Anomaly Detection on Web-User Behaviors Through Deep Learning

Jiaping Gui(✉), Zhengzhang Chen, Xiao Yu, Cristian Lumezanu, and Haifeng Chen

NEC Laboratories America, Inc., Princeton, USA
{jgui,zchen,xiao,lume,Haifeng}@nec-labs.com

Abstract. The modern Internet has witnessed the proliferation of web applications that play a crucial role in the branding process among enterprises. Web applications provide a communication channel between potential customers and business products. However, web applications are also targeted by attackers due to sensitive information stored in these applications. Among web-related attacks, there exists a rising but more stealthy attack where attackers first access a web application on behalf of normal users based on stolen credentials. Then attackers follow a sequence of sophisticated steps to achieve the malicious purpose. Traditional security solutions fail to detect relevant abnormal behaviors once attackers login to the web application. To address this problem, we propose *WebLearner*, a novel system to detect abnormal web-user behaviors. As we demonstrate in the evaluation, *WebLearner* has an outstanding performance. In particular, it can effectively detect abnormal user behaviors with over 96% for both precision and recall rates using a reasonably small amount of normal training data.

Keywords: Web application · Abnormal behavior · Sequence-based attack · Deep learning

1 Introduction

Web applications (or apps) have become an important part of many companies' digital marketing strategy. In 2018, nearly two-thirds of small businesses rely on their web applications to connect with customers [2], not to mention those large enterprises. In spite of business values, web apps can be the target of attackers due to vulnerable interfaces in these apps. We continue to see a trend of increasing number of vulnerabilities in web apps. The overall number of new web-app vulnerabilities in 2018 increased by 23% compared to 2017 and by 162% compared to 2016 [10]. Among all attack targets, web apps are the top two to be hit in 2018 [13].

To defend against web-related attacks, different solutions (e.g., [4,14,16]) have been proposed. Among these solutions, there are two major directions:

N. Park et al. (Eds.): SecureComm 2020, LNICST 335, pp. 467–473, 2020.
https://doi.org/10.1007/978-3-030-63086-7_25

rule-based and learning-based. The former extracts unique patterns from attack traces and design specific rules to detect future attacks that have the same patterns. The latter explores a large amount of attack traces and learns traditional attack patterns. For example, some methods group these attacks into clusters in the hope that future attacks fall into these clusters. Some leverage statistical information (e.g., traffic volume) to detect anomalies related to user behaviors.

Despite the effectiveness of detection on specific web attacks, existing solutions suffer from a critical problem that limits their usability and application. In particular, existing solutions fail to detect sophisticated web attacks that involve multiple steps (as a sequence), since these solutions typically target single-step attacks. For example, in cross-site scripting (XSS), attackers directly inject malicious scripts into benign web applications viewed by other users. To detect such an attack, the solution is straightforward by detecting whether there exists any executable-script keyword in the application. However, there are two key characteristics in sequence-based attacks that existing solutions do not take into account.

The first missing characteristic is the sequence-based patterns. In sophisticated web attacks, a sequence of steps are conducted by attackers to achieve the malicious goal. Each single step may be benign, while as a whole, these steps are suspicious. We call this *local-benign-global-malicious*. For example, in a shopping web app, an infrequent user who makes tens of orders at one time is suspicious. One order itself should be normal but a sequence of such orders are abnormal. The second missing characteristic is to be able to detect attacks with unknown patterns. In the real world, attackers can easily bypass existing solutions once they locate the set of patterns that are leveraged by security solutions. This set of patterns, though can be upgraded iteratively, do not accommodate the dynamic behaviors of attackers. The capability of existing solutions is greatly restricted when new unknown attacks are encountered.

In this paper, we propose *WebLearner*, an intelligent web security system that detects abnormal web-user behaviors through deep learning. In particular, *WebLearner* learns session-based sequences of user visits (e.g., URLs) on a web application. To address the first characteristic, *WebLearner* utilizes a sliding window to generate representations of sequences of user visits. The whole sequence in each window is used for training. In other words, each single visit is not trained individually. To address the second characteristic, *WebLearner* leverages a deep learning technique to model normal user behaviors that takes advantage of information in a guided way. Considering attackers have no access to the web-visit history of normal users, suspicious behaviors from attackers are expected to be different from those normal ones, and thus be detected as abnormal.

The contributions of this paper can be summarized below:

- We propose *WebLearner*, an intelligent web security system that enables UBA on web applications to detect sequence-based anomalies.

– We have implemented *WebLearner* and deployed it into a real-world environment we set up. Our experimental results are promising, demonstrating that *WebLearner* can detect abnormal web-user behaviors effectively.

2 Approach

To detect sequence-based abnormal user behaviors in web attacks (e.g.., credential stuffing, logic attack, and bot-related attacks), *WebLearner* adopts a training-prediction-retraining mode that incorporates feedback from security analysts in an efficient and effective manner. *WebLearner* has following advantages: (1) it takes a representation of web-user behaviors directly from raw user access log, which keeps the inherent relationship among visits in the sequence. (2) it has a flexible framework that can train the model corresponding to each of the groups of user behaviors. (3) it has the capability to detect new, unknown and sophisticated attack behaviors, since *WebLearner* depends on a training data set that consists of sequences of normal user visits on the web app.

In the high level, it has two main components: log parser and deep learning model. There are two different types of inputs: normal user access log for training, and new access log for prediction.

In the first step, the log parser takes as input the normal user access log, and outputs representations of sequences of user visits in different dimensions. In particular, *WebLearner* first extracts effective page requests (in terms of URL) from the user access log, and filters out noisy requests (e.g., resource-related image, JS, CSS). Then *WebLearner* takes into account both the page directory (also known as page topic) and parameters. For example, if a URL is `BrowseCategories.php?nickname=u1&password=p1`, then the page directory of this URL is `BrowseCategories.php`, and parameters are `nickname` and `password`. Let $\mathcal{S}_\mathcal{D} = \{d_1, d_2, ...d_m\}$ be the set of all directories in the normal training log, and P_i be the set of all parameters to the directory d_i. *WebLearner* indexes element strings in $\mathcal{S}_\mathcal{D}$ according to the alphabetical order. For a new URL with the directory d_n and parameter set P_n, the representation of this new URL is generated according to Eq. 1. Finally, *WebLearner* forms value representations into corresponding vectors.

$$URL\_representation = \begin{cases} i, & \text{if } d_n = d_i \in \mathcal{S}_\mathcal{D} \text{ and } P_n \subset P_i \\ n, & \text{if } d_n \notin \mathcal{S}_\mathcal{D} \\ max\{int\}, & \text{otherwise} \end{cases} \tag{1}$$

The deep learning model is used to learn complex web-user behaviors in terms of page visits on the web app. To model user-behavioral sequences in terms of URL, we leveraged a recurrent neural network with long short-term memory (LSTM) [8] due to its superior performance in learning long-term dependencies over sequences [5,6,9]. The sequence of page visits is much more diverse than the environmental information. To do the learning, this model utilizes a sliding

window to get the input sequence with fixed length, and uses the next visit following the sequence as the output prediction of the sequence.

Finally, after modeling, *WebLearner* can predict whether a new access log has abnormal user behaviors. When new log entries come, the log parser first generates corresponding representations. Then the sequence of page visits goes to the deep learning model. After prediction, if the deep learning model detects the sequence as an anomaly, security analysts further investigate it to take corresponding actions for defense. However, it is possible the sequence that is detected as suspicious is benign. In this case, *WebLearner* uses the sequence to retrain the deep learning model for incremental learning.

3 Evaluation

In this section, we show the performance of *WebLearner* in detecting abnormal web-user behaviors. The high-level research question is how effective *WebLearner* is to detect anomalies.

3.1 Experiment Setup

WebLearner is implemented using JAVA (for log parser) and PyTorch (for deep learning model). We deployed *WebLearner* on a server with 12 cores (Intel Core i7-8700K CPU @ 3.70 GHz) and 32 GB memory.

To evaluate *WebLearner*, we have two requirements on the data set: (1) there are a large amount of user visits on a web app. (2) there is labeling information about normal and abnormal user behaviors. To achieve both requirements, we set up a benchmark based on RUBiS [1], an auction site prototype modeled after eBay.com. To meet the first requirement, we automated the generation of workloads (i.e., user interaction traces) in a large scale. To meet the second requirement, we control the workload generation to create synthetic abnormal behaviors or attack cases. In RUBiS, each group of user behaviors is controlled by a state transition matrix, where each element specifies the probability of page transition that goes from one interaction to another. In the end, we generated five different groups of normal web-user behaviors with in total 11,677 sessions. In the deep learning model, we set the size of sliding window $w = 10$. Table 1 summarizes the values of other relevant parameters. After filtering out the sessions that have no more than ten visits, we have in total 10,910 effective sessions for evaluation.

Table 1. Values of parameters used in deep learning model

# Directories	# Hidden dimensions	# Layers	# Epochs	Batch size
24	64	2	300	2,048

3.2 Effectiveness of *WebLearner*

To validate the effectiveness of *WebLearner*, we randomly selected 30% of sessions in each of the groups for training. The data used for prediction consists of two parts: normal (i.e., the rest 70% of sessions) and abnormal. To collect data related to abnormal behaviors, we randomly generated sessions whose sequence length conforms with the length distribution among those of group behaviors.

For evaluation, we use standard metrics as follows. A false positive (FP) denotes a normal session sequence that is detected as abnormal, and a false negative (FN) denotes an abnormal session sequence that is detected as normal. A true positive (TP) means an abnormal session sequence is detected correctly as abnormal. Metrics for evaluation include: $Precision = \frac{TP}{TP+FP}$, $Recall = \frac{TP}{TP+FN}$, and $F1\text{-}measure = \frac{2 \cdot Precision \cdot Recall}{Precision+Recall}$.

After modeling and predicting, *WebLearner* could achieve the precision 96.75%, the recall 96.54%, and the F1-measure score 96.63%. This indicates that *WebLearner* works well when the training data set is limited.

4 Related Work

In this section, we discuss the overlaps and differences of the related work with our work.

Web Application Defense: Many products or solutions have been implemented to defend against web-related attacks. The Open Web Application Security Project (OWASP) [12], an online community that focuses on web application security, lists the top web attacks (e.g., injection, cross-site scripting) and suggests best practices for developers. In the literature, researchers also propose various methodologies (e.g., [4,14]) for more secure web applications. All pieces of above work focus on traditional web attacks, or model the runtime behavior of web applications. In contrast, our work investigates abnormal web-user behaviors consisting of sequences of visits. We believe our approach is complementary to the related work, and both are beneficial to achieving a more secure web application.

Deep Learning Application: Deep learning models such as LSTM have been widely applied to various scenarios in the real world. Want et al. [15] proposed attention-based LSTM to do sentiment classification in NLP. Du et al. [3] implemented DeepLog, a deep learning model based on LSTM to detect anomalies in system logs. Researchers also use LSTM for other purposes, such as speech recognition [7] and time series study [11]. However, all above studies focus on tasks different from ours. We designed and implemented *WebLearner* to achieve anomaly detection on sequence-based web-user behaviors.

5 Conclusion

In this paper, we propose a novel system, *WebLearner*, to detect abnormal web-user behaviors. This work is motivated by a rising type of web attacks that are based on a sequence of visits. Such an attack is more stealthy than before, and thus cannot be detected by traditional web-security solutions. To address this problem, *WebLearner* utilizes a deep learning model to learn normal user behaviors, and detects abnormal behaviors that are unknown and different from normal ones. Our evaluation results demonstrate *WebLearner* has an outstanding performance. In particular, it can effectively detect abnormal user behaviors with over 96% for both precision and recall rates using a reasonably small amount of normal training data.

References

1. Amza, C., et al.: Specification and implementation of dynamic Web site benchmarks. In: 5th Workshop on Workload Characterization. No. CONF (2002)
2. Clutch: Small Business Websites in 2018. https://clutch.co/website-builders/resources/small-business-websites-2018
3. Du, M., Li, F., Zheng, G., Srikumar, V.: DeepLog: anomaly detection and diagnosis from system logs through deep learning. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, pp. 1285–1298. ACM (2017)
4. García, V.H., Monroy, R., Quintana, M.: Web attack detection using ID3. In: Debenham, J. (ed.) Professional Practice in Artificial Intelligence. IIFIP, vol. 218, pp. 323–332. Springer, Boston, MA (2006). https://doi.org/10.1007/978-0-387-34749-3_34
5. Gers, F.A., Schraudolph, N.N., Schmidhuber, J.: Learning precise timing with LSTM recurrent networks. J. Mach. Learn. Res. **3**, 115–143 (2002)
6. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural Netw. **18**(5–6), 602–610 (2005)
7. Han, S., et al.: ESE: efficient speech recognition engine with sparse LSTM on FPGA. In: Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, pp. 75–84 (2017)
8. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
9. Huang, Z., Xu, W., Yu, K.: Bidirectional LSTM-CRF models for sequence tagging. arXiv preprint arXiv:1508.01991 (2015)
10. Imperva: the state of Web application vulnerabilities in 2018. https://www.imperva.com/blog/the-state-of-web-application-vulnerabilities-in-2018/
11. Karim, F., Majumdar, S., Darabi, H., Chen, S.: Lstm fully convolutional networks for time series classification. IEEE Access **6**, 1662–1669 (2017)
12. OWASP: OWASP top ten project. https://www.owasp.org/index.php/Main_Page
13. PTSecurity: cybersecurity threatscape 2018: trends and forecasts. https://www.ptsecurity.com/ww-en/analytics/cybersecurity-threatscape-2018/

14. Seo, J., Kim, H.S., Cho, S., Cha, S.: Web server attack categorization based on root causes and their locations. In: International Conference on Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004, vol. 1, pp. 90–96. IEEE (2004)
15. Wang, Y., Huang, M., Zhu, X., Zhao, L.: Attention-based LSTM for aspect-level sentiment classification. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 606–615 (2016)
16. Xie, Y., Yu, S.Z.: A large-scale hidden Semi-Markov model for anomaly detection on user browsing behaviors. IEEE/ACM Trans. Netw. **17**(1), 54–65 (2008)