A Deep Learning-based System for DDoS Attack Anticipation

Gabriel Lucas F. M. e Silva[*], **Anderson Bergamini de Neira**[†], **Michele Nogueira**[*†]
[*]Department of Computer Science - Federal University of Minas Gerais, Brazil
[†]Department of Informatics - Federal University of Paraná, Brazil
Emails: {gabriel.silva, michele}@dcc.ufmg.br, abneira@inf.ufpr.br

2022 IEEE Latin-American Conference on Communications (LATINCOM) | 978-1-6654-8225-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/LATINCOM56090.2022.10000427

***Abstract*—Among various threats in the Cyberspace, Distributed Denial of Service (DDoS) attacks stand out for interrupting essential services, denying access to legitimate users, and causing financial losses. The literature presents defense mechanisms against DDoS attacks. However, there are limitations in these mechanisms' response time to prevent damage effectively because of their late detection. Hence, this work presents a system based on unsupervised deep learning to identify signals of an attack in its preparation phase. Identifying early signals of an attack aims to increase the security administrator's time to stop it. The system extracts early signals from the network traffic and processes them through a deep neural network. Performance evaluation employs as input the CTU-13 dataset that contains the traffic of two different DDoS attacks. The system had anticipated the launch of the attack 46 minutes before it effectively started.**

***Index Terms*—Security Management, DDoS attack prediction, Machine Learning.**

I. Introduction

The Internet and the constant development of networking technologies allow thousands of people worldwide to connect and generate massive data volume. Services and applications offer innovative businesses and increasingly play a vital role in our personal and professional lives. However, the growing importance of services increases the incentive for criminals to attack them, aiming at illicit personal gain. One of the most severe threats in Cyberspace is the Distributed Denial of Service (DDoS) attack [1]. DDoS attacks disrupt services; nowadays, attackers even sell them as a service on the Internet, increasing their disruptive potential.

Due to DDoS attacks being sudden threats and rapidly reaching a massive volume of traffic against their targets, defense mechanisms have a short time to detect and react when they are in advanced stages. DDoS attacks take many compromised devices connected to the Internet and use them to send massive messages, connection requests, or malformed packets coordinated against their targets. The attack aims to consume the victim's computing resources, reducing performance and denying services to legitimate users [2]. Defense mechanisms need to identify an attack as soon as possible, for instance, in the preparation phase, to take actions to reduce damages.

This work was supported by National Council for Scientific and Technological Development (CNPq/Brazil), grants #309129/2017-6 and #432204/2018-0, by São Paulo Research Foundation (FAPESP), grant #2018/23098-0, by the Coordination for the Improvement of Higher Education Personnel (CAPES/Brazil), grant #88887.509309/2020-00.

Different strategies against DDoS attacks exist, such as prediction, detection, and mitigation [3]. Deep learning is prominent for creating cybersecurity solutions to combat DDoS attacks. For instance, attack detection solutions have applied deep learning techniques [4]. Furthermore, some works operate on predicting DDoS attacks; they complement the other defense mechanisms and increase the time victims would have to react to the attack [5], [6]. These last papers argued that deep learning could significantly contribute to predicting DDoS attacks. However, the literature presents only a few deep learning-based studies to predict DDoS attacks.

This work presents a system based on deep learning to anticipate signals of DDoS attacks. Unlike the literature on DDoS attack detection, this work evolves the literature by predicting DDoS attacks allowing more time to defend against attacks. The system searches for early signals of DDoS attacks on the victim's network traffic, helping to anticipate changes in network traffic. This system is the first in the literature that uses a trained artificial neural network based on early signals computed through statistical metrics and attributes extracted from network traffic. After training, the neural network compresses and reconstructs the signals under a small reconstruction cost. The system uses this cost's value to define a threshold and identify early signals; this is possible when the network receives input data that follows a different distribution. Then, it tries to reconstruct them. Therefore, the signals of attack preparation indicate its imminent launch.

The system evaluation uses the CTU-13 dataset from the Czech Technical University [7]. This scenario contains network traffic captured from a simulated attack. The system extracts the size in bytes, the time elapsed between the previous and the current packet during Transmission Control Protocol (TCP) communication, and the TCP communication window size. The system calculates each feature's skewness and kurtosis measures for a one-second traffic window to construct early signals. The system uses the constructed signals to train and test a Long Short-Term Memory (LSTM) Autoencoder neural network [8] to predict DDoS attacks. The prediction anticipates the attack 46 minutes before its launch, reaching an accuracy of 91%.

This work proceeds as follows. Section II presents works related to the prediction of DDoS attacks. Section III details the proposed system. Section IV presents the evaluation of the system. Finally, Section V concludes this work.

II. Related Works

The literature presents several methods for DDoS attack detection. The study of [9] proposes a study for detecting DDoS attacks in software-defined networks. The solution uses flow features to define entropy as new packets reach. The solution detects attacks in progress when the calculated entropy surpasses a threshold. The solution showed a 98.2% of detection rate. In [10], the authors developed a study for detecting DDoS attacks on smart homes. The authors recognized that Smart Homes contain four device profiles determined by the difficulty of detecting attacks from these devices. The solution uses these different profiles to train a specialized boosting method of logistic model trees for each profile. The results reached an accuracy of 99.99% in the tests.

Despite the diversity of detection solutions, the literature on DDoS attack prediction is limited. The study of [5] aims to identify and model the typical behavior of botnets in a Markov chain. Thus, the authors presented a methodology for predicting attacks based on the evolution probability from the current state to an attack state. In order to train the Markov chain, the author centrally obtains and processes several alerts. The solution predicts C&C communication with 99% accuracy in the tested scenarios. This result shows the prediction of DDoS attacks. It also shows that the prediction varies from a few seconds to 18 hours before the attack begins. The study in [6] utilizes the metastability theory to discover signals before the onset of the attack. The solution captures network traffic, prepares the captured data, and calculates indicators. The solution analyzes these indicators to obtain evidence about attacks and produce DDoS attack alerts. The author evaluates the solution on two datasets. The solution identifies evidence of the attack two hours before the attacker launched the attack.

In [11], the authors use data collected from the Internet to train the Support Vector Machine and predict security events. The collected data are reputation blacklists and security events. Blacklists represent the reputation of devices that may be involved in attacks. Security events reported occurrences of attacks on the Internet. The authors predict cybersecurity-related events with an average true positive of 69%. Although the work does not focus on DDoS attacks, this solution may work in a DDoS attack scenario. In [12], the authors use alerts produced by an Intrusion Detection System (IDS) to predict attacks utilizing the Hidden Markov Chain (HMM). The proposal performs offline training where the solution compares IDS alerts with previously defined historical data. After training, the solution defines the states and probabilities of changes. The solution predicts the attack when there is a probability that the HMM will evolve from a non-attacked state to an attack state. The authors have found that the technique predicts an attack 11 minutes ahead.

Despite the existence of these works, much can still be done to evolve the literature on DDoS attack prediction. Deep learning solutions can evolve the DDoS attack prediction literature, presenting new and advanced solutions. This technique can overcome the limitation of depending on labeled data.

III. Deep Learning Approach

This section describes the proposed system to predict DDoS attacks based on Unsupervised Deep Learning. The system uses an LSTM Autoencoder deep neural network capable of identifying Early Warning Signals (EWSs). EWSs are statistical signals extracted from time series, capable of anticipating critical transitions before they happen. These transitions, in this work, are associated with the actions of DDoS attacks. When the system detects an Early Warning Signal, it notifies network administrators. Therefore, the network administrators will have more time to stop the attack.

A. Data Collection

The first step of the system consists of collecting the network traffic. This step is essential because the system uses the network traffic to train the deep learning model and perform the attack prediction. The firewall forwards the data to a proxy collector to perform the collection of network traffic. The proxy collector must have a processing capacity compatible with the volume of packets trafficked on the network of users. The proxy has a built-in network traffic capture tool to intercept all incoming and outgoing traffic from the user's network. The proxy exports all raw data, i.e., all intercepted packets, to type Packet Capture (PCAP) files and stores them on a file server. Users configure the maximum limit of traffic stored per capture, but each capture stored by the system contains 1 second of traffic by default. Moreover, the number of packets can define the maximum capture size. The saved files will be available on the file server to the system process in the next step. The file server periodically monitors storage capacity and, by default, deletes the oldest captures.

B. Data Preprocessing

After collecting network traffic data, the preprocessing server is responsible for verifying the captures. The preprocessing server periodically monitors the file server for new captures. When a new capture is available, the server downloads it. By default, the system performs this query every second. Then, preprocessing server loads the data into memory and extracts attributes from the packet headers: the size in bytes, the time elapsed between the previous and current packet during TCP communication, and the TCP communication window size because these attributes are some of the most relevant in detecting DDoS attacks [13]. A two-dimensional array $m \times n$ stores the extracted values, in which m equals the total number of packets collected and n the number of attributes extracted. After extraction, the preprocessing server stores the array and reports when it has completed the preprocessing.

The following preprocessing step transforms the obtained matrices into EWSs associated with Skewness and Kurtosis; both statistical measures are also employed in other works for predictions [14], [15]. These measures follow the concept of time series to organize the data. Skewness (Eq. 2) measures the asymmetry of the distribution, an ideally symmetric probability indicates how much the probability distribution of a random variable deviates from its normal distribution. In Eq. 2, x_t refers to each item observed in the time series. N represents the total amount of observed items. $\bar{x}$ refers to the arithmetic mean, and s is the standard deviation. Kurtosis (Eq. 1) characterizes the flattening of the curve of distribution. It indicates how much a variable is in the tail of the distribution. In Eq. 1, N represents the number of items in the time series. Kurtosis relies on $\hat{y}$, defined by $\hat{y} = \frac{N\sum(x_t-\bar{x})^4}{[\sum(x_t-\bar{x})^2]^2}$. x_t refers to each item in the time series, and $\bar{x}$ refers to the arithmetic mean.

$$Kurtosis = \frac{(N-1)}{(N-2)(N-3)}(N-1)\hat{y}+6 \quad (1)$$

$$Skewness = \frac{N\sum_{t=1}^{N}(x_t-\bar{x})^3}{(N-1)(N-2)s^3} \quad (2)$$

Disturbances or oscillations affect data distribution and generate variations that characterize EWS. This variation occurs when the system changes state, for example, from a negative value to a positive value. As the transition from one state to the other presents a deceleration, it is possible to observe an increase or decrease in the skewness of a time series since the distribution of values in the time series will become skewed. Oscillations or strong perturbations also make it more likely that the state of a system can reach extreme values close to a transition. These effects can lead to an increase in the kurtosis of a pre-transition time series. Thus, the tail of the time series distribution becomes wider due to the increased presence of rare values. Thus, the attributes extracted from each packet are transformed into signals, utilizing skewness and kurtosis to identify critical transitions during the preparation of attacks.

Windows of 1-second aggregate entries of the matrix composed of network traffic extracted data. Hence, the system generates two new values for each of the three attributes utilized for each second. As a result, the system obtains a second array $m' \times n'$, where m' equals the total number of seconds, and n' equals the total number of new attributes after the transformation. This work normalizes the data to keep all values between 0 and 1 to speed up the learning process and obtain faster convergence. For this, the system uses Min-Max (Eq. 3). Where $X(j)$ represents the value of each sample of the analyzed attribute, min_A and max_A are the smallest and largest observed value for the attribute. The min'_A and max'_A represent the new range. Finally, for the sake of compatibility, inputs are formatted in a three-dimensional matrix in the format $m' \times 1 \times n'$.

$$Min - Max = \frac{X(j) - min_A}{max_A - min_A}(max'_A - min'_A) + min'_A \quad (3)$$

C. Reconstruction of Signals

After preprocessing, this work uses data to train and test a deep neural network called LSTM Autoencoder. This neural network learns to generate a compressed version of the processed data and uses it to reconstruct the original signals at the cost of reconstruction. This reconstruction cost results from the difference between the original signal and the reconstructed signal at the network output. When the cost value is equal to zero, the reconstructed signal is identical to the input signal; otherwise, the values are different if the reconstruction cost is greater than or less than zero. The system uses the cost of signal reconstruction to identify EWSs and predict DDoS attacks. During an attack preparation traffic, its cost value stands out concerning the rest of the traffic because the system does not recognize these signals.

LSTMs handle sequential data inputs. These networks save information, storing states for later use utilizing internal memory. They do this with a set of gates controlling when information enters, is emitted, or is deleted. LSTMs are suitable for solving problems whose data evolves [16].

An autoencoder is a type of neural network that uses unsupervised learning to generate a compact representation of input data and, from that representation, reconstruct the data entered in the input. The autoencoder architecture has an Encoder-Decoder structure, which has a bottleneck at its midpoint. This bottleneck works as an Encoder output and Decoder input simultaneously. The network weights are usually randomly initialized and then updated iteratively during training via backpropagation to train the autoencoder. Once fitted, the output of the network at the bottleneck is a fixed-length vector that provides a compact representation of the input data. The output of the network in the Decoder is the reconstructed version of the data inserted in the Encoder from the compressed representation emitted in the bottleneck. After training, the system discards the Decoder and the Encoder stacks with other types of neural networks for diverse applications.

The LSTM Autoencoder is an LSTM network organized on the Encoder-Decoder architecture of the autoencoder. This architecture leverages the capabilities of the traditional autoencoder by allowing the reconstruction of time series through the learning of sequential datasets made possible by the cells of the LSTM layers. The LSTM Autoencoder architecture has five layers. The first and second layers of the neural network form the Encoder and create the compact representation of the input data. The Encoder stacks two LSTM layers containing 16 and 4 neurons, respectively. Then, a repeating vector layer distributes the compressed vector across the Decoder time steps. The Decoder has two more LSTM layers stacked with 4 and 16 neurons. The output of the f neuron follows the Rectified Linear Unit (ReLU), as the activation function ($f(x) = x^+ = \max(0, x)$). The final output layer of the Decoder provides the reconstructed input data. Finally, the neural network has Adam as an optimizer and the Mean Absolute Error to calculate the loss function. Fig. 1 illustrates the neural network architecture.

D. Identification of Early Warning Signals

The LSTM Autoencoder learns to create a compact representation of temporal signals inserted in the input of the neural network. From this representation, it recreates temporal signals similar to those introduced in the input without generating overfitting. However, these signals are not the same, and the system calculates a reconstruction cost by comparing the reconstructed signal at the neural network's output with the

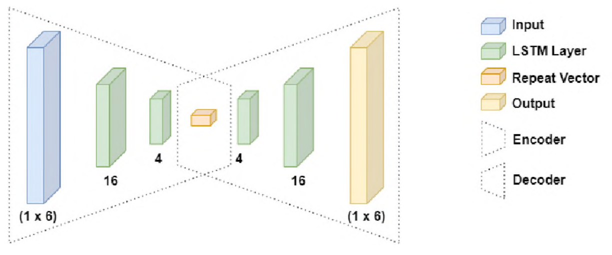


Fig. 1. LSTM Autoencoder Architecture.

input signal. Thus, this work uses the Mean Absolute Error (MAE) as the reconstruction cost. MAE (Eq. 4) is the mean of the absolute errors $|e_i| = |y_i - x_i|$, where y_i is the reconstructed value and x_i is the original value. The term n is the number of signs.

$$\text{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n} = \frac{\sum_{i=1}^{n} |e_i|}{n} \tag{4}$$

The system monitors the reconstruction cost to identify the pre-attack signals and anticipate them. The cost to reconstruct a normal signal without attack preparation signals is small since it has similar data to those utilized in the training stage. However, when the model receives a signal with malicious actions, the cost of reconstructing this signal is higher and may present discrepant values compared to the cost values obtained in training. Through this behavior, this work defines a threshold L for the cost function's results to separate the normal signal from the attack preparation signal. The signals that exceed the threshold are the signals that the system utilizes to anticipate DDoS attacks. Fig 2 details this process. When the system receives a normal signal, the LSTM Autoencoder compresses and reconstructs the normal signal, and the difference between the reconstructed signal and the original is less than the threshold. When the LSTM Autoencoder compresses and reconstructs a bots-influenced signal, the difference between the reconstructed signal and the original malicious signal is greater than the threshold.

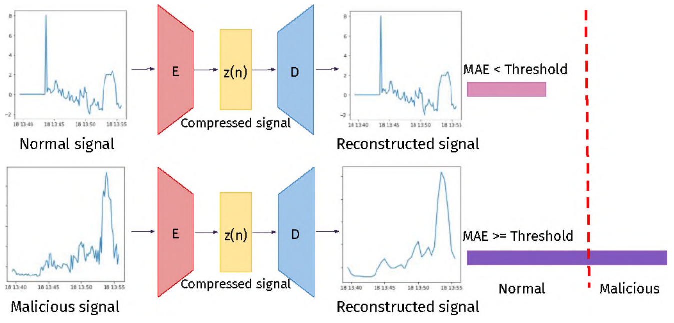


Fig. 2. Identifying Early Warning Signals.

E. Attack Notification

In the last step, the system uses the output of the LSTM Autoencoder to notify about the occurrence of possible DDoS attacks. The system sends messages to the network administrators when the signals analyzed by the LSTM Autoencoder exceed the configured threshold value. Potential attack notification comprises time cycles where EWSs change abnormally. LSTM Autoencoder uses this anomaly as a signal of a possible DDoS attack. The normal cycle indicates that the cycle has no anomalies; therefore, there are no signs of a possible attack happening, and the system does not notify the network administrators. If the system identifies any signal of an attack, it notifies the network administrators to take appropriate action. There are options for this notification, such as sending text messages in instant messengers or enterprise communications platforms. One option for automating incident response is communicating with a web Application Programming Interface to transmit incident data.

IV. Performance Evaluation

This section describes the methodology and results of the performance evaluation of the proposed system. The evaluation follows one experiment to analyze the performance of the model. The experiment utilized the Google Colaboratory platform. It provides a virtual machine with 12 GB of RAM, 107 GB of storage, and NVIDIA Tesla K80. This work implements the code utilizing the Python programming language.

A. Methodology

The evaluation utilizes the CTU-13 dataset. The Czech Technical University (CTU) researcher created this dataset in 2011. The dataset has 13 scenarios containing traffic from benign hosts and hosts infected by different botnets. The employed scenario was Scenario 10, which has 4.75 hours and approximately 1.3 million network flows. This scenario contains botnet traffic before and after the DDoS attack starts.

Preprocessing starts with separating the original network capture into smaller files containing 1,000,000 packets each. The system extracts the header attributes of each packet from the network traffic files. As a result, the system obtains a CSV file in the format $p \times q$ generated for each of the 46 chunks of the capture, where p equals the number of packets per file and q equals the number of attributes extracted. Then, the system preprocesses all the CSV files and transforms each of their entries into the Skewness and Kurtosis time signals according to the procedures of Subsection III-B. The system obtains $p' \times q'$, where p' is equal to the total number of seconds before the attack and q' is equal to the number of new attributes. In this case, $q' = 6$ since the Skewness and Kurtosis values were computed for the three attributes extracted from each packet.

Although LSTM Autoencoder is an unsupervised algorithm, it is necessary to label every second of the dataset as normal or malicious traffic; this helps to verify at which moment bots influence the attributes collected in the network traffic. Also, this work uses labels to calculate the evaluation metrics described above. To label the dataset, every second with sent and received packets by the bots were considered potential signals of attack preparation and malicious traffic. The remaining seconds were considered normal traffic.

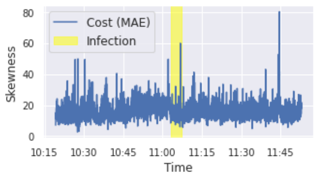
As the solution aims to identify the initial signals of the attack preparation, this work only uses the traffic that precedes the first DDoS attack to train and test the neural network. Hence, all traffic prior to the first attack was selected, which started at 10:19:13 AM on 08/18/2011 and ended at 11:52:53 AM on the same day, totaling 5680 seconds of traffic. This work divides this subset into three equal parts. This work uses the first part for training and the other for testing. In addition, this work trains the model utilizing 100 epochs.

The metrics accuracy, precision, and recall assist in the system evaluation. It is necessary to calculate the total of positive (TP) and negative (TN) correct classifications, the total of positive (FP) and negative errors (FN), and the total observations (N) to obtain the metrics. Accuracy (Eq. 5) measures the proportion of correct predictions among the total number of cases examined. Precision (Eq. 6) measures the fraction of positive predictions that belong to the set of positive predictions. Recall (Eq. 7) quantifies the number of positive predictions made among all positive examples in the dataset.

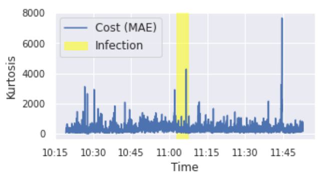
$$a = \frac{TP + TN}{N} \quad (5) \qquad p = \frac{TP}{TP + FP} \quad (6) \qquad r = \frac{TP}{TP + FN} \quad (7)$$

B. Results

The system uses the LSTM Autoencoder designed for six-time signals as input. Each of the signals was obtained by aggregating the packet size values, the time between packets, and TCP communication window size, per second of the traffic subset preceding the first attack and calculating the Skewness and Kurtosis value for each aggregate second. Fig. 3 shows the constructed signals. The figure shows Skewness (Fig. 3(a)) and Kurtosis (Fig. 3(b)) obtained through the analysis of the time between packets to simplify the presentation of the results. However, all data regarding Skewness and Kurtosis for the other attributes are available online[1]. Fig. 3 shows that the generated signals present few continuous values, a high frequency, and high granularity of the information.



(a) Skewness



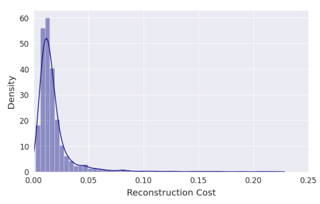
(b) Kurtosis

Fig. 3. Skewness and Kurtosis of TCP packets in Experiment 1.

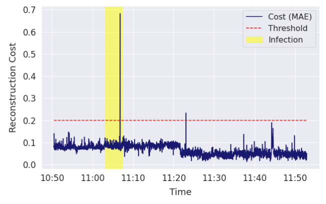
Fig. 4(a) presents the distribution of cost values to reconstruct the signal utilized during training. This work verified that most of the reconstruction cost values are between 0.0 and 0.1. Furthermore, in a few cases, the model obtained a reconstruction cost between 0.1 and 0.2 for a few traffic seconds. Thus, to ensure that only anomalies are identified as EWSs, the threshold was set to notify the attack when the cost to reconstruct the signals exceeds 0.2. Therefore, every second of the test subset signal that had a reconstruction cost value above 0.2 was considered an attack preparation signal

[1]https://github.com/gsilv4/latincom-2022-silva-neira-nogueira

and generated an alert for the network administrators. Fig. 4(b) illustrates the cost values to reconstruct all test set signals in Experiment 1. The cost values to reconstruct a few seconds of the test set signals are above the configured threshold, but the vast majority are below 0.2. Furthermore, most of the cost values are close to each other and are between 0.0 and 0.1, reflecting the same behavior indicated by Fig. 4(a) to reconstruct the signals utilized during training.



(a) Training set reconstruction cost.



(b) Test set reconstruction cost.

Fig. 4. Time reconstruction cost for Experiment 1.

When the reconstruction cost crosses the threshold, the system notifies system administrators of the possibility of an imminent DDoS attack. To check if these notifications have malicious traffic, the system has labeled every second of the original traffic that has bot traffic as malicious and the rest as normal traffic. All seconds that have packets sent or received by the bots were considered malicious traffic. The remaining seconds were considered normal traffic. Then, it was verified if the seconds that generated a reconstruction cost above the threshold (Fig. 4(b)) had malicious traffic. The proposed system identified three seconds of bot activity during the botnet infection period. In these three seconds, the system notifies administrators about the possibility of a future DDoS attack. The first notification came 46 minutes before the launch of the first attack. Despite this result, the system generated an attack notification with no malicious traffic, a false positive. Also, bot traffic affects 255 seconds, and the reconstruction cost has not crossed the threshold; this made the system classify these seconds as normal, showing 255 false negatives. This work believes that the low impact of bots on metrics causes this false positive rate.

Finally, the system has correctly classified 2737 normal seconds (TN). From these results, this work calculates the evaluation metrics described above. This work gets 91% Accuracy, 75% Precision, and 1% Recall. The metrics show that the model has a high percentage of hits. In addition, three predictions were performed correctly out of four times the system identified an anomaly. However, low recall demonstrates the need for adjustments that the model classifies more attack preparation signals and increases the number of notifications.

C. Discussion

The evaluation results indicate that a system can predict the DDoS attack. Fig. 5 shows the moment the system predicted the attack and compares it with the bot infection and attack launch phases. The figure indicates that the prediction occurred at 11:06:47 and 11:06:48 during the bot infection phase performed by the attackers. Therefore, the system emitted two alerts during the bot infection stage, 46 minutes before the attack started. Furthermore, Fig. 5 shows the system predicted the DDoS attack during its preparation; this prediction occurred at 11:23:03. This prediction occurred during the communication of the command and control server with the bots. Therefore, the system notified the network administrators again 16 minutes after the end of the infection stage and 30 minutes before the attack launch. This prediction occurred because some bots communicate with each other and caused slight variations in the EWSs, although the system could identify these variations. Identifying this anomalous traffic demonstrates the system's capacity to predict DDoS attacks by recognizing signs of attack preparation.

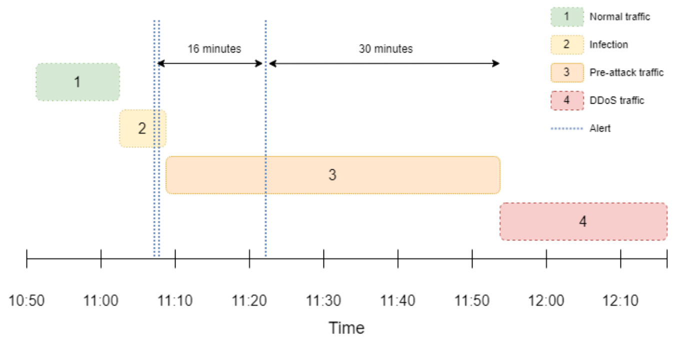


Fig. 5. Attack and anticipation timeline in the test set.

This work compares the proposed solution with the ANTE system [17]. The ANTE system analyzes network traffic to identify bots and notify network administrators before the consequences of the attack generated by the botnet are irremediable. Although not focused on predicting DDoS attacks, such an action can happen if the ANTE system notifies network administrators about the existence of bots before the attack starts. The comparison between the systems is possible because both use scenario 10 of the CTU-13 dataset in their evaluation. The ANTE system anticipates the possibility of a DDoS attack occurring 30 seconds before the launch of the attack. The system proposed in this work notifies network administrators 46 minutes before the start of the attack. Therefore, the system proposed in this work provides more time for administrators to stop the attack and avoid losses.

Although the system predicts the DDoS attack and anticipates it at 46 minutes, some limitations and refinements provide insights to improve the system. The system misclassified a few seconds of traffic produced by the bots, which increased its error rate and affected its accuracy and recall metrics; this occurred because, in most bot interactions, the amount of traffic generated is insufficient for the system to identify the signals of attack preparation. An alternative to improve the model's performance is to use a dynamic threshold for adapting to different scenarios, increasing the number of correct predictions, and increasing the prediction time. Another future action is to evaluate other deep learning techniques to reduce the error rate presented in this work.

V. Conclusion

Attack prediction is a defense mechanism gaining attention in the literature. Predictive techniques identify signals of a DDoS attack in its preparation phases. Its goal is to offer network administrators more time to stop the attack. This work presented a DDoS attack prediction system utilizing the LSTM Autoencoder, a deep learning technique allied to the theory of Early Warning Signals. Evaluation results on the CTU-13 dataset indicate that the system anticipated DDoS attacks 46 minutes in advance. This result is significant as attacks are evolving in sophistication and volume. Hence, the proposed system helps network administrators to avoid losses related to DDoS attacks. Future works intend to evolve the proposed system by creating an adaptive strategy for thresholds that the system uses to determine different signals from the predicted ones, thus indicating possible signals of attack preparation.

References

[1] N. Jyoti and S. Behal, "A meta-evaluation of machine learning techniques for detection of DDoS attacks," in *INDIACom*, 2021, p. 5.

[2] A. Bhardwaj, G. V. B. Subrahmanyam, V. Avasthi, H. Sastry, and S. Goundar, "DDoS attacks, new DDoS taxonomy and mitigation solutions — A survey," in *SCOPES*, 2016, p. 5.

[3] A. Keshariya and N. Foukia, "DDoS defense mechanisms: A new taxonomy," in *DPM*. Springer Berlin Heidelberg, 2010, pp. 222–236.

[4] R. Doriguzzi-Corin, S. Millar, S. Scott-Hayward, J. Martínez-del Rincón, and D. Siracusa, "Lucid: A practical, lightweight deep learning solution for DDoS attack detection," *TNSM*, vol. 17, no. 2, pp. 876–889, 2020.

[5] Z. Abaid, D. Sarkar, M. A. Kaafar, and S. Jha, "The early bird gets the botnet: A markov chain based early warning system for botnet attacks," in *LCN*. IEEE, 2016, pp. 61–68.

[6] M. Pelloso, A. Vergutz, A. Santos, and M. Nogueira, "A self-adaptable system for DDoS attack prediction based on the metastability theory," in *GLOBECOM*, 2018, pp. 1–6.

[7] S. García, M. Grill, J. Stiborek, and A. Zunino, "An empirical comparison of botnet detection methods," *C&S*, vol. 45, p. 24, 09 2014.

[8] P. Kromkowski, S. Li, W. Zhao, B. Abraham, A. Osborne, and D. E. Brown, "Evaluating statistical models for network traffic anomaly detection," in *SIEDS*, 2019, pp. 1–6.

[9] A. Mishra, N. Gupta, and B. B. Gupta, "Defense mechanisms against DDoS attack based on entropy in SDN-cloud using POX controller," *Telecommun. Syst.*, vol. 77, no. 1, pp. 47–62, Jan. 2021.

[10] I. Cvitić, D. Perakovic, B. B. Gupta, and K.-K. R. Choo, "Boosting-based ddos detection in internet of things systems," *IEEE Internet of Things Journal*, vol. 9, no. 3, pp. 2109–2123, 2022.

[11] Y. Liu, J. Zhang, A. Sarabi, M. Liu, M. Karir, and M. Bailey, "Predicting cyber security incidents using feature-based characterization of network-level malicious activities," in *IWSPA*, 2015.

[12] P. Holgado, V. A. Villagrá, and L. Vázquez, "Real-time multistep attack prediction based on hidden markov models," *IEEE Trans. Dependable Secure Comput*, vol. 17, no. 1, pp. 134–147, 2020.

[13] A. Koay, I. Welch, and W. Seah, "(Short Paper) Effectiveness of entropy-based features in high- and low-intensity DDoS attacks detection," in *IWSEC*, 07 2019, pp. 207–217.

[14] V. Dakos, S. R. Carpenter, W. A. Brock, A. M. Ellison, V. Guttal, A. R. Ives, S. Kéfi, V. Livina, D. A. Seekell, E. H. van Nes, and M. Scheffer, "Methods for detecting early warnings of critical transitions in time series illustrated using simulated ecological data," *PLOS ONE*, vol. 7, no. 7, pp. 1–20, 07 2012.

[15] X.-q. Xie, W.-P. He, B. Gu, Y. Mei, and S.-s. Zhao, "Can kurtosis be an early warning signal for abrupt climate change?" *Climate Dynamics*, vol. 52, 06 2019.

[16] B. Lindemann, T. Müller, H. Vietz, N. Jazdi, and M. Weyrich, "A survey on long short-term memory networks for time series prediction," *Procedia CIRP*, vol. 99, pp. 650–655, 2021.

[17] A. B. de Neira, A. M. Araujo, and M. Nogueira, "Early botnet detection for the internet and the internet of things by autonomous machine learning," in *MSN*, Japan, 2020, pp. 516–523.