Detecting Server-Side Request Forgery (SSRF) Attack by using Deep Learning Techniques

Khadejah Al-talak[1]
Department of Information Security
University of Tabuk, KSA. Tabuk, KSA
KSA. Tabuk, KSA

Dr. Onytra Abbass[2]
Department of Information Technology
University of Tabuk
Tabuk, KSA

***Abstract*—Server-side request forgery (SSRF) is a security vulnerability that arises from a vulnerability in web applications. For example, when the services are accessed via URL the attacker supply or modify a URL to access services on servers that he is not permitted to use. In this research, various types of SSRF attacks are discussed, and how to secure web applications are explained. Various techniques have been used to detect and mitigate these attacks, most of which are concerned with the use of machine learning techniques. The main focus of this research was the application of deep learning techniques (LSTM networks) to create an intelligent model capable of detecting these attacks. The generated deep learning model achieved an accuracy rate of 0.969, which indicates the strength of the model and its ability to detect SSRF attacks.**

Keywords*—Server-side request forgery (SSRF); machine learning (ML); deep learning (DL); long short-term memory (LSTM)

I. Introduction

With the development and increasing number of applications on the World Wide Web, security violations have increased, as these applications are characterized by being public, making them vulnerable to attacks [1]. Despite the great progress in methods to protect web applications, hackers are searching using advanced technology for loopholes to overcome these protection methods [2]. These applications inherently contain huge and sensitive data that must be protected from intrusion. SSRF attacks exploit any vulnerability within the web application to enter the server and obtain data in illegal ways, so we need mechanisms to defend against these attacks. Web applications use Authentication Systems to confirm the identity of the client communicating with the web application on the server. The username and password are sent from the client to the server in an encrypted form via HTTP, this information can be compromised while it is being sent. One of the main characteristics of the web applications is that it should work without interruption, so when designing these applications, you must take into account that it works even if it is attacked by hackers. Therefore, several security rules and controls must be put in place to protect web applications from attacks that are widespread with the development of technology. The development of technology helps hackers search for potential vulnerabilities in web applications, making the web application vulnerable with the data contained within it being corrupted, lost, or hijacked. There are now basic plans and concepts to reduce the risk of these attacks and protect data [3]. Web applications are exposed to multiple attacks, the most common of these attacks is Cross-Site Scripting (XSS), SQL Injection, DDoS Attack, Malware, Bots, Cross-Site Request Forgery (CSRF) and Server-Side Request Forgery (SSRF). To detect and mitigate attacks against web applications, machine learning techniques are used because of their ability to learn from data. Server-side request forgery (SSRF) is a security vulnerability that arises from a vulnerability in web applications**.**

In this research, different methods are presented in several literatures to detect SSRF attacks that use URLs to perform the attack. in addition to, the LSTM deep learning network was used to build an intelligent model for detecting SSRF attacks. The training of this model was based on a set of data that represents normal data and others infected with attacks. The test results of this model were good, with an accuracy of 96%.

A. Background of SSRF

In a Server-Side Request Forgery (SSRF) attack, the attacker reads and controls internal server resources by using the available functions on the server through web applications [4], as shown in Fig. 1.

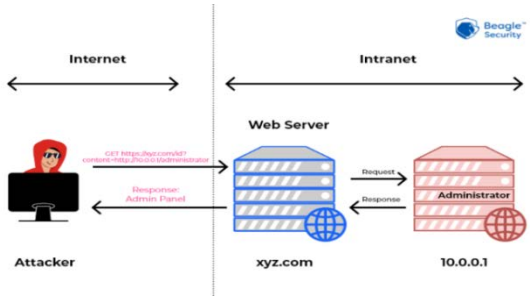

Fig. 1. Server-Side Request Forgery (SSRF) Attack [**4**].

Internal servers behind firewalls can be accessed by the attackers by submitting a URL within a web request to the web application.

A real example of an SSRF attack is the Capital One breach, where the database of Capital One Bank was hacked, and the information of more than 100 million customers was stolen. The attacker uses the Amazon Web Services credentials that were then used to access the Capital One database. Now a list of the main three types of SSRF attacks are explained [5].

- Non-Blind SSRF: As shown in Fig. 2, an attacker can access the data via the HTTP response. Server retrieves the contents of the resource located at the URL submitted, without verification, in an HTTP response to the user is given.

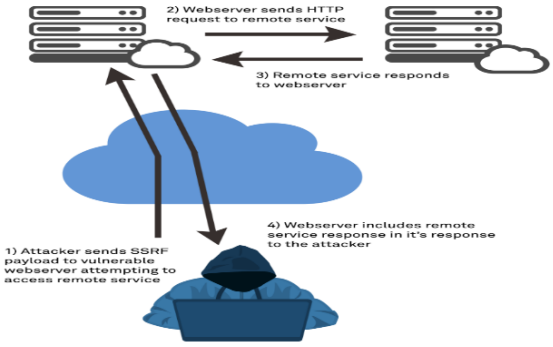

Fig. 2. Non-Blind SSRF [5].

- Blind SSRF: Fig. 3 illustrates this type of SSRF attack. When a web application has SSRF vulnerability but at the same time there is no HTTP response to the attacker. Here the attacker sends his own URL, he can access it, and the server sends an HTTP response to this URL. However, this method detects vulnerability, but it is possible that sensitive data will not be obtained.

Machine learning techniques greatly support artificial intelligence, as computer systems learn to perform human tasks such as classification and prediction. There are many machine learning algorithms and statistical models that are used to analyze data and extract knowledge from it. The process of training machine models produces a system capable of making decisions or recognizing objects. Machine learning algorithms are divided into several types according to the training method and the data available for this process.

Deep learning is a machine learning technique, and it is also considered as developing neural networks by adding new layers to them. Compared to neural networks, deep learning networks improve classifiers, especially as the volume of data increases [6]. As shown in Fig. 5, the performance of deep learning networks generally improves with increasing amounts of training data, unlike other types of machine learning techniques [7].

- Semi-Blind SSRF: Fig. 4 illustrates this type of attack. In the HTTP response, the server does not display all the details but only some of them. Data can be contained in error messages that enable the attacker to learn more information such as request response times, allowing the attacker to validate if a request succeeds.

Deep learning techniques are now used in all areas of artificial intelligence, recognition of voices, faces, text analysis, etc. The process of training deep learning networks often takes longer compared to other machine learning techniques, due to the presence of many parameters in deep learning algorithms.

The difference between neural networks and Deep learning is mainly found in the hidden layer structures. Neural networks contain one layer while deep learning contains several hidden layers. These multiple layers enable deep learning to recognize features in the data without human intervention [8]. Deep learning techniques vary according to several factors, as shown in Fig. 6 [9].

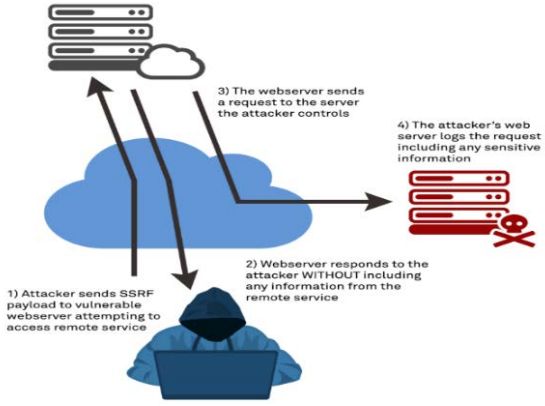

Fig. 3. Blind SSRF [5].

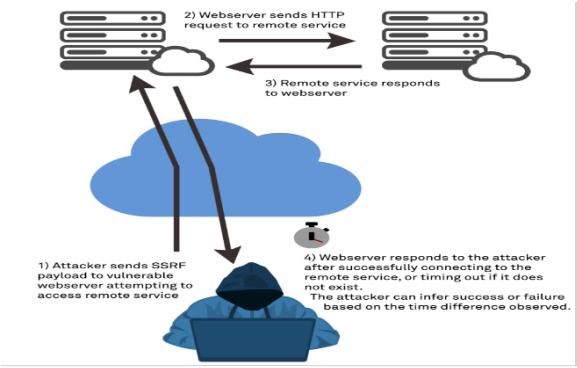

Fig. 4. Semi-Blind SSRF [5].

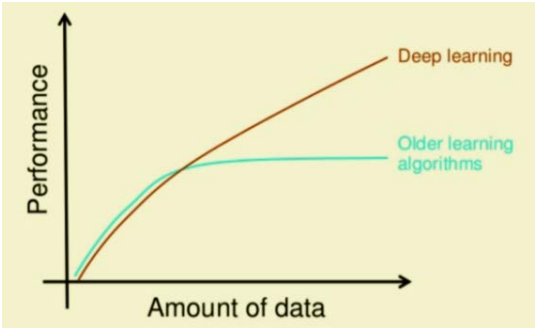

Fig. 5. Machine Learning Techniques Scale with Amount of Data (source [7]).

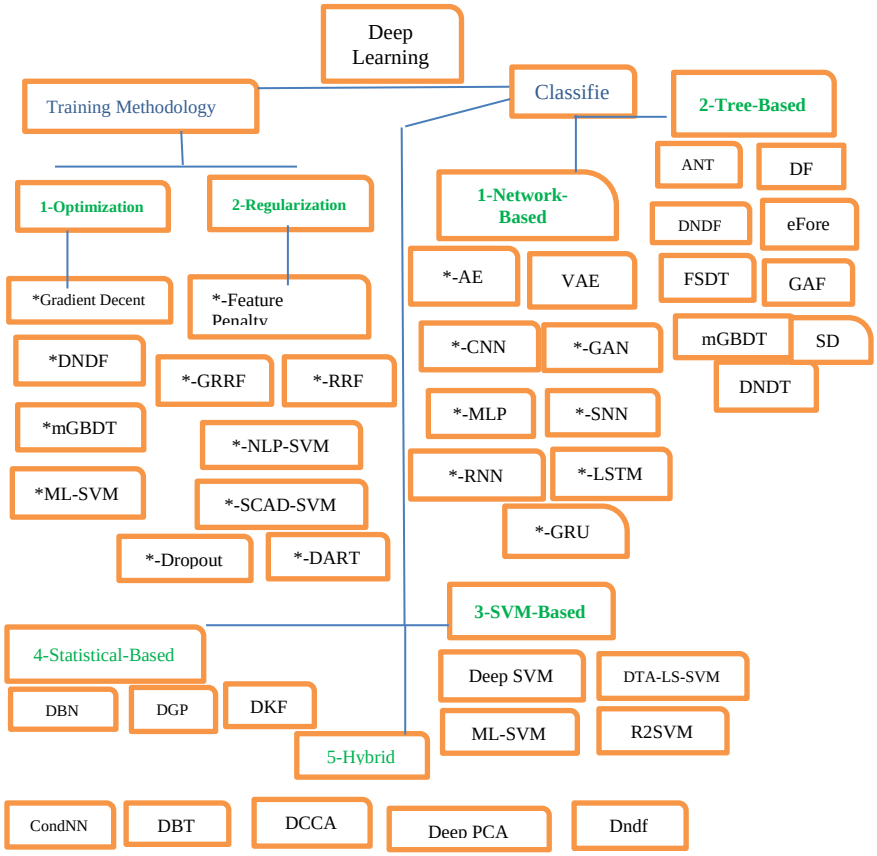

Fig. 6. Deep Learning Methods (source [9]).

II. Related Work

There are several studies concerned with how to detect attacks against web applications, as well as how to face these attacks and overcome them. Various techniques have been used to detect and mitigate these attacks, most of which are concerned with the use of machine learning techniques. we discuss many of the researches concerned with Preventing Server-Side Request Forgery Attacks by discovering them and preventing them from making any threat. Several methods have been used to detect this type of attack. Most of this research was based on artificial intelligence techniques because of their flexibility in application.

In [10], a defensive method was shown to protect web servers from SSRF attacks, and the researcher divided this method into several steps as shown in Fig. 7.

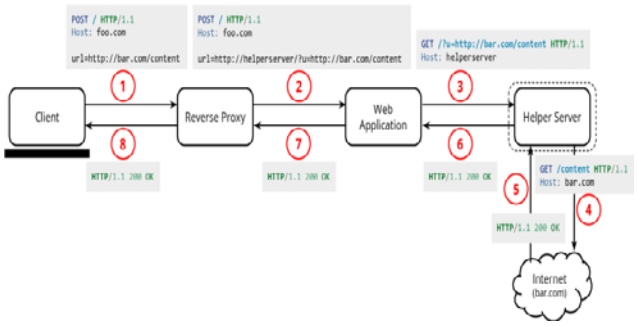

Fig. 7. Overall Architecture to Defenses from SSRF [10].

The reverse proxy located in front of the web application server, in the first step of the methodology, checks all the signals received from service seekers before allowing them to log in and use the application. If the reverse proxy finds a URL embedded in the request from the client to log in to the web application services, it automatically modifies the address and transfers this modified address to the web application server. Helper service takes the original value of the URL and executes it on the server since it cannot use services that are running on the internal network Experimental results show that the proposed solution can prevent all in-band SSRF attacks. Only one of them requires minimal developer collaboration In fact, little increase is observed in the response time in applications for the rest of the applications in the response time.

The researcher at [11] presented a deep learning-based model to discover attacks against web application. This model uses a deep learning technique which is an auto-encoder that is able to learn from the presence of a sequence of words while giving weight to these words according to their presence in the sequence. The model receives an entry request for the web application and then decodes and encodes the requests vector and calculates the reconstruction or loss error. If the loss error value is large then it classifies this request as anomalous requests, and conversely if the value is low then the request is classified as normal requests. The threshold θ is set to determine how small or large the loss error is. The experimental results show that the proposed model can detect web applications attack with low false positive rate and true

positive rate is 1. Because of less volume of labeled categorized anomalous dataset, the proposed classification engine is not 100 percent accurate; however, the classification can be improved with optimized training with a large volume of dataset, which is left as the future scope of the work.

In [12] end-to-end deep learning is applied to detect cyber-attacks astronomically in real-time. The intelligent part of the proposed framework is illustrated in Fig. 8. Authors evaluate the feasibility of an unsupervised/semi-supervised approach for web attack detection based on the Robust Software Modeling Tool (RSMT), which autonomically monitors and characterizes the runtime behavior of web applications. RSMT operates as a late-stage (post-compilation) instrumentation-based toolchain targeting languages that run on the Java Virtual Machine (JVM). It extracts arbitrarily fine-grained traces of program execution from running software and constructs its models of behavior by first injecting lightweight shim instructions directly into an application binary or byte code. These shim instructions enable the RSMT runtime to extract features representative of control and data flow from a program as it executes, but do not otherwise affect application functionality. shows the high-level workflow of RSMT's web attack monitoring and detection system. This system is driven by one or more environmental stimuli, which are actions transcending process boundaries that can be broadly categorized as either manual or automated.

In [13], an accurate and light-weight Android malware detection method is proposed. This method takes an APK-formatted Android app and passes it on to a deep learning network (1-D CNN) to be analyzed to discover the extent of the app's damage. The training and testing steps for this method are shown in Fig. 2. The training process is dependent on 5,000 malwares and 2,000 good wares. We confirmed our method using only the last 512-1K bytes of APK file achieved 95.40% in accuracy discriminating their malignancy under the 10-fold cross-validation strategy as shown in Fig. 9.

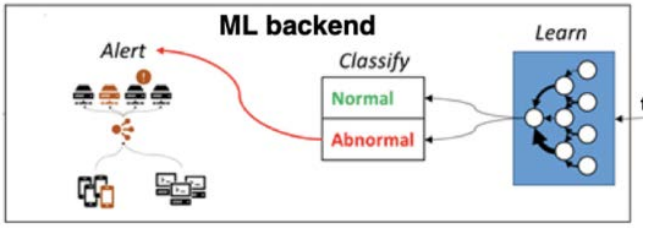

Fig. 8. Train the Classifier to Detect [12].

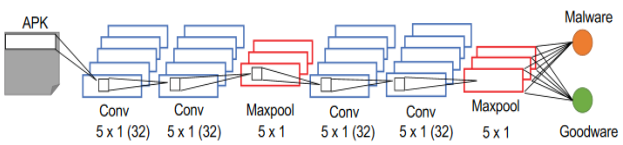

Fig. 9. 1-D CNN Model for Malware Discrimination [13].

In [14] a strategy is presented to discover real-time SSRF activities in the Amazon Web Services environment. This strategy consisted of several steps as follows:

- Detection using VPC Traffic Mirroring
- Detecting SSRF using Zeek
- Detecting SSRF using Suricata
- Detection using iptables

In [15], the researcher introduced a network-based intrusion detection system (NIDS) based on deep learning and machine learning techniques. This system is aimed at detecting intrusion on the network by examining the traffic through it. The general AI-based NIDS methodology for this paper is illustrated in Fig. 10 [16]. Table I covers the literature review evaluation.

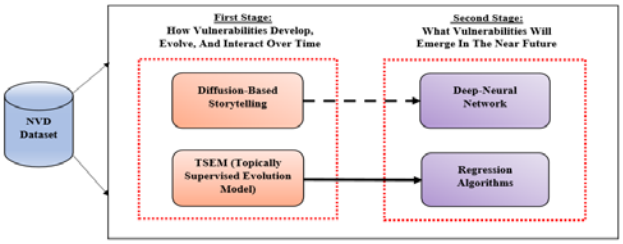

Fig. 10. Proposed Framework in Paper [16].

TABLE I. LITERATURE REVIEW EVALUATION

Paper	Contribution	Tool
[10]	a defensive method using was shown to protect web servers from SSRF attacks	Reverse Proxy
[11]	Detect attacks against web application.	Deep Learning
[12]	Detect cyber-attacks astronomically in real-time.	Deep Learning
[13]	detect malware targeting the Android system	1-D CNN
[14]	A strategy to Detect real-time SSRF activities in the Amazon Web Services environment	VPC Traffic Mirroring, Zeek, Suricata, iptables
[15]	a network-based intrusion detection system	deep learning and machine learning

III. METHODOLOGY

In this Paper, Long Short-Term Memory (LSTM) is used as a deep learning technique to build a model to detect SSRF attacks. Long Short-Term Memory (LSTM) architecture overcomes vulnerabilities in RNN networks [17]. In the first part of our methodology, we collect a dataset. Dataset obtained from the Canadian Institute for Cybersecurity of the University of New Brunswick. This dataset contains many features; we can mention some of them.

This dataset covered all types of SSRF attacks:

- Domain token count, avgpathtokenlen, tld, Arg URL Ratio.
- Number of DotsinURL, Arguments Longest Word Length.
- Spchar URL delimeter Doman, delimeter path, Number Rate.

- Directory Name, Symbol Count Domain, Entropy Domain.

Pre-processing and transformation the data before it is used to train and test machine learning algorithms is essential for creating high-accuracy models. Also, the data that determines whether there is an attack or not, be converted from text to numerical form to facilitate the training process of the deep learning algorithm. Then The text 'benign', 'Defacement' values have been converted to (0-1) to fit the LSTM deep learning network training process Among the important processes of data processing in this Paper is the transformation of data into a form suitable for analysis. Where scaling techniques were used to improve the possibility of recognizing data patterns by deep learning model. All data items is scaled to (-1,1) to enhance the training process.

A. *Learning Methodology*

The LSTM network that used to build the model in this Paper be explained in Fig. 11.

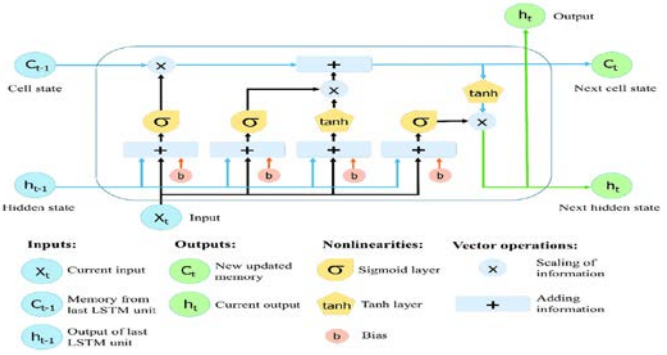

Fig. 11. LSTM Network.

LSTM networks are specifically designed to learn long-range dependencies and avoid problems with RNN. The idea of this type of network is to have a repeating edge inside each cell with weight w = 1. This eliminates the problem of the vanishing gradient problem, since repeated multiplication by 1 neither diverges nor converges to zero. Information flows on two levels from one step to the next while updating the weights is controlled by so-called gates. Gateways use various activation functions to redirect or stop the flow of information. The central change within the LSTM model as opposed to the RNN model or the feedforward model is a more complex type of hidden neuron. LSTM recursive networks contain 'LSTM cells' which have an internal repeat (self-loop), in addition to the external redundancy of the RNN. As can be seen from the previous figure, each element represents a vector with multiple properties that are placed across the grid showing three time steps and a dataset containing three sequential data samples (xt-1, xt, xt+1). The central LSTM cell is shown in detail to reveal the processes within it. The grid appears only until the hidden layers are output (ht-1, ht, ht+1).

LSTM network be used in this Paper to design an intelligent model to predict the SSRF attacks.

In the part of our methodology, we learn a deep learning model to be able to detect SSRF attacks as illustrated in Fig. 12.

In the part of our methodology, we will learn a deep learning model to be able to detect SSRF attacks as illustrated in Fig. 12. To train the deep learning model, the dataset is divided into two parts, the first for the training process and the second for the testing process. The percentage of training data was 80% of the entire dataset, while the percentage of test data was 20%. Then the Feature scaling method was applied to normalize the range of independent variables, where the data range became between -1 and 1. Data now is ready to train and test the model.

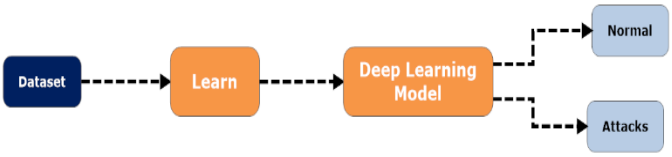

Fig. 12. Training Deep Learning Model.

After that, we can use our proposed model to detect any attacks in the HTTP Request and allow or block this entry according to the absence or presence of an attack as illustrated in Fig. 13.

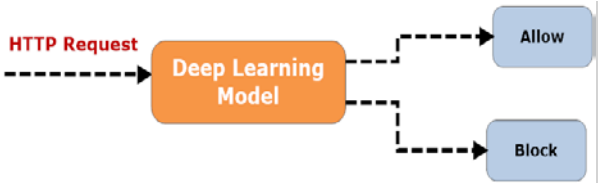

Fig. 13. Use Deep Learning Model to Detect SSRF Attack.

B. *Metrics Measures*

We rely on a set of measures to evaluate the model presented in this Paper, which is based on deep learning techniques. These measures can be summarized as follows:

1) True Positive (TP): The data instances correctly predicted as an Attack by the classifier.

2) False Negative (FN): The data instances wrongly predicted as Normal instances.

3) False Positive (FP): The data instances wrongly classified as an Attack.

4) True Negative (TN): The instances correctly classified as Normal instances.

The confusion matrix, which is generated at the end of the above-mentioned training and testing process, is used to calculate the preceding metrics. The four types (true positives, false negatives, false positives, and true negatives) as well as the positive and negative classifications are used in the template for any binary confusion matrix. Table II shows a 2x2 confusion matrix that can be used to express the four outcomes.

TABLE II. CONFUSION MATRIX PARAMETERS

Actual class	**Predicted class**		
		Class= yes	***Class=no***
	Class =yes	True positive	False negative
	Class=no	Fales positive	True negative

IV. Result and Discussion

Request Forgery (SSRF) attacks on web applications. The basic architecture of web applications and how this architecture is threatened are explained. Also, the types of SSRF attacks are described in this Paper. The proposed system is based on a technique of machine learning, which is deep learning technique, as it has the ability to learn, especially with the availability of a large amount of data.

A. Methods and Tools

The software and libraries needed to construct the LSTM model, which is utilized to identify SSRF assaults, will be detailed in this section. The programming language used is Python, which contains a huge number of external libraries. The diversity of these libraries and their easy handling of data has led to the widespread use of the Python language in all research Paper. Among these libraries used in the proposed Paper:

1) Numpy: This library is for scientific computing to work on the creation, editing and calculation of N-dimensional array objects.

2) Pandas: This library is used to handle large numerical tables with high performance. It can handle data that contains more than 2000 columns.

3) Scikit-learnrn: This library contains many methods that are used for data processing and evaluation of machine learning and deep learning models. It has been used in this Paper to evaluate the proposed deep learning model, using accuracy, precision, Recall, F1 score metrics.

4) Keras: It is an interface for programming neural networks and deep learning applications, by accessing the top of TensorFlow to create, train and test models. This interface supports the training of neural network models on the CPU or GPU. It also supports most types of deep learning networks such as CNN, RNN, and LSTM. In this Paper, keras was used to build an LSTM model for detecting SSRF attacks [18].

B. Development of the LSTM Model

In this part, a description of how to develop, train and evaluate an LSTM network be presented. At first, the data be read from the file that contains the dataset, then a figure be drawn showing the percentage of the data containing the attack as shown in Fig. 14.

After that, the data is scaled between 1 and -1 to enhance the accuracy of the deep learning model. To prepare dataset for training and testing process, it split to the training and testing parts using the following python code [19].

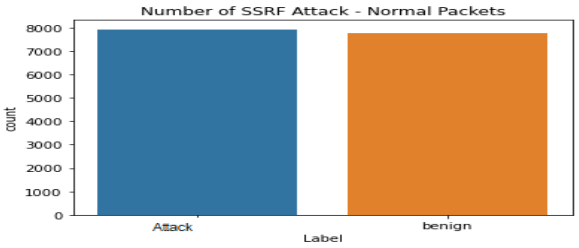

Fig. 14. Attack and Benign Percentage in the Dataset.

As a result of this code, the first 70% of the dataset is classified as training and the last 30% is used as testing data. The `X_train, X_test, Y_train, Y_test = train_test_split(X,y,test_size=0.3, random_state=6)` parameter values of LSTM network is optimized by testing its values for many times, and the final values are:

Batch size: 25

Epochs: 100

Hidden units: 20

Learning rate: 0. 005

Loss Function: MSE

A loss function quantifies how "good" or "bad" a given predictor is at classifying the input data points in a dataset. If the gap between training loss and validation loss is large its means that your model is overfitting and if training loss is large its means your model is underfitting. If your training loss and validation loss are overlapping or close to each other means your model is now good for prediction. Here the model is look good as illustrated in Fig. 15.

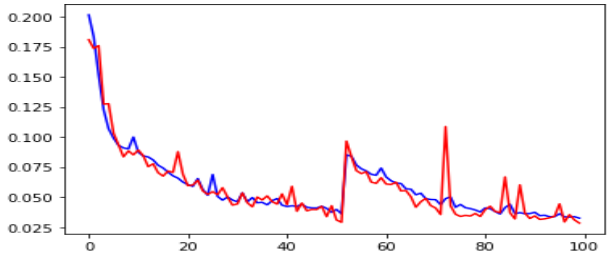

Fig. 15. Training and Validation Loss through Epoch Values.

C. Evaluation of the LSTM Model

The results of the attack detection on dataset are shown in Fig. 16 (Confusion matrix). The values of TP, FN, FP, TN is obtained from confusion matrix and can be summarized as follows:

1) True Positive (TP): 3015
2) False Negative (FN): 80
3) False Positive (FP): 112
4) True Negative (TN): 3078

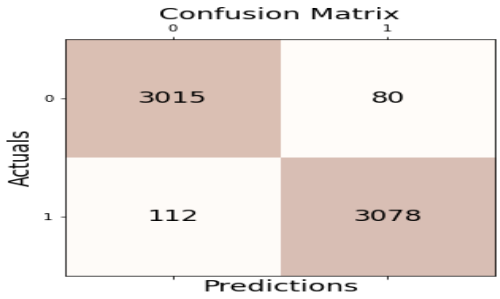

Fig. 16. Confusion Matrix for the Testing Results for LSTM Model.

From these values the metrics for evaluation the proposed model can be calculated as follows:

Precision: 0.975

Recall: 0.965

Accuracy: 0.969

F1 Score: 0.970

V. Conclusion

Server-side request forgery (SSRF) is a web application vulnerability that attackers exploit to access services on servers that an attacker is not allowed to use. In this paper a comprehensive review of the types of SSRF attacks and how they occur is presented. How to protect against these attacks is also explained with a review of literature reviews that provide various solutions to counter these attacks. Some machine learning and deep learning techniques have been demonstrated with an emphasis on the LSTM deep learning network. The LSTM network was used to design a deep learning model to detect SSRF attacks contained in URLs. This network was trained using a set of data after preprocessing operations were performed on it, and the data was scaled between values 1 and -1. This model was tested using several metrics such as accuracy to measure the accuracy of the proposed model in this Paper. The results of the deep learning model test were good, reaching 96%.

VI. Future Work

In the future, we will try to design other machine learning and deep learning models and compare them to get the most accurate model. Also, searching for another preprocessing operation to increase the accuracy of the deep learning model will be the focus of our next work.

References

[1] Michael Cross. 2007. Developer's Guide to Web Application Security. Syngress Publishing.

[2] Hoffman, A. (2020). Web Application Security: Exploitation and Countermeasures for Modern Web Applications.

[3] Web Application Security, accessed, March 2021, https://www.synopsys.com/glossary/what-is-web application-security.html.

[4] https://beaglesecurity.com/blog/article/server-side-request-forgery-attack.html.

[5] Server-Side Request Forgery (SSRF) & the Cloud Resurgence, (2020), accessed 2021, https://appcheck-ng.com/server-side-request-forgery-ssrf/.

[6] Palash Goyal, Sumit Pandey, and Karan Jain. Introduction to natural language processing and deep learning. In Deep Learning for Natural Language Processing, pages 1–74. Springer, 2018.

[7] Ng, A. (2016). Machine learning yearning: Technical strategy for AI Engineers, in the era of deep learning, draft version 0.5. Harvard Business Publishing.

[8] Schmidhuber, J. (2015). Deep learning in neural networks: An overview. Neural Networks, 61, 85–117.

[9] Ghods, A., & Cook, D. (2019). A Survey of Techniques All Classifiers Can Learn from Deep Networks: Models, Optimizations, and Regularization. ArXiv, abs/1909.04791.

[10] Jabiyev, B., & Kharraz, A. (2020). Preventing Server-Side Request Forgery Attacks.

[11] Alma, T., & Das, M. (2020). Web Application Attack Detection using Deep Learning. ArXiv, abs/2011.03181.

[12] Pan, Y., Sun, F., Teng, Z., White, J., Schmidt, D., Staples, J., & Krause, L. (2019). Detecting web attacks with end-to-end deep learning. Journal of Internet Services and Applications, 10, 1-22.

[13] C. Hasegawa and H. Iyatomi, "One-dimensional convolutional neural networks for Android malware detection," 2018 IEEE 14th International Colloquium on Signal Processing & Its Applications (CSPA), Penang, Malaysia, 2018, pp. 99-102.

[14] Sean McElroy, Detecting Server-Side Request Forgery Attacks on Amazon Web Services ISSA Journal February 2020, volume 18, issue 2.

[15] Ahmad, Z., Khan, A., Cheah, W.S., Abdullah, J., & Ahmad, F. (2021). Network intrusion detection system: A systematic study of machine learning and deep learning approaches. Trans. Emerg. Telecommun. Technol., 32.

[16] Williams, M., Barranco, R.C., Naim, S.M., Dey, S., Hossain, M.S., & Akbar, M. (2020). A vulnerability analysis and prediction framework. Comput. Secur., 92, 101751.

[17] Wichers, D.: OWASP Top Ten Project. https://owasp.org/www-project-top-ten/ Online; accessed March 2021].

[18] Gulli, A., & Pal, S. (2017). Deep learning with Keras. Packt Publishing Ltd.

[19] Van Rossum, G., & Drake, F. L. (2009). Python 3 Reference Manual. Scotts Valley, CA: CreateSpace.