SeEagle: Semantic-Enhanced Anomaly Detection for Securing Eagle

Wu Xin[1,3], Qingni Shen[2,3], Yahui Yang[2,3], and Zhonghai Wu[2,3](✉)

[1] School of Electronics and Computer Engineering, Peking University, Shenzhen, China
xinwu@pku.edu.cn
[2] School of Software and Microelectronics, Peking University, Beijing, China
{qingnishen,yhyang,wuzh}@ss.pku.edu.cn
[3] Lab for Big Data Technology, Peking University, Beijing, China

Abstract. In order to ensure data security and monitor data behavior, eBay has developed Eagle, which can detect anomalous user behavior based on user profiles and can intelligently protect data security of Hadoop ecosystem in real-time. By analyzing the kernel density estimation (KDE) algorithm and source code implemented in Eagle, we recognize that there are two security risks: One is that user profiles are models of operations, but the objects of operations are not analyzed; The other is that the owner of HDFS audit log files is not authenticated. Consequently, the attacker can bypass Eagle and form attack of APT combined with default permissions of Hadoop. In this paper, we analyze the two risks of Eagle, propose two kinds of attack methods that can bypass anomaly detection of Eagle: co-frequency operation attack and log injection attack, and establish threat model of which feasibility is verified experimentally. Finally, we present SeEagle, a semantic-enhanced anomaly detection for securing Eagle, including user authentication and file tagging modules. Our preliminary experimental evaluation shows that SeEagle works well and extra overhead is acceptable.

Keywords: Semantic-enhanced · User authentication · Tagging · APT
User profile · Eagle · Anomaly detection · User activity monitoring
Machine learning

1 Introduction

In recent years, Hadoop [1] has become the most popular distributed system in both industry and academia. For data security, HDFS provides access control to prevent unauthorized access to file data. But in the era of big data, the access control is facing significant challenges [2]: To partition roles for users and to define permissions for roles is difficult.

In response to the challenges of data access control in the era of big data, Molloy et al. [3] proposed to extract roles from the access logs, based on machine-learning algorithms. Zeng et al. [4] proposed an access control model based on the content. Their methods are mostly verified by experimental prototypes, but they are not being in practice. Gupta et al. [5] designed Eagle [6], which can further ensure the security of

HDFS data through user profile-based anomaly detection. Eagle, which has aroused widespread concern in both industry and academia, has been announced to be a Top Level Project (TLP) of Apache Software Foundation (ASF) [7].

The idea of Eagle is extracting audit logs from applications running on Hadoop systems, such as HDFS which is concerned in this paper, and using machine-learning algorithms to generate user profiles depending on the users' history logs. Based on user profiles, Eagle can detect malicious activities when a user action does not match with the user profile.

Several approaches dealing with anomaly detection for operating system, networks, Web applications and database have been developed, but the behaviors deemed malicious for HDFS are not necessarily malicious for them.

In the database domain, Karma et al. [8] and Spalka and Lehnhardt [9] proposed the method of detecting anomalies respectively. Their work is complementary. [8] focuses on the syntactic aspects by detecting anomalous access patterns in a DBMS, while [9] focuses on the semantic aspects of the SQL queries. So a mature anomaly detection system that designed to better monitor user behaviors should focus on both syntactic and semantic aspects.

However, we notices that the approach in Eagle is closer to that of [8], which both use machine-learning algorithms and focus on syntactic aspects, but there is the lacks of sematic analysis and the authentication of log files owner. If the risks cannot be effectively resolved, they may form the data security issues and attack of APT. Therefore, we propose SeEagle, a semantic-enhanced anomaly detection for securing Eagle, to deal with the risks.

The contribution of this paper can be summarized as follows:

- By analyzing the machine-learning algorithms, we realize that user profiles are models of user operations for operated files, and the KDE algorithm, which only statistically analyzes user operations and does not analyze the objects of operations, focuses on the syntactic analysis.
- Through the analysis of source code, tracking the processes of reading and processing HDFS audit log data in Eagle, we observe that the owner of log files is not authenticated during the process of HDFS log data flow into.
- Based on the two security risks and combined with the default permissions of Hadoop, co-frequency operation attack and log injection attack (see Sect. 2.2) which can bypass the anomaly detection of Eagle are proposed. And the threat model is established to verify their feasibility.
- In order to deal with the two kinds of attack methods, SeEagle,a semantic-enhanced anomaly detection for securing Eagle, is proposed. Based on the general policy framework of Eagle, the user authentication module is added to the entrance of log data flow, and the file tagging module, which based on semantic analysis, is added to the offline training that generate user profiles. Finally, SeEagle which is evaluated experimentally can effectively defend against the attacks and the extra overhead is acceptable.

The paper is organized as follows. Next section analyzes the security risks of Eagle and describes two kinds of attack methods. Section 3 describes SeEagle and shows the results of the experimental evaluation. Finally, we conclude the paper by discussing future work.

2 Challenges of Eagle

2.1 Security Risks

A. The lack of semantic analysis

Through the analysis of machine-learning algorithms in Eagle, we realize that there is a security risk in the offline training of KDE algorithm which lacks semantic analysis. The idea of KDE algorithm is to calculate the probability density of sample data points to evaluate each user by the Gaussian distribution function [10].

By analyzing the KDE algorithm, it is understood that only user operations are analyzed statistically while the objects of operations are not.

For a HDFS user, the HDFS files can be categorized into authorized files and unauthorized ones. Authorized files can be divided into operated files and non-operated files. Figure 1 shows the categorization of HDFS files.

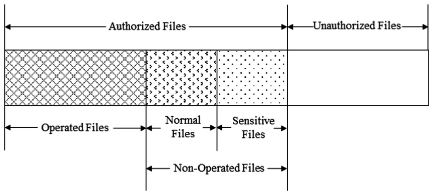


Fig. 1. The categorization of HDFS files

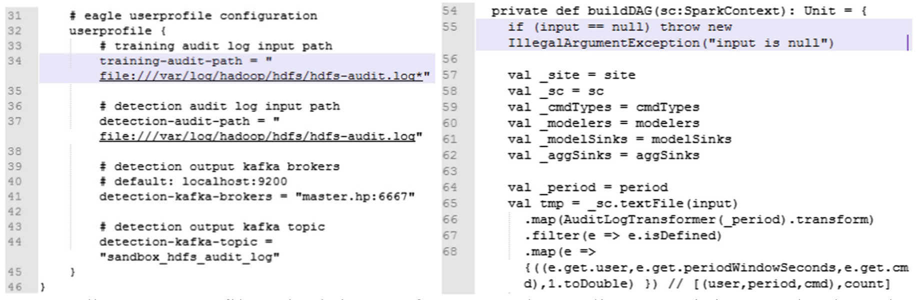
Through the analysis of the machine-learning algorithms in Eagle, it is learned that user profiles are models of operations for operated files. User profiles can effectively detect anomaly for operated files, but they may not defend against the internal threats for non-operated files because the operations for the former may be abnormal for the latter, especially for the sensitive data.

B. The lack of log files owner authentication

By analyzing the source code, there is also a security risk in the process of reading and analyzing HDFS audit logs: the owner of HDFS log files is not authenticated. We illustrate the process of reading HDFS logs as follows:

Configure the path for training dataset of user profiles in the *conf/sandbox-user-profile-scheduler.conf* in the Eagle home directory. From the 34th line in Fig. 2(a), we can know that the training dataset of user profiles is all local HDFS log files whose names start with *hdfs-audit.log* in */var/log/hadoop/hdfs/*directory.

User profiles are generated through reading and analyzing the HDFS logs in *AuditLogTrainingSparkJob.scala*. From the 55th and 65th lines in Fig. 2(b), we can learn that Eagle only judges whether the path is empty, and then reads and analyzes the HDFS logs. However, the owner of HDFS audit log files is not authenticated.



```
31      # eagle userprofile configuration
32      userprofile {
33          # training audit log input path
34          training-audit-path = "
            file:///var/log/hadoop/hdfs/hdfs-audit.log*"
35
36          # detection audit log input path
37          detection-audit-path = "
            file:///var/log/hadoop/hdfs/hdfs-audit.log"
38
39          # detection output kafka brokers
40          # default: localhost:9200
41          detection-kafka-brokers = "master.hp:6667"
42
43          # detection output kafka topic
44          detection-kafka-topic =
            "sandbox_hdfs_audit_log"
45      }
46  }
```

a. sandbox-userprofile-scheduler.conf

```
54    private def buildDAG(sc:SparkContext): Unit = {
55      if (input == null) throw new
        IllegalArgumentException("input is null")
56
57      val _site = site
58      val _sc = sc
59      val _cmdTypes = cmdTypes
60      val _modelers = modelers
61      val _modelSinks = modelSinks
62      val _aggSinks = aggSinks
63
64      val _period = period
65      val tmp = _sc.textFile(input)
66        .map(AuditLogTransformer(_period).transform)
67        .filter(e => e.isDefined)
68        .map(e =>
          {((e.get.user,e.get.periodWindowSeconds,e.get.cm
          d),1.toDouble) }) // [(user,period,cmd),count]
```

b. AuditLogTrainingSparkJob.scala

Fig. 2. The source code of Eagle

2.2 Attack Methods

We propose two kinds of attack methods based on the above two security risks:

- *Co-frequency operation attack*: Due to the lack of semantic analysis in Eagle, the malicious behavior that the objects of operation are different can be performed based on the same frequency of operation when an attacker obtains the authority of a legitimate user.
- *Log injection attack*: As Eagle lacks log owner authentication, the attacker can forge the HDFS audit logs according to the operational requirements of getting the HDFS data, and inject them into the Eagle. Once the mendacious user profile is generated, it will cause failure of anomaly detection.
- The relationship between co-frequency operation attack & log injection attack: The former is invalid when the conventional operations in the user profile cannot meet the needs of the attacker. The latter is needed to generate mendacious user profile to meet the operational requirements of the former.

3 SeEagle

3.1 Overview

According to the security risks in Eagle and the two kinds of attacks proposed in this paper, SeEagle, a semantic-enhanced anomaly detection for securing Eagle, has been designed as shown in Fig. 3, including the user authentication and file tagging modules.

The user authentication module is used to defend the log injection attack. We exploit the HDFS audit logs can only be generated by *hdfs* that is the super user of HDFS. Therefore, we increase the user authentication module to authenticate the owner of the HDFS log files whether *hdfs*, which can effectively defend against the log injection attack.

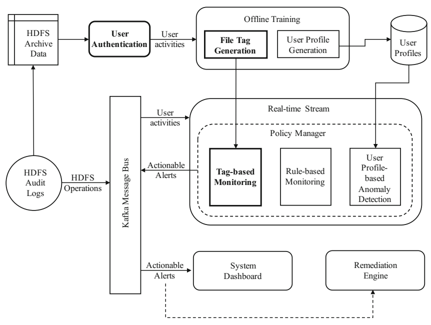


Fig. 3. SeEagle architecture

The file tagging module, which is based on the semantic analysis, is used to protect co-frequency operation attack. In the process of offline training, not only the user operations are statistically analyzed, but also the operated files of the user are tagged with the user name. Then a default policy that an alert is triggered when a user accesses any file without tag of the user name is created for each user through the general policy management framework of Eagle.

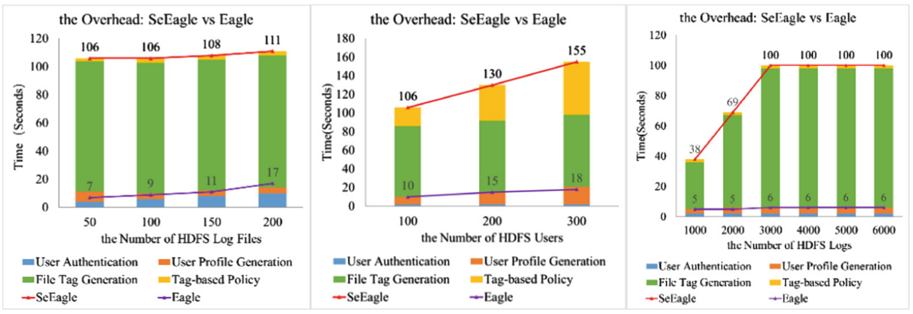
The file tagging can effectively protect from the co-frequency operation attacks to access the non-operated files. However, it still cannot avoid the co-frequency operation attack to access the operated files.

In order to protect the operated files from co-frequency operation attack, the default permissions of HDFS log directory and files should be changed and the log files should be defined more granular ACL to prevent the attacker from acquiring the HDFS logs.

3.2 Experimental Evaluation

We mainly from three aspects to test the Eagle and SeEagle overhead: the number of HDFS log files, the number of HDFS users and the number of HDFS logs in Hadoop system. From a large number of experimental data, we draw the following three charts in Fig. 4 to illustrate.

Through the analysis above, we observe that the extra overhead of SeEagle mainly in the generation of file tags and tag-based policies. By combining source code and log output analysis, it is realized that the main overhead is I/O. Considering that on the basis of Eagle, SeEagle has improved its security and has no effect on the performance of online detection anomalies, and the extra overhead is mainly in off-line training. So the extra overhead of SeEagle is acceptable.



a. The number of log files b. The number of users c. The number of logs

Fig. 4. The overhead: SeEagle vs Eagle

4 Conclusions and Future Work

In this paper, we aware the security risks by analyzing the machine-learning algorithms and the source code in Eagle and propose co-frequency operation attack and log injection attack that can bypass anomaly detection of Eagle and form the attack of APT combined with the default permissions of the Hadoop. Finally, we present SeEagle, a semantic-enhanced anomaly detection for securing Eagle, including the user authentication and the file tagging modules. The SeEagle cannot only effectively defend against the above two kinds of attacks, but also the extra overhead is acceptable.

In the future, we plan to further research the response of Eagle when an anomaly is detected. At present, Eagle just generates an alert and informs the related person by e-mail after detecting an abnormal user behavior. It just makes a response after the occurrence of abnormal events rather than making judgment in advance. In addition, during the offline training, the logs of abnormal behavior are regarded as the regular HDFS logs to generate user profiles and Eagle cannot remove them from the HDFS logs. Therefore, we intend to add the appropriate function so that Eagle can generate more accurate user profiles.

Acknowledgements. This work is supported by the National High Technology Research and Development Program (“863” Program) of China under Grant No. 2015AA016009 and the National Natural Science Foundation of China under Grant No. 61232005. The authors would like to acknowledge Xiaoyi Chen, Bin Yang, Dong Huo and Xuxin Fan for their support for our preliminary experiments. We are also grateful to Fenmei Li for her valuable suggestions and thorough proofread for this paper.

References

1. Hadoop. https://hadoop.apache.org/
2. Feng, D.G., Zhang, M., Li, H.: Big data security and privacy protection. Chin. J. Comput. **37** (1), 246–258 (2014)

3. Molloy, I., Park, Y., Chari, S.: Generative models for access control policies: applications to role mining over logs with attribution. In: ACM Symposium on Access Control Models and Technologies, pp. 45–56 (2012)
4. Zeng, W., Yang, Y., Luo, B.: Access control for big data using data content. In: IEEE International Conference on Big Data, pp. 45–47 (2013)
5. Gupta, C., Sinha, R., Zhang, Y.: Eagle: user profile-based anomaly detection for securing Hadoop clusters. In: IEEE International Conference on Big Data, pp. 1336–1343 (2015)
6. Eagle. http://eagle.apache.org/
7. Apache Software Foundation (ASF). http://www.apache.org/
8. Kamra, A., Terzi, E., Bertino, E.: Detecting anomalous access patterns in relational databases. VLDB J. **17**(5), 1063–1077 (2008)
9. Spalka, A., Lehnhardt, J.: A comprehensive approach to anomaly detection in relational databases. In: Jajodia, S., Wijesekera, D. (eds.) DBSec 2005. LNCS, vol. 3654, pp. 207–221. Springer, Heidelberg (2005). https://doi.org/10.1007/11535706_16
10. Gaussian Distribution. https://en.wikipedia.org/wiki/Gaussian_function