DL-SAFE: Proteção Baseada em Aprendizado Profundo para Detecção de Botnets na Borda

Lucas Chagas de Brito Guimarães[1], **Rodrigo de Souza Couto**[1]

[1]Grupo de Teleinformática e Automação (GTA/PEE/COPPE)
Universidade Federal do Rio de Janeiro (UFRJ)

Abstract. *IoT (Internet of Things) devices are fundamental to multiple sectors, such as smart homes, cities, and grids. However, the existence of billions of devices with limited computing power makes them ideal targets for botnets. This paper proposes DL-SAFE, a tool for real-time traffic classification in edge environments using Open Argus and Pytorch. The tool also allows the evaluation of neural network architectures using 3 types of layers. The results demonstrate the tools' effectiveness, obtaining precision and recall values greater than 99% for multiple models. With the throughput tests it can be seen that performance varies greatly according to the architecture, with half the evaluated models processing more than 1000 network flows per second.*

Resumo. *Dispositivos de Internet das Coisas (IoT) são fundamentais para setores como casas, cidades e redes inteligentes. Entretanto, a existência de bilhões de dispositivos com poder computacional limitado os torna alvos para botnets. Este artigo propõe DL-SAFE, uma ferramenta para classificação de tráfego em ambientes de borda usando Open Argus e Pytorch. A ferramenta também permite a avaliação de arquiteturas de redes neurais usando 3 tipos de camadas. Os resultados demonstram a eficácia da ferramenta, obtendo precisão e sensibilidade superior a 99% para diversos modelos. Os teste de vazão apontam que o desempenho varia de acordo com a arquitetura, com metade dos modelos avaliados processando mais de 1000 fluxos por segundo.*

1. Introdução

A Internet das coisas (*Internet of Things* - IoT) refere-se a um paradigma marcado pelo número crescente de dispositivos interconectados capazes de serem gerenciados remotamente, muitas vezes equipados com processadores leves [Koroniotis et al. 2019]. A IoT tem se tornado cada vez mais essencial, sendo adotada em setores como energia, água, transporte, saúde e habitação, entre outros. O relatório anual da Internet da Cisco prevê que 14,7 bilhões de dispositivos IoT estarão conectados à Internet até 2023 [Cisco 2018].

Apesar de seus inúmeros casos de uso, o paradigma IoT apresenta desafios em relação à segurança do dispositivo. Muitos dispositivos IoT sofrem de vulnerabilidades como autenticação inadequada, portas abertas desnecessariamente e controle de acesso insuficiente [Neshenko et al. 2019]. A proliferação de dispositivos vulneráveis levou ao

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior Brasil (CAPES) - Código de Financiamento 001. O trabalho também foi financiado pelo CNPq, FAPERJ (E-26/010.002174/2019 e E-26/201.300/2021), PR2/UFRJ, e pela Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), auxílio no. 2015/24494-8.

surgimento de botnets capazes de executar poderosos ataques de negação de serviço distribuído (*Distributed Denial of Service* - DDoS), como visto nos ataques feitos contra Yandex e Microsoft em 2021 [Catalin Cimpanu 2021, Alicia Hope 2021]. Essas *botnets* tiram proveito de vulnerabilidades para se propagar, permitindo assim variados tipos de ataques à rede.

Para proteger dispositivos IoT, práticas simples de segurança como alterar senhas padrão e manter dispositivos atualizados podem ser implementadas. No entanto, essas medidas servem apenas como uma primeira linha de defesa. Soluções avançadas são necessárias para proteger dispositivos contra ataques sofisticados e detectar quando um dispositivo é comprometido. Os sistemas de detecção de intrusão (*Intrusion Detection Systems* - IDSs) são uma dessas soluções, e podem ser categorizados como baseados em hospedeiro ou em rede. Os sistemas baseados em hospedeiro operam diretamente nos dispositivos, contando com a análise de *logs* para detectar atividades suspeitas. Como os dispositivos IoT geralmente têm memória e recursos de processamento limitados, a implementação de IDSs baseados em hospedeiro geralmente é evitada, pois pode interferir no desempenho dos dispositivos; assim, IDSs para ambientes IoT tendem a ser baseados em rede.

IDSs podem utilizar modelos de classificação derivados de algoritmos de aprendizado de máquina supervisionados para minimizar a intervenção humana e acelerar a detecção de ataques. Esse processo envolve a seleção de um conjunto de dados rotulado contendo tanto tráfego regular quanto de ataques a serem detectados; também é necessário realizar o ajuste dos hiperparâmetros do algoritmo para otimizar seu desempenho.

Este artigo propõe e implementa DL-SAFE[1]: Proteção Baseada em Aprendizado Profundo para Detecção de Botnets na Borda, um IDS para análise e classificação de tráfego em tempo real em ambientes de borda. O conjunto de dados BoT-IoT [Koroniotis et al. 2019] é selecionado para construir os modelos de classificação, dado que é recente e contém tráfego de rede IoT rotulado com ataques de *botnet*. A ferramenta Open Argus é usada para converter, em tempo real, o tráfego de rede em fluxos, a biblioteca Pandas para extrair características relevantes dos fluxos de rede convertidos, e a biblioteca Pytorch para realizar a classificação dos fluxos. Além disso, a ferramenta também permite a construção e o teste de arquiteturas de redes neurais usando 3 tipos de camadas: perceptron multicamadas (*Multilayer Perceptron* - MLP), rede neural recorrente (*Recurrent Neural Network* - RNN) e memória longa de curto prazo (*Long Short-Term Memory* - LSTM). O treinamento inclui o ajuste de hiperparâmetros usando o método da busca em grade, e a avaliação considera o impacto da quantização pós-treinamento (*Post-Training Quantization* - PTQ) de 8 bits no desempenho do modelo resultante. Os resultados demonstram a eficácia dos modelos, com sete das oito arquiteturas avaliadas alcançando valores de precisão e sensibilidade superiores a 95% para todas as classes. Com os testes de vazão pode-se observar que, embora o desempenho dos modelos varie de acordo com a arquitetura implementada, a maioria dos modelos são capazes de suportar o uso médio de rede de um dispositivo IoT, que varia de 10 a 3.000 pacotes por segundo [Mainuddin et al. 2021].

O restante deste artigo está organizado da seguinte forma. A Seção 2 apresenta

[1]Deep Learning-based SAFeguard for Edge botnet detection

outros trabalhos com foco na detecção de *botnets* usando aprendizado de máquina. A Seção 3 descreve a arquitetura da ferramenta proposta, assim como os experimentos e os resultados obtidos. Por fim, a Seção 4 apresenta as considerações finais dos autores, descreve a demonstração no salão de ferramentas assim como os equipamentos necessários, e indica onde o código e documentação da ferramenta podem ser acessados.

2. Trabalhos Relacionados

A detecção de ataques de *botnet* usando aprendizado de máquina é uma questão já trabalhada por alguns autores. No entanto, embora muitos desses trabalhos incluam artigos e *surveys* que avaliam o desempenho da classificação de modelos em conjuntos de dados tanto novos quanto já disponíveis, existem poucas ferramentas visando realizar essa classificação em tempo real.

Por exemplo, Ferrag *et al.* conduz uma *survey* de sistemas de detecção de intrusão centrada em técnicas de aprendizagem profunda [Ferrag et al. 2020]. Ferrag e Maglaras também propõem um novo *framework* de troca de energia para redes inteligentes chamado DeepCoin, baseado em corrente de blocos e aprendizado profundo [Ferrag and Maglaras 2019]. Alkandi *et al.* propõem uma estrutura de IDS colaborativa baseada em corrente de blocos para proteger a privacidade em ambientes de nuvem, garantindo um processo seguro de troca de dados [Alkadi et al. 2020]. Popoola *et al.* propõem uma abordagem de aprendizado profundo federado para detectar ataques de *botnet* de dia zero, priorizando a privacidade e a segurança dos dados de tráfego de rede em dispositivos IoT [Popoola et al. 2021b]. Popoola *et al.* também propõem SMOTE-DRNN, projetado especificamente para detecção de *botnets* em ambientes IoT, com forte ênfase no tratamento de dados altamente desbalanceados [Popoola et al. 2021a]. Saurabh *et al.* propõem o LBDMIDS, um IDS baseado em rede que emprega dois tipos de LSTM para treinar modelos preditivos [Saurabh et al. 2022]. Sualihah *et al.* propõem um IDS projetado para detectar ataques em ambientes IoT, empregando uma arquitetura baseada em LSTM em conjunto com camadas totalmente conectadas [Jan et al. 2022].

Quanto aos trabalhos que implementam a detecção de *botnets* em tempo real, Shao *et al.* propõem uma estratégia para detectar botnets usando aprendizado adaptativo online (*online adaptive learning*) e aprendizado por agrupamento online (*online ensemble learning*) [Shao et al. 2021]. O treinamento é implementado usando 2 algoritmos: a árvore adaptativa de Hoeffding e a floresta aleatória adaptativa. Por se tratar de treinamento adaptativo, um aspecto central de seu trabalho é como lidar com a deriva de conceito (*concept drift*) decorrente de mudanças nos padrões de tráfego da rede IoT. Velasco-Mata *et al.* realiza detecção de *botnets* usando aprendizado de máquina em redes de alta velocidade [Velasco-Mata et al. 2023]. O trabalho emprega uma árvore de decisão e usa um conjunto de quatro características simples acoplados a uma janela de tempo de um segundo, com o objetivo de otimizar o desempenho da proposta. Há um foco na identificação dos requisitos de *hardware* necessários para que a proposta funcione em ambientes com variados requisitos de rede.

Ao contrário das propostas de classificação em tempo real mencionadas acima, este trabalho foca na aplicação de métodos de aprendizado profundo para a detecção de ataques. Como múltiplos trabalhos relacionados propõem arquiteturas de redes neurais para modelos de classificação, este trabalho implementa essas arquiteturas a fim de ve-

rificar seu desempenho quando implementadas para o mesmo propósito em uma única aplicação. Essas arquiteturas estão listadas na Tabela 1; a tabela descreve cada arquitetura pela sequência de camadas que a compõem, com o número de neurônios presentes em cada camada apresentado entre parênteses. Assim, além de detectar botnets em redes IoT, este trabalho também permite a comparação de diferentes propostas de arquiteturas de redes neurais.

Tabela 1. Arquiteturas de redes neurais usadas nos experimentos, incluindo a ordem e o tipo de camadas, assim como o número de neurônios por camada.

Nome	Arquitetura
MLP1	MLP (100) $\rightarrow$ MLP (100) $\rightarrow$ MLP (100) $\rightarrow$ *Softmax*
MLP2	MLP (100) $\rightarrow$ MLP (100) $\rightarrow$ MLP (100) $\rightarrow$ MLP (100) $\rightarrow$ *Softmax*
RNN1	RNN (60) $\rightarrow$ *Softmax*
RNN2	RNN (100) $\rightarrow$ *Softmax*
RNND	RNN (100) $\rightarrow$ MLP (100) $\rightarrow$ MLP (100) $\rightarrow$ MLP (100) $\rightarrow$ *Softmax*
LSTM1	LSTM (32) $\rightarrow$ LSTM (32) $\rightarrow$ *Softmax*
LSTMD	LSTM (128) $\rightarrow$ LSTM (128) $\rightarrow$ MLP (32) $\rightarrow$ MLP (10) $\rightarrow$ *Softmax*
BLSTM1	BLSTM (12) $\rightarrow$ *Softmax*

3. DL-SAFE

Esta seção descreve a arquitetura do IDS proposto, bem como os resultados obtidos em termos de precisão, sensibilidade e capacidade de processamento em fluxos por segundo.

3.1. Arquitetura Proposta

A ferramenta consiste em 4 módulos: o módulo de treinamento de modelos, o módulo de captura de dados, o módulo de tratamento de dados e o módulo de processamento de dados. A Figura 1 ilustra esses módulos.

O módulo de treinamento de modelos é responsável por criar os modelos de classificação utilizados pelo módulo de processamento de dados. O processo de treinamento inicia limpando o conjunto de dados BoT-IoT com a remoção de dados incompletos ou nulos. Os dados são então divididos em dois conjuntos de dados: um conjunto de treinamento, contendo 70% dos dados, e um conjunto de teste contendo os 30% restantes. O conjunto de treinamento é usado para obter os modelos de classificação; este treinamento usa validação cruzada k-fold, com um valor k igual a 5, e é realizado usando o PyTorch em conjunto com o Ray Tune. As arquiteturas apresentadas na Tabela 1 são usadas como um dos hiperparâmetros durante o treinamento, enquanto os outros são definidos usando a grade de hiperparâmetros do Ray Tune. Após a obtenção dos modelos, o desempenho de classificação é avaliado por meio do módulo de avaliação offline, utilizando o conjunto de testes. Dessa forma, é possível obter resultados de acurácia, precisão, sensibilidade, F1-Score e perda. Diferentemente dos outros módulos, o módulo de treinamento de modelos não precisa estar ativo durante a classificação de tráfego em tempo real; assim, é possível treinar os modelos com antecedência antes de utilizar a ferramenta para classificar dados em tempo real.

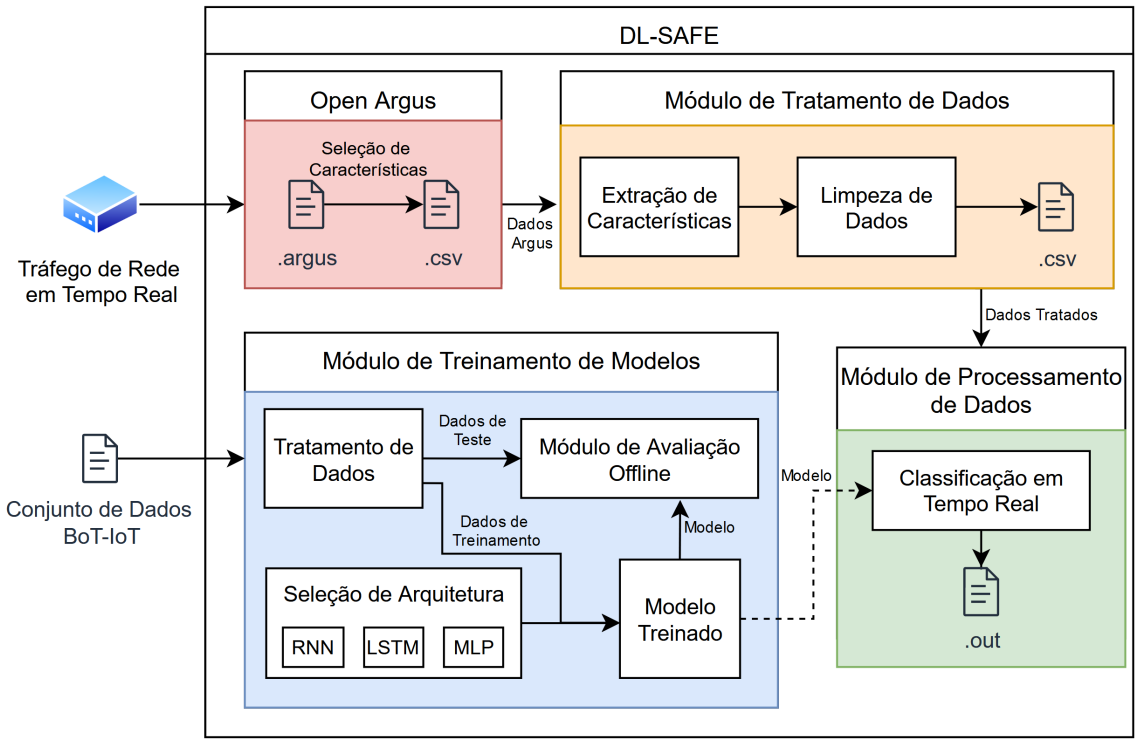

Figura 1. Arquitetura da ferramenta, apresentando os quatro módulos.

O módulo de captura de dados é implementado através do Open Argus, a mesma ferramenta utilizada na criação do conjunto de dados BoT-IoT. Open Argus lê o tráfego de rede e converte os dados em um arquivo ARGUS. Utilizando métodos do próprio Open Argus, um documento CSV é extraído do arquivo ARGUS contendo os fluxos e suas respectivas características em um formato legível pelo módulo de tratamento de dados. O módulo de captura de dados agrupa os dados de rede usando uma janela de tempo configurável, gerando por padrão um arquivo CSV a cada cinco segundos.

O módulo de processamento de dados lê o arquivo CSV gerado pelo módulo de captura de dados, extrai características adicionais para que o conjunto final possua as mesmas características utilizadas durante o treinamento, e realiza a limpeza de dados utilizando o mesmo método implementado no módulo de treinamento de modelos. Este módulo aguarda até receber um arquivo CSV, realiza o tratamento desses dados e envia o arquivo CSV resultante para o módulo de processamento de dados.

Finalmente, o módulo de processamento de dados recebe os dados tratados no formato CSV, executa o modelo de classificação selecionado usando a biblioteca PyTorch e salva os resultados da classificação para análise posterior do usuário. O usuário pode empregar o modelo de classificação em seu formato regular ou, alternativamente, usar o formato de quantização pós-treinamento (PTQ) de 8 bits para obter melhor vazão em troca de um desempenho de classificação ligeiramente inferior.

3.2. Análise de Desempenho

Para avaliar o desempenho dos modelos o DL-SAFE obtém múltiplas métricas, como acurácia, precisão, sensibilidade, F1-Score e perda durante o treinamento e o teste. A Figura 2 apresenta os resultados de precisão e sensibilidade obtidos para cada classe após

o ajuste de hiperparâmetros, considerando todas as arquiteturas de redes neurais avaliadas. Uma escala de cores foi utilizada para representar o valor de cada célula, na qual a cor tende para o vermelho para valores baixos e para o verde para valores altos.

		MLP1	MLP2	RNN1	RNN2	RNND	LSTM1	LSTMD	BLSTM1
Normal	Precisão	99.52	99.76	99.98	99.94	99.7	99.99	99.98	99.99
	Sensibilidade	100	99.92	100	100	100	100	100	100
DoS-TCP	Precisão	99.98	99.81	98.85	98.6	94.2	99.67	98.52	99.98
	Sensibilidade	99.98	99.98	99.59	99.67	97.48	99.8	99.08	99.99
DoS-UDP	Precisão	99.91	99.97	100	99.96	99.9	99.99	99.94	100
	Sensibilidade	100	100	100	100	98.67	99.98	99.46	100
DoS-HTTP	Precisão	99.97	99.99	96.04	96.01	92.72	99.98	99.18	99.99
	Sensibilidade	99.89	99.84	98.92	99.75	81.44	99.99	99.08	100
DDoS-TCP	Precisão	100	100	99.58	99.67	97.37	99.8	99.06	99.99
	Sensibilidade	99.96	99.88	98.8	98.56	93.53	99.67	98.46	99.97
DDoS-UDP	Precisão	100	100	100	100	98.71	99.98	99.47	100
	Sensibilidade	100	100	100	99.96	99.93	99.99	99.94	100
DDoS-HTTP	Precisão	99.98	99.95	98.98	99.84	83.55	99.99	99.09	99.99
	Sensibilidade	100	100	95.97	95.88	93.98	99.98	99.23	100
Keylogging	Precisão	100	100	100	100	99.98	100	99.98	100
	Sensibilidade	100	100	100	100	100	100	100	100
Data Exfiltration	Precisão	100	100	99.97	99.97	100	100	99.99	100
	Sensibilidade	100	100	100	100	99.99	100	99.98	100
OS Fingerprinting	Precisão	99.99	100	99.95	99.97	99.87	100	100	100
	Sensibilidade	99.97	99.99	100	100	99.95	100	99.99	100
Service Scan	Precisão	100	99.92	100	100	100	100	100	100
	Sensibilidade	99.53	99.77	99.98	99.99	99.73	99.99	99.98	99.99

Figura 2. Precisão e sensibilidade para cada arquitetura de rede neural após o ajuste de hiperparâmetros, considerando cada classe do conjunto de dados.

É possível observar que os modelos treinados possuem alta precisão e sensibilidade, sendo os modelos baseados em RNN os únicos que apresentam desempenho abaixo de 98% entre as classes avaliadas. Os resultados também mostram que os modelos têm maior dificuldade em identificar ataques de negação de serviço, com todas as outras classes de ataque superando 99,87% de precisão.

Todos os experimentos de vazão são realizados com modelos ajustados, cujos resultados em métricas de classificação estão apresentados na Figura 2. A Tabela 2 apresenta a vazão de processamento de cada modelo, obtido pela divisão do número de amostras do conjunto de teste pelo tempo que cada modelo leva para classificar todas as amostras. Os experimentos são realizados em um NVIDIA Jetson Nano Developer Kit, de modo a avaliar o desempenho dos modelos de classificação ao operarem em um dispositivo de borda, e consideram um intervalo de confiança de 95%.

Conforme observado na Tabela 2, a vazão dos modelos otimizados se encontra entre 232,60 a 1.423,84 fluxos por segundo. O uso de rede de dispositivos IoT varia de acordo com o tipo de dispositivo, com fluxos TCP de dispositivos IoT enviando em média de 10 a 3.000 pacotes por segundo [Mainuddin et al. 2021]. Como os fluxos de rede extraídos pelo Open Argus comumente contêm centenas de pacotes por fluxo, uma taxa de processamento de mais de 1000 fluxos por segundo é suficiente para lidar com os requisitos padrões de rede de um dispositivos IoT. Para modelos com menor capacidade de processamento e para ambientes com maiores requisitos de tráfego, é necessário avaliar o desempenho de cada modelo para garantir que ele se adéque aos requisitos do ambiente.

Tabela 2. Vazão dos modelos otimizados em fluxos processados por segundo.

Arquitetura	Vazão (fluxos/segundo)
MLP1	1.423,84 $\pm$ 6,56
MLP2	1.236,54 $\pm$ 3,81
RNN1	1.413,88 $\pm$ 6,73
RNN2	1.338,63 $\pm$ 11,11
RNND	943,86 $\pm$ 6,33
LSTM1	742,02 $\pm$ 28,62
LSTMD	232,60 $\pm$ 12,19
BLSTM1	793,72 $\pm$ 36,97

4. Conclusão, Demonstração e Código Fonte

Este artigo propõe e implementa DL-SAFE, um IDS para classificação de tráfego em tempo real em ambientes de borda construídos usando Open Argus e Pytorch. Além de classificar o tráfego em tempo real, a ferramenta também permite construir e testar arquiteturas de redes neurais usando 3 tipos de camadas. A partir dos resultados obtidos foi possível observar que, com o ajuste de hiperparâmetros, sete dos oito modelos avaliados obtêm mais de 95% de precisão e sensibilidade. Os resultados de vazão também demonstram que a maioria dos modelos é capaz de lidar com os requisitos de tráfego de rede dos dispositivos IoT.

A versão da ferramenta que será utilizada para demonstração roda em uma máquina virtual para facilitar o processo de instalação. A demonstração exigirá um computador pessoal com processador Core i7 de sexta geração ou superior, no mínimo 8 GB de RAM e HDD com mais de 10 GB disponíveis, além de monitor e teclado. A demonstração da ferramenta classificará o tráfego em tempo real, usando uma máquina com o sistema operacional Kali Linux para simular ataques à rede. Os resultados da classificação apresentarão, se um ataque for detectado, os identificadores de fluxo (endereço IP de origem e destino, portas de origem e destino, e protocolo) e o tipo de ataque. Também será demonstrado o módulo de avaliação offline, usando o conjunto de dados BoT-IoT para apresentar o processo de treinamento e teste dos modelos usados para classificação em tempo real. Essa avaliação incluirá como informações a acurácia, precisão, sensibilidade, F1-Score e perda, bem como o tempo necessário para treinar o modelo e testar suas versões quantizada e regular.

A ferramenta, assim como sua documentação, licença e código, estão disponíveis em: https://github.com/GTA-UFRJ-team/neuralnetwork-IoT. A documentação da ferramenta inclui um guia detalhando o processo de instalação, assim como um manual sobre como configurar e executar o DL-SAFE.

Referências

Alicia Hope (2021). Russian Internet Giant Yandex Wards off the Largest Botnet DDoS Attack in History. available at: *https://www.cpomagazine.com/cyber-security/russian-internet-giant-yandex-wards-off-the-largest-botnet-ddos-attack-in-history/*.

Alkadi, O., Moustafa, N., Turnbull, B., and Choo, K.-K. R. (2020). A deep blockchain framework-enabled collaborative intrusion detection for protecting iot and cloud networks. *IEEE Internet of Things Journal*, 8(12):9463–9472.

Catalin Cimpanu (2021). Microsoft said it mitigated a 2.4 Tbps DDoS attack. available at: *https://therecord.media/microsoft-said-it-mitigated-a-2-4-tbps-ddos-attack-the-largest-ever/*.

Cisco (2018). Cisco Annual Internet Report (2018–2023). available at: *https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html*.

Ferrag, M. A. and Maglaras, L. (2019). Deepcoin: A novel deep learning and blockchain-based energy exchange framework for smart grids. *IEEE Transactions on Engineering Management*, 67(4):1285–1297.

Ferrag, M. A., Maglaras, L., Moschoyiannis, S., and Janicke, H. (2020). Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study. *Journal of Information Security and Applications*, 50:102419.

Jan, S., Masoodi, F., and Bamhdi, A. (2022). Effective intrusion detection in iot environment: Deep learning approach. In *SCRS Conference Proceedings on Intelligent Systems*, pages 495–502.

Koroniotis, N., Moustafa, N., Sitnikova, E., and Turnbull, B. (2019). Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset. *Future Generation Computer Systems*, 100:779–796.

Mainuddin, M., Duan, Z., and Dong, Y. (2021). Network traffic characteristics of iot devices in smart homes. In *International Conference on Computer Communications and Networks (ICCCN)*, pages 1–11.

Neshenko, N., Bou-Harb, E., Crichigno, J., Kaddoum, G., and Ghani, N. (2019). Demystifying iot security: An exhaustive survey on iot vulnerabilities and a first empirical look on internet-scale iot exploitations. *IEEE Communications Surveys & Tutorials*, 21(3):2702–2733.

Popoola, S. I., Adebisi, B., Ande, R., Hammoudeh, M., Anoh, K., and Atayero, A. A. (2021a). smote-drnn: A deep learning algorithm for botnet detection in the internet-of-things networks. *Sensors*, 21(9):2985.

Popoola, S. I., Ande, R., Adebisi, B., Gui, G., Hammoudeh, M., and Jogunola, O. (2021b). Federated deep learning for zero-day botnet attack detection in iot-edge devices. *IEEE Internet of Things Journal*, 9(5):3930–3944.

Saurabh, K., Sood, S., Kumar, P. A., Singh, U., Vyas, R., Vyas, O., and Khondoker, R. (2022). LBDMIDS: LSTM based deep learning model for intrusion detection systems for iot networks. In *IEEE World AI IoT Congress (AIIoT)*, pages 753–759.

Shao, Z., Yuan, S., and Wang, Y. (2021). Adaptive online learning for iot botnet detection. *Information Sciences*, 574:84–95.

Velasco-Mata, J., González-Castro, V., Fidalgo, E., and Alegre, E. (2023). Real-time botnet detection on large network bandwidths using machine learning. *Scientific Reports*, 13(1):4282.