimizar a eficácia do modelo.

As características léxicas utilizadas neste trabalho são um subconjunto das características usadas no sistema *FANCI* [Schüppen et al. 2018]. É importante ressaltar que existem famílias de DGAs, como *Matsnu* [Skuratovich 2015], que são baseadas em dicionário e utilizam listas de palavras com o objetivo de imitar características de domínios legítimos, o que justifica nossa abordagem baseada em múltiplas características combinadas. As características implementadas incluem:

- **Comprimento do nome de domínio**: Domínios gerados por DGAs podem possuir comprimentos distintos dos domínios legítimos.
- **Proporção de caracteres alfanuméricos**: Representa a fração de caracteres que são letras ou números. Domínios legítimos tendem a ter proporções diferentes de caracteres especiais quando comparados a domínios gerados por DGA.
- **Proporção de vogais**: Razão entre o número de vogais e o comprimento total do domínio. Domínios legítimos geralmente seguem padrões fonéticos da linguagem natural, resultando em proporções de vogais diferentes dos domínios gerados algoritmicamente.
- **Características sobre dígitos**
  - **Proporção de dígitos**: Razão entre o número de caracteres numéricos e o comprimento total do domínio.
  - **Presença de dígitos**: Booleano indicando se possuí dígito no domínio
- **Quantidade de subdomínios**: Número de níveis no nome de domínio.
- **Entropia do nome de domínio**: Mede o grau de aleatoriedade na distribuição de caracteres, calculada através da fórmula de entropia de Shannon. Domínios legítimos como "google.com" tipicamente apresentam baixa entropia devido à sua natureza previsível e mnemônica, enquanto domínios gerados algoritmicamente como "raorbyjptkjwaivax.su" exibem alta entropia devido à distribuição mais aleatória de caracteres.
- **Características baseadas em n-gramas**: Extraímos estatísticas sobre subcadeias de caracteres de tamanho n (no nosso caso, $n = 2$). Durante o treinamento, identificamos os 75 n-gramas mais comuns em domínios benignos; este valor foi determinado empiricamente durante testes preliminares. Testamos valores acima de 75 (100, 125, 150) e não observamos melhoria significativa nos resultados, o que estabeleceu 75 como um bom ponto de equilíbrio entre complexidade e desempenho para novos domínios, calculamos:
  - **Proporção de n-gramas raros**: Fração de n-gramas no domínio que não aparecem entre os 75 mais frequentes em domínios benignos.
  - **Quantidade de n-gramas raros**: Número absoluto de n-gramas raros no domínio.
  - **Frequência média de n-gramas**: Média aritmética das frequências absolutas com que cada n-grama do domínio apareceu no conjunto de treinamento. Um valor baixo indica que o domínio é composto principalmente de padrões de caracteres que raramente aparecem em tráfego benigno.
  - **Pontuação de raridade**: Métrica derivada que normaliza a frequência média em relação à frequência máxima observada no conjunto de treinamento, e então calcula seu complemento (1 - frequência_normalizada).

Valores próximos a 1 indicam domínios com padrões de caracteres extremamente raros, enquanto valores próximos a 0 representam domínios com padrões comuns em tráfego legítimo.

Realizamos três experimentos com diferentes configurações de características para identificar o conjunto ideal. O primeiro experimento utilizou um subconjunto mais básico, focando nas características de comprimento, proporção alfanumérica, entropia e n-gramas; o segundo experimento contém o subconjunto do primeiro experimento adicionando as características de proporção de vogais, características de dígitos e quantidade de subdomínios e, o terceiro experimento utilizou apenas as duas características que obtiveram o melhor desempenho isoladamente.

Optamos por trabalhar exclusivamente com características léxicas dos domínios devido a uma limitação do conjunto de teste, que complementa o tráfego real com domínios do *dataset* 25-DGA. Visto que este dataset consiste apenas em uma lista de nomes de domínio, ele não possui informações temporais associadas. Por essa razão, a tentativa de simular um comportamento temporal foi descartada, pois poderia introduzir padrões artificiais criados pelo próprio ambiente experimental. Dessa forma, a decisão de focar nas características léxicas garante que os resultados não sejam enviesados e reflitam uma avaliação mais fiel da capacidade do modelo em generalizar para cenários de infecção reais.

3.2. Abordagem de Aprendizado de Máquina

Utilizamos o algoritmo *One-class SVM* para detecção de anomalias, considerando que em cenários reais temos acesso principalmente a dados benignos, enquanto comportamentos maliciosos são raros e existem diversas variações de ataques. O *One-class SVM* aprende a fronteira que engloba o comportamento normal, classificando como anomalias os padrões que desviam significativamente deste comportamento.

O processo de treinamento e avaliação seguiu uma abordagem de divisão dos dados baseada na ordenação cronológica:

1. Os dados são inicialmente ordenados cronologicamente para manter a sequência temporal das consultas DNS.
2. Após a ordenação, 80% dos primeiros registros são utilizados para treinamento. Este conjunto contém apenas tráfego normal, representando o comportamento legítimo dos dispositivos IoT.
3. Os 20% restantes dos dados são reservados para teste. Para avaliação da capacidade de detecção de anomalias, este conjunto foi complementado com consultas DNS para domínios gerados por DGAs, realizando testes para cada uma das 25 famílias de DGA presente no *dataset*. Esta abordagem permite avaliar isoladamente o desempenho do modelo para cada família de DGA específica, identificando possíveis variações nos resultados entre diferentes tipos de DGA e garantindo resultados estatisticamente significativos para uma análise comparativa robusta.

O sistema foi desenvolvido em *Python* 3.9 [Python Core Team 2001], utilizando um conjunto de bibliotecas especializadas: *scikit-learn* (versão 1.3.0) [scikit-learn Developers 2007] para a implementação do algoritmo *One-class SVM*;

NumPy (1.24.3) [NumPy Developers 2005] para manipulação eficiente de arrays e operações matemáticas; *Pandas* (2.0.3) [The Pandas Development Team 2008] para processamento e análise de dados estruturados; *Joblib* (1.3.1) [Varoquaux 2008] para persistência de modelos treinados; além de *pytest* (8.3.5) [Krekel et al. 2004] e *pytest-benchmark* (5.1.0) [Măriești 2014] para testes automatizados e avaliação de desempenho. Todo o ambiente de execução foi containerizado usando *Docker*, garantindo reprodutibilidade e facilitando a implantação.

3.3. Configuração dos Experimentos

Para avaliar a eficácia da abordagem proposta, realizamos uma série de experimentos com o objetivo de avaliar o desempenho de diferentes configurações de características e compreender a contribuição de cada característica para o processo de detecção. A seguir são apresentados três experimentos realizados com diferentes configurações de características e, após, sete experimentos para cada característica isolada avaliada.

3.3.1. Experimentos com Múltiplas Características

Realizamos três experimentos para avaliar sistematicamente o desempenho de diferentes combinações de características na detecção de DGAs. Esta estratégia experimental permite uma avaliação progressiva do impacto das diferentes características, partindo de uma linha de base, testando expansões do conjunto de características e, finalmente, explorando uma abordagem minimalista focada nas características mais eficazes:

- **Experimento 1**: O objetivo deste experimento foi estabelecer uma linha de base utilizando um conjunto básico de características lexicais simples. Este conjunto inclui comprimento do domínio, proporção alfanumérica, entropia e características baseadas em n-gramas.
- **Experimento 2**: O objetivo deste experimento foi verificar o impacto da adição de novas características em relação a linha de base. Para isso, expandimos o conjunto do Experimento 1 adicionando características de proporção de vogais, características de dígitos e quantidade de subdomínios.
- **Experimento 3**: Baseado nos resultados da análise de características individuais discutida abaixo, este experimento avaliou o modelo utilizando apenas as duas características que demonstraram melhor desempenho isoladamente: proporção de caracteres alfanuméricos e características baseadas em n-gramas. Esta abordagem visa verificar se um conjunto reduzido de características pode alcançar desempenho comparável ou superior aos conjuntos mais amplos, oferecendo maior eficiência computacional.

3.3.2. Experimentos com Características Individuais

Para compreender o impacto específico de cada característica, foram conduzidos experimentos utilizando-as de forma isolada. O objetivo desta abordagem foi analisar como cada característica léxica contribui individualmente para o processo de detecção, permitindo identificar quais características são mais eficazes contra diferentes famílias de DGA. As características avaliadas foram:

- Proporção de caracteres alfanuméricos
- Características baseadas em n-gramas
- Quantidade de subdomínios
- Comprimento do nome do domínio
- Entropia do nome do domínio
- Proporção de vogais
- Características de dígitos

3.4. Ambiente Experimental

Para validar nossa abordagem em um cenário realista, implementamos um ambiente experimental utilizando um Raspberry Pi 4 Model B [Raspberry Pi Trading Ltd. 2019] executando *Home Assistant Operating System* [Home Assistant 2018] como plataforma de automação IoT, e uma máquina virtual executando o resolvedor de DNS *dnsmasq* [Kelley 2001] configurado para registrar todas as consultas DNS.

O conjunto de dados utilizado neste trabalho consiste em logs deste resolvedor de DNS coletados durante o período de 11 de março a 7 de abril de 2025. Estes logs registraram um total de 492.243 consultas DNS originadas do dispositivo IoT em condições normais de operação, tendo períodos de maior atividade com picos de até 10,394 consultas por segundo em janelas de 5 minutos, enquanto momentos de baixa atividade apresentaram apenas 0,003 consultas por segundo.

Adicionalmente, para fins de validação e *benchmark*, utilizamos o *dataset* 25-DGA [Cucchiarelli et al. 2021] contendo 13.500 domínios gerados por 25 famílias de DGA diferentes, frequentemente utilizados por *botnets* para comunicação C&C, resultando em 337.500 domínios maliciosos utilizados no conjunto de teste.

Este *dataset* foi selecionado por sua abrangência, pois engloba famílias de DGA de *malwares* significativos, com diferentes propósitos, períodos de atividade e mecanismos técnicos. Entre os exemplos estão: o ***Conficker*** [Barabosch et al. 2012], que no final dos anos 2000 utilizava um DGA baseado em tempo para gerar milhares de domínios com aparência aleatória diariamente; o ***Matsnu*** [Skuratovich 2015], ativo em meados da década de 2010 e que utilizava um DGA baseado em dicionário para mesclar palavras e formar domínios plausíveis; e o trojan bancário ***Emotet*** [Shulmin 2015, Global Research & Analysis Team 2023], uma das ameaças mais resilientes da história recente. Surgida em 2014 como um trojan bancário, ela evoluiu para uma infraestrutura massiva de distribuição de *malware* que, apesar de ter sido desmantelada em uma grande operação policial em 2021, ressurgiu posteriormente, demonstrando a persistência de suas operações. Essa diversidade técnica, histórica e funcional reforça a importância do *dataset* para avaliar a capacidade da abordagem em detectar tanto ameaças clássicas quanto as mais atuais.

É importante ressaltar que, embora muitas dessas famílias de DGA tenham se originado em *malwares* que visavam computadores tradicionais, sua utilização neste estudo é fundamentalmente relevante para o contexto de IoT. A detecção proposta baseia-se na análise de características léxicas dos nomes de domínio, uma abordagem agnóstica à plataforma de origem da consulta DNS. A técnica de DGA é um método de comunicação resiliente que pode ser, e frequentemente é, adaptado e reutilizado por atores maliciosos em diferentes tipos de *malware*, incluindo aqueles que visam o ecossistema IoT. Portanto,

validar o modelo contra um *dataset* diversificado com famílias de DGA conhecidas e eficazes serve como um *benchmark* robusto para aferir a capacidade do método em identificar a estrutura algorítmica desses domínios, independentemente do dispositivo infectado.

Os experimentos foram conduzidos em três fases:

1. Coleta de tráfego normal durante o período especificado
2. Treinamento e otimização do modelo *One-class SVM*
3. Avaliação com tráfego benigno e simulações de tráfego malicioso

Para avaliar o desempenho computacional da nossa abordagem, utilizamos as bibliotecas *pytest* [Krekel et al. 2004] em conjunto com *pytest-benchmark* [Măriești 2014]. Realizamos medições executando cada teste 1.000 vezes para garantir resultados estatisticamente significativos. Foram analisadas duas métricas principais: o tempo de inferência, obtido pelo tempo médio de predição de uma única consulta DNS, e o *throughput*, que representa a quantidade de domínios que o sistema consegue analisar por segundo (domínios/s), foi calculado com base no processamento de lotes de 100 domínios por execução. Os testes de desempenho foram realizados no Raspberry Pi 4B, com o objetivo de avaliar a viabilidade de implementação em ambientes com recursos limitados.

Esta metodologia nos permite avaliar tanto a eficácia do modelo na detecção de atividades maliciosas quanto seu desempenho computacional em condições reais. A análise combinada de métricas de detecção (acurácia, precisão, *recall*, *F1-Score* e taxa de falsos positivos) com métricas de performance (tempo de inferência, *throughput*) fornece uma visão abrangente sobre a viabilidade prática da solução proposta, especialmente considerando as restrições de recursos típicas em ambientes IoT.

4. Resultados

Nesta seção, são apresentados e discutidos os resultados dos experimentos descritos na metodologia. A análise avalia a abordagem proposta sob duas óticas principais: a eficácia na detecção de domínios maliciosos, baseada em métricas como acurácia e *recall*, e o desempenho computacional, essencial para a viabilidade em ambientes com recursos limitados. Os resultados são apresentados de forma comparativa entre as diferentes configurações de características testadas, permitindo identificar o conjunto mais equilibrado e eficaz.

4.1. Métricas de Avaliação

Para uma avaliação completa do modelo, foram empregados dois conjuntos de métricas que se complementam. O primeiro conjunto avalia a eficácia da detecção, medindo a capacidade do modelo de distinguir corretamente entre tráfego benigno e malicioso. O segundo avalia o desempenho computacional, quantificando a eficiência e a viabilidade da implementação da solução em dispositivos com recursos restritos. A combinação dessas métricas oferece uma visão holística da aplicabilidade prática da abordagem.

Para avaliar a eficácia da detecção, utilizamos as seguintes métricas:

- **Acurácia**: Proporção de classificações corretas (verdadeiros positivos + verdadeiros negativos) em relação ao total de amostras.
- **Precisão**: Proporção de detecções corretas de anomalias (verdadeiros positivos) em relação ao total de anomalias detectadas.

- ***Recall***: Proporção de anomalias reais que foram corretamente detectadas.
- ***F1-Score***: Média harmônica entre precisão e *recall*, fornecendo uma métrica balanceada para casos de distribuição desigual de classes.
- **Taxa de Falsos Positivos (*False Positive Rate - FPR*)**: Proporção de amostras benignas incorretamente classificadas como anomalias.

Para avaliar o desempenho computacional, utilizamos as seguintes métricas:

- **Tempo de Inferência**: Tempo médio necessário para classificar uma única consulta DNS, medido em milissegundos (ms).
- ***Throughput***: Quantidade de domínios que o sistema consegue analisar por segundo (domínios/s), medido com base no processamento de lotes de 100 domínios.

4.2. Análise de Características Individuais

Para compreender a importância relativa de cada característica no desempenho do modelo, realizamos experimentos utilizando cada uma das características isoladamente. Esta análise permite identificar quais características possuem maior impacto para a detecção de domínio maligno.

Tabela 1. Desempenho de características individuais

Característica	**Acurácia (%)**	**Precisão (%)**	***Recall* (%)**	***F1-Score* (%)**	**Falsos Positivos (%)**	**Tempo de Inferência (ms)**	***Throughput* (domínio/s)**
Proporção de caracteres alfanuméricos	$\mathbf{93,27}$	$99,99$	$93,27$	$96,52$	$4,73$	$\mathbf{6,7 \pm 0,2}$	$\mathbf{11.226 \pm 134}$
Características baseadas em n-gramas	$\mathbf{92,46}$	$99,99$	$92,46$	$96,08$	$7,27$	$\mathbf{8,0 \pm 0,1}$	$\mathbf{6.401 \pm 73}$
Quantidade de subdomínios	$\mathbf{87,63}$	$99,99$	$87,62$	$93,40$	$\mathbf{3,27}$	$6,9 \pm 0,1$	12.172 ± 85
Comprimento do nome do domínio	$76,17$	$99,99$	$76,15$	$86,46$	$5,82$	$6,8 \pm 0,1$	12.254 ± 178
Entropia do nome do domínio	$57,71$	$99,98$	$57,68$	$73,16$	$12,00$	$6,8 \pm 0,2$	9.328 ± 124
Proporção de vogais	$63,96$	$99,93$	$63,98$	$78,01$	$58,91$	$\mathbf{6,7 \pm 0,1}$	$\mathbf{10.761 \pm 109}$
Proporção de dígitos	$13,18$	$99,63$	$13,16$	$23,25$	$60,36$	$7,6 \pm 0,2$	10.002 ± 103

Analisando o desempenho de cada característica isoladamente, conforme detalhado na Tabela 1, os resultados indicam que a proporção de caracteres alfanuméricos

apresenta o melhor desempenho geral, com 93,27% de acurácia. Este resultado é notável considerando que se trata de uma métrica de relativamente baixo custo computacional.

A característica baseada em n-gramas obteve acurácia similar (92,46%), indicando que ambas capturam informações significativas para a discriminação entre domínios legítimos e maliciosos.

Um resultado relevante foi o desempenho da quantidade de subdomínios, que alcançou 87,63% de acurácia, apresentando também a menor taxa de falsos positivos (3,27%).

Em termos de desempenho computacional, observamos variações significativas entre as características. A característica baseada em n-gramas apresentou o maior tempo de inferência médio (8,0 ms) e o menor *throughput* médio (6.400 domínios/s), o que é esperado considerando a complexidade computacional envolvida no processamento dessas características. Em contrapartida, características mais simples como proporção de vogais e proporção de caracteres alfanuméricos apresentaram desempenho superior, com tempos de inferência de aproximadamente 6,7 ms e *throughput* acima de 10.000 domínios/s.

Tabela 2. *Recall* (%) por família de DGA usando características individuais

Família DGA	prop. alfanum.	n-gramas	subdomínios	comprimento	entropia	prop. vogais	prop. dígitos
corebot	**100,00**	99,96	**0,00**	66,46	89,82	71,36	99,31
conficker	**51,70**	89,59	64,21	**100,00**	85,03	71,14	0,00
cryptolocker	92,79	92,62	85,71	89,35	36,57	61,23	0,00
dircrypt	100,00	98,93	100,00	76,52	47,09	67,78	0,00
emotet	100,00	95,97	100,00	100,00	28,33	76,92	0,00
fobber	100,00	99,99	100,00	98,81	50,01	72,83	0,00
gozi	100,00	78,13	100,00	44,55	31,30	72,49	0,00
kraken	**65,75**	**96,35**	49,99	86,80	64,70	70,44	0,00
matsnu	98,18	98,28	100,00	81,23	37,23	56,58	0,00
murofet	100,00	99,14	100,00	75,87	74,53	68,78	33,16
necurs	100,00	98,94	100,00	80,53	53,02	68,50	0,00
nymaim	**69,37**	**66,83**	**100,00**	80,56	45,77	53,21	0,00
padcrypt	100,00	87,60	93,84	37,68	83,85	71,27	0,00
pushdo	100,00	100,00	100,00	100,00	94,85	49,53	0,00
pykspa	100,00	90,58	100,00	100,00	61,19	68,13	0,00
qadars	100,00	96,22	100,00	100,00	48,17	66,85	**97,82**
ramdo	100,00	94,11	100,00	0,00	51,13	78,00	0,00
ramnit	100,00	98,68	100,00	75,30	45,24	66,41	0,00
ranbyus	100,00	99,08	100,00	4,61	37,00	74,13	0,00
rovnix	100,00	99,92	100,00	60,00	73,59	78,32	**98,63**
simda	94,52	99,99	100,00	100,00	75,41	58,36	0,00
suppobox	100,00	76,41	100,00	82,89	66,65	55,00	0,00
symmi	62,73	58,17	0,00	62,69	34,92	24,21	0,00
tinba	96,69	96,05	96,69	99,97	55,81	73,11	0,07
vawtrak	100,00	99,99	100,00	100,00	70,81	24,87	0,00

A análise do *recall* por família de DGA, apresentada na Tabela 2, revela particularidades que demonstram como certas características são mais eficazes contra tipos específicos de DGA. Por exemplo:

- A família *kraken* possui *recall* de 96,35% usando apenas n-gramas, enquanto a proporção alfanumérica alcança apenas 65,75% de recall.
- A família *nymaim* possui *recall* excelente de 100% pela quantidade de subdomínios, mas obtém resultados medianos com proporção alfanumérica (69,37%) e n-gramas (66,83%).
- Casos como *conficker* são mais eficientemente detectados pelo comprimento do domínio (100%), enquanto a proporção alfanumérica mostra-se ineficaz (51,70%).
- Para *corebot*, a contagem de subdomínios é completamente ineficaz (0%), mas a proporção alfanumérica alcança 100% de *recall*.
- Famílias que utilizam dígitos em seus domínios, como *qadars* e *rovnix*, obtêm bons resultados com a proporção de dígitos, característica que é ineficaz para a maioria das outras famílias.

Esta variabilidade no comportamento das famílias de DGA evidencia a necessidade de uma abordagem de múltiplas características para a detecção eficaz. Na seção seguinte, analisaremos como a combinação de características proporciona uma cobertura mais abrangente contra as diversas técnicas de geração de domínios.

4.3. Desempenho Geral do Modelo

Tabela 3. Comparação de desempenho entre os experimentos

Métrica	**Experimento 1**	**Experimento 2**	**Experimento 3**
Acurácia (%)	$99,41529$	$96,29013$	$95,99349$
Precisão (%)	$99,99553$	$99,99569$	$99,99506$
Recall (%)	$99,41926$	$96,29126$	$95,99496$
F1-Score (%)	$99,70656$	$98,10852$	$97,95419$
Taxa de Falsos Positivos (%)	$5,45455$	$5,09091$	$5,81818$
Tempo de Inferência (ms)	$8,49 \pm 0,15$	$9,56 \pm 0,16$	$8,18 \pm 0,13$
Throughput (domínios/s)	4757 ± 37	3954 ± 24	5.840 ± 45

Conforme detalhado na Tabela 3, os resultados mostram diferenças significativas no desempenho dos três experimentos. O Experimento 1, com o conjunto básico de quatro características, alcançou o melhor desempenho geral com 99,42% de acurácia, 99,42% de *recall* e 99,71% de *F1-score*. O Experimento 3, utilizando apenas um subconjunto das características do Experimento 1, obteve resultados intermediários com 95,99% de acurácia e *recall*, demonstrando que mesmo um subconjunto reduzido das características principais consegue capturar elementos essenciais para a detecção.

O Experimento 2, que adicionou mais características ao conjunto básico do Experimento 1, apresentou o desempenho mais baixo em termos de acurácia (96,29%) e *recall* (96,29%), indicando que a inclusão de características adicionais não resultou em melhoria na capacidade de detecção. No entanto, este experimento obteve a menor taxa de falsos positivos (5,09%), comparado a 5,45% do Experimento 1 e 5,82% do Experimento 3.

Estes resultados sugerem que o Experimento 1 oferece o melhor equilíbrio entre capacidade de detecção e precisão, aprimorando significativamente o desempenho do subconjunto utilizado no Experimento 3. A taxa de falsos positivos ligeiramente menor no Experimento 2 merece consideração em cenários onde falsos positivos precisam ser minimizados, mesmo que às custas de uma redução na capacidade geral de detecção.

A avaliação do desempenho computacional demonstrou que a quantidade de características utilizadas tem impacto direto no tempo de processamento e *throughput*. O modelo com menor número de características (Experimento 3) apresentou melhor desempenho computacional comparado com os outros dois modelos.

Para contextualizar estes resultados em cenários reais, utilizamos como referência o pico de atividade do dispositivo IoT monitorado que realizou 10,394 consultas DNS por segundo em janelas de 5 minutos. Com base nesta métrica, podemos estimar a capacidade de monitoramento de cada experimento: os experimentos 1, 2 e 3 suportam aproximadamente 457, 380 e 561 dispositivos IoT respectivamente. Embora o Experimento 3 ofereça maior capacidade de processamento, o Experimento 1 demonstra ser o mais adequado ao apresentar o melhor equilíbrio entre eficácia de detecção e capacidade operacional, onde a diferença de 104 dispositivos a menos em relação ao Experimento 3 é amplamente compensada pela melhoria significativa na acurácia (99,42% vs 95,99%) e *recall* (99,42% vs 95,99%).

4.4. Desempenho por Família de DGA

A análise de *recall* por família de DGA, apresentada na Tabela 4, mostra que em todos os experimentos, diversas famílias de DGA foram efetivamente identificadas pelos modelos, porém com diferenças significativas de desempenho. No Experimento 1, 17 das 25 famílias obtiveram *recall* de 100%, enquanto no Experimento 2 esse número caiu para apenas 6 famílias. O Experimento 3, utilizando apenas proporção alfanumérica e n-gramas, conseguiu detectar 16 das 25 famílias com 100% de *recall*, demonstrando a eficácia dessa combinação simplificada de características.

A análise dos resultados mostra um padrão interessante: enquanto o Experimento 3 mantém desempenho excelente para a maioria das famílias de DGA, existe degradação significativa para algumas famílias específicas como *nymaim* (78,63%), *conficker* (81,83%) e *symmi* (52,17%). Isto sugere que, para certas famílias de DGA com características particulares, outras características além da proporção alfanumérica e n-gramas são necessárias para detecção eficiente.

O caso da família *symmi* é particularmente notável, pois tanto no Experimento 2 quanto no Experimento 3 seu *recall* é consideravelmente menor em comparação com o Experimento 1. Este comportamento reforça a ideia de que diferentes famílias de DGA podem exigir abordagens personalizadas para uma detecção eficiente.

5. Conclusão

Este trabalho apresentou uma abordagem leve e eficaz para detecção de *botnets* em dispositivos IoT baseada na análise de consultas DNS utilizando o algoritmo *One-class SVM*. A metodologia proposta oferece vantagens significativas para cenários IoT, destacando-se pela implementação em pontos de concentração da rede sem necessidade

Tabela 4. *Recall* (%) por família de DGA

Família DGA	Experimento 1	Experimento 2	Experimento 3
emotet	100,00000	100,00000	100,00000
murofet	100,00000	100,00000	100,00000
qadars	100,00000	100,00000	100,00000
ramdo	100,00000	100,00000	100,00000
ranbyus	100,00000	100,00000	100,00000
rovnix	100,00000	100,00000	100,00000
gozi	100,00000	99,93333	100,00000
fobber	100,00000	99,88889	100,00000
padcrypt	100,00000	99,87407	99,56296
dircrypt	100,00000	99,84444	100,00000
ramnit	100,00000	99,80000	100,00000
necurs	100,00000	99,57778	100,00000
pykspa	100,00000	99,20741	100,00000
simda	100,00000	96,51852	100,00000
pushdo	100,00000	96,22222	100,00000
suppobox	100,00000	94,12593	100,00000
vawtrak	100,00000	92,71111	100,00000
corebot	99,98519	100,00000	99,94815
tinba	99,98519	99,54074	98,71852
cryptolocker	99,96296	98,06667	95,16296
kraken	99,85926	97,42222	95,14815
matsnu	99,62222	99,62222	98,70370
nymaim	97,48889	94,48889	78,62963
symmi	96,45186	54,11111	52,17037
conficker	92,12593	86,32593	81,82963

de instrumentação direta nos dispositivos, aspecto crucial considerando as limitações de recursos destes equipamentos.

Os resultados experimentais evidenciam aspectos relevantes sobre a seleção de características para detecção de domínios DGA. Ao comparar os três experimentos, observamos que o conjunto básico de quatro características (Experimento 1) proporcionou o melhor desempenho geral, superando tanto o subconjunto de duas características mais relevantes (Experimento 3) quanto o conjunto expandido (Experimento 2). Esta observação indica que existe um ponto ótimo na seleção de características, onde um conjunto pode proporcionar melhor capacidade de detecção do que outros com mais ou menos características.

Os resultados obtidos apontam viabilidade de utilização da solução proposta em redes com mais de 400 dispositivos IoT, demonstrando que nossa abordagem não é apenas eficaz na acurácia de detecção, mas também altamente eficiente em termos computacionais. A baixa latência de processamento garante, também, que o monitoramento ocorra em tempo real sem introduzir atraso perceptível a resolução DNS.

Ao analisar o desempenho por família de DGA, foi possível observar que o mo-

delo do Experimento 1 obteve 100% de *recall* em 17 das 25 famílias de DGA, demonstrando a robustez da abordagem mesmo frente à diversidade de algoritmos de geração de domínios. O menor desempenho no Experimento 2 para certas famílias (como *symmi*, cujo *recall* diminuiu de 96,45% para 54,11%) sugere que diferentes famílias de DGA possuem características distintas que podem ser afetadas de maneiras diferentes pela seleção de atributos.

Embora nossa abordagem não exija acesso direto aos dispositivos - diferentemente de métodos como o proposto por Bezerra et al. [Bezerra et al. 2018], que requerem monitoramento de recursos como CPU e memória diretamente nos equipamentos - os resultados alcançados são satisfatórios, demonstrando que a análise exclusiva de consultas DNS pode ser tão eficaz quanto métodos mais intrusivos, com a vantagem adicional da não interferência na operação dos dispositivos. Enquanto o trabalho de Bezerra et al. alcançou acurácia entre 94% e 99%, precisão entre 96% e 97%, e *recall* variando entre 75% a 100% utilizando características do *host*, nossa abordagem atingiu 99,42% de acurácia, 99,996% de precisão e 99,42% de *recall* utilizando apenas características léxicas de consultas DNS, reforçando a viabilidade de métodos não-intrusivos para detecção de *botnets* em ambientes IoT.

Contudo, é importante reconhecer uma limitação inerente a esta abordagem. Embora a solução demonstre robustez para a detecção de comunicação C&C que utiliza DGA, existem malwares que não utilizam esta técnica. Nesses casos, a detecção baseada em consultas DNS é insuficiente, sendo necessário o uso de estratégias complementares, como a análise de padrões de tráfego de rede mais amplos, como as abordagens de Meidan et al. e Hooshmand et al.

Como trabalhos futuros, identificamos algumas direções promissoras. Planejamos uma investigação mais aprofundada sobre o impacto das características, utilizando análises estatísticas como matriz de correlação e VIF para refinar a seleção de atributos e determinar a combinação mínima ideal para cada ambiente IoT. Além disso, pretendemos comparar a abordagem com outras técnicas de aprendizado de máquina, como autoencoders ou modelos híbridos de baixa complexidade, incorporar a análise temporal das consultas DNS para capturar padrões comportamentais, e realizar uma avaliação de desempenho em paralelo com o modelo de Bezerra et al. para uma análise comparativa mais robusta.

Os resultados obtidos demonstram que a análise de consultas DNS com *One-class SVM* representa uma abordagem promissora e viável para a detecção de *botnets* em dispositivos IoT, oferecendo um equilíbrio entre eficácia, simplicidade e não-intrusividade. Esta contribuição é particularmente significativa no contexto atual, onde a segurança de infraestruturas IoT se torna cada vez mais crítica para a sociedade digital e para a defesa cibernética nacional.

Referências

[Angrishi 2017] Angrishi, K. (2017). Turning internet of things (IoT) into internet of vulnerabilities (IoV): IoT botnets. https://arxiv.org/abs/1702.03681.

[Antonakakis et al. 2017] Antonakakis, M., April, T., Bailey, M., Bernhard, M., Bursztein, E., Cochran, J., Durumeric, Z., Halderman, J. A., Invernizzi, L., Kallitsis, M., Kumar,

D., Lever, C., Ma, Z., Mason, J., Menscher, D., Seaman, C., Sullivan, N., Thomas, K., and Zhou, Y. (2017). Understanding the mirai botnet. In *26th USENIX Security Symposium (USENIX Security 17)*, pages 1093–1110, Vancouver, BC. USENIX Association.

[Barabosch et al. 2012] Barabosch, T., Wichmann, A., Leder, F., and Gerhards-Padilla, E. (2012). Automatic extraction of domain name generation algorithms from current malware.

[Bezerra et al. 2018] Bezerra, V. H., da Costa, V. G. T., Junior, S. B., Miani, R. S., and Zarpelão, B. B. (2018). One-class classification to detect botnets in IoT devices. In *Anais do XVIII Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais*, pages 43–56, Porto Alegre, RS, Brasil. SBC.

[Cucchiarelli et al. 2021] Cucchiarelli, A., Morbidoni, C., Spalazzi, L., and Baldi, M. (2021). Algorithmically generated malicious domain names detection based on n-grams features. *Expert Systems with Applications*, 170:114551.

[Drichel et al. 2024] Drichel, A., Meyer, M., and Meyer, U. (2024). Towards robust domain generation algorithm classification. In *Proceedings of the 19th ACM Asia Conference on Computer and Communications Security*, ASIA CCS '24, page 2–18. ACM.

[Global Research & Analysis Team 2023] Global Research & Analysis Team (2023). Emotet, darkgate, lokibot: crimeware report. https://securelist.com/emotet-darkgate-lokibot-crimeware-report/110286/.

[Home Assistant 2018] Home Assistant (2018). Home assistant operating system. https://github.com/home-assistant/operating-system. Acessado em: 08 de abril de 2025.

[Hooshmand et al. 2024] Hooshmand, M. K., Huchaiah, M. D., Alzighaibi, A. R., Hashim, H., Atlam, E.-S., and Gad, I. (2024). Robust network anomaly detection using ensemble learning approach and explainable artificial intelligence (xai). *Alexandria Engineering Journal*, 94:120–130.

[Kelley 2001] Kelley, S. (2001). Dnsmasq. http://www.thekelleys.org.uk/dnsmasq/doc.html. Acessado em: 09 de abril de 2025.

[Krekel et al. 2004] Krekel, H., Oliveira, B., Pfannschmidt, R., Bruynooghe, F., Laugher, B., and Bruhin, F. (2004). pytest 8.3. https://github.com/pytest-dev/pytest.

[Meidan et al. 2018] Meidan, Y., Bohadana, M., Mathov, Y., Mirsky, Y., Breitenbacher, D., Shabtai, A., and Elovici, Y. (2018). N-baiot: Network-based detection of IoT botnet attacks using deep autoencoders. *CoRR*, abs/1805.03409.

[Ministério da Defesa 2023] Ministério da Defesa (2023). Doutrina militar de defesa cibernética. (MD31-M-07). Acessado em: 10 de abril de 2025.

[Mărieș 2014] Mărieș, I. C. (2014). pytest-benchmark. https://github.com/ionelmc/pytest-benchmark.

[NumPy Developers 2005] NumPy Developers (2005). Numpy: The fundamental package for scientific computing with python. https://numpy.org/.

[Plohmann et al. 2016] Plohmann, D., Yakdan, K., Klatt, M., and Bader, J. (2016). A comprehensive measurement study of domain generating malware. In *25th USENIX Security Symposium (USENIX Security 16)*.

[Python Core Team 2001] Python Core Team (2001). *Python: A dynamic, open source programming language*. Python Software Foundation.

[Raspberry Pi Trading Ltd. 2019] Raspberry Pi Trading Ltd. (2019). Raspberry pi 4 model b datasheet. https://datasheets.raspberrypi.org/rpi4/raspberry-pi-4-datasheet.pdf. Acessado em: 14 de abril de 2025.

[Schüppen et al. 2018] Schüppen, S., Teubert, D., Herrmann, P., and Meyer, U. (2018). FANCI : Feature-based automated NXDomain classification and intelligence. In *27th USENIX Security Symposium (USENIX Security 18)*, pages 1165–1181, Baltimore, MD. USENIX Association.

[Schölkopf et al. 2001] Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., and Williamson, R. C. (2001). Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471.

[scikit-learn Developers 2007] scikit-learn Developers (2007). scikit-learn: machine learning in python. https://github.com/scikit-learn/scikit-learn.

[Shulmin 2015] Shulmin, A. (2015). The banking trojan emotet: Detailed analysis. https://securelist.com/the-banking-trojan-emotet-detailed-analysis/69560/.

[Skuratovich 2015] Skuratovich, S. (2015). Matsnu: A deep dive. https://blog.checkpoint.com/wp-content/uploads/2015/07/matsnu-malwareid-technical-brief.pdf.

[The Pandas Development Team 2008] The Pandas Development Team (2008). pandas: powerful python data analysis toolkit. https://pypi.org/project/pandas/.

[Varoquaux 2008] Varoquaux, G. (2008). Joblib. https://pypi.org/project/joblib/.

[Whitmore et al. 2015] Whitmore, A., Agarwal, A., and Da Xu, L. (2015). The internet of things—a survey of topics and trends. *Information Systems Frontiers*, 17(2):261–274.