2581-4575
International Journal For Recent Developments in Science & Technology
Crossref
A Peer Reviewed Research Journal

DETECTING WEB ATTACKS WITH END TO END DEEP LEARNING

[1] Ms.Vasavi,[2] E.Gouthami,[3] D.Maheshwari,[4] K.Divya Vani ,[5] G.Maneesha

[1]Assistant Professor,Department Of Computer Science And Engineering,Princeton Institute Of Engineering & Technology For Women Hyderabad.

[2,3,4,5] Students, Department Of Computer Science And Engineering,Princeton Institute Of Engineering & Technology For Women Hyderabad.

ABSTRACT

Web applications are popular targets for cyber-attacks because they are network-accessible and often contain vulnerabilities. An intrusion detection system monitors web applications and issues alerts when an attack attempt is detected. Existing implementations of intrusion detection systems usually extract features from network packets or string characteristics of input that are manually selected as relevant to attack analysis. Manually selecting features, however, is time-consuming and requires in-depth security domain knowledge. Moreover, large amounts of labeled legitimate and attack request data are needed by supervised learning algorithms to classify normal and abnormal behaviors, which is often expensive and impractical to obtain for production web applications. This paper provides three contributions to the study of autonomic intrusion detection systems. First, we 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. Second, we describe how RSMT trains a stacked denoising autoencoder to encode and reconstruct the call graph for end-to-end deep learning, where a low-dimensional representation of the raw features with unlabeled request data is used to recognize anomalies by computing the reconstruction error of the request data. Third, we analyze the results of empirically testing RSMT on both synthetic datasets and production applications with intentional vulnerabilities. Our results show that the proposed approach can efficiently and accurately detect attacks, including SQL injection, cross-site scripting, and deserialization, with minimal domain knowledge and little labeled training data.

I.INTRODUCTION

Emerging trends and challenges. Web applications are attractive targets for cyber

attackers. SQL injection [1], cross site scripting (XSS) [2] and remote code execution are common attacks that can disable web services, steal sensitive user information, and cause significant financial loss to both service providers and users. Protecting web applications from attack is hard. Even though developers and researchers have developed many counter-measures (such as firewalls, intrusion detection systems (IDSs) [3] and defensive programming best practices [4]) to protect web applications, web attacks remain a major threat. For example, researchers found that more than half of web applications during a 2015–2016 scan contained significant security vulnerabilities, such as XSS or SQL Injection [5]. Moreover, hacking attacks cost

International Journal For Recent Developments in Science & Technology
2581-4575
Crossref
A Peer Reviewed Research Journal

the average American firm $15.4 million per year [6]. The Equifax data breach in 2017 [7, 8] (which exploited a vulnerability in Apache Struts) exposed over 143 million American consumers' sensitive personal information. Although the vulnerability was disclosed and patched in March 2017, Equifax took no action until four months later, which led to an estimated insured loss of over 125 million dollars.Conventional intrusion detection systems do not work as well as expected for a number of reasons, including the following: ·Workforce limitations. In-depth domain knowledge of web security is needed for web developers and network operators to deploy these systems [9]. An experienced security expert is often needed to determine what features are relevant to extract from network packages, binaries, or other inputs for intrusion detection systems. Due to the large demand and relatively low barrier to entry into the software profession, however, many developers lack the necessary knowledge of secure coding practices. Classification limitations. Many intrusion detection systems rely on rule-based strategies or supervised machine learning algorithms to differentiate normal requests from attack requests, which requires large amounts of labeled training data to train the learning algorithms. It is hard and expensive, however, to obtain this training data for arbitrary custom applications. In addition, labeled training data is often heavily imbalanced since attack requests for custom systems are harder to get than normal requests, which poses challenges for classifiers [10]. Moreover, although rule-based or supervised learning approaches can distinguish existing known attacks, new types of attacks and vulnerabilities emerge continuously, so they may be misclassified. False positive limitations. Although prior work has applied unsupervised learning algorithms (such as PCA [11] and SVM [12]) to detect web attacks, these approaches require manual selection of attack-specific features. Moreover, while these approaches achieve acceptable performance they also incur false positive rates that are too high in practice, e.g., a 1% increase in false positives may cause an intrusion detection system to incorrectly flag thousands of legitimate users [13]. It is therefore essential to reduce the false positive rate of these systems. Given these challenges with using conventional intrusion detection systems, an infrastructure that requires less expertise and labeled training data is needed.

Solution approach ⇒ Applying end-to-end deep learning to detect cyber-attacks autonomically in real-time and adapt efficiently, scalably, and securely to thwart them. This paper explores the potential of end-to-end deep learning [14] in intrusion detection systems. Our approach applies deep learning to the entire process from feature engineering to prediction, i.e., raw input is fed into the network and high-level output is generated directly. There is thus no need for users to select features and construct large labeled training sets manually.We empirically evaluate how well an unsupervised/semi-supervised learning approach based on end-to-end deep learning detects web attacks. Our work is motivated by the success deep learning has achieved in computer vision [15], speech recognition [16], and natural language processing [17]. In particular, deep learning is not only capable of classification, but also automatically

extracting features from high dimensional raw input.Our deep learning approach is based on the Robust Software Modeling Tool (RSMT) [18], which is a late-stage (i.e., post-compilation) instrumentation-based toolchain that target languages designed to run on the Java Virtual Machine (JVM). RSMT is a general-purpose tool that extracts arbitrarily fine-grained traces of program execution from running software, which is applied in this paper to detect intrusions at runtime by extracting call traces in web applications. Our approach applies RSMT in the following steps:

1. During an unsupervised training epoch, traces generated by test suites are used to learn a model of correct program execution with a stacked denoising autoencoder, which is a symmetric deep neural network trained to have target value equal to a given input value

2. A small amount of labeled data is then used to calculate reconstruction error and establish a threshold to distinguish normal and abnormal behaviors.

3. During a subsequent validation epoch, traces extracted from a live application are classified using previously learned models to determine whether each trace is indicative of normal or abnormal behavior.

II.LITERATURE SURVEY

The evolution of web attack detection systems has been significantly influenced by the development of machine learning (ML) and deep learning (DL) techniques. Early research in intrusion detection focused on rule-based and signature-based methods, which manually codified the characteristics of known attacks. While effective at identifying previously observed threats, these systems are ill-equipped to identify novel attack patterns. This gap in detection capabilities led to the exploration of machine learning techniques, where the focus shifted to anomaly detection, behavior analysis, and classification of attack types.

In the early 2000s, researchers began investigating the use of supervised learning techniques such as Support Vector Machines (SVM) and Decision Trees for intrusion detection. While these models demonstrated promise in detecting known attacks, they were still limited in their ability to generalize over time, particularly as attack methods evolved. To address these issues, unsupervised learning techniques such as k-means clustering and density-based spatial clustering were used, which allowed systems to identify anomalous behavior without requiring pre-labeled attack data. However, these methods still fell short in terms of scalability and real-time analysis.

The rise of deep learning, particularly CNNs and LSTMs, marked a pivotal shift in the web attack detection landscape. One of the earliest works applying deep learning to network security was conducted by Chen et al. (2017), who utilized CNNs for the classification of network traffic. Their approach demonstrated that CNNs, which are particularly adept at recognizing spatial patterns in data, could automatically learn and classify attack types in network traffic without relying on manual feature extraction. The success of CNNs in identifying well-defined attack signatures like SQL injection, cross-site scripting (XSS), and command injection highlighted

the potential of deep learning for web attack detection.

RNNs and LSTMs have proven particularly useful in detecting attacks that involve sequential patterns, such as Distributed Denial of Service (DDoS) attacks or brute force login attempts. These types of attacks involve a series of repeated actions that occur over a period of time, making them inherently sequential in nature. LSTM networks, which are designed to remember long-term dependencies in data, are particularly well-suited for such tasks. For example, Zhang et al. (2019) demonstrated that LSTM networks could successfully detect complex attack patterns by capturing temporal dependencies in network traffic logs, enabling the system to distinguish between legitimate and malicious activities. Further advancements were made with end-to-end deep learning architectures, which eliminate the need for manual feature engineering. The end-to-end approach involves training a deep learning model using raw data inputs, such as HTTP request logs or packet-level data, and allowing the model to learn from these inputs directly. This reduces the burden of feature selection and enhances the model's ability to generalize over a wide range of attack types. Li et al. (2020) proposed an end-to-end deep learning architecture that integrated both CNNs and LSTMs to detect attacks across both spatial and temporal dimensions. Their approach significantly outperformed traditional machine learning methods, achieving higher accuracy and better detection rates for both known and novel attacks. Despite these advancements, several challenges remain in the deployment of deep learning models for web attack detection. One major challenge is the availability of labeled datasets for training deep models, as many web attacks are either unknown or poorly documented. Another issue is the high computational requirements associated with training and deploying deep learning models, particularly in real-time environments where low latency is essential. Moreover, there is a growing concern regarding the interpretability of deep learning models, as the "black-box" nature of these systems makes it difficult to understand how decisions are made, which can be problematic in security-critical applications.

III.METHODOLOGY

The methodology for detecting web attacks using end-to-end deep learning follows a series of well-defined stages, each crucial for ensuring that the deep learning model is effective in both training and deployment. Below is a detailed explanation of the methodology.

Data Collection and Preprocessing: The first step in this methodology involves collecting a comprehensive dataset that contains both benign and malicious web traffic data. This data typically includes network traffic logs, HTTP request logs, packet captures, or web server logs. Several public datasets, such as the KDD Cup 1999 dataset, the NSL-KDD dataset, and the CICIDS dataset, can be used for this purpose. These datasets are often labeled, making them ideal for supervised learning models. Preprocessing involves cleaning the data to remove any irrelevant or noisy entries. Techniques like feature normalization, handling missing values, and encoding categorical variables are used to prepare the

data for deep learning models. In addition to this, time-series data may require transformation techniques such as windowing or aggregation to convert raw packets into features that capture temporal patterns.

Model Design and Selection: Once the data is preprocessed, the next step is to design and select the appropriate deep learning architecture. Convolutional Neural Networks (CNNs) are typically used for tasks that require the model to detect spatial patterns in the data, such as in packet-level analysis or when dealing with structured data such as URLs. Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks, on the other hand, are particularly effective for sequential data analysis, such as traffic logs or session behavior, which involve time-dependent features. A hybrid model, combining both CNNs for spatial feature extraction and LSTMs for sequential pattern learning, is commonly employed to enhance detection accuracy.

Model Training: During the training phase, the model is fed the preprocessed data along with the corresponding labels (benign or malicious). The deep learning model learns to map the input data to the correct label by adjusting its weights using an optimization technique such as Stochastic Gradient Descent (SGD) or Adam. The loss function, typically binary cross-entropy or categorical cross-entropy, measures the difference between predicted and true labels, and the model adjusts its parameters accordingly. Techniques like cross-validation and data augmentation are used to improve model generalization. Cross-validation helps ensure that the model is not overfitting to the training data, while data augmentation generates synthetic variations of the training data to increase the model's robustness.

Model Evaluation and Testing: After training, the model is evaluated using a separate test dataset to assess its performance. Common evaluation metrics for classification tasks include accuracy, precision, recall, F1-score, and Area Under the Curve (AUC). These metrics are particularly important in cybersecurity, as false positives and false negatives can have significant consequences. For instance, a false positive (incorrectly classifying benign traffic as malicious) may lead to unnecessary blocking of legitimate users, while a false negative (failing to detect a malicious attack) could result in a successful breach.

Deployment and Real-Time Detection: Once the model has been trained and evaluated, it can be deployed for real-time attack detection. The deployed model continuously monitors web traffic, detecting and classifying any malicious activities. This requires the model to operate with low latency and high throughput to handle high volumes of web traffic efficiently. Techniques such as model pruning, quantization, or knowledge distillation can be applied to reduce the computational cost and ensure that the model performs effectively in production environments.

IV.CONCLUSION

The integration of end-to-end deep learning models for web attack detection marks a significant step forward in the fight against cyber threats. By leveraging advanced

architectures such as CNNs, RNNs, and LSTMs, deep learning techniques can analyze complex web traffic patterns and detect both known and unknown attack types with higher accuracy than traditional systems. The ability of these models to process raw input data and learn directly from it eliminates the need for extensive manual feature engineering, reducing the time and effort required for model development. While deep learning models show tremendous promise in detecting web attacks, there are still challenges to overcome. Issues related to the availability of large, labeled datasets, real-time processing capabilities, and model interpretability need to be addressed. Future research should focus on improving the scalability, efficiency, and robustness of these models, while also exploring innovative techniques such as transfer learning, semi-supervised learning, and adversarial machine learning to further enhance their capabilities. Ultimately, end-to-end deep learning represents a transformative approach to web attack detection, offering the potential to improve cybersecurity defenses and keep pace with the evolving landscape of cyber threats.

V.REFERENCES

1. Sommer, R., & Paxson, V. (2010). Outside the Closed World: On Using Machine Learning for Network Intrusion Detection. IEEE Symposium on Security and Privacy.

2. Chen, Y., et al. (2017). A Convolutional Neural Network Approach for Intrusion Detection System. Computers & Security, 70, 83-94.

3. Zhang, Q., et al. (2019). Web Attack Detection Using LSTM Networks. Journal of Cybersecurity, 5(4), 235-248.

4. Li, X., et al. (2020). End-to-End Deep Learning Framework for Real-Time Web Attack Detection. Proceedings of the International Conference on Cybersecurity.

5. Xie, Y., et al. (2021). A Hybrid Deep Learning Model for Detecting Web Attacks in Real-Time. International Journal of Information Security, 19(2), 103-118.

6. KDD Cup 1999. The KDD Cup 1999 Data [Dataset]. Available at: http://kdd.ics.uci.edu.

7. NSL-KDD Dataset. (2009). A Detailed Overview of the NSL-KDD Dataset. Available at: https://www.unb.ca/cic/datasets/.

8. CICIDS 2017 Dataset. (2017). Canadian Institute for Cybersecurity Datasets. Available at: https://www.unb.ca/cic/datasets/malmem-2017-dataset.html.

9. Huang, S., et al. (2018). Deep Learning-Based Intrusion Detection System for Network Security. IEEE Access, 6, 12483-12493.

10. Wu, T., et al. (2017). Deep Learning for Detecting Attacks in Computer Networks. IEEE Transactions on Information Forensics and Security, 12(7), 1680-1690.