Detecção de *Botnets* em Dispositivos IoTs baseado em *LSTM Autoencoder*

Caio Maciel[1], **Anderson B. de Neira**[2], **Ligia F. Borges**[1], **Michele Nogueira**[1,2]

[1]Departamento de Ciência da Computação - Universidade Federal de Minas Gerais

[2]Departamento de Informática - Universidade Federal do Paraná

caioalex2018@ufmg.br, abneira@inf.ufpr.br

{ligia.borges,michele}@dcc.ufmg.br

Resumo. *Com a ascensão da Internet das Coisas, atores maliciosos exploram brechas para a criação de redes infectadas por softwares (a.k.a, botnet). As botnets geram uma série de ameaças à segurança dos serviços. Soluções existentes especializam-se em cenários de ataques. Isto diminui a eficácia em cenários sem ataque, aumentando o número de detecções incorretas, gerando mais custos computacionais e retrabalho. Para resolver isso, este trabalho emprega o LSTM Autoencoder em conjunto com a técnica de voto majoritário. A proposta melhora o desempenho em cenários com e sem ataques de botnets, diminuindo o número de falsos positivos e evitando custos. Resultados preliminares indicam uma acurácia de 99,42% na detecção de botnets, superando a atual literatura.*

Abstract. *In the face of the increasing adoption of Internet of Things (IoT), malicious actors exploit system loopholes to create software-infected networks (botnets). Botnets generate a series of service threats to security. Existing solutions specialize in attack scenarios. This decreases effectiveness in non-attack scenarios, increasing the number of incorrect detections, and generates more computational costs and rework. To deal with this problem, this work employs the LSTM Autoencoder combined with the majority vote technique. The proposal improves performance in scenarios with and without botnet attacks, reducing the number of false positives and avoiding costs. Preliminary results indicate an accuracy of 99.42% in detecting botnets, surpassing the current literature.*

1. Introdução

Os dispositivos da Internet das Coisas (em inglês, *Internet of Things* — IoT) disponibilizam uma ampla gama de automações, como sistemas de segurança doméstica, iluminação inteligente, controle climático entre outros. Entretanto, esses dispositivos trazem também ameaças à segurança do usuário. A exploração de vulnerabilidades por atores maliciosos leva ao vazamento de dados e violação da privacidade. Exemplos de ataques de rede comuns no meio IoT, tratados neste trabalho, são provenientes das *botnets* Mirai e Bashlite (ou Gafgyt). Uma *botnet* consiste de uma rede de dispositivos infectados coordenados por *botmasters* capazes de gerar ataques. Um dos ataques gerados por *botnets* é o ataque de Negação de Serviço Distribuído (em inglês, *Distributed Denial of Service* — DDoS) [Jyoti and Behal 2021], cujo objetivo é causar interrupção de serviços por horas/dias e prejuízos financeiros que sejam significativos para a empresa alvo.

Devido à característica heterogênea de um ambiente IoT (*i.e.*, dado a falta de padronização de arquiteturas e protocolos), criar um modelo de segurança específico para cada dispositivo IoT da rede torna-se custoso e inviável. Este trabalho aplica modelos de aprendizado de máquina para a detecção dos ataques atuando na camada de rede [Meidan et al. 2018]. Assim, a ausência de padronização não representa um problema. Em Cunha *et al.* (2022), os autores utilizam um modelo de *Variational Autoencoder* para detecção de ataques de *botnets* em IoT. A solução detecta ataques em tempo real, contudo apresenta alto número de falso positivo (FP) (*i.e.,* identificação incorreta de *bots*) ocasionando retrabalho por parte da equipe de segurança e maior custo computacional.

Para diminuir os FPs e evitar custos adicionais, este artigo propõe uma abordagem baseada na rede neural *Long Short-Term Memory Autoencoder* (LSTM-AE). A rede LSTM-AE detecta o tráfego de rede provenientes de *botnets* através do treinamento com informações do tráfego normal. Após o treinamento, a LSTM-AE comprime e reconstrói o tráfego de rede sob um custo de reconstrução. O custo de reconstrução é calculado usando do Erro Médio Absoluto (*i.e.*, média dos erros absolutos tomados instância a instância). Quando a LSTM-AE recebe um tráfego de *botnet*, o valor de reconstrução é alto devido ao padrão incomum em relação ao treinamento. Nos casos em que o valor de reconstrução ultrapassa um limiar predefinido, a abordagem identifica que o tráfego de rede foi gerado por um membro da *botnet*. Para maior precisão a LSTM-AE é combinada com a estratégia de votação por maioria, pois reduz a ocorrência de FPs [Meidan et al. 2018].

A abordagem foi avaliada em dois experimentos. O primeiro detecta *botnets* em um cenário onde o tráfego de dispositivos infectados está combinado com o tráfego regular. O segundo experimento avalia a proposta onde há somente tráfego regular. A base de dados utilizada nos experimentos é a N-BaIoT [Meidan et al. 2018]. A base de dados contém tráfego de rede de dispositivos infectados com as famílias de *botnets* Mirai e Bashlite (malígno) e tráfego regular. Os resultados obtidos indicam uma acurácia de 99,91% no cenário sob ataque e uma taxa de 0,57% de FPs no cenário regular, superando o trabalho Cunha *et al.* (2022) que atingiu 99,63% de acurácia no primeiro cenário e o trabalho de Meidan *et al.* (2018) que obteve 0,70% de taxa de FPs no segundo cenário.

2. Trabalhos Relacionados

Existem trabalhos na literatura com o foco de detectar *botnets* em redes IoT. Baruah et al. (2023) combinam algoritmos de aprendizado de máquina supervisionados para fazer a detecção de *botnets*. Os autores obtiveram 99,99% na detecção de *botnets* que usam a arquitetura *peer-to-peer*. Apesar da acurácia próxima a 100%, os autores necessitaram de dados rotulados para a detecção. Além disso, caso as *botnets* não usem a arquitetura *peer-to-peer* a solução pode não funcionar. Em Mirsky *et al.* (2018), os pesquisadores construíram uma solução de detecção de intrusão baseada em um conjunto de *Autoencoders*. Apesar da baixa taxa de FPs na detecção de intrusão, a solução proposta não foca em *botnets* e pode ficar instável quando a rede defendida sofre um ataque baseado na inundação de pacotes com endereços falsificados. O trabalho de Wang *et al.* (2020) apresenta uma abordagem baseada na utilização de uma Rede Neural com *Autoencoder*, para detectar ataques de injeção falsa de dados. O problema tratado pelos autores é complexo pois a solução deveria operar em infraestruturas críticas (*smart grid*). Apesar da solução apresentar uma acurácia acima de 90%, a taxa de erros para os dados de ataques é de 6,4%, longe do ideal para o contexto em que a solução deve operar. Para evoluir a

literatura diminuindo o uso de dados rotulados e aumentar as classificações corretas, este trabalho propõe uma abordagem de detecção de *botnets* baseada na rede LSTM-AE.

3. Aprendizado Profundo para a Detecção de *Botnets*

Esta seção detalha a abordagem proposta para a detecção de *botnets* baseada em aprendizado profundo semi-supervisionado usando a LSTM-AE. A LSTM-AE reúne as características de uma rede LSTM com a arquitetura codificador-decodificador de um *autoencoder*. As redes LSTM são um tipo especial de *Recurrent Neural Networks* [Hochreiter and Schmidhuber 1997]. A LSTM lida com dados de entrada sequenciais. O diferencial das redes LSTMs é que elas possuem uma memória interna capaz de conservar informação, armazenando o estado das células para uso futuro. Elas conseguem controlar quando as informações entram, quando podem ser utilizadas ou apagadas das células. O *autoencoder* é uma rede neural baseada na estrutura codificador-decodificador para gerar representações de uma entrada. O *autoencoder* aplica representações para reconstruir os novos dados que serão avaliados. O treinamento de um *autoencoder* inicia com valores aleatórios para pesos da rede e são atualizados iterativamente através do algoritmo de *backpropagation*. O resultado da saída da rede no decodificador é o dado reconstruído pelo *autoencoder*, o qual simula os dados inseridos no codificador.

A LSTM-AE reconstrói dados temporais do tráfego de rede através do aprendizado de dados sequenciais, o qual recorre às células das camadas da rede LSTM. A LSTM-AE atua na desconstrução e reconstrução dos dados de entrada, aprendendo as informações existentes sobre eles. Essa abordagem possui uma dinâmica de modelo generativo, diferentemente dos modelos discriminativos. Assim, busca-se predizer uma saída correta olhando apenas para a natureza dos dados de entrada. Ou seja, a LSTM-AE visa simular como os dados de entrada de seu treinamento são construídos. Portanto, a LSTM-AE pode auxiliar na detecção de anomalias na rede como as *botnets* [e Silva et al. 2022].

A LSTM-AE foi escolhida pois não depende de dados rotulados para realizar a detecção das *botnets*. Isso simplifica a adoção da solução em ambientes reais e reduz custos com a obtenção de rótulos para o treinamento da proposta. Ao usar a LSTM-AE, a abordagem proposta lida com a heterogeneidade em *IoT*. Pois a solução proposta usa uma única rede LSTM-AE para analisar o tráfego de rede e detectar as *botnets*. Isso economiza tempo de treinamento e recursos computacionais, ao evitar o uso de várias redes LSTM-AE. Assim, a abordagem foi projetada para não depender de dados rotulados e proporcionar a detecção *botnets* em diferentes cenários.

Para realizar a detecção das *botnets* a abordagem coleta o tráfego de rede e aplica um pré-processamento sobre ele. A coleta do tráfego de rede pode ser realizada quando o roteador ou um *firewall* envia uma cópia dos pacotes para o dispositivo que executa a abordagem proposta. Observa-se em outros trabalhos da literatura que algoritmos de aprendizado de máquina tendem a atingir, em média, melhor desempenho quando as variáveis de entrada são escaladas (pré-processadas). Por este motivo, a abordagem proposta aplica a normalização do tráfego de rede. Esta normalização acontece ao aplicar um escalador máximo-mínimo sobre as informações de entrada, com o intuito de proporcionar uma melhor performance ao modelo limitando os dados para um intervalo padrão (Equação 1). O termo $X(j)$ indica o valor de cada amostra do atributo analisado, o min_A e o max_A são respectivamente o menor e o maior valor observado para o atributo. O min'_A e

o max'_A representam o novo intervalo.

$$máximo - m\acute{i}nimo = \frac{X(j) - min_A}{max_A - min_A}(max'_A - min'_A) + min'_A \quad (1)$$

Após normalização do tráfego de rede, a solução realiza o treinamento da rede LSTM-AE e começa a detectar as *botnets*. A rede LSTM-AE analisa a entrada e gera uma representação dos dados processados em outro espaço latente, para depois utilizá-los no processo de reconstrução do tráfego original, obtendo um erro de reconstrução na saída da rede. Ao aplicar a rede LSTM-AE em um novo conjunto de dados semelhante ao do treinamento como entrada, o valor do erro seria baixo. Por outro lado, caso o erro fosse maior que um limiar pré-estabelecido, poderia se afirmar que o dado passado para a rede difere do dado no qual a rede LSTM-AE foi treinada. Dessa forma, o valor do erro obtido na saída da rede pode ser utilizado para identificar ataques de *botnets*, pois ao treinar a rede LSTM-AE apenas com dados benignos, ao receber um tráfego malicioso, ela aponta um erro de reconstrução maior, podendo-se assinalar o tráfego como anômalo.

A abordagem proposta usa a LSTM-AE para recriar o tráfego de entrada na saída da abordagem. Ao comparar os valores reais do tráfego de rede com os valores preditos, a abordagem proposta identifica o erro de reconstrução. Este trabalho usa o Erro Médio Absoluto (em inglês, *Mean Absolute Error* - MAE) para calcular o custo de reconstrução. O MAE é a média dos erros absolutos $|e_i| = |y_i| - |x_i|$, onde y_i é o valor reconstruído e x_i o valor original, para cada ponto do vetor de reconstrução. Assim, quando a abordagem proposta recebe um tráfego malicioso (oriundo de *botnet*), o valor de reconstrução é alto por ser um padrão incomum em relação ao tráfego regular de treinamento. Com isso, foi definido um limiar *L* responsável por separar o tráfego regular do tráfego anormal. Visando ser uma abordagem flexível para as equipes de segurança, este trabalho não predefiniu o valor do limiar. Assim, as equipes de segurança podem optar por um limiar brando, onde a abordagem proposta irá detectar *botnets* menos ativas podendo gerar mais FPs ou usar um limiar mais rígido implicando em detecção de *botnets* que estão mais ativas podendo gerar menos FPs.

Para evitar a geração de FPs, este trabalho utilizou a estratégia de voto majoritário através de uma janela deslizante [Meidan et al. 2018]. A técnica combina as decisões do modelo dentro de uma janela que percorre o vetor de reconstrução partindo da premissa de que em um cenário anômalo a votação da maioria na janela indicaria o ataque. No caso de instâncias regulares elas receberiam poucos votos de anomalia. Esta abordagem auxilia assim na redução de FPs e aumenta a confiabilidade da detecção de tráfego malicioso.

4. Avaliação

Esta seção apresenta avaliações referentes à abordagem proposta. A abordagem foi avaliada em dois experimentos, os quais buscam simular as condições e a metodologia proposta por [Meidan et al. 2018]. O primeiro experimento foca na detecção de *botnets* em um cenário onde o tráfego de dispositivos infectados está junto com o tráfego regular (dispositivos não infectados). O segundo experimento avalia o modelo em um cenário onde não há tráfego de *botnets*, apenas tráfego regular. O objetivo dessa decisão foi avaliar um cenário de um ambiente com ações de *botnets* e um cenário regular de um ambiente *IoT*.

Ambos os experimentos avaliaram a base de dados N-BaIoT, disponibilizada em [Meidan et al. 2018]. Esta base é composta por informações do tráfego de rede de

nove dispositivos *IoT* diferentes. A metodologia aplicada pelos autores da base foi a de coletar os pacotes de tráfego gerados pelos dispositivos observados. Os pacotes possuem atributos como endereço de origem e destino, porta, protocolo, tamanho total entre outros. Os autores extraíram 115 características estatísticas, como a média e variância das informações, que foram agrupadas por intervalos de tempo diferentes. Dentre as diferentes formas de que os autores agruparam os dados, optou-se pelos dados de entrada agrupados no intervalo de 100 *ms*. Com essa escolha, o espaço da base foi reduzido para 23 características, o que proporcionou a realização de diversas iterações do modelo em um tempo menor. Por fim, em ambos os experimentos foram utilizados 70% dos dados benignos para o treinamento, sendo 10% deles destinados para o conjunto de validação.

A LSTM-AE usada nos dois experimentos possui o Adam como otimizador e o Erro Absoluto Médio como função de perda. A arquitetura da rede é composta por cinco camadas. As duas primeiras camadas, L1 e L2, representam o codificador, capazes de criar uma representação dos dados de entrada. Ademais, L1 e L2 contém 16 e 4 neurônios, respectivamente. A camada de vetor de repetição (L3) distribui o vetor de representação nas etapas de tempo do decodificador. Este é formado por outras duas camadas LSTM (L4 e L5), com 4 e 16 neurônios, respectivamente. Entre uma camada e a próxima, aplica-se na saída dos neurônios a função de ativação Unidade Linear Retificada (em inglês, *Rectified Linear Unit — ReLU*) na forma ($f(x) = x^+ = max(0, x)$). A camada de saída recebe as informações do decodificador, retornando os dados de entrada reconstruídos. Por fim, a arquitetura proposta foi treinada com um tamanho de *batch* igual a 32 e um total de 100 épocas. A configuração usada neste trabalho foi obtida empiricamente. Trabalhos futuros irão pesquisar métodos para automatizar esse processo.

Outro aspecto importante é a definição do valor do limiar *L* e do tamanho da janela deslizante. Este trabalho verificou de forma empírica que o valor de quantil 95 do vetor erro de reconstrução do treino maximiza a separação do tráfego regular e malicioso. Na fase final da detecção foi aplicada a técnica de voto majoritário, através de uma janela deslizante de tamanho 96. Este foi o valor ótimo dos testes, pois, caso fosse superior, a classificação final era penalizada devido ao tamanho maior de janela.

No primeiro experimento, foi utilizado os 30% restantes de dados benignos e todo o tráfego das *botnets*. Assim, foi simulado um cenário onde o ambiente *IoT* estivesse inundado de tráfego malicioso, com 2,5% de tráfego benigno e o restante dados de ataque. O limiar *L* com valor 0,091 foi utilizado para classificar a saída da reconstrução como regular ou maliciosa. O modelo LSTM-AE obteve neste caso, após a aplicação da janela deslizante, 99,91% de acurácia, 98,66% de precisão e um *recall* de 99,66% para a classe de ataque. As métricas obtidas revelam que o modelo foi capaz de detectar o tráfego de rede malicioso em um cenário onde quase toda a rede já estava inundada de tráfego oriundo de *botnets*. Assim, a arquitetura proposta conseguiu reproduzir os dados regulares e apontar um erro de reconstrução maior quando se deparou com os dados maliciosos.

O segundo experimento usou um conjunto de teste formado apenas por dados regulares (sem dados de ataque). O objetivo foi analisar o comportamento da solução somente com dados normais, já que uma solução satisfatória deveria gerar a menor quantidade de FPs, evitando custos adicionais. Ademais, buscou-se testar o modelo apenas com dados benignos diferentes daqueles utilizados no treinamento, avaliando assim o comportamento em relação à taxa de FPs gerados pelo sistema. A proposta obteve uma

taxa de 0,05% de FPs e 99,42% de acurácia usando a técnica de janela deslizante na última fase da classificação. Como nesta situação não havia falso negativo (FN), optou-se por analisar somente as métricas citadas já que o teste continha apenas os rótulos de uma das classes. O modelo mostrou-se, mais uma vez, robusto à variação no tráfego de entrada e capaz de distinguir com alta precisão os dados na rede oriundos de *botnets*.

Considerando trabalhos que utilizam o mesmo conjunto de dados, os resultados obtidos neste ultrapassam os apresentados por [Meidan et al. 2018, Cunha et al. 2022]. A melhoria das métricas, especialmente a acurácia, podem ser pouco representativas. Porém em cenários de rede real contendo tráfego massivo, esta diferença é significativa. Considerando a taxa de erro (FPs + FNs) da proposta, este trabalho errou 5.504 instâncias no primeiro experimento. Enquanto que o modelo apresentado no trabalho Cunha *et al.* (2022) identificou incorretamente 24.367 instâncias no mesmo cenário. Assim, a solução proposta diminui a geração de alertas incorretos auxiliando o trabalho das equipes de segurança. Este trabalho avaliou o tráfego malicioso de duas *botnets* (Mirai e Bashlite) originado por dispositivos IoT. Assim, o resultado apresentado limita-se a esse contexto.

5. Considerações Finais e Trabalhos Futuros

Este trabalho usou uma estratégia de aprendizado profundo para detectar *botnets* através de informações de tráfego em dispositivos *IoTs*. O modelo LSTM-AE foi implementado de forma semi-supervisionada, diminuindo o custo de rotular o tráfego de entrada. Conforme resultados obtidos, foi possível estender a utilização do modelo para outros cenários e superar a literatura. Trabalhos futuros avaliarão outras bases de dados com arquiteturas e *botnets* diferentes e compararão o custo computacional da proposta com a literatura.

Agradecimentos

Este trabalho foi financiado pelo CNPq (#309129/2017-6 e #432204/2018-0) pela FAPESP (#2018/23098-0 e #2022/06840-0) e pela CAPES (#88887.501287/2020-00).

Referências

Baruah, S., Borah, D. J., and Deka, V. (2023). Detection of peer-to-peer botnet using machine learning techniques and ensemble learning algorithm. *IJISP*, 17(1):1–16.

Cunha, A. A., Borges, J. B., and Loureiro, A. A. (2022). Detecção de ataques de botnets em IoT via variational autoencoder. In *CoUrb*, pages 238–251. SBC.

e Silva, G. M., Neira, A., and Nogueira, M. (2022). Aprendizado profundo para a predição de ataques de negação de serviço distribuído. In *SBRC*, pages 475–488, Brasil. SBC.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Jyoti, N. and Behal, S. (2021). A meta-evaluation of machine learning techniques for detection of ddos attacks. In *INDIACom*, pages 522–526. IEEE.

Meidan, Y., Bohadana, M., Mathov, Y., Mirsky, Y., Shabtai, A., Breitenbacher, D., and Elovici, Y. (2018). N-baiot—network-based detection of IoT botnet attacks using deep autoencoders. *IEEE Pervasive Computing*, 17(3):12–22.

Mirsky, Y., Doitshman, T., Elovici, Y., and Shabtai, A. (2018). Kitsune: an ensemble of autoencoders for online network intrusion detection. *arXiv preprint arXiv:1802.09089*.

Wang, C., Tindemans, S., Pan, K., and Palensky, P. (2020). Detection of false data injection attacks using the autoencoder approach. In *PMAPS*, pages 1–6. IEEE.