An Attack Detection System for Multiple Web Applications Based on Big Data Platform

Xiaohui Jin[1,2], Congxian Yin[1,2], Pengpeng Yang[1,2], Baojiang Cui[1,2]
[1] School of Computer Science and Technology,
Beijing University of Posts and Telecommunications, Beijing, China
[2] National Engineering Laboratory for Mobile Network Security
Emails: jinxiaohui@bupt.edu.cn, yincongxian@foxmail.com,
yp1129_@bupt.edu.cn, cuibj@bupt.edu.cn

Abstract. Considering the protection requirements of large organizations for multiple web applications, we design and implement an attack detection system. The system is built on the big data platform, which is highly scalable. It adopts the network-traffic-based detection, capturing, parsing and analyzing the HTTP packets passing by in real time. By analyzing historical data, we are able to get application-specific access patterns, which can help domain experts find out anomalies efficiently. Besides, based on the labels given by domain experts, semi-supervised learning is applied to build attack detection classifier. The system is deployed in the real network of our university and has detected dozens of attacks.

1 Introduction

Large organizations, such as enterprises, colleges and governments, usually have to maintain multiple web applications. The traditional firewall and intrusion detection systems work at network layer, while web application firewall (WAF) just provides rule-based filtering, which cannot fulfill the security protection requirements of various web applications.

The development of big data technology makes it possible to gain massive storage and powerful computing ability at a relatively low cost. It is regarded as an ideal solution for coping with unknown attacks. By combining big data technology and machine learning algorithms in the field of intrusion detection, the protection ability of web applications can be effectively improved.

We design and implement an attack detection system based on big data platform, which can provide attack detection service for multiple web applications. The system is consisted of six subsystems, which are data collection, protocol parsing, big data storage, offline learning, online detection and visualization respectively. The system is deployed in the real network of our university——Beijing University of Posts and Communications (BUPT), providing attack detection service for multiple web applications maintained by the university. The statistic data shows that the system works stably and is able to detect dozens of attacks targeting at web applications.

L. Barolli et al. (eds.), *Advances on Broad-Band Wireless Computing, Communication and Applications*, Lecture Notes on Data Engineering and Communications Technologies 2, DOI 10.1007/978-3-319-49106-6_35

The following chapters are organized as follows: Chapter 2 introduces the related work, mainly focusing on the application of machine learning in the field of intrusion detection. Chapter 3 describes the system's overall architecture, and explains the functionality and implementation of each subsystem in detail. The conclusion and future work is in Chapter 4.

2 Related Work

The system described in this paper belongs to the category of intrusion detection system. As an important member of the security appliances, intrusion detection system [1] has a long history. It protects computer systems and networks from abuse, and is another generation of security technology following up the firewall. Intrusion detection is usually regarded as a classification issue since its main objective is to distinguish malicious access behavior. Machine learning is an effective way to solve such problems, and has been applied in the field of intrusion detection for a long period [2]. Supervised and unsupervised learning are two commonly used methods. Supervised learning is introduced in [3][4][5], and maximum entropy, support vector machine (SVM) and random forest algorithms are adopted respectively. In general, supervised learning requires a great deal of labeled training samples. Some researchers label the training sets manually, while a large number of other researchers tend to use some publicly available labeled sets, such as the famous KDD CUP ’99 data set. An unsupervised learning method is proposed in [6], while [7] makes attempt to combine the two learning methods. Overall, unsupervised learning is easily affected by the data distribution, whose effect is inferior to supervised learning.

With the development of the Internet, it is now quite easy to collect a large number of unlabeled samples. However, the number of labeled samples is relatively small, and the cost of manual annotation is very expensive. For those reasons, many researchers are turning to semi-supervised learning and active learning. The whole iterative process of semi-supervised learning needs no manual intervention. It is based on a small number of labeled samples and tries to make use of unlabeled data [8][9]. On the other hand, the general idea of active learning is to imitate the learning process of human. It extracts a small number of most uncertain samples, and asks domain experts for a correct label. Active learning classifier is established based on these samples, and is used to classify other unlabeled samples [10]. The system designed in this paper adopts semi-supervised learning. Meanwhile, we also draw lessons from the idea of active learning, and ask domain experts to judge normal access patterns and abnormal access behavior right before the semi-supervised learning starts.

From another point of view, the changing focus of machine learning research reflects the fact that the amount of available data is increasing dramatically. Big data era comes. It can provide adequate fuel for machine learning, so the combination of big data technology and machine learning can effectively improve the ability of existing intrusion detection system. Therefore, we argue that it is quite necessary to design and implement an attack detection system for a variety of web applications based on big data platform.

3 The Proposed System

By utilizing big data and machine learning technologies, we design and implement a system which can provide attack detection service for many kinds of web applications simultaneously. The system's overall architecture is shown in Figure 1. We will introduce the six subsystems respectively.

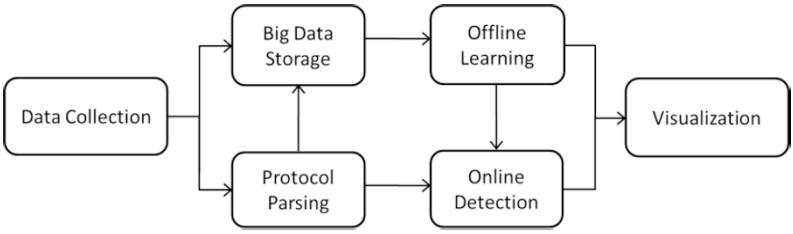

Fig. 1. The system's overall architecture

3.1 Data Collection Subsystem

The data collection subsystem adopts the client/server architecture. A high-performance distributed message queues named Kafka [11] is used in server side. Kafka is intended to cache text log messages originally. However, we make some extension in our system and use it to cache the high-speed network traffic.

The client monitors network traffic on the bypath. It consists of high-performance network traffic capture procedures and the producer client of Kafka. In practice, the server side is usually deployed close to the data source. Because of the client does not do any parsing, and its memory usage is optimized in the code level, the system can collect data at the rate of several gigabit per second.

3.2 Protocol Parsing Subsystem

The protocol parsing subsystem is used to transform the captured network traffic data to an object type, so that the following subsystem can read and process it. We define the HTTPSession class, its members cover all information of a HTTP session, including source IP, source port, destination IP, destination port, protocol type, as well as the main fields in request headers, request body, the main fields in response header, response body and response time. The object of HTTPSession class is the basic processing unit in the whole system.

The protocol parsing subsystem is based on the Spark Streaming [12]. First of all, it reads traffic packets cached in data collection subsystem using Kafka consumer client written in Scala, and uses the 5-Tuple (source IP, source port, destination IP, destination port, protocol type) as the key, packing and restoring the complete HTTP session data with join and reduce operations. Then HTTP protocol analysis module is called by the map operation to map network traffic data to HTTPSession object. To

improve performance, we implement HTTP protocol parsing module in C language, and use JNI to interact with Scala.

3.3 Big Data Storage Subsystem

The big data storage subsystem contains two storage platforms——HDFS and HBase.

HDFS is the open source implementation of Google File System (GFS) [13], and is suitable for storing large volume of unstructured data. HDFS is used for storing the original traffic data. The specific implementation is using Spark Streaming tasks to call Kafka consumer client and save the received data to HDFS.

Based on the design principle of Google Bigtable [14], HBase is a column-oriented distributed database. The subsystem uses HBase to storage HTTPSession object sent from protocol parsing subsystem. We build index for key fields, so that the offline learning subsystem can perform effective queries.

3.4 Offline Learning Subsystem

Offline learning subsystem is a machine learning platform based on the Spark MLlib [15], as is shown in Figure 2. By analyzing the historical web access data, we discover following rules:

(1) Normal access takes more than 90% of total access, and they are the high similar to each other.

(2) The access to specific domain and URI shows certain statistical characteristics in several dimensions, including depth, width, number of visits, duration, response status code and response time. The object that deviates from statistical center is more likely to be an attack.

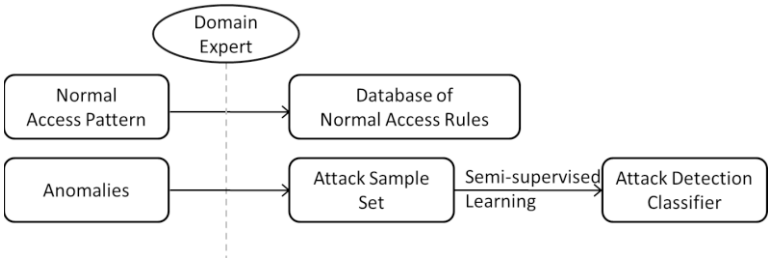

Fig. 2. The Offline Learning subsystem

For (1), we extract the URI and parameters from URL, and generalize it to some normal access patterns. Domain experts make judgment to the extracted normal access patterns to form the database of normal access rules.

The subsystem discovers anomalies according to (2), and submits the HTTPSession object to domain experts for further judgment. The objects marked as attack by domain experts will be added into the attack sample set. The subsystem extracts

features from attack sample set and use semi-supervised learning algorithm to train intrusion detection classifier.

3.5 Online Detection Subsystem

The online detection subsystem uses misuse-based and anomaly-based detection methods jointly. Figure 3 shows the work flow.

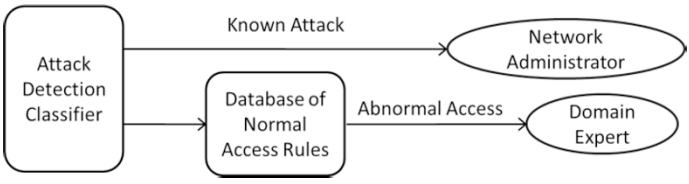

Fig. 3. The Online detection subsystem

Newly-received HTTPSession objects first go through the attack detection classifier. The subsystem informs network administrators if any known attack is detected. Other normal HTTPSession objects will be passed to the database of normal access rules. If an object fails matching the rules, the object will be treated as an abnormal access and submitted to domain experts for further judgment. If domain experts mark it as an attack, then the object will be added to the attack sample set.

3.6 Visualization Subsystem

The main purpose of visualization subsystem is to provide a convenient operation interface to network administrators and domain experts. We will neglect the details here.

4 Conclusion and Future Work

Generally speaking, the system mainly has three advantages. First of all, being built on big data platform, the system can effectively parse web application protocols and extract access patterns leveraging massive storage and powerful processing ability. Secondly, the results of big data analysis make the labeling work of domain experts more specific and efficient. Besides, semi-supervised learning method is very suitable for such application scenarios. Finally, the system uses the misuse-based and anomaly-based detection jointly, which reduces false alarms and improves the recall rate at the same time.

Our statistic data shows that the system processes about 1.8 million web requests per day, and the proportion of detected web attacks in all requests is about 1%. SQL injection, directory traversal and cross site scripting (XSS) are the most common attacks.

Moreover, a lot of work can be done to improve the system. The future work will be carried out in the following two aspects. First, web protocol should be parsed in depth. Currently, the offline learning subsystem can only extract simple request parameters. The parameters coded in JSON or XML format cannot be parsed yet. Besides, IPv6 and HTTPS support should also be involved. Second, the feature selection strategy needs to be optimized. In order to obtain a better performance of classifiers, it is indispensable to do in-depth research jobs on the theoretical knowledge involved in this field.

References

1. J Mchugh，A Christie，J Allen. Defending Yourself: The Role of Intrusion Detection Systems. IEEE Software, 2000, 17(5):42-51
2. D Barbará，J Couto，S Jajodia，L Popyack，N Wu. ADAM: Detecting Intrusions by Data Mining. Proceedings of the IEEE Workshop on Information Security, 2001:11--16
3. Y Gu, A Mccallum, D Towsley. Detecting anomalies in network traffic using maximum entropy estimation. ACM Sigcomm Conference on Internet Measurement, 2005:345-350
4. J Yu, H Lee, MS Kim, D Park. Traffic flooding attack detection with SNMP MIB using SVM. Computer Communications, 2008, 31(17):4212-4219
5. J Zhang, M Zulkernine. A Hybrid Network Intrusion Detection Technique Using Random Forests. International Conference on Availability, 2006, 37(8):262-269
6. G Jia, G Cheng, DM Gangahar，DK Agrawal. Traffic anomaly detection using k-means clustering. In. GI/ITG workshop MMBnet
7. SR Gaddam, VV Phoha, KS Balagani. K-Means+ID3: A Novel Method for Supervised Anomaly Detection by Cascading K-Means Clustering and ID3 Decision Tree Learning Methods. IEEE Transactions on Knowledge & Data Engineering, 2007, 19(3):345-354
8. X. Zhu. Semi-supervised learning literature survey. Technical Report 1530, Department of Computer Sciences, University of Wisconsin at Madison, Madison, WI, Apr. 2006.
9. O. Chapelle, B. Schölkopf, A. Zien, eds. Semi-Supervised Learning, Cambridge, MA: MIT Press, 2006
10. M Almgren, E Jonsson. Using active learning in intrusion detection. Proceedings of the 17th IEEE Computer Security Foundations Workshop (CSFW’04)
11. J Kreps, L Corp, N Narkhede, J Rao, L Corp: Kafka: a distributed messaging system for log processing. NetDB'11, Athens, 2011
12. http://spark.apache.org/streaming/
13. S Ghemawat: The Google file system. ACM SIGOPS Operating Systems Review, 2003, 37(5):29-43
14. F Chang, J Dean, S Ghemawat, WC Hsieh, DA Wallach: Bigtable:a distributed storage system for structured data. ACM Transactions on Computer Systems, 2008, 26(2):205--218
15. http://spark.apache.org/mllib/