Identificação Antecipada de Botnets por Aprendizagem de Máquina

Anderson Bergamini de Neira[1], **Alex Medeiros**[1], **Michele Nogueira**[1]

[1]Centro de Ciência de Segurança Computacional (CCSC)
Universidade Federal do Paraná (UFPR)

{abneira,michele}@inf.ufpr.br, aelxmedeiros@gmail.com

Resumo. *O envio de spam, o roubo de dados pessoais e o ataque de negação de serviço são exemplos de ações resultantes da exploração de vulnerabilidades em dispositivos inseguros conectados à Internet. A constante evolução dos ataques, o aumento na quantidade de dispositivos vulneráveis devido à Internet das Coisas (IoT) e os elevados custos com os danos causados reforçam a necessidade de antecipar a ação de redes de dispositivos infectados (bots) geradoras de ataques. Neste contexto, os algoritmos de aprendizagem de máquina são relevantes para identificar essas redes, pois oferecem adaptação e tratamento de uma quantidade massiva de dados. Este trabalho apresenta o sistema ANTE, identificação ANTecipada de botnEts com base em algoritmos de aprendizagem de máquina. Instanciamos o sistema e comparamos os resultados obtidos de diferentes cenários e sob a análise de dados, como as bases de dados CTU-13 (cenários 10 e 11), CICDDoS2019, ISOT HTTP Botnet, CAIDA DDoS Attack 2007 e CSE-CIC-IDS2018. Embora as instâncias do sistema sejam capazes de identificar os bots, não existe uma única capaz de atender todos os cenários.*

1. Introdução

A "Internet das Coisas" ou IoT (do inglês, *Internet of Things*) tem destaque na academia e na indústria, produzindo cada vez mais avanços. Os dispositivos da IoT possuem limitações na largura de banda, baixo poder de processamento e pouca memória. Tais aspectos, agregados a vulnerabilidades no projeto de *hardware* e de *software*, tornam esses dispositivos alvos fáceis de atacantes, que se beneficiam comprometendo não apenas a privacidade de dados sensíveis, mas também com o uso desses dispositivos infectados na construção de ataques massivos e mais danosos. Apesar destas limitações, as estimativas são de crescimento na quantidade de dispositivos conectados à Internet, podendo chegar a um marco de 30 bilhões até 2020 e tráfego de 40 trilhões de gigabytes até 2020, 57 exabytes em 2021 e 77 exabytes por mês em 2022 [Nordrum 2016, Cisco 2019]. Tal crescimento, quando explorado por atacantes, tende a aumentar o potencial danoso das chamadas redes de *bots* (*botnets*).

As *botnets* são um dos principais problemas relacionados à segurança . Um *bot* é um dispositivo conectado à Internet, infectado por um *malware*, permitindo ao atacante executar ações remotamente. Uma *botnet* compreende dispositivos infectados capazes de realizar comandos dos *botmasters* [Mane 2017]. Um dos principais fatores propulsores da periculosidade das *botnets* consiste no fato da mesma executar diferentes tipos de ataques. É possível utilizar *botnets* para: (i) o envio de correspondência eletrônica em massa também conhecidos com spam; (ii) captura de informações pessoais; (iii) e

ataques distribuídos de negação de serviço. Além da diversidade de ataques, detectar e identificar *botnets* para limitar ou bloquear seu acesso a um servidor ou uma rede não é trivial [Khanchi et al. 2018].

Diante da diversidade geográfica, computacional e tecnológica, a identificação de *botnets* é um desafio. Karim et al. (2014) afirmam que os *malwares* escondem o código malicioso de forma tão eficiente e várias abordagens de detecção baseada em assinaturas não são capazes de identificá-los. Além disso, com o advento de novas tecnologias e com o aperfeiçoamento das técnicas de invasão, os *botmasters* estão conseguindo evitar as técnicas de detecção. Gupta and Badve (2017) citam que, em geral, os atacantes conseguem inundar os servidores com pacotes similares aos reais, sendo difícil de classificar o fluxo malicioso do real. Chang et al. (2018) verificaram que os atacantes estão intercalando entre enviar pacotes e permanecer temporariamente inativo para dificultar a identificação dos *bots*. Assim, faz-se necessário o avanço nas investigações para não apenas detectar *botnets*, mas sim antecipar sua formação como um caminho para evitar danos maiores aos serviços e às redes ocasionados por ataques gerados a partir de *botnets*.

Este trabalho apresenta o sistema ANTE para identificação ANTecipada de *botnEts*, cuja base é um conjunto de algoritmos de aprendizagem de máquina. Os algoritmos de aprendizagem de máquina são relevantes para identificar essas redes, pois oferecem adaptação e tratamento de uma quantidade massiva de dados. Instanciamos o modelo para o fim de antecipar e detectar *botnets* e comparamos os resultados das instâncias extraídos de diferentes cenários. Para as análises, utilizamos as bases de dados CTU-13, CICDDoS2019, ISOT HTTP Botnet, CAIDA DDoS Attack 2007 e CSE-CIC-IDS2018 a fim de oferecer uma diversidade de cenários e situações para a avaliação. Os testes seguiram a mesma metodologia que compreende ações como: dividir as bases em treinamento, antecipação e teste; aplicar quatro algoritmos de aprendizagem de máquina; e comparar os resultados utilizando métricas como *precision*, *recall* e *F1-score*.

Os resultados demonstram que, em geral, os dispositivos maliciosos estão trafegando dados na rede antes mesmo do início do ataque, deste modo, é possível utilizar as técnicas de aprendizagem de máquina para identificar os dispositivos maliciosos antes que esteja concluída a coordenação dos *bots*, objetivando evitar o início de um ataque, ou ainda diminuir a probabilidade do mesmo obter sucesso em interromper o serviço. Por outro lado, também é notável que não houve uma instância específica do modelo que prevalecesse em todos os cenários avaliados, sendo que dependendo das características das janelas temporais os algoritmos tornavam-se mais precisos ou não. Por fim, foi possível verificar que a Regressão Logística é a técnica mais eficiente para janelas de tempo menores, enquanto o Random Forest e o Gradient Boosting necessitam de janelas maiores, o que leva a mais dados coletados, para separar corretamente os dispositivos.

O restante deste trabalho está organizado como segue. A Seção 2 apresenta esforços similares ao proposto neste trabalho. A Seção 3 detalha o sistema ANTE. A Seção 4 descreve a metodologia de avaliação e os resultados. Por fim, a Seção 5 conclui o trabalho e direciona os trabalhos futuros.

2. Trabalhos Relacionados

Diferentes trabalhos na literatura investigaram o problema de detectar dispositivos infectados e *botnets*. Esses trabalhos tratam desse problema tanto no nível de sistema,

quanto no nível de rede. As técnicas mais utilizadas para detecção de *botnets* seguem a classificação: detecção baseada em anomalias; detecção baseada em assinaturas de tráfego; grafos; aprendizagem de máquina supervisionada; e aprendizagem de máquina não supervisionada. As técnicas de detecção baseadas em anomalias têm por objetivo identificar irregularidades a partir da observação de características como portas utilizadas no tráfego de dados, alta latência de rede ou aumento do volume de tráfego.

As técnicas baseadas em assinaturas de tráfego abstraem o funcionamento dos *malwares* para identificar os dispositivos que estão sendo controlados. As técnicas baseadas em grafos geram modelos matemáticos que visam mostrar a relação entre diferentes objetos da rede para abstrair características ou comportamentos maliciosos com o intuito de identificar os *bots*. Além destas, existem outras estratégias para a identificação de *botnets* que são menos similares ao trabalho proposto. São exemplos de outras estratégias o uso da entropia para mapear o grau de aleatoriedade da rede e o uso de *botnets* benignas para interceptar a comunicação entre os dispositivos maliciosos. Esta seção foca nas técnicas de aprendizagem de máquina porque elas são capazes de aprender com grandes quantidades de dados sendo rápidas e assertivas em automatizar tarefas como a classificação de novas entradas em diferentes cenários, ou ainda estimar valores de variáveis contínuas. Além disso, devido à diversidade de técnicas de aprendizagem de máquina existente é possível que o problema da identificação antecipada de *botnet* possa ser resolvido com o uso dessas técnicas.

Com relação ao uso de técnicas de aprendizagem de máquina, o mais comum até então na literatura consistia em utilizar o poder de adaptação dessas técnicas para treinar modelos capazes de distinguir o fluxo de dados oriundo de usuários reais do fluxo originado por *bots* [Saad et al. 2011]. Mais recentemente essas técnicas começaram a ser aplicadas para antecipar ataques baseados nas ações dos *bots* [Abaid et al. 2016, Lu et al. 2017]. Nos casos em que existe o treinamento dos modelos, a aprendizagem de máquina é denominada supervisionada, pois utiliza um conjunto de dados rotulados para previamente sugerir ao modelo como seria a classificação de um novo registro. Existem diversos algoritmos de aprendizagem de máquina supervisionada, dentre eles as redes neurais e os métodos baseados em árvore de decisão têm sido utilizados com frequência na literatura [Bansal and Mahapatra 2017, Chen et al. 2018, Lu et al. 2017].

Uma outra abordagem visa agrupar dispositivos com fluxo de dados similares a fim de distinguir os dispositivos legítimos dos *bots* [Li et al. 2015, Wang et al. 2018]. Diferentemente dos algoritmos de aprendizado de máquina supervisionado, os algoritmos de clusterização não necessitam de treinamento antes de serem usados na diferenciação dos fluxos de dados, deste modo estes algoritmos são classificados como não supervisionados. Abaid et al. (2016) catalogaram diferentes estágios do comportamento das *botnets* e criaram um modelo de infecção para predizer ataques. Os autores citam que a principal limitação é a quantidade de estados mapeados, caso uma *botnet* não siga o mapeamento o modelo tende a não funcionar. Diferente do trabalho de Abaid et al. (2016), para o sistema ANTE proposto neste trabalho, o comportamento de infecção das *botnets* não influencia na identificação antecipada de *bots*. O trabalho de Lu et al. (2017) foca em identificar sessões de comando e controle de botnet (C&C), para isso um vetor de 55 características das sessões é analisado seguindo o algoritmo *Random Forest* com o intuito de identificar os C&C. A desvantagem dessa abordagem é que existem tipos de *botnet* que não centra-

lizam o comando dos ataques como a peer-to-peer (P2P). Deste modo, qualquer *botnet* que não utilize a arquitetura C&C tende a não ser identificada. O trabalho de Wang et al. (2018) utiliza uma abordagem similar, onde o objetivo é agrupar as sessões P2P dos dispositivos infectados. Diferentemente, para o sistema ANTE, a arquitetura da *botnet* não é um pré-requisito para que a antecipação seja feita com sucesso, ou seja, nosso sistema independe do tipo do ataque.

3. Aprendizado de Máquina para Identificação Antecipada de Botnets

Esta seção apresenta o sistema ANTE de identificação ANTecipada de *botnEts*. O sistema tem como base algoritmos de aprendizagem de máquina devido à sua flexibilidade e fácil adaptação. O modelo do sistema é instanciado considerando os principais algoritmos conhecidos na literatura, tal como os algoritmos KMeans [Haq and Singh 2018], Random Forest [Lu et al. 2017], Regressão Logística [Bapat et al. 2018] e Gradient Boosting [Indre and Lemnaru 2016]. Estes algoritmos foram escolhidos pois seguem a literatura e possuem características diferentes podendo variar seu desempenho nos diferentes cenários. A Figura 1 apresenta o posicionamento do ANTE em uma rede, bem como a relação entre os componentes que serão detalhados nas próximas subseções.

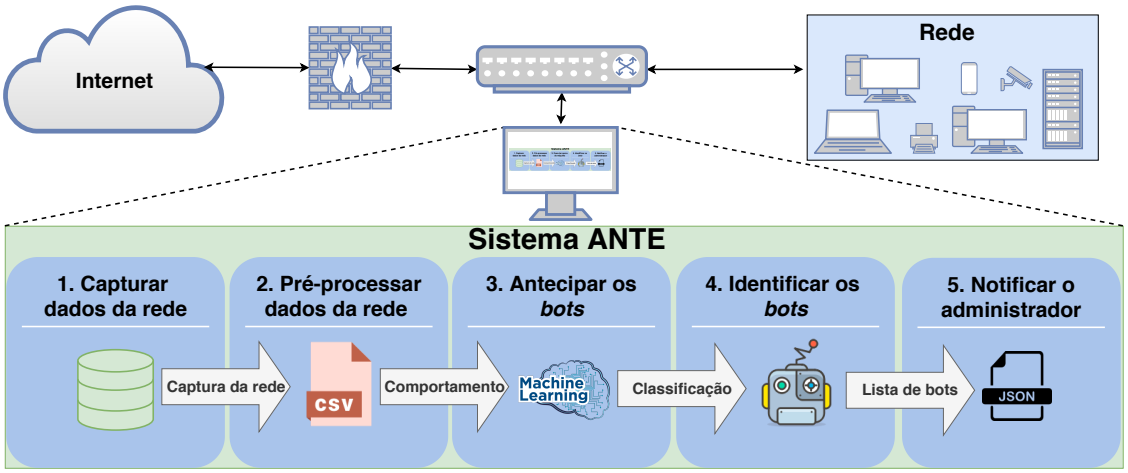

Figura 1. Arquitetura do sistema ANTE

3.1. O Sistema ANTE

O sistema ANTE analisa o tráfego da rede em busca de *bots* para notificar os administradores da rede antes que os efeitos e as consequências do ataque gerado pela *botnet* sejam irremediáveis, por exemplo, exaustão de um servidor devido a um ataque de negação de serviço. Esta seção apresenta as cinco etapas da arquitetura que do sistema proposto sendo elas: (i) captura e (ii) pré-processamento do tráfego de rede; (iii) antecipação e (iv) identificação de botnets com aprendizado de máquina; e (v) notificação do administrador.

3.1.1. Captura e Pré-processamento do Tráfego de Rede

Uma das premissas para que o sistema funcione é a necessidade da análise dos dados trafegados na rede. Assim, todo o tráfego da rede é espelhado para uma central de análise

interna. Essa central pode ser um máquina virtual ou física, responsável por receber e processar os dados extraindo as características e aplicar os modelos de aprendizagem de máquina. Em cenários em que a quantidade de informação trafegada é exorbitante, é possível configurar a central de análise para utilizar amostras do fluxo ao invés de processar todo o conjunto de dados. Essa ação não afetaria o sistema, pois mesmo na amostragem o comportamento anômalo dos *bots* seria diferente do comportamento normal.

Existem diferentes abordagens para a definição das características que podem ser coletadas do tráfego das redes. Por exemplo, Seo and Lee (2016) utilizam 14 atributos para representar o fluxo normal e malicioso em seu trabalho de detecção de ataques DDoS, enquanto Lu et al. (2017) utilizam 22 atributos para detecção de seções de C&C. Como o objetivo deste trabalho é identificar a diferença comportamental dos dispositivos, é necessário utilizar características que sejam representativas. A quantidade de pacotes enviados, o tamanho dos pacotes, a frequência de envio e recebimento dos pacotes são características importantes para diferenciar usuários reais dos *bots*. Neste trabalho, a seleção das características foi feita manualmente com base nos atributos que representam acessos rápidos e constantes ou a quantidade de tráfego enviado ou recebido acima do normal. As informações são gravadas na central de análise para serem utilizadas no treinamento e antecipação dos *bots*, depois são descartadas para liberar recursos. Apesar da literatura apresentar métodos como o de Osanaiye et al. (2016) para redução da quantidade de características diminuindo a redundância entre elas e, consequentemente, diminuindo a quantidade de dados, o escopo deste trabalho não contempla tal seleção de atributos.

A captura do tráfego da rede segue ciclos compostos por janelas de tempo. Cada ciclo consiste nas janelas de treinamento, de antecipação e de testes, como ilustrado na Figura 2. A cada ciclo de captura o sistema cria uma série temporal para cada uma das características da rede. A janela de treinamento tem por objetivo utilizar os dados capturados para criar os modelos que foram utilizados nas outras etapas. Na janela de antecipação uma série temporal é criada e servirá de entrada para a análise dos algoritmos de aprendizagem de máquina, ou seja, para que seja possível identificar os *bots* em suas fases iniciais. Nas janelas de teste, novas séries temporais são formadas a partir das coletas de tráfego para verificar se o modelo criado anteriormente é eficiente em identificar os dispositivos maliciosos durante o ataque.

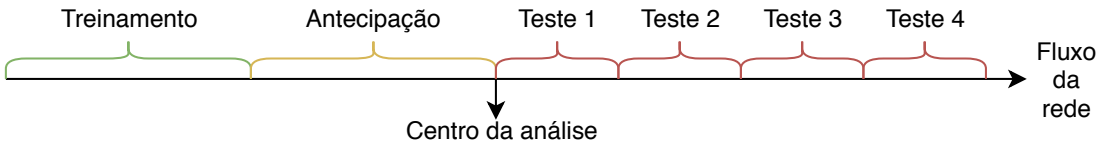

Figura 2. Ciclo do sistema de identificação antecipada de botnets

Após a coleta dos dados durante as janelas temporais é necessário aplicar um pré-processamento para extrair as características que serão utilizadas para diferenciar o fluxo normal do fluxo malicioso. O primeiro passo para tal ação é capturar o fluxo da rede e transformá-lo em uma entrada para o algoritmo de pré-processamento. Esta entrada pode ser um arquivo do tipo PCAP, arquivo do tipo CSV ou qualquer padrão para que seja possível extrair as características dos dispositivos da rede. Utilizam-se ferramentas como o TCPDump. Essa entrada é necessária para que todas as trocas de pacotes na rede

possam ser mapeadas e fiquem temporariamente disponíveis na central de análise. Com o fluxo da janela salvo é necessário extrair o comportamento dos dispositivos durante a janela temporal, ou seja, agrupar os envios e os recebimentos de pacotes durante o período analisado. Deste modo, é necessário que um algoritmo percorra todo o arquivo contando a quantidade de pacotes enviados, somando o tamanho dos pacotes e gerando as demais características definidas. Ao fim do pré-processamento tem-se o comportamento dos dispositivos mapeados. Com esses dados, é possível gerar um arquivo CSV, ou ainda enviá-los como *stream* de entrada para o algoritmo de antecipação de *botnets*.

3.1.2. Antecipação e Identificação de Botnets com Aprendizagem de Máquina

O sistema ANTE foi idealizado para ser flexível através da possibilidade de usar diferentes técnicas de aprendizagem de máquina tanto supervisionadas quanto não-supervisionadas. Particularmente, para que as técnicas de aprendizagem supervisionada funcionem é necessário que, durante a janela de treinamento, os dados estejam rotulados indicando se o dispositivo é confiável ou malicioso. Em geral a etapa para rotular os dados de treinamento é realizada manualmente analisando quais dispositivos são *bots*, podendo ser um processo complexo e custoso. Deste modo, os algoritmos desta classe verificam se as novas entradas possuem semelhança com os dados utilizados no treinamento e classificam os rótulos correspondentes. Existem diferentes modos para calcular a semelhança e classificar as novas entradas dependendo do algoritmo utilizado. Detalhes sobre os algoritmos de classificação são apresentados na Subseção 3.2.

Para que uma *botnet* seja identificada é necessário que um algoritmo de aprendizagem de máquina classifique um ou mais dispositivos como *bots*. A saída desta classificação retorna para o ANTE que verifica se algum dispositivo foi classificado como *bot*. Quando o ANTE identifica uma *botnet* ele deve produzir alguma ação com o intuito de evitar a interrupção dos serviços disponibilizados. A literatura aborda algumas opções como a limitação ou bloqueio do acesso do dispositivo podendo ser realizado manualmente ou automaticamente no *firewall*. Neste trabalho, a ação vislumbrada é a comunicação com os administradores da rede através de um e-mail com a listagem dos *bots*. Uma opção para a automatização da resposta a incidentes é o envio de uma mensagem do tipo JavaScript Object Notation (JSON) contendo os dados da ocorrência como a listagem de *bots* e a probabilidade do dispositivo ser *bot*. Ele arquivo pode servir como entrada para automatizar um firewall, um sistema de intrusão ou um sistema de alertas.

3.2. Algoritmos de Aprendizagem de Máquina

Algumas técnicas baseadas em aprendizagem não-supervisionada não necessitam de identificadores, assim esta classe de algoritmos cria agrupamentos dividindo os dispositivos em grupos por semelhança entre suas ações na rede. O algoritmo KMeans é um clusterizador que divide as instâncias similares em agrupamentos garantindo que a quantidade de grupos não ultrapasse o parâmetro K. Para criar os agrupamentos o KMeans distribui todas as instâncias que devem ser classificadas em um espaço com a quantidade de dimensões igual ao número de atributos. Após a distribuição o algoritmo define aleatoriamente K pontos que serão considerados os centros dos agrupamentos. O valor de K deve ser definido antes da execução do algoritmo, no caso deste trabalho o valor de K é 2. Este valor foi definido com o objetivo de criar um grupo com os dispositivos não-infectados e

outro com os *bots*, refletindo o comportamento o anômalo (exemplo: frequência de acesso diferente dos seres humanos) e o normal (exemplo: frequência normal para seres humanos). Após os centros serem definidos, as instâncias com menor Distância Euclidiana para cada centro são agrupadas formando os dois grupos. Após todas as instâncias serem agrupadas, um novo ponto central é calculado utilizando a média de todas as instâncias pertencentes ao agrupamento. Esse processo é repetido até que os pontos centrais não mudem. Porém, os grupos criados por este algoritmo não possuem classificação, ou seja, o resultado separa os dispositivos, mas não indica qual grupo é composto por *bots*. Além disso, a escolha de um valor diferente para o parâmetro K poderia criar resultados diferentes, como por exemplo separar duas famílias de *bots* dos dispositivos normais, porém essa análise não foi avaliada neste trabalho.

Os métodos baseados em árvores de decisão costumam ser assertivos e de rápido desempenho, deste modo o *Random Forest* e o *Gradient Boosting* foram escolhidos por utilizarem técnicas de combinação de árvores. O *Random Forest* constrói árvores utilizando amostras selecionadas aleatoriamente, enquanto o *Gradient Boosting* cria uma árvore de cada vez sempre com o objetivo de melhorar a anterior. O *Random Forest* é um algoritmo de aprendizado de máquina supervisionado que consiste de um conjunto de árvores de decisões criadas independentes usando uma sequência de reamostragens. Durante o treinamento um subconjunto das características iniciais é selecionado aleatoriamente usando a mesma distribuição probabilística. Cada subconjunto é então utilizado para criar uma árvore diferente. Para a classificação, a decisão final é obtida por um voto de maioria, onde cada árvore tem direito a um voto.

Regressão logística é amplamente popular para a predição de variáveis dicotômicas. A regressão logística foi utilizada, pois ela disponibiliza probabilidades para suas classificações, deste modo, em cenários reais, é possível propor um limiar para que somente as predições com alta probabilidade sejam concluídas, objetivando diminuir as taxas de erros. A regressão logística é similar à regressão linear, porém a regressão logística utiliza a *sigmoid* ou "*logistic*" como função de ativação. Deste modo a função que representa os dados não é uma reta e sim uma curva. Assim, a regressão logística classifica as novas observações estimando a probabilidade delas serem *bots* ou não.

4. Avaliação

O primeiro passo da avaliação do sistema ANTE foi definir as janelas de treinamento, antecipação e testes (representadas na Figura 2). Para isso, foram extraídos dois minutos imediatamente anteriores ao início da ação das *botnets*. Para as bases com ataques DDoS, o início do ataque foi definido como identificador das ações das *botnets*. Para as bases referentes a outros ataques, o pico de transferência de pacotes foi utilizado como ponto central da análise, por ser o momento mais propenso a causar danos na rede. Caso não exista um pico de transferência, o primeiro segundo após dois minutos do início da troca de pacotes foi definido como o ponto inicial. Os detalhes relativos às ações das *botnets* foram obtidos na documentação das bases. Os primeiros dois minutos foram divididos em duas partes iguais, o primeiro minuto foi nomeado de janela de treinamento, o segundo foi nomeado de janela de antecipação. Após o ponto central da análise, mais dois minutos foram utilizados para os testes de identificação durante o ataque. Esse período foi dividido em quatro janelas de teste com 30 segundos cada. Essas janelas foram definidas para que fosse possível analisar diferentes etapas do ciclo de vida das *botnets*.

Para todas as janelas temporais, foram gerados arquivos de extensão CSV contendo dados relacionados aos dispositivos e suas trocas de pacotes na rede representando o comportamento dos dispositivos agrupados pelo endereço IP (do Inglês, *Internet Protocol*), porém é possível utilizar outras combinações como o IP e endereço MAC (do Inglês, *Media Access Control*). Esse comportamento é composto por 20 atributos detalhados na Tabela 1. Esses atributos foram definidos utilizando o trabalho de Lu et al. (2017) como inspiração. Apesar da existente literatura em relação a seleção de atributos, tal ação não será conduzida neste trabalho, assim utilizaremos os mesmos atributos em todos os cenários avaliados. Portanto, cada linha do CSV representa um dispositivo diferente sendo identificado pelos atributos IP e rótulo (*isbot*). É oportuno ressaltar que todos os atributos presentes no CSV (Tabela 1) podem ser extraídos de todas as bases, garantindo a imparcialidade para condução dos testes propostos. Além disso, os campos IP e rótulo somente são utilizados para caracterizar o comportamento e não são usados como característica de fluxo. O identificador de dispositivo malicioso (rótulo) foi obtido nas documentações das respectivas bases e adicionadas aos arquivos CSV para garantir que a aprendizagem supervisionada possa ser treinada. Nos casos da aprendizagem não supervisionada esse rótulo não foi utilizado para a clusterização dos dados.

Atributo	**Descrição**
IP	Identificador do dispositivo
isBot	Variável binária indicando se o dispositivo é bot (0) ou não (1)
Tsr	Soma em bytes do tamanho dos pacotes recebidos
Tss	Soma em bytes do tamanho dos pacotes enviados
Rpc	Contagem de pacotes recebidos
Spc	Contagem de pacotes enviados
Savg	Média dos pacotes enviados
Smin	Tamanho em bytes do menor pacote enviado
Smax	Tamanho em bytes do maior pacote enviado
Svar	Variação em bytes dos pacotes enviados
Ravg	Média dos pacotes recebidos
Rmin	Tamanho em bytes do menor pacote recebido
Rmax	Tamanho em bytes do maior pacote recebido
Rvar	Variação em bytes dos pacotes recebidos
SITmin	Menor intervalo entre o envio dos pacotes
SITmax	Maior intervalo entre o envio dos pacotes
SITavg	Média do intervalo entre o envio dos pacotes
RITmin	Menor intervalo entre o recebimentos dos pacotes
RITmax	Maior intervalo entre o recebimento dos pacotes
RITavg	Média do intervalo entre o recebimento dos pacotes

Tabela 1. Atributos utilizados para a representação dos dispositivos

Para testar o sistema ANTE, foram utilizados 8 cenários diferentes descritos ao longo desta seção. Os cenários possuem fluxos oriundos de dispositivos maliciosos e não maliciosos e foram coletados por diferentes projetos de universidades ao redor do mundo. Os quatro algoritmos citados na Seção 3 foram testados em todos os cenários, porém apenas os melhores resultados foram reportados nesta seção, mas todos os resultados bem

como a extração de características estão disponíveis no repositório do projeto[1]. Os testes foram conduzidos em um *desktop* com um processador I5, SSD e 8 GB de memória *ram*.

A primeira métrica reportada é a acurácia que relaciona o total de acertos com o total de itens classificados. Porém em casos onde a distribuição das classes é desbalanceada esta métrica pode apresentar falso sentimento de bons resultados. Pois o classificador pode acertar todas as previsões para a classe majoritária e errar todas as previsões para a classe minoritária e mesmo assim possuir alta acurácia. Para a plena análise dos resultados dos testes, foram calculados os valores para as métricas *precision*, *recall* e o *F1-Score*. Essas métricas são baseadas na quantidade de verdadeiros positivos (TP), falsos positivos (FP), falsos negativos (FN), verdadeiros negativos (TN). Deste modo, é possível obter essas métricas para a classe dos não *bots* e dos *bots*. Porém, na apresentação dos resultados, foram expostas as médias das métricas para ambas as classes. Por exemplo, nos testes foi calculado a *precision* para a classe dos *bots* e dos não *bots* e nos resultados foi apresentada a média desses valores. Devido à variação dos TP, FP, FN e TN é necessário convencionar o uso deles neste trabalho, deste modo, a expressão "verdadeiros positivos" representa os elementos que foram classificados como não *bots* e realmente não eram *bots*. Os "falsos positivos" são os elementos que foram classificados com não *bot*, porém eram *bots*. Os "falsos negativos" correspondem aos elementos que foram classificados como *bots*, porém não eram *bots*. Os "verdadeiros negativos" são os elementos que eram *bots* e foram classificados como *bots*. As fórmulas das métricas estão apresentadas a seguir.

$$\textit{Acurácia} = \frac{TP + TN}{TP + FP + FN + TN}; Precision = \frac{TP}{TP + FP};$$

$$Recall = \frac{TP}{TP + FN}; F1 = 2.\ \frac{precision\ .\ recall}{precision + recall}$$

Cenário 1: O projeto Stratosphere[2] disponibiliza um conjunto de bases de dados denominado CTU-13 [García et al. 2014]. Esse conjunto de bases possui 13 capturas de tráfego de *botnets* gravados na Universidade Técnica Tcheca na República Tcheca. Utilizando parte dos dados do projeto, foi definido que o primeiro cenário do presente trabalho seria correspondente a captura 52 ou cenário 11 da CTU-13[3]. Essa base possui aproximadamente 4 GB de fluxo e 3 dispositivos infectados, porém um dos *bots* pouco atua durante o ataque. O ponto inicial do ataque ocorreu às 10:52:39 do dia 18 de agosto e foi obtido por meio da documentação da base. Este cenário foi treinado com a presença de dois *bots*, pois antes do início do ataque esse *bots* já estavam ativos enviando e/ou recebendo informações. Apesar da quantidade de fluxo capturado, as etapas de treinamento, antecipação e testes foram executadas rapidamente, deste modo a etapa mais onerosa foi a extração de características. Para o KMeans, o resultado foi ruim em todas as janelas. Para a janela antes do ataque o algoritmo que melhor identificou a presença de bots foi a Regressão Logística. O Gradient Boosting e o Random Forest apresentaram bons resultados no primeiro e no segundo teste após o início do ataque, quando existiam mais dispositivos na rede (9 e 12 dispositivos respectivamente). Porém quando a quantidade de dispositivos diminuiu no terceiro e no quarto período (7 e 3 dispositivos, respectivamente) eles passaram a errar mais que a Regressão logística. A Tabela 2 apresenta a acurácia e a média da *precision*, *recall* e *F1-score* para os principais resultados.

[1]https://github.com/andersonneira/sbrc-2020-neira-medeiros

[2]www.stratosphereips.org Acessado em: 23/03/2020

[3]https://mcfp.felk.cvut.cz/publicDatasets/CTU-Malware-Capture-Botnet-52 Acessado em: 23/03/2020

Algoritmo	Métricas	Antecipação	Teste 1	Teste 2	Teste 3	Teste 4
Gradient Boosting	**Acurácia**	75%	**89%**	**83%**	71%	67%
	Precision	65%	**93%**	**80%**	36%	33%
	Recall	61%	**83%**	**89%**	50%	50%
	F1-Score	62%	**86%**	**81%**	42%	40%
Regressão Logística	**Acurácia**	**92%**	78%	75%	**100%**	**100%**
	Precision	**88%**	88%	75%	**100%**	**100%**
	Recall	**94%**	67%	83%	**100%**	**100%**
	F1-Score	**90%**	68%	73%	**100%**	**100%**

Tabela 2. Principais resultados para o cenário 1.

Cenário 2: Como no cenário anterior este também faz uso de uma base do conjunto CTU-13. Neste caso foi escolhida a captura 51 ou cenário 10 da CTU-13[4]. Esta base é bem maior que a anterior visto que possui aproximadamente 66 GB de fluxo. Foram coordenados ataques no dia 18 agosto durante dois períodos, onde foram testados ataques com características diferentes. No primeiro período foram conduzidos os ataques SYN Flood e ACK DDoS e no segundo período foram testados ataques do tipo ICMP Flood. Deste modo, este cenário utiliza apenas os dados do primeiro período, sendo que a nossa análise teve início às 12:16:44 no horário de verão da Europa Central (CEST). Como era esperado, este cenário conta com mais dispositivos que o cenário anterior, deste modo o treinamento contou com dados de 33 dispositivos sendo 10 *bots*. O Gradient Boosting e o Random Forest identificaram todos os *bots* na janela anterior ao início do ataque e nas duas primeiras janelas do ataque. Porém, quando a quantidade de dispositivos diminuiu esses algoritmos foram incapazes de diferenciar os dispositivos normais dos *bots*. A Tabela 3 apresenta os principais resultados obtidos com a análise deste cenário.

Algoritmos	Métricas	Antecipação	Teste 1	Teste 2	Teste 3	Teste 4
Random Forest e *Gradient Boosting*	**Acurácia**	100%	100%	100%	9%	9%
	Precision	100%	100%	100%	5%	5%
	Recall	100%	100%	100%	50%	50%
	F1-Score	100%	100%	100%	9%	9%

Tabela 3. Principal resultado para o cenário 2.

Cenário 3: Este cenário também faz uso da captura 51 do conjunto CTU-13 mencionado no cenário 2. Porém os dados analisados fazem referência ao segundo período onde um ataque do tipo ICMP Flood foi conduzido. Este ataque foi iniciado no dia 18 agosto de 2011 às 14:43:27 CEST e conta com 10 dispositivos infectados. Durante o treinamento 30 dispositivos foram identificados, sendo que os 10 que são *bots* já estavam ativos. Os algoritmos não supervisionados apresentaram resultados ruins quando comparados com algoritmos supervisionados. Dos quatro algoritmos testados, a Regressão Logística e o Random Forest acertaram todas as classificações.

Cenário 4: A Universidade de New Brunswick (UNB) possui um projeto são disponibilizados conjuntos de dados que contém a presença de *bots*. Assim, este trabalho utiliza o conjunto CICDDoS2019[5] para definir o quarto cenário. Este conjunto foi es-

[4]https://mcfp.felk.cvut.cz/publicDatasets/CTU-Malware-Capture-Botnet-51 Acessado em: 23/03/2020
[5]www.unb.ca/cic/datasets/ddos-2019.html Acessado em: 23/03/2020

colhido por ter sido publicado no ano de 2019, possuir 20 GB e contar com diferentes tipos de ataques, coletados em dois dias [Sharafaldin et al. 2019]. Dos 12 ataques reportados nessa base, este cenário analisa o ataque *PortMap* que, segundo a documentação, ocorreu no primeiro dia de coleta durante o período de 9:43 a 9:51. Porém, durante a análise dos dados coletados não foi possível encontrar fluxo de rede no horário reportado. Então como o objetivo era analisar o primeiro ataque conduzido no primeiro dia, foi verificado que o primeiro pico de tráfego de pacotes ocorreu no dia 12 de janeiro às 14:36:10 Tempo Universal Coordenado (UTC), sendo esse o momento base para as análises deste cenário. As características deste cenário fazem dele um desafio para os algoritmos de aprendizagem de máquina, visto que, no treinamento, apenas um *bot* estava ativo entre os 36 dispositivos que compunham a rede. Essa diferença torna-se ainda maior ao longo dos testes quando a rede possui 523 dispositivos normais e um malicioso. A principal diferença está relacionada com a grande quantidade de informação originada por este dispositivo, sendo que no início do ataque, o *bot* enviou 6.135.105 pacotes a mais do que o segundo dispositivo mais ativo. Apesar disso todos os métodos encontraram dificuldades em identificar o *bot* neste contexto. Na janela de antecipação, o Random Forest e a Regressão Logística foram os que melhor conseguiram separar o *bot* do fluxo normal, gerando 1 e 2 falsos negativos respectivamente. Ainda na fase anterior ao início do ataque, o KMeans gerou dois agrupamentos, um com 8 dispositivos incluindo o *bot* e outro com 31 dispositivos. Considerando que todos deste agrupamento fossem classificados com *bots* o KMeans geraria 7 falsos negativos. Apesar desta alta taxa do erro, o KMeans foi o único capaz de separar o *bot* depois do início do ataque. A Tabela 4 foi construída considerando sempre o agrupamento que o *bot* estava inserido como todos sendo *bots*.

Algoritmo	Métricas	Antecipação	Teste 1	Teste 2	Teste 3	Teste 4
KMeans	**Acurácia**	82%	97%	99%	99%	99%
	Precision	56%	57%	60%	75%	62%
	Recall	91%	99%	100%	100%	99%
	F1-Score	56%	62%	66%	83%	70%

Tabela 4. Principal resultados para o cenário 4.

Cenário 5: Este cenário utiliza outro contexto do CICDDoS2019. O ataque selecionado para este cenário foi o NetBIOS sendo o segundo ataque conduzido no primeiro dia do experimento. Segundo a documentação este ataque ocorreu entre às 10:21 e 10:30 do primeiro dia. Porém analisando os dados disponibilizados não foi possível encontrar fluxo de rede neste horário, deste modo, o pico de tráfego ocorrido em 12 de janeiro às 15:06:10 UTC foi escolhido para ser o ponto base da análise. Algumas características do cenário anterior também são verificadas neste. Por exemplo, a presença de apenas um *bot* ativo, sendo que este dispositivo gera muito tráfego na rede e ele está presente antes do início do ataque. Contudo, este cenário possui menos dispositivos conectados na rede ao longo da análise chegando a um pico de 110 dispositivos. Essa mudança causou resultados diferentes da análise anterior, sendo que neste caso a Regressão logística e Gradient Boosting acertaram todas as classificações na janela de antecipação e nas janelas de testes.

Cenário 6: O cenário 6 é derivado do trabalho da Universidade de Victoria no Canadá e foi nomeado como ISOT HTTP Botnet Dataset[6]. Este cenário possui 780 MB

[6]www.uvic.ca/engineering/ece/isot/assets/docs/ISOT%20HTTP%20Botnet%20Dataset.pdf Acessado

de fluxo de rede e 9 dispositivos infectados [Alenazi et al. 2017]. O ponto central da análise foi definido sendo as 21:39:31 do dia 30 maio 2017, visto que, o objetivo do ataque era o roubo de informações. Este cenário foi treinado com a presença de 6 *bots* entre os 33 dispositivos que possuíam troca de pacotes antes do início do ataque. Em todas as janelas o Kmeans criou os agrupamentos com a maioria dos dispositivos normais junto com os *bots*, e os agrupamentos que possuíam poucos dispositivos não possuíam *bots*. O algoritmo que melhor classificou a janela de antecipação foi o Random Forest acertando todas as classificações. Após o início do ataque o Random Forest produziu apenas 2 falsos positivos no teste 2 e 1 falso positivo no teste 4, todos os resultados estão na Tabela 5.

Algoritmo	Métricas	Antecipação	Teste 1	Teste 2	Teste 3	Teste 4
Random Forest	**Acurácia**	100%	100%	95%	100%	97%
	Precision	100%	100%	97%	100%	98%
	Recall	100%	100%	86%	100%	94%
	F1-Score	100%	100%	90%	100%	96%

Tabela 5. Principal resultados para o cenário 6.

Cenário 7: Este cenário foi definido com base em uma captura frequentemente utilizada na literatura, a base de dados denominada DDoS Attack 2007 disponibilizada pela CAIDA[7]. Este cenário é diferente dos demais, visto que, apenas o fluxo malicioso e as respostas do servidor foram disponibilizados. Isso significa que, tirando o servidor, todos os outros dispositivos são *bots*. O ponto central da análise é dia 4 de agosto de 2007 às 21:14:00. O treinamento contou com 12 dispositivos, sendo 11 *bots* e a vítima, porém ao longo dos testes foi possível verificar um crescimento na quantidade de dispositivos trocando informações, chegando em um pico de 1286 dispositivos na última janela de teste. O KMeans e o Random Forest conseguiram separar o servidor dos dispositivos infectados, tanto na janela de antecipação quanto nas janelas de teste.

Cenário 8: O último cenário escolhido é baseado em um projeto disponibilizado pela UNB e é denominado CSE-CIC-IDS2018. Este cenário conta com sete ataques diferentes e com mais de 200 GB de fluxo de dados. Entre os ataques tem-se a ação de *bots* para quebrar senhas utilizando força bruta, *bots* para roubar dados sensíveis e ataques de negação de serviço. O ataque escolhido foi o DDOS-HOIC conduzido no dia 21/02/2018 sendo que o ponto central da análise ocorreu às 14:09:00. Apesar da existência de dez *bots*, este foi o único cenário que não contou com a presença de *bots* no treinamento. Provavelmente esse fator influenciou no ruim resultado dos algoritmos de aprendizagem de máquina supervisionado, visto que, nenhum algoritmo foi capaz de classificar corretamente um *bot* sequer. O algoritmo KMeans conseguiu separar parte do fluxo malicioso do normal. Utilizando o menor grupo como aquele dos *bots* foi possível observar falsos positivos e falsos negativos, porém, exceto na janela de antecipação, o algoritmo classifica mais corretamente os *bots* do que erra, como é possível verificar na Tabela 6.

em: 23/03/2020

[7]Tivemos acesso a esta base devido à parceria entre a Universidade Federal do Paraná (UFPR)/Brasil e a Universidade Cernegie Mellon (CMU)/EUA. URL: `https://www.caida.org/data/passive/ddos-20070804_dataset.xml` Acessado em: 23/03/2020

Algoritmo	Métricas	Antecipação	Teste 1	Teste 2	Teste 3	Teste 4
KMeans	**Acurácia**	99,85%	99,89%	99,89%	99,93%	99,93%
	Precision	70%	89%	88%	92%	92%
	Recall	83%	94%	100%	100%	100%
	F1-Score	75%	91%	93%	95%	95%

Tabela 6. Principais resultados para o cenário 8.

5. Conclusão

Diante do potencial destrutivo das *botnets* e da dificuldade em detectar seus componentes, este artigo apresentou o sistema ANTE, cujo objetivo é identificar antecipadamente as *botnets*. Uma constatação é que, em geral, os *bots* estão presentes e já começaram a troca de pacotes antes do ataque acontecer. Essa observação é importante, pois foi possível mostrar que utilizando a aprendizagem de máquina em diferentes cenários, o sistema ANTE identificou as *botnets* antes que o ataque seja iniciado visando o bloqueio do consumo desnecessário de recursos e evitando que o serviço seja comprometido. Também verificamos a influência das características dos cenários na assertividade dos modelos, porém mais estudos são necessários para compreender plenamente esse fenômeno. Em ambientes reais, cujo fluxo de dados é contínuo e potencialmente infinito, dificilmente os dois minutos que antecedem um ataque serão utilizados para o treinamento do modelo, visto que esse período pode conter a ação dos primeiros *bots* podendo melhorar o desempenho do sistema. Outra limitação está relacionada ao consumo de recursos para a extração de características. Este trabalho utiliza 20 atributos para treinar e classificar os dispositivos.Em cenários reais, a extração de todas as características pode ser lenta e prejudicar a classificação *online*, pois durante os experimentos conduzidos, a etapa mais onerosa em todos os cenários foi a extração de características. Isso ocorreu pois os arquivos possuíam diversos gigas de informação. Uma vez com as características extraídas o processo de detecção foi instantâneo em todos os casos. Para contornar os limitantes os próximos passos irão focar na seleção de atributos, aprendizado incremental, técnicas que não usem de dados rotulados e teste com diferentes formas de janelamento.

Referências

Abaid, Z., Sarkar, D., Kaafar, M. A., and Jha, S. (2016). The early bird gets the botnet: A markov chain based early warning system for botnet attacks. In *IEEE on LCN*, pages 61–68.

Alenazi, A., Traore, I., Ganame, K., and Woungang, I. (2017). Holistic model for http botnet detection based on DNS traffic analysis. In Traore, I., Woungang, I., and Awad, A., editors, *ISDDC*, pages 1–18. SIP.

Bansal, A. and Mahapatra, S. (2017). A comparative analysis of machine learning techniques for botnet detection. In *SINCONF*, pages 91–98, New York, NY, USA. ACM.

Bapat, R., Mandya, A., Liu, X., Abraham, B., Brown, D. E., Kang, H., and Veeraraghavan, M. (2018). Identifying malicious botnet traffic using logistic regression. In *SIEDS*, pages 266–271.

Chang, W., Mohaisen, A., Wang, A., and Chen, S. (2018). Understanding adversarial strategies from bot recruitment to scheduling. In Lin, X., Ghorbani, A., Ren, K., Zhu, S., and Zhang, A., editors, *SecureComm*, pages 397–417, Cham. SIP.

Chen, S., Chen, Y., and Tzeng, W. (2018). Effective botnet detection through neural networks on convolutional features. In *IEEE TrustCom*, pages 372–378.

Cisco, V. N. I. (2019). Global mobile data traffic forecast update, 2017-2022. *https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-738429.pdf*.

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

Gupta, B. and Badve, O. P. (2017). Taxonomy of DoS and DDoS attacks and desirable defense mechanism in a cloud computing environment. *Neural Comp.and Appl.*, 28(12):3655–3682.

Haq, S. and Singh, Y. (2018). Botnet detection using machine learning. In *International Conference on Parallel, Distributed and Grid Computing*, pages 240–245.

Indre, I. and Lemnaru, C. (2016). Detection and prevention system against cyber attacks and botnet malware for information systems and Internet of Things. In *IEEE ICCP*, pages 175–182.

Karim, A., Salleh, R. B., Shiraz, M., Shah, S. A. A., Awan, I., and Anuar, N. B. (2014). Botnet detection techniques: review, future trends, and issues. *Journal of Zhejiang University SCIENCE C*, 15(11):943–983.

Khanchi, S., Vahdat, A., Heywood, M. I., and Zincir-Heywood, A. N. (2018). On botnet detection with genetic programming under streaming data, label budgets and class imbalance. In *GECCO*, pages 21–22, New York, NY, USA.

Li, S.-H., Kao, Y.-C., Zhang, Z.-C., Chuang, Y.-P., and Yen, D. C. (2015). A network behavior-based botnet detection mechanism using PSO and K-means. *ACM Trans. Manage. Inf. Syst.*, 6(1):3:1–3:30.

Lu, L., Feng, Y., and Sakurai, K. (2017). C&c session detection using random forest. In *IMCOM*, pages 34:1–34:6, New York, NY, USA. ACM.

Mane, Y. D. (2017). Detect and deactivate p2p zeus bot. In *ICCCNT*, pages 1–7.

Nordrum, A. (2016). Popular internet of things forecast of 50 billion devices by 2020 is outdated (2016). *https://spectrum.ieee.org/tech-talk/telecom/internet/popular-internet-of-things-forecast-of-50-billion-devices-by-2020-is-outdated[Acesso em: 29/7/2019]*.

Osanaiye, O., Cai, H., Choo, K.-K. R., Dehghantanha, A., Xu, Z., and Dlodlo, M. (2016). Ensemble-based multi-filter feature selection method for ddos detection in cloud computing. *JWCN*, 2016(1):130.

Saad, S., Traore, I., Ghorbani, A., Sayed, B., Zhao, D., Lu, W., Felix, J., and Hakimian, P. (2011). Detecting P2P botnets through network behavior analysis and machine learning. In *PST*, pages 174–180.

Seo, J. W. and Lee, S. J. (2016). A study on efficient detection of network-based ip spoofing ddos and malware-infected systems. *SpringerPlus*, 5(1):1878.

Sharafaldin, I., Lashkari, A. H., Hakak, S., and Ghorbani, A. A. (2019). Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy. In *ICCST*, pages 1–8.

Wang, C.-Y., Ou, C.-L., Zhang, Y.-E., Cho, F.-M., Chen, P.-H., Chang, J.-B., and Shieh, C.-K. (2018). Botcluster: A session-based P2P botnet clustering system on netflow. *Computer Networks*, 145:175 – 189.