
UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO


Francisco Queiroga da Silva Neto

# Sistema para detecção de intrusão de botnets utilizando aplicações de machine learning

Recife

2021

Francisco Queiroga da Silva Neto

# Sistema para detecção de intrusão de botnets utilizando aplicações de machine learning

Artigo apresentado ao Curso de Bacharelado em Sistemas de Informação da Universidade Federal Rural de Pernambuco, como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Universidade Federal Rural de Pernambuco – UFRPE

Departamento de Estatística e Informática

Curso de Bacharelado em Sistemas de Informação

Orientador: Rodrigo Elia Assad

Recife

2021

Dados Internacionais de Catalogação na Publicação
Universidade Federal Rural de Pernambuco
Sistema Integrado de Bibliotecas
Gerada automaticamente, mediante os dados fornecidos pelo(a) autor(a)

N469s Neto, Francisco Queiroga da Silva
Sistema para detecção de intrusão de botnets utilizando aplicações de machine learning: Estudo exploratório / Francisco Queiroga da Silva Neto. - 2021.
22 f. : il.

Orientador: Rodrigo Elia Assad.
Inclui referências.

Trabalho de Conclusão de Curso (Graduação) - Universidade Federal Rural de Pernambuco, Bacharelado em Sistemas da Informação, Recife, 2021.

1. intrusion detection. 2. cybersecurity. 3. machine learning. 4. data science. 5. botnet. I. Assad, Rodrigo Elia, orient. II. Título

CDD 004

FRANCISCO QUEIROGA DA SILVA NETO

# SISTEMA PARA DETECÇÃO DE INTRUSÃO DE BOTNETS UTILIZANDO APLICAÇÕES DE MACHINE LEARNING

Artigo apresentado ao Curso de Bacharelado em Sistemas de Informação da Universidade Federal Rural de Pernambuco, como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Aprovada em: 13 de Dezembro de 2021.

BANCA EXAMINADORA

Rodrigo Elia Assad (Orientador)
Departamento de Estatística e Informática
Universidade Federal Rural de Pernambuco

Marcelo Gama
Departamento de Estatística e Informática
Universidade Federal Rural de Pernambuco

# Agradecimentos

Este trabalho foi realizado com o apoio de familiares, amigos e de todos os professores da Coordenação de Estatísticas e Informática (DEINFO) que mantêm o curso de Bacharelado em Sistemas de Informação da Universidade Federal Rural de Pernambuco.

Sou filho de Adriana e Anibal, neto de Dona Branca e Livin, criado por minha bisavó Dilurdes. Sou grato.

# Sistema para detecção de intrusão de botnets utilizando aplicações de machine learning

**Francisco Queiroga da Silva Neto**[1], **Rodrigo Elia Assad**[1]

[1]Departamento de Estatística e Informática - Universidade Federal Rural de Pernambuco (UFRPE)
52171-900 – 81 3320-6001 – Recife – PE – Brasil

franciscofqueiroga@gmail.com, rodrigo.assad@gmail.com

***Abstract.*** *Communication tools and the continuous advancement of the Internet have also resulted in the sophistication of tools and methods to carry out attacks against users and their computers, with features that facilitate criminal activities in the cyber environment. Among cyber threats, botnets have characteristics and advantages that have expanded their use in recent years, becoming a tool employed extensively by attackers to conduct attacks and gain control of various devices connected to computer networks. The way these threats behave and are updated brings several challenges to the intrusion detection area. In this paper, a study is presented on the application of machine learning techniques in detecting botnets by analyzing network traffic flows. The study aims to show how pattern classification techniques can be applied in intrusion detection systems to identify similarities between the infrastructure of botnets, where works in the literature were studied to address an application that aims to improve the problems related to the attribute selection steps and the data processing, crucial steps in machine learning models*

***Resumo.*** *As ferramentas de comunicação e o avanço contínuo da Internet resultaram também na sofisticação de ferramentas e métodos para realizar ataques contra usuários e seus computadores, com recursos que facilitam atividades criminosas no ambiente cibernético. Dentre as ameaças cibernéticas, as botnets possuem características e vantagens que expandiram seu uso nos últimos anos, se tornando uma ferramenta empregada de forma abrangente por atacantes para conduzir ataques e obter o controle de diversos dispositivos conectados a redes de computadores. A forma como estas ameaças se comportam e são atualizadas traz diversos desafios para a área de detecção de intrusão. Nesse documento, é apresentado um estudo sobre a aplicação de técnicas de machine learning na detecção de botnets ao analisar fluxos de tráfego de rede. O estudo visa mostrar como técnicas de classificação de padrões podem ser aplicadas em sistemas de detecção de intrusão para identificar similaridades entre a infraestrutura de botnets, onde foram estudados trabalhos da literatura para abordar uma aplicação que visa melhorar os problemas relacionados às etapas de seleção de atributos e o processamento de dados, etapas cruciais em modelos de machine learning.*

## 1. Introdução

O advento de mecanismos de comunicação cada vez mais sofisticados vem alterando o uso das redes de computadores em ambientes corporativos, domésticos e em redes públicas,

introduzindo novos desafios para a proteção da privacidade, integridade e segurança dos dados dos usuários. Cada vez mais, indivíduos se expõem e passam a utilizar serviços digitais oferecidos por bancos, lojas de *e-commerce* e saúde. Estes serviços passaram a integrar um número vertiginoso de dispositivos inteligentes (IoT) conectados à Internet e, nos últimos anos, vêm se tornando alvos de ataques que abusam de vulnerabilidades que expõem seus usuários a um número periclitante de ameaças cibernéticas que, por sua vez, também estão em constante evolução. [Stephan 2019].

Uma das principais ameaças à segurança de usuários conectados à *Internet* advém do desenvolvimento de *softwares* maliciosos, também conhecidos como *malwares*, programas que implementam atividades ilegais, comprometendo o uso legítimo de um computador e a segurança do usuário [Alenezi et al. 2020]. O desenvolvimento de *malwares* acompanha de forma paralela o crescimento e expansão de serviços na *Internet*, nos quais agentes maliciosos vêm melhorando mecanismos de propagação de suas atividades, tornando as ameaças cada vez mais resilientes a métodos de proteção e detecção.

Dentre as classes de *malwares*, as botnets são ferramentas populares para automatizar a propagação de outros artefatos maliciosos, como *trojans*, *rootkits* e *worms*, e realizar ataques em larga escala que impactam a disponibilidade de serviços [Imam et al. 2014]. Adicionalmente, as *botnets* também são empregadas por agentes maliciosos (indivíduos que procuram violar a privacidade de outros indivíduos e entidades para obter lucros através de atividades criminosas) para realizar varreduras em busca de dispositivos vulneráveis e disseminar *malwares* através de campanhas criminosas para invadir sistemas de organizações visando informações sensíveis. Muitas vezes, as atividades de uma *botnet* estão associados com outras ameaças para roubo de dados (*infostealers*), nos quais os dispositivos infectados pela rede de *bots* se tornam pontos vulneráveis a outros ataques cibernéticos [Feily et al. 2009].

Para os agentes maliciosos, as *botnets* são vantajosas pela capacidade de comunicação construída através do canal de comando e controle (C&C), que centraliza o manuseio da rede e permite que sejam enviadas ações para realizar ataques de maneira remota. Entidades e instituições de segurança da informação têm se preocupado com o avanço dessas redes de *bots* [Spamhaus 2021], atividade que se intensificou durante a pandemia do COVID-19. Dados de telemetria do ano de 2021 de entidades de segurança da informação permitiram observar um crescimento no número servidores C&C e, em média, foram criadas 553 novas *botnets*, somente para o primeiro trimestre de 2021.

A principal forma de infecção das *botnets* ocorre pelo envio de *spams*. No ano de 2018, relatórios de segurança apontavam que 55% dos e-mails enviados pela *Internet* eram *spams*, sendo 1 mensagem maliciosa a cada 412 *e-mails* enviados e 1 a cada 54 domínios de *e-mails* e URLs relacionados a atividade de *botnets* [Symantec 2019]. O Brasil é um alvo bastante popular destas campanhas maliciosas, tendo um aumento de 150% — do último quadrimestre de 2020 para o primeiro quadrimestre de 2021 — no número de redes de *botnets* reconhecidas disseminando *spams* para usuários brasileiros [A10Networks 2020]. O Brasil também é um dos principais países de origem de *bots* em todo o mundo. Somente no segundo trimestre de 2020, existiam mais de 757 mil *bots* em uso no Brasil para realizar ataques de *denial of service* (DoS).

O crescimento das redes *bots* e seus ataques contra usuários no Brasil e no mundo

é preocupante, tornando os ambientes de trabalho remoto, padrão adotado durante a pandemia da COVID-19, muito atrativos para grupos criminosos. Portanto, a pesquisa em questão propõe analisar métodos de detecção de *botnets* baseados através de modelos de *machine learning* (ML). O documento apresenta uma visão geral sobre a estrutura de uma *botnet* e sobre métodos de detecção com objetivo de elaborar um sistema que aplica ML para classificar a atividade de *botnets* através da análise do tráfego de rede de computadores, também estudando como melhorar a eficácia do processo de seleção e filtragem de características úteis dos fluxos de tráfego através de recursos da área de *data mining* (DM).

Este artigo é organizado da seguinte forma. A Seção II apresenta um estudo do estado da arte que mostra pesquisas relacionadas à cibersegurança, DM e ML. A Seção III examina alguns pontos da estrutura de uma *botnet* e seu ciclo de vida a partir de suas implementações mais comuns, e na sequência, na Seção IV, é mostrado o método e os passos seguidos para selecionar os dados utilizados nos experimentos, o processamento dos dados e os modelos de ML que foram aplicados. Os resultados obtidos e os experimentos realizados são apresentados na Seção V. Por fim, a Seção VI apresenta as considerações finais, revisando as descobertas feitas durante a pesquisa e oportunidades para trabalhos futuros na detecção de *botnets* aplicando ML.

## 2. Trabalhos Relacionados

As *botnets* se tornaram ferramentas complexas e cada vez mais sofisticadas, e a aplicação de técnicas de ML para detectar comportamentos anômalos vêm sendo bastante explorada no meio acadêmico em diversos trabalhos que se diferenciam por aspectos como as técnicas empregadas, as características consideradas e o foco da detecção. Os trabalhos citados nesta seção fazem parte de um ramo de pesquisa mais abrangente, sendo estes selecionados pela sua importância, ajudando a definir e desenvolver a pesquisa em questão.

O trabalho *"A Survey of Network-based Internet Intrusion Detection Data Sets"* [Ring et al. 2019] é uma revisão da literatura sobre pesquisas de detecção de ameaças que levanta os principais conjuntos de dados (*datasets*) utilizados para treinar e avaliar sistemas de detecção. São enumeradas algumas das características mais utilizadas pela literatura, principais aplicações, relações entre os conjuntos de dados e onde estão disponibilizados. Este trabalho é de suma importância para a comunidade acadêmica, principalmente sobre a relevância do uso de dados de tráfego de rede realistas. O artigo detalha um grupo de propriedades distintas que podem ser utilizadas para avaliar e adequar *datasets* a cenários e aplicações diversas. Estas propriedades são agrupadas em categorias, como volume de dados e o ambiente de onde foram gravados, o que permite classificar e comparar *datasets*, fornecendo uma visão sobre recomendações para seu uso. A pesquisa é tratada aqui como um manual utilizado para selecionar o *dataset* e ferramentas para seleção de características.

A pesquisa intitulada *"Design of multiple-level hybrid classifier for intrusion detection system using Bayesian clustering and decision trees"* [Xiang et al. 2008] propõe um sistema híbrido que combina modelos de ML supervisionados e não-supervisionados através da aplicação em um modelo de classificação em camadas para detecção de diversos tipos de ataques realizados através de redes de computadores. A pesquisa usa como base o projeto vencedor da KDDCUP99, um sistema híbrido de detecção de in-

trusão baseado em um algoritmo de conjuntos para geração de uma floresta de árvores de decisão. A proposta do projeto é melhorar a taxa de detecção de ataques *User to Root* (U2R) e *Remote to Local* (R2L). O sistema de detecção em múltiplos níveis mostra como a aplicação de modelos de árvore de decisão em estágios iniciais podem melhorar o desempenho da avaliação posterior de dados para classificação da origem e objetivos de ataques cibernéticos. Isso é performado e medido através do conjunto de dados KDDCUP99 [ISC 1999], base de referência bastante difundida e aplicada na validação de modelos de detecção de intrusão, mesmo tratando-se de um *dataset* antigo. Diversos problemas podem ser enumerados (dimensionalidade, generalização de dados, tradução dos dados, importância dos rótulos de dados, tipos de ataques, dentre outros) ao utilizar os dados da KDDCUP, sendo a base do sistema híbrido utilizada para entender como elaborar e como aplicar uma estrutura similar, dando foco na detecção de *botnets*.

O artigo *"Android Botnets: What URLs are Telling Us"* [Stakhanova et al. 2017] ressalta que as *botnets* se tornaram ameaças comuns contra computadores pessoais, ressaltando que o crescimento do uso de aparelhos móveis, assim como de dispositivos *Internet of Things* (IoT), aumentou o interesse de agentes operando *botnets* contra estes alvos. A pesquisa combina análises estática e dinâmica de endereços C&C e URLs utilizadas por *botnets* com foco em aparelhos móveis, mostrando a relação entre famílias de *botnets* e suas aplicações, enumerando algumas características da infraestrutura de cada rede de *bots* reconhecida, mas não chega a aplicar diretamente o *dataset* em modelos de detecção de intrusão. O processo de criação e coleta do *dataset*, assim como as informações sumarizadas na pesquisa foram utilizados para escolha das características mais importantes a serem consideradas na seleção de atributos do *dataset* utilizado durante os experimentos realizados.

Das pesquisas que buscaram aplicar algoritmos híbridos no processo de detecção de intrusão, o trabalho *"AdaBoost-Base Algoritnm for Netowrk Intrusion Detection"* [Hu et al. 2008] propõe uma sistema de detecção que utiliza um algoritmo de ML baseado no AdaBoost para criar classificadores melhores através de troncos de decisão. Troncos são árvores de decisão com um único nó raiz que, através da definição de rótulos iniciais, procura todas as possibilidades dentre árvores geradas através dos parâmetros decisórios para selecionar um caminho com um mínimo de falsos positivos no processo de classificação de intrusos. O modelo foi testado através do *dataset* KDDCUP99, segmentado em uma base de treino e de teste. A aplicação do AdaBoost se torna eficiente ao ser aplicada em modelos preditivos nomeados como classificadores fracos, e é utilizado para melhorar o processo de treinamento destes a partir dos rótulos dados no processo de geração do modelo inicial, permitindo atingir modelos menos suscetíveis ao *overfitting*, na prática, mas que são sensíveis a *outliers*.

Já na pesquisa intitulada *"Towards Effective Feature Selection in Machine Learning-Based Botnet Detection Approaches"* [Beigi et al. 2014], é apresentada uma visão sobre como sistemas de cibersegurança estão avançando e como as *botnets* estão se tornando mais sofisticadas. O objetivo do trabalho é mostrar a eficiência de diferentes características de dados de tráfego de rede para detecção de *botnets*. A extração de características aplicada por diversos pesquisadores de maneira intuitiva pode gerar resultados difusos, de acordo com a aplicação do modelo de classificação no processo de detecção. A seleção das características é um problema comum e existem diversas estratégias para

melhorar a filtragem e avaliar a qualidade dos resultados, e nesta pesquisa, foi aplicado um modelo de seleção de recursos usando um algoritmo de exaustão para seleção de atributos. Os métodos de seleção baseados em *wrapper* buscam validar e selecionar o melhor conjunto de atributos que melhore o resultado final da classificação, técnica geralmente conhecida como seleção por exaustão. A popularidade destes métodos tem aumentado em aplicações técnicas para análises de dados, e o trabalho usa um modelo para descobrir subconjuntos de atributos gerados a partir de uma seleção intuitiva, visando principalmente melhorar a etapa de processamento dos dados.

Existem problemas atrelados ao uso de sistemas de detecção que advém da necessidade de reduzir taxas de falsos positivos para garantir uma boa eficácia da ferramenta sem que o seu funcionamento (eficiência) afete dispositivos adjacentes ao sistema de detecção [Sommestad et al. 2021]. Portanto, as pesquisas citadas nesta seção são utilizadas para o estudo proposto neste trabalho, que visa tratar técnicas de *machine learning* em sistema de detecção que possam melhorar a eficácia dos resultados para detecção de intrusão. Sendo assim, foram estudadas técnicas de ML supervisionadas que são empregadas de forma abrangente na literatura, focando os esforços no entendimento dos desafios relacionados a detecção de *botnets* e como mensurar o desempenho dos modelos gerados, principalmente no que diz respeito ao pré-processamento de dados para obtenção dos modelos de ML. Mais detalhes serão abordados nas próximas seções.

## 3. Botnet

Diferente de outros *malwares*, as *botnets* possuem uma estrutura de comando e controle através de um ou mais servidores que recebem comandos reproduzidos para os nós da rede que executam as instruções recebidas, criando uma plataforma de ataques distribuídos [Stephan 2019]. Uma *botnet* é uma rede de *hosts* (bots ou zombies) infectados e controlados por um atacante (*botmaster*) para realização de alguma atividade maliciosa. O *botmaster* controla essa rede de zombies para iniciar ciberataques direcionados e automatizados, tais como ataques distribuídos de DoS, envio de *spams*, *phishing*, *click fraud*, roubo de informações e disseminação de outras ameaças [Xing et al. 2021]. Nos últimos anos, muitas *botnet* foram aprimoradas e passaram a ser operadas como ferramentas de ataque confeccionadas com características únicas, integrando funcionalidades mais específicas e sofisticadas através de estruturas distintas.

### 3.1. Ciclo de Vida e Estrutura

A organização de uma *botnet* se inicia pela entrega de um *payload* malicioso que cria de um canal de comunicação, a rede, que basicamente é composta por três elementos: o *botmaster*, o servidor de C&C, e o *bot* (Figura 1). Essa arquitetura se distingue de outras ameaças pela capacidade dada aos *bots* de operar ações coordenadas para atingir seus objetivos com pouca ou nenhuma intervenção do atacante, o *botmaster* [Craig A. Schiller 2007].

O funcionamento da *botnet* pode ser analisado a partir de seu ciclo de vida (Figura 2), no qual a *botnet* segue fases funcionais e observáveis durante sua operação [Craig A. Schiller 2007]. Tais fases são descritas como um conjunto de estados definidos [Feily et al. 2009], e autores conseguiram categorizar ciclos semelhantes de funcionamento entre *botnets* distintas [Silva et al. 2013]: a fase da infecção, a fase da comunicação e a fase do ataque.

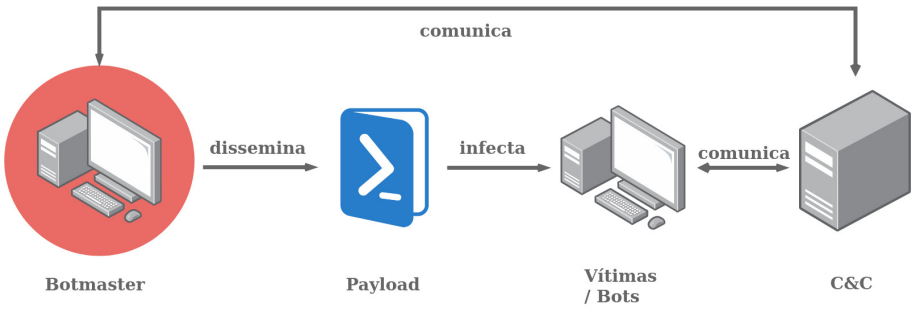


**Figura 1. Estrutura primária de uma botnet.**

A primeira fase é a infecção, na qual ataques são realizados contra computadores vulneráveis, em que as brechas de segurança são exploradas para comprometer o *host* por um *malware* que integrará o dispositivo como um novo *bot* da rede. Normalmente, essa fase pode ser dividida em subfases conhecidas como infecção inicial e infecção secundária. Durante a infecção inicial, os *hosts* são comprometidos por um software malicioso conhecido como *loader* que realiza tarefas como o *download* indesejado de outros programas maliciosos, códigos que podem ser utilizados para explorar outras vulnerabilidades e realizar buscas dentro do *host* e da rede da vítima [Craig A. Schiller 2007]. O papel principal do *loader* é auxiliar o atacante a obter uma posição e carregar os próximos binários da *botnet*. Sendo a primeira etapa da infecção bem sucedida, os binários baixados são instalados na máquina vulnerável para dar início à próxima etapa.

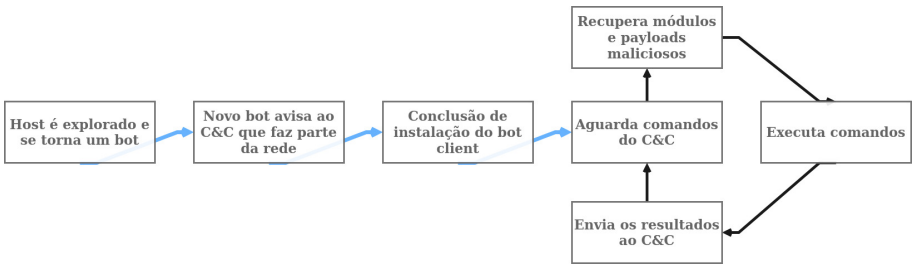


**Figura 2. Ciclo de vida de uma botnet.**

A segunda fase, de comunicação, ocorre quando o código malicioso da infecção inicia a busca por portas e serviços que conectarão a máquina comprometida com o servidor C&C e abrange a coleta de informações para o *botmaster* sobre o estado atual dos *bots* e o recebimento de instruções e atualizações da *botnet*. Nessa fase, podem ser realizadas diversas tentativas de comunicação com a *botnet* após o comprometimento e a comunicação com o servidor CC pode ser implementada de formas diferentes.

Com a comunicação estabelecida, a terceira fase é iniciada, quando os *bots* comunicam-se com o servidor comando e controle para receber instruções e realizar ataques ordenados pelo *botmaster* [Craig A. Schiller 2007], assim, correspondendo ao uso dos *bots* para chegar aos objetivos da rede maliciosa, e durante a fase de ataque os zombies geralmente são empregados em campanhas de *spams*, ataques DoS e disseminação de outros *malwares* [Piotr Białczak e Mattioli 2020].

### 3.2. Canal C&C

O canal de comando e controle é a principal característica das redes de *bots* e faz parte da infraestrutura empregada por atacantes cibernéticos, através de um conjunto de ferramentas e técnicas de invasão para penetrar nas redes de suas vítimas e manter a comunicação com os dispositivos comprometidos após a exploração de falhas de segurança. Os mecanismos utilizados para comunicação variam de acordo com o objetivo do atacante e também pela forma com o qual o canal de C&C é mantido, muitas vezes sendo a comunicação realizada através de múltiplas vias de comunicação por protocolos de rede distintos (SMB, IRC ou DNS) [Craig A. Schiller 2007].

Segundo o ATT&CK [MITRE 2019], *framework* de segurança que tem como objetivo mapear padrões de táticas, técnicas e procedimentos (TTP) de atores criminosos, são mapeadas mais de 16 técnicas distintas que adversários empregam para estabelecer e controlar vários níveis diferentes de canais de comunicação furtivos a depender da infraestrutura emprega, como o uso de esteganografia para ocultar comandos em tráfego de rede benignos, a adição de *junk data* no cabeçalho de protocolos de rede ou o uso de *Domain Generation Algorithms* (DGAs) para identificar dinamicamente domínios de destinos maliciosos.

Como parte principal de uma *botnet*, o servidor C&C encaminha os comandos executados pelo *botmaster* para realizar suas ações através de serviços como IRC, HTTP e *Peer to Peer* (P2P) [Xing et al. 2021], sendo as mais preponderantes as redes centralizadas, com base no protocolo IRC ou protocolo HTTP, e de redes descentralizadas, baseadas no protocolo P2P [Stephan 2019].

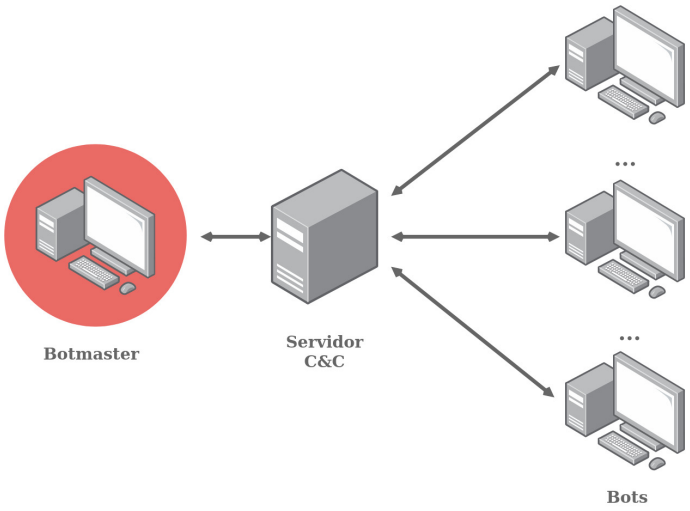


**Figura 3. Exemplo de topologia centralizada de uma botnet.**

Canais de C&C baseados em uma topologia centralizada (Figura 3) são muito semelhantes a serviços *client-server* tradicionais. O serviço instalado nos *bots* envia requisições de rede a servidor C&C afim de realizar atualizações, receber novos comandos e tarefas para serem executadas no *client* do *malware*, os nós mais extremos da *botnet* [Xing et al. 2021]. A infraestrutura muitas vezes contém mais de um servidor e pode incluir redirecionadores de rede para evitar mecanismos mediadores de defesa. É comum também que Serviços de nuvem sejam utilizados para entregar dados advindos de *Con-*

*tent Delivery Networks* () e mascarar a atividade do servidor de comando e controle. Os *botmasters* também realizam ataques para comprometer portais web legítimos, passando a utilizá-los para hospedar suas ferramentas maliciosas sem o conhecimento de seus proprietários. Em ataques mais sofisticados, os *malwares* de *botnets* introduzem *payloads* adicionais ofuscados com listas de servidores C&C, ou possuem módulos que executam DGAs [MITRE 2019].

Em topologias descentralizadas, as *botnets* são desenvolvidas com o objetivo de serem mais resilientes e manterem suas funcionalidades básicas [Craig A. Schiller 2007]. *Botnets* com infraestrutura de C&C descentralizadas (Figura 4) adotam protocolos P2P como principal meio de comunicação da rede de *bots*, implicando que os *bots* pertencentes a rede P2P formam uma sobreposição na qual o *botmaster* pode utilizar qualquer *bot* para torná-lo um disseminador de comandos a outros pares da rede. Essa abordagem reduz a velocidade no envio de comandos e na gestão da rede devido ao número de mensagens que devem ser trocadas pelos *bots* na rede P2P, em que a principal ideia da topologia é que cada *bot* possua uma lista de pares que é atualizada sempre que um novo *bot* é inserido na rede [Chang et al. 2009]. Mesmo que sejam redes de bosts mais complexas e mais custosas de gerir, se partes da rede forem desligadas, os *bots* restantes podem ser capazes de se comunicar e assim persistir com o propósito malicioso da *botnet*.

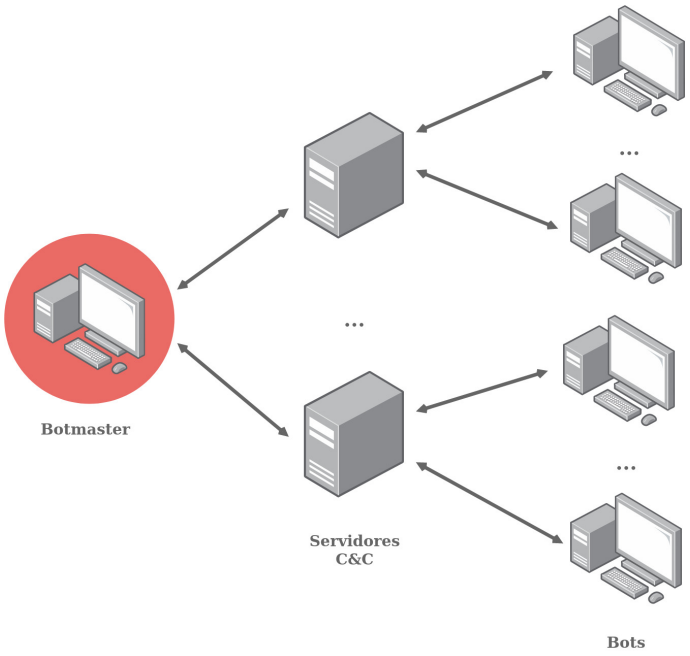


**Figura 4. Estrutura baseada em uma topologia com múltiplos servidores.**

Algumas das *botnets* descobertas nos últimos anos adotaram uma organização mais avançada, implementando uma arquitetura híbrida [Silva et al. 2013] que combina princípios de um canal de comando e controle centralizado e descentralizado. Essas *botnets* usam protocolos de comunicação P2P e possuem grupos distintos de *bots*, um capaz de atuar como servidor e *bot* comum da rede P2P e outro que somente recebe configurações e instruções de outros *bots*. Os dois grupos possuem funções nas redes, sendo o primeiro grupo responsável por garantir que as instruções e mensagens do C&C sejam enviados por toda a *botnet* e o segundo grupo responsável por inciar os coman-

dos recebidos para realizar suas atividades maliciosas e aguardando novas instruções, tornando a rede mais resistente a esforços que visem derrubar a *botnet*.

## 4. Metodologia

Diversos sistemas experimentais de detecção de intrusão foram desenvolvidos nas últimas décadas com vários objetivos, aplicando técnicas e princípios para classificar padrões de comportamento de *botnets* [Silva et al. 2013]. O método de pesquisa seguido neste trabalho se baseia em modelos de detecção a partir da análise do fluxo de tráfego que é coletado da "borda" de redes de computadores, através dispositivos roteadores ou *firewalls*, por exemplo. Portanto, para realização deste trabalho foi utilizada a seguinte estrutura de pesquisa:

1. Inicialmente, foi feito um levantamento bibliográfico do estado da arte da área de detecção de *botnets*, principalmente sobre a aplicação de técnicas de ML;
2. Com o estudo inicial, foi selecionada uma base de dados (*dataset*) que agrega dados de tráfego de diversas *botnets*;
3. O estudo da literatura também permitiu visualizar quais são as técnicas mais utilizadas, a fim de implementá-las durante os experimentos;
4. Em seguida, foi iniciada uma análise exploratória do *dataset* selecionado, sendo a base utilizada para todos os experimentos posteriores, dando sequência ao pré-processamento da base de dados, permitindo chegar a uma visão sumarizada dos dados para uso nos modelos de ML escolhidos;
5. Com o processamento concluído, foi feita a execução dos experimentos entre as técnicas de aprendizado em duas abordagens, uma utilizando os dados com as características escolhidas no pré-processamento e outra com um processo de busca por características de maior relevância a partir de uma estratégia de seleção por por força bruta;
6. Em seguida, os modelos de classificação gerados foram executados em um meta-algoritmo, técnica que usa classificadores "fracos" para gerar modelos preditivos "fortes".

Diversos trabalhos da literatura discutem como melhorar [García et al. 2014] a construção de ambientes de desenvolvimento para mecanismos de segurança cibernética, criando uma base vasta sobre projetos desenvolvidos pela comunidade de cibersegurança. Dos desafios discutidos [Aviv e Haeberlen 2011], é importante pensar em como estruturar bases de dados do tráfego de *botnets* para sua aplicação na área de detecção. Para este trabalho, os seguintes aspectos técnicos e práticos de um conjunto de dados precisaram ser correspondidos. O *dataset* precisa ser: generalista, contendo dados de natureza abrangente; ser realista, sendo a base seja composta por uma mescla de dados reais e dados capturados em experimentos; e ser representativo, com a sobreposição de diversas informações unificadas em um conjunto experimental.

Tendo isso em mente, neste trabalho foi utilizada a base ICSX 2014 [Beigi et al. 2014], conjunto que agrupa dados de diversas *botnets* associadas a fluxos de tráfego de rede benignos obtidos através de um gerador de pacotes de rede que simula o comportamento de uma rede de computadores real. Como resultado, foram gerados dois conjuntos de dados, um *dataset* de treinamento com 5.1 GB de tamanho, com 48,25%

(Tabela 1) dos pacotes representando uma parcela maliciosa e os demais dados representam tráfego de rede benigno. O segundo *dataset*, utilizado para testes, têm 2.1 GB de tamanho e 53,05% representa a parcela maliciosa dos dados (Tabela 2).

| Tráfego Malicioso | | |
|---|---|---|
| Nome da Botnet | Percentual | Pacotes |
| Neris | 8,75% | 227432 |
| Rbot | 3,05% | 79388 |
| Menti | 5,87% | 152575 |
| Sogou | 0,34% | 8867 |
| Murlo | 5,65% | 146920 |
| Virut | 13,13% | 341132 |
| IRCBot& BH | 2,41% | 62745 |
| Tbot | 13,04% | 338883 |
| Weasel | 6,80% | 176735 |
| Zeus | 3,18% | 82536 |
| OSX Trojan | 0,04% | 1124 |
| Zero Access | 7,55% | 196268 |
| Smoke | 1,60% | 41380 |
| IRC Attack | 28,58% | 742600 |

**Tabela 1. Distribuição do dataset de treinamento.**

Após a caracterização e análise exploratória, a fase de pré-processamento dos dados levou em consideração como filtrar características importantes para as próximas etapas. A composição inicial dos dados não possui rotulação, ou caracteriza informações como duração de uma comunicação, quantidade de pacotes trafegados ou o direcionamento de uma comunicação. Foi necessário aplicar um mecanismo para rotular informações relevantes para o processo de detecção. Isso foi realizado através da ferramenta CicFlowMeter [Mai e Park 2016] para caracterizar e analisar fluxos de tráfego de rede. A ferramenta integra recursos para extrair informações de pacotes de tráfego e permitir a rotulagem de fluxos de comunicação baseados na continuidade temporal e na quantidade de informações trocadas entre origem e destino.

| Tráfego Malicioso | | |
|---|---|---|
| Nome da Botnet | Percentual | Pacotes |
| IRC Attack | 18,26% | 818375 |
| Neris | 9,63% | 431573 |
| Rbot | 71,76% | 3217575 |
| Virut | 0,34% | 15305 |

**Tabela 2. Distribuição do dataset de teste.**

Após o uso da ferramenta, foi necessário selecionar características importantes para a etapa de aprendizado. Neste trabalho, a abordagem seguida por [Beigi et al. 2014] é referenciada em outros trabalhos [Heuer et al. 2016], justamente em etapas da seleção de características ou no processamento dos dados. O método foi escolhido pela

| Atributo | Descrição |
|---|---|
| Origem | Informação sobre o IP de origem da comunicação. |
| Destino | Informação sobre o IP de destino da comunicação. |
| PO | Informação sobre a porta de origem (PO) da comunicação. |
| PD | Informação sobre a porta de destino (PD) da comunicação. |
| Protocolo | Especificação do protocolo utilizado de transporte das informações durante a comunicação. |
| Duração | Duração total de uma comunicação. |
| TP | Total de pacotes trocados (TP), recebidos e enviados, durante uma comunicação. |
| PPT | Proporção de pacotes (PPT) trocados (recebidos e enviados) durante uma comunicação. |
| TBT | Total de bytes trafegados (TBT) durante uma comunicação. |
| TMP | Tamanho médio de um pacote (TMP). |
| DPT | Desvio padrão do tamanho de um pacote (DPT). |
| PS | Média de pacotes trafegados por segundo. |
| BS | Média de bytes trafegados por segundo. |
| ICP | Intervalo entre a chegada de novos pacotes (ICP). |
| NS | Número de pacotes com a flag SYN (NS) trocados durante uma comunicação. |
| PDU | Proporção entre download e upload (PDU) de dados durante uma comunicação. |
| AM | Tempo de atividade média (AM) de um fluxo de dados antes da comunicação se tornar ociosa. |
| DPA | Desvio padrão de atividade (DPA) baseado no tempo de um fluxo de dados antes da comunicação se tornar ociosa. |
| OM | Tempo de ociosidade média (OM) de um fluxo de dados antes da comunicação se tornar ativa. |
| DPO | Desvio padrão de ociosidade (DPO) baseado no tempo de um fluxo de dados antes da comunicação se tornar ativa. |

**Tabela 3. Atributos selecionados após o pré-processamento dos datasets.**

comparação de informações que são utilizadas no processo de detecção de diferentes estruturas de *botnets*, sendo selecionados os atributos descritos na Tabela 3.

Cada atributo foi selecionado pelo que é mais difundido na literatura sobre como o processo de preparação dos dados é realizado para serem aplicadas técnicas de detecção, se diferindo entre cenários oriundos das topologias mais comuns de *botnets* — centralizadas ou descentralizadas [Chang et al. 2009]. Alguns dos atributos selecionados são utilizados na área de detecção como identificação inicial de comunicações de rede, sobre a sua origem e seu destino (Origem, Destino, PO, PD e Protocolo). Mais especificamente, na detecção de *botnets*, esses atributos são aplicados independentemente do cenário de detecção [Haddadi e Zincir-Heywood 2017]. Existe uma discussão ampla sobre as principais características que devem ser consideradas em métodos de detecção, como é o caso dos atributos Duração, DPT, PS e BS, que são utilizados na caracterização de comunicações similares e aplicados para discernir tráfego IRC de outros protocolos de rede [Beigi et al. 2014]. Seguindo o processo, foram selecionados atributos considerando

a melhor caracterização do tráfego advindo de *botnets*: a partir do tamanho dos pacotes trafegados (TBT, TMP, DPT); de seu comportamento ao longo do tempo (OM, AM, ICP, DPA e DPO); e da distinção de como os dados são trafegados (TP, PPT, NS, PDU).

### 4.1. Apurando a seleção de atributos

As etapas de um processo de seleção de atributos geralmente focam em tipos de *botnets* e cenários de detecção específicos, tornando a seleção enviesada para a aplicação escolhida. Enquanto alguns procedimentos podem se basear na estrutura de comunicação de *botnets* e como as mesmas utilizam protocolos de comunicação de rede [Chang et al. 2009], o desenvolvimento de métodos modernos de detecção baseados em fluxos de tráfego leva em consideração como técnicas de ML que podem ser aplicadas para diversas abordagens [Haddadi e Zincir-Heywood 2017]. Com isso, foi escolhida uma aproximação bastante comum das áreas de DM e ML para melhorar a eficiência computacional e reduzir erros de generalização a partir de um algoritmo de seleção sequencial de características (SFA). SFAs também são conhecidos como algoritmos de busca gulosa ou busca por exaustão, populares em sistemas sendo necessária ênfase na eficiência do processo de aprendizado de modelos de detecção [Ding e Fu 2018]. A aplicação de um algoritmo de busca é um dos focos desta pesquisa e é utilizada pensando na melhora dos resultados obtidos durante a detecção. O processo seguido nesta etapa foi:

1. Analisar métodos de busca de SFAs;
2. Selecionar os algoritmos de detecção a partir do que é mais comum e amplamente empregado na literatura;
3. Realizar o treinamento e teste dos modelos sem a aplicação do SFA;
4. Realizar o treinamento e teste dos modelos com o SFA;
5. Filtrar o *dataset* a partir do resultado da etapa anterior, classificando as técnicas escolhidas em um algoritmo de impulsionamento (*boosting*).

Essas etapas representam a composição de aplicações comuns de outros trabalhos pesquisados, que também dão ênfase na melhoria dos resultados de etapas preliminares ao processo de treinamento e teste de classificadores [Ding e Fu 2018]. O algoritmo de seleção de atributos escolhido é baseado em uma busca sequencial direta [Raschka 2018], que utiliza a estrutura inicial dos dados e, a partir do melhor resultado encontrado para um atributo que potencialize a acurácia do modelo, é gerado um subconjunto ótimo (K). O algoritmo, então, passa a comparar todos os resultados exaustivamente e adiciona atributos sequencialmente ao subconjunto K, até que o critério (c) seja atingido. Esse critério é definido na inicialização da seleção e, para os testes realizados, foi estabelecido como 5. A adição de atributos ao subconjunto ótimo (K) é finalizada quando só existam características que representem o melhor resultado obtido pelos testes com o modelo de ML escolhido.

### 4.2. Detecção de botnets e machine learning

*Machine learning* é um ramo da área de inteligência artificial que tem por objetivo construir e estudar sistemas que podem aprender a partir de dados, e o processo de aprendizado implica na habilidade de reconhecer padrões que qualificam a decisão ou a classificação de novos dados a partir dos dados vistos anteriormente [Dua e du 2016]. Sistemas de detecção de intrusão que aplicam métodos de ML, a partir do tráfego de rede, assumem que o comportamento de uma *botnet* pode ser distinguido e catalogado através da

aplicação de técnicas de aprendizado que, na prática, podem seguir diversas abordagens sendo as mais populares a de aprendizado supervisionado e a de aprendizado não supervisionado [Craig A. Schiller 2007].

Métodos de aprendizado supervisionado operam com suporte fornecido por um supervisor, que rotula cada padrão conhecido — possibilidades a serem aprendidas — sobre o conjunto de dados de treinamento, de modo que o objetivo do problema passa a ser reduzir o custo operacional do modelo no processo de reconhecimento de padrões. A natureza dos problemas que aplicam algoritmos de aprendizado supervisionado geralmente tem seus modelos comumente conhecidos como classificadores [Silva et al. 2013]. Portanto, o cenário inicial deste trabalho segue um sistema de aprendizado supervisionado para detecção que se baseia em classificadores que primeiro são treinados usando um conjunto de dados de treinamento para, a partir do modelo gerado, serem realizados os testes que mensurarão sua eficácia (Figura 5).

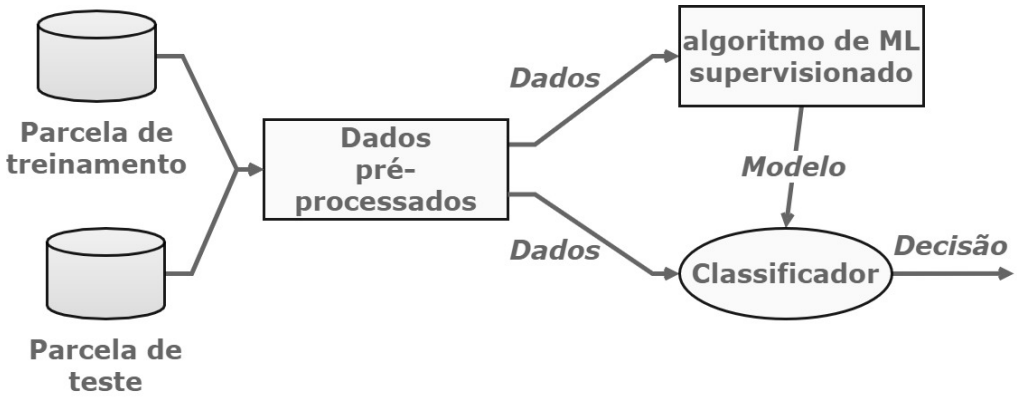


**Figura 5. Etapas de um modelo supervisionado.**

### 4.3. Técnicas de aprendizado supervisionado

Cada método tem seu próprio domínio de aplicação e suas próprias vantagens e desvantagens. Neste trabalho, foram utilizadas técnicas de aprendizado supervisionado consideradas comuns, que inclui a árvore de decisão, SVM e AdaBoost, algoritmos de ML já descritos e estudados de forma abrangente [Dua e du 2016]. O Scikit-learn [Pedregosa et al. 2011] foi utilizado para implementar os classificadores para cada experimento realizado.

#### 4.3.1. Árvore de decisão

O método de classificação de uma árvore de decisão funciona como um fluxograma em forma de árvore, onde cada nó indica um teste feito sobre um valor e as ligações entre os nós representam os valores possíveis do teste do nó superior [Dua e du 2016]. Esse método é simples e não necessita de nenhum parâmetro de configuração, onde as folhas indicam a classe a qual o registro pertence e, desta forma, após a construção do modelo, é possível classificar um novo registro seguindo o fluxo da árvore do nó raiz até chegar a uma folha.

Dentre as vantagens da aplicação desta técnica de classificação, o processo decisório tomado através da árvore gerada leva em consideração as regras que são mais relevantes, na sua grande maioria tendo um bom grau de precisão. O algoritmo de árvore de decisão utilizado para os experimentos deste trabalho é o CART [Scikit-learn 2021].

### 4.3.2. SVM

*Support Vector Machine* (SVM) é uma técnica aplicada em problemas de regressão e classificação capaz de transformar dados complexos e separá-los em classes ordenadas definidas em um plano bidimensional [Dua e du 2016]. O modelo gerado pelo SVM espalha as características aprendidas em um vetor através dos rótulos conhecidos. Portanto, essa técnica permite visualizar os limites entre grupos de características, representados através de uma função decisória para classificar novos dados a partir de sua posição em um plano que separa duas classes distintas de dados.

O SVM é bastante consolidado em pesquisas sobre detecção de intrusão por causa de sua boa performance de generalização, assim como a fácil aplicação e mínima necessidade de configuração [Enache e Patriciu 2014]. O algoritmo aplicado nos experimentos é o C-SVC [Hsu et al. 2008].

### 4.3.3. AdaBoost

Métodos de classificação baseados em conjuntos — uma mistura de múltiplos classificadores de aprendizado — são consistentemente aplicados em pesquisas que visam melhorar o processo decisório e reduzir o enviesamento em modelos de classificação [Hu et al. 2008]. O AdaBoost é uma técnica que realiza uma combinação de classificadores para criar um mecanismo de ponderação, um tronco de decisão gerado pelo treino de diversos modelos "fracos". Cada representante do conjunto treina classificadores fracos para compor a estrutura de ponderação que transforma os modelos em classificadores melhores, "fortes", com base no critério obtido entre métricas de desempenho comparadas com a taxa de erro da etapa de predição.

Nos experimentos, foram utilizadas dua abordagens do AdaBoost do algoritmo AdaBoost-SAMME, formando uma comparação de desempenho do SAMME discreto, que escolhe os melhores resultados a partir da função de erro, e o SAMME-R, que utiliza estimadores probabilísticos que diminuem o número de iterações necessárias para atingir os modelos fortes [Pedregosa et al. 2011].

## 4.4. Avaliando os classificadores

A métrica considerada para avaliar os modelos de classificação usados nos experimentos foi a acurácia (AC), mas também foram consideradas a medida F1, a precisão (P) e a revocação (R) para caracterizar o desempenho de cada modelo gerado. Cada métrica foi obtida através dos recursos do Scikit-learn, que calcula cada métrica pelas fórmulas:

$$AC = \frac{VP + VN}{VP + FP + VN + FN} * 100\%$$

$$F1 = \frac{2VP}{2VP + FP + FN} * 100\%$$

$$P = \frac{VP}{2VP + FN} * 100\%$$

$$R = \frac{VP}{VP + FN} * 100\%$$

Sendo VP (Verdadeiros Positivos) é o número de dados rotulados como "maliciosos" que foram classificados corretamente, VN (Verdadeiros Negativos) é o número de dados rotulados como "benignos" e foram classificados corretamente, FP (Falsos Positivos) é o número de dados rotulados como "benignos" e foram classificados como "maliciosos" e FN (Falsos Negativos) é o número de dados rotulados como "maliciosos" que foram classificados como "benignos".

## 5. Resultados

Na Tabela 4, são mostrados os resultados obtidos pelos primeiros testes com os classificadores, partindo dos parâmetros padrões de cada modelo. Neste primeiro momento, sem o uso de algoritmos para seleção dos atributos, o SVM teve uma melhor performance para detecção, com uma taxa de acurácia de 72% e sua precisão se mantém em um patamar similar. Já o modelo de árvore de decisão, teve uma baixa acurácia e uma taxa de revocação de somente 46%.

| Classificador | Acurácia | Medida F1 | Precisão | Revocação |
|---|---|---|---|---|
| Árvore de decisão | 54% | 81% | 71% | 46% |
| SVM | 72% | 81% | 70% | 97% |

**Tabela 4. Resultados do primeiro conjunto de experimentos.**

O próximo conjunto de testes foi realizado aplicando o SFA, que ao final indica a melhor taxa de acurácia obtida pela combinação de atributos que potencializam os resultados dos modelos, destacados na Tabela 5. A proporção de pacotes trocados (PPT), o intervalo de tempo entre pacotes (ICP) e o números de pacotes com flags SYN (NS) foram selecionados em ambos os modelos, sendo foi possível observar uma melhora significativa na acurácia do modelo da árvore de decisão.

| Classificador | Melhor acurácia | Atributos selecionados |
|---|---|---|
| Árvore de Decisão | 94% | TP, PPT, BS, ICP, NS |
| SVM | 64% | PPT, ICP, NS, AM, PDU |

**Tabela 5. Resultados do segundo conjunto de experimentos**

Os resultados da aplicação do algoritmo de seleção sequencial (SFA) destacou atributos semelhantes aos resultados de [Beigi et al. 2014], onde os atributos de intervalo

entre a chegada de pacotes (IPC) e da proporção de pacotes trocados (PPT) também foram selecionados no melhor resultado obtido pelo classificador baseado em uma arvore de decisão. No entanto, o processo de seleção de atributos do foi realizado com testes isolados por cenários. Já no trabalho atual, os resultados foram obtidos levando em consideração o número máximo de interações necessárias para que o SFA obtivesse o subconjunto ótimo, chegando a uma taxa de acurácia de 94% para o classificador baseado na árvore de decisão, e 64% para o modelo baseado no SVM.

Os próximos testes utilizaram os dados filtrados a partir dos conjuntos de características selecionadas pelo SFA, sendo estes TP, PPT, BS, ICP, NS, AM e PDU. Os testes com o modelo AdaBoost foram realizados com 250 estimadores distintos para as duas implementações do SAMME. Pelos resultados descritos na Tabela 6 é possível visualizar que o melhor modelo foi obtido utilizando o padrão SAMME-R.

| Classificador | Base | Acurácia | Medida F1 | Precisão | Revocação |
|---|---|---|---|---|---|
| SAMME | Árvore de Decisão | 63% | 68% | 74% | 62% |
| | SVM | 62% | 68% | 71% | 68% |
| SAMME-R | Árvore de Decisão | 65% | 69% | 70% | 71% |
| | SVM | 64% | 76% | 69% | 74% |

**Tabela 6. Resultados do AdaBoost.**

Ao comparar os resultados do trabalho realizado por [Hu et al. 2008], os classificadores obtidos através do treinamento com o AdaBoost obtiveram taxas de detecção significativamente menores, onde o melhor desempenho de um classificador chegou a ter uma acurácia de 65% na aplicação baseada em árvore de decisão. A redução na taxa de detecção do terceiro conjunto de experimentos (Tabela 6) pode ser explicada, já que o conjunto de dados aplicado por [Hu et al. 2008] possui pouca variação de dados provenientes de *botnets*, com *dataset* do KDD99 [ISC 1999] possuindo uma porção de dados de *botnets* quase que totalmente proveniente de comunicações de baseadas nos protocols SMTP e UDP. O *dataset* aplicado nos experimentos desse trabalho, o ICSX 2014 [Beigi et al. 2014] se baseia em diversas estruturas e canais distintos utilizados por *botnets*, ponto que precisa ser levado em consideração na avaliação dos resultados.

O comitê de estimadores do AdaBoost permitiu observar que os resultados obtidos, tanto para a avaliação da taxa de detecção de falsos positivos quanto para uma precisão, estão mais equilibrados, criando dessa maneira uma consistência que não caracteriza problemas relacionados ao *overfitting* dos modelos, mesmo que as taxas de erro possam chegar a taxas próximas de 30% do conjunto de dados testados. Ao final, foi possível estruturar um modelo balanceado de um sistema de detecção de intrusão de *botnets* (Figura 6), com fases bem definidas correspondidas através do conjunto de testes realizados.

## 6. Conclusões

Este trabalho apresentou um estudo sobre a utilização de técnicas de aprendizado de máquina para a detecção de *botnets* baseadas no tráfego de rede. Os resultados obtidos permitiram validar características que foram utilizadas em estudos anteriores, com destaque para o uso de métodos para melhorar as etapas de pré-processamento de dados.

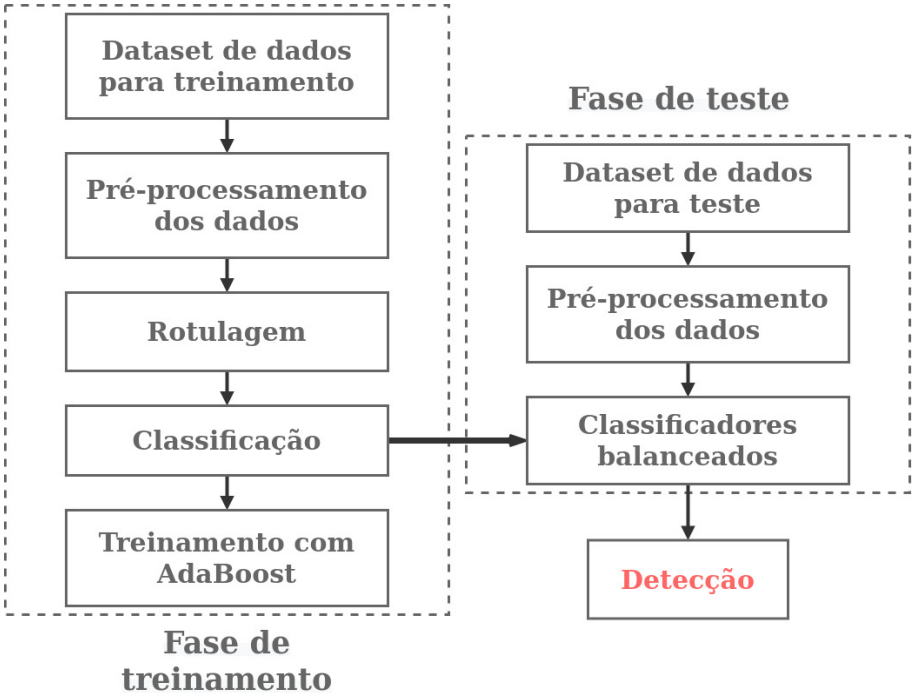


**Figura 6. Modelo do sistema de detecção de intrusão.**

Estudos como este são cada vez mais necessários e relevantes para a área devido ao crescente número de aparelhos e na quantidade de pessoas conectadas em redes públicas ou privadas, possíveis alvos de *botnets* e agentes maliciosos.

Os resultados obtidos pelos modelos gerados com o AdaBoost, com acurácia média de 63,5%, permitiram observar que classificadores de aprendizado supervisionado podem ser aplicados em sistemas reais, sendo também possível uma melhor filtragem nas características dos dados utilizados para treinar e avaliar métodos de detecção. A ênfase na melhoria das etapas de pré-processamento é crucial para diminuir problemas relacionados a performance computacional e ao tempo de resposta de sistemas de detecção, tendo em vista a necessidade de que sistemas de detecção precisam ser capazes de criar indicadores cruciais para a tomada de medidas proativas, antes que danos sejam gerados a indivíduos e organizações.

Para continuidade das pesquisas iniciadas nesse trabalho, é necessário expandir o sistema para trabalhar com dados coletados em tempo real, validando se o mesmo seria eficiente em um mecanismo de proteção. Isso pode ser feito mudando a abordagem *offline* do sistema e aplicando redes neurais artificiais ou algoritmos de *machine learning* não-supervisionados, para encontrar padrões de comportamento monitorando e analisando fluxos de rede em tempo real. Outras linhas de trabalho podem ser iniciadas através da segmentação de cenários de detecção, focando na análise de *botnets* específicas ou em tipos de dados padrões de comunicações como P2P, IRC ou TCP.

Os resultados e os teste realizados não levaram e consideração uma avaliação profunda de métricas baseadas no tempo e no uso de recursos computacionais. Avaliação de gestão de recursos e tempo de resposta de sistemas para detecção são cruciais para validar a aplicação do sistema em um componente de rede real, se tornando necessário um estudo

mais profundo sobre métricas de eficiência do modelo baseada no seu desempenho computacional. Os experimentos realizados podem ser utilizados para avaliar o desempenho dos classificadores e, a partir do resultado obtido, ajustar os parâmetros dos modelos ou até mesmo validar outros classificadores que não afetem o desempenho do sistema.

## Referências


A10Networks, T. I. T. (2020). Ddos attack mitigation: A threat intelligence report. Technical report, A10 Networks.

Alenezi, M., Alabdulrazzaq, H., Alshaher, A., e Alkharang, M. (2020). Evolution of malware threats and techniques: A review. *International Journal of Communication Networks and Information Security*, 12:326.

Aviv, A. e Haeberlen, A. (2011). Challenges in experimenting with botnet detection systems. pages 6–6.

Beigi, E., Jazi, H., Stakhanova, N., e Ghorbani, A. (2014). Towards effective feature selection in machine learning-based botnet detection approaches. *2014 IEEE Conference on Communications and Network Security, CNS 2014*, pages 247–255.

Chang, S., Zhang, L., Guan, Y., e Daniels, T. (2009). A framework for p2p botnets. pages 594 – 599.

Craig A. Schiller, Jim Binkley, D. H. G. E. T. B. C. W. M. C. (2007). *Botnets: The Killer Web App*. Syngress.

Ding, J. e Fu, L. (2018). A hybrid feature selection algorithm based on information gain and sequential forward floating search. *Journal of Intelligent Computing*, 9:93.

Dua, S. e du, X. (2016). Data mining and machine learning in cybersecurity.

Enache, A. e Patriciu, V.-V. (2014). Intrusions detection based on support vector machine optimized with swarm intelligence.

Feily, M., Shahrestani, A., e Ramadass, S. (2009). A survey of botnet and botnet detection. pages 268–273.

García, S., Grill, M., Stiborek, J., e Zunino, A. (2014). An empirical comparison of botnet detection methods. *Computers  Security*, 45:100–123.

Haddadi, F. e Zincir-Heywood, A. (2017). Botnet behaviour analysis: How would a data analytics-based system with minimum a priori information perform? *International Journal of Network Management*, 27:e1977.

Heuer, T., Schiering, I., Klawonn, F., Gabel, A., e Seeger, M. (2016). Recognizing time-efficiently local botnet infections - a case study.

Hsu, C., Chang, C., e Lin, C. (2008). A practical guide to support vector classification. *BJU International*, 101:1396–1400.

Hu, W., Hu, W., e Maybank, S. (2008). Adaboost-based algorithm for network intrusion detection. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 38:577–583.

Imam, M., Nir, M. P., e Matrawy, A. (2014). A survey on botnet architectures, detection and defences. *International Journal of Network Security*, 17.

ISC (1999). Kdd cup 1999 data. Disponível em: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

Mai, L. e Park, M. (2016). A comparison of clustering algorithms for botnet detection based on network flow. pages 667–669.

MITRE (2019). Mitre attck: Command and control. Disponível em: https://attack.mitre.org/tactics/TA0011/.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., e Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Piotr Białczak, Paweł Pawliński, K. R. e Mattioli, R. (2020). Proactive detection – measures and information sources. Technical report, European Union Agency for Cybersecurity.

Raschka, S. (2018). Mlxtend: Providing machine learning and data science utilities and extensions to python's scientific computing stack. *The Journal of Open Source Software*, 3(24).

Ring, M., Wunderlich, S., Scheuring, D., Landes, D., e Hotho, A. (2019). A survey of network-based intrusion detection data sets. *Computers  Security*, 86.

Scikit-learn (2021). Tree algorithms: Id3, c4.5, c5.0 and cart. Disponível em: https://scikit-learn.org/stable/modules/tree.html#id2.

Silva, S., Silva, R., Pinto, R., e Salles, R. (2013). Botnets: A survey. *Computer Networks*, 57:378–403.

Sommestad, T., Holm, H., e Steinvall, D. (2021). Variables influencing the effectiveness of signature-based network intrusion detection systems. *Information Security Journal: A Global Perspective*, 0(0):1–18.

Spamhaus, M. T. (2021). Spamhaus botnet threat update: Q1-2021. Technical report, SPAMHAUS.

Stakhanova, N., Ghorbani, A., e Abdul kadir, A. F. (2017). Android botnets: What urls are telling us.

Stephan, C. (2019). An analysis of the evolution of botnets. Creative components, Iowa State University.

Symantec, S. R. T. (2019). Internet security threat report. Technical Report 24, Symantec.

Xiang, C., Yong, P., e Meng, L. (2008). Design of multiple-level hybrid classifier for intrusion detection system using bayesian clustering and decision trees. *Pattern Recognition Letters*, 29:918–924.

Xing, Y., Shu, H., Zhao, H., Li, D., e Guo, L. (2021). Survey on botnet detection techniques: Classification, methods, and evaluation. *Mathematical Problems in Engineering*, 2021:1–24.