An Insider Threat Detection Method Based on User Behavior Analysis

Wei Jiang, Yuan Tian, Weixin Liu, Wenmao Liu

An Insider Threat Detection Method based on User Behavior Analysis

Wei Jiang1,2, Yuan Tian1, Weixin Liu3, Wenmao Liu3
1Beijing University of Technology, No. 100 Pingleyuan, Chaoyang District, Beijing, China,
jw@bjut.edu.cn, ty_cdream@163.com
2Chinese Academy of Cyberspace Studies, Beijing, 100010 China
3NSFOCUS Information Technology, No. 4, Beiwa Street, Haidian District,
Beijing, China,
jack18jack@gmail.com

ABSTRACT. Insider threat has always been an important hidden danger of information system security, and the detection of insider threat is the main concern of information system organizers. Before the anomaly detection, the process of feature extraction often causes a part of information loss, and the detection of insider threats in a single time point often causes false positives. Therefore, this paper proposes a user behavior analysis model, by aggregating user behavior in a period of time, comprehensively characterizing user attributes, and then detecting internal attacks. Firstly, the user behavior characteristics are extracted from the multi-domain features extracted from the audit log, and then the XGBoost algorithm is used to train. The experimental results on a user behavior dataset show that the XGBoost algorithm can be used to identify the insider threats. The value of F-measure is up to 99.96% which is better than SVM and random forest algorithm.

KEYWORDS: Insider threat, user behavior, machine learning

1 Introduction

In recent years, insider threats have become a critical issue in the field of information security, which can have a serious impact on the organization. Insider threats are individuals or organizations that have a legitimate right to access an organization's internal system and pose a threat to the organization.

Insider threats have the following characteristics: transparency, concealment and high-risk. Insider threat detection is more difficult than many other anomaly detection problems, as insiders are often familiar with the company's information system and can easily circumvent the detection of safety equipment. In addition, malicious act by insiders is often hidden in a large number of normal activities which is difficult to detect. And more importantly, insiders often master the core assets of the organization. As a result, the damage is enormous even if the number of insider threats is much smaller than the external attack. CERT database shows that insider threats cause an

average loss of 1.7 million dollars, so the threat posed by insiders requires serious attention.

This research is accomplished through the analysis of user audit log data. In this paper, an insider threat detection model based on user behavior analysis is proposed, which can detect the attack behavior or potential threat behavior of users with certain privileges within the company. The proposed system is divided into three parts: feature extraction for original log data, feature re-extraction based on aggregated data and classifier training.

The key contributions of this article are as follows: We aggregate user behavior events over a period of time as subsequent detection data to avoid detecting false positives caused by individual event streams. In addition, this model overcomes the problem of information loss in traditional feature extraction. We also solved the problem of data imbalance and effectively reduced the false alarm rate of insider threat detection. As far as we know, this is the first time to use the XGBoost algorithm for insider threat detection.

The remainder of this paper is organized as follows: Section 2 reviews prior literatures related to user behavior analysis and insider threat detection. Section 3 explains our feature extraction methods and detection algorithms. The experimental results are presented in Section 4. Finally, Section 5 concludes this article.

2 Related work

To address the challenges of insider threat, the research community has proposed various systems and models. They first put forward the conceptual problem of what insider threat is and the conceptual model of insider behavior.

Some scholars have trained a specific attack behavior as a user model. Helen Ashman(2012) proposes that attributes such as an attacker keystroke type, web browsing behavior, or a preference application are stored in the user model to capture the user's behavior and accurately describe an individual user. Liu Xuan(2009) considers the possible destructive behavior by establishing user behavior Library. Yifeng Lian(2002) proposes a model based on recursive correlation function, which detects anomalies in user behavior according to the comparison similarity between user's historical behavior pattern and current behavior pattern. By analyzing the network data packet, Liping Wang excavates the frequent behavior patterns in the network system, and uses the pattern similarity to detect the behavior of the system, and then automatically establishes the pattern Library of abnormal and misuse behavior(2004). They only consider the abnormal act that exists for each timestamp. However, the user's unusual behavior often occurs within a time period. Exceptions that occur at a single time step tend to produce false positives.

Gamachchi(2017) proposes an insider threat detection framework, which utilizes the attributed graph clustering techniques and outlier ranking mechanism for enterprise users. Empirical results also confirm the effectiveness of the method by

achieving the best area under curve value of 0.7648 for the receiver operating characteristic curve. To classify users, Kandias(2013) applied Naive Bayes, SVM, and logistic regression algorithms on their dataset and evaluated using precision, recall, F-score and accuracy as metrics. Logistic regression gave the highest scores, achieving 81% for both F-score and accuracy. But the detection accuracy of these algorithms is not enough high.

Some researchers construct a behavioral model of user behaviour and built a "baseline" model for each user. To do this, they tracked a series of system-related activities, such as activities related to the Windows registry, access to various dynamic-link library (DLL) actions, creating processes and terminating processes, and so on, and then using OCSVM to identify the exception user. They have tested and evaluated individual detectors individually, but have not yet created an integrated End-to-end solution. (Bowen, et al. 2009, Maloof, et al. 2007). Eberle(2010) proposes a method of detecting insider threat by developing a graphical anomaly detection algorithm. They present an anomaly detector based on the modification of the activity in the graph. However, the ability to discover the anomalies is sometimes limited by the allocated resources.

Myers(2009) considers how to use web server log data to identify malicious insiders who want to exploit internal systems, but he ignores user behavior characteristics. Eldardiry(2013) also proposes an insider threat detection system based on user activity feature extraction. However, they did not consider role-based evaluation. Rashid(2016) uses hidden Markov models to learn what normal behavior is and then use them to detect deviations from normal behavior. The results show that this method has successfully detected the insider threat, but he ignores the specific operation of each user's behavior, resulting in imperfect detection rate. Andropov(2017) proposes a method of identifying and classifying network anomalies, using the artificial neural network to analyze the data, describing the potential anomalies and their characteristics, and using the multilayer perceptron to train with the reverse propagation algorithm. The output of the neural network shows whether there is an exception. But the neural network algorithm is apt to produce the over fitting problem.

3 Our approach

Audit log mainly involves system logon/logoff, file access, device usage, HTTP access, mail sending and receiving records. In addition, we take full advantage of the LDAP (Lightweight Directory Access Protocol) file which records all the user metadata for each month in the organization, such as their role in the organization, the projects assigned to them, and the team they are working on.

We first merge the events in each audit log and sort them chronologically. The features are then extracted in two ways: based on the features of the new observations and statistical features. We need to establish a baseline of normal behavior for each user and each role, and get an exception value when deviating from the baseline for

the new observations. The behavior of each user in a period of time is then aggregated into an event packet, and the statistical feature dimension is obtained while aggregating the event.

3.1 Feature extraction

Insider threat detection mainly analyzes the abnormal behavior of internal personnel, such as employee's unauthorized behavior, malicious attack, etc. These abnormal behaviors consist of two main categories: comparisons between the user's daily activity and their previous activity, comparisons between the user's daily activity and the previous activity of their role. Since the day-to-day behavior of each user consists of a series of events, we define a set of rules based on each type of event in the dataset that takes into account the use of new devices, the execution of new activity types, and new operations for an activity and each exception value can be interpreted as a specific exception behavior, which is significant for insider threat detection. For example, the logon system time can indicate a user's behavior anomaly, which could be an unusual behavior if the user suddenly logs in late at night.

3.1.1.Form baseline

First, we select n events that do not contain malicious behavior as a baseline. For example, the user's normal operating time set is u_t, means that normal operation time should be in the working hours, we will get a time interval according to historical behavior. If operations from 7 to 19 are considered as normal behavior, then u_t = {7~19}. Our rule set also includes u_{pc}, u_{type}, $u_{activity}$. u_{pc}, u_{type} and $u_{activity}$.

u_{pc}: a set of fixed computer used by each user.

u_{type} (for example, access to a web page): a set of all types of operations performed by a user on the computer.

$u_{activity}$ (download, upload, visit page):a set of specific activities for each type of operation performed by the user in the device.

After analyzing the first n events that are not marked by malicious behavior, we establish the baseline for all users.

For example, a user u_1 does a type of activity on pc_1 at t_1, we first look at the rules table of u_1, if $pc_1 \notin u_{pc}$ or $t_1 \notin u_t$, we will set the corresponding dimension value to 1. Similarly, we establish baselines for each role.

3.1.2.Generate fundamental eigenvector

We have also extracted statistical based features for each user and each role, counting the number of times a user uses each specific device or type of activity, such as the number of times a user visit the Web page. We divide the feature dimension based on statistics into six categories, as shown in table 1：

Table 1. *The feature dimension based on statistics*

Index	**Feature**
1	number of logon/logoff

Index	Feature
2	number of downloads (page uploads, page visits)
3	number of emails sent(recipients, attachments, address
4	number of removable media connect (disconnect)
5	number of files copy(delete, move)
6	number of devices used for one particular activity

The feature matrix we are considering has 68 columns. All rules are mainly evaluated in three sections: the statistics for the user's current behavior, the user's comparison with the previous activity and the comparison between the user and the activities of their role. The latter two parts are compared to the baseline, if within the baseline, the feature value is 0. However, if it exceeds the baseline, then the feature value is 1, indicating a new exception value. And then we get fundamental eigenvector as is shown in Figure 1 and [1].

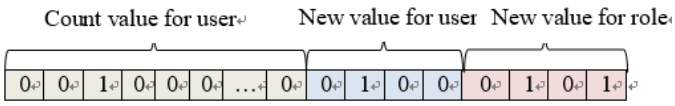

Figure 1. *Fundamental eigenvector*

$$F_u = (f_u counts , f_u new, f_r new) \quad [1]$$

3.1.3.Get aggregation eigenvector

Each basic eigenvector represents the user's behavior at a given moment, and then aggregates all the basic eigenvectors based on time T to get the user's behavioral characteristics over time. The aggregation eigenvectors are shown in Figure 2.

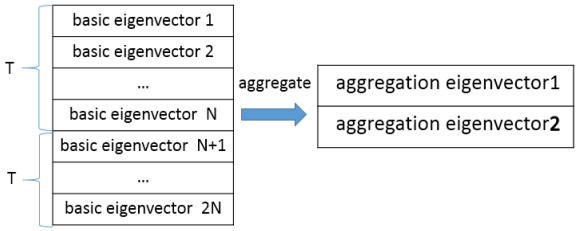

Figure 2. *Aggregation in time T*

Finally, each aggregation feature vector that is aggregated in time T is shown in Figure 3.

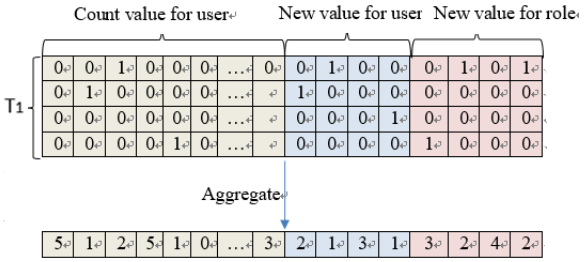

Figure 3. *Aggregation eigenvectors*

As a result, we successfully converted the original log into a digital eigenvector and aggregated it by time T.

3.2 Detection algorithm

Internal attacks only account for a tiny proportion of all the behaviors conducted by all users and it is hard to obtain enough samples for training. Thus, we use smote algorithm to deal with the unbalanced data, using XGBoost (extreme gradient boosting) algorithm for insider threat detection, so as to achieve a good detection effect.

In order to solve the problem of imbalanced data, sampling, weight adjustment and kernel function correction are generally used. And the sampling is divided into over-sampling and under-sampling. The disadvantage of weight adjustment is that it cannot control the proper weight ratio which needs several attempts. The use of kernel function correction is very limited and the adjustment cost is high. However, sampling will result in loss of data information, so we chose an over-sampling algorithm smote. The main idea is to increase the number of samples by inserting new synthetic samples into the sparse samples, so that the dataset is balanced.

Table 2 shows the sum of normal samples and negative samples. We can see that positive and negative samples are seriously unbalanced, so we need to use the smote algorithm to get the balanced dataset before training model.

Table 2. The *sum of normal and abnormal samples*

Sum	***Normal***	***Anomaly***	***Normal :Anomaly***
135,117,169	135,116,741	428	315,693

4 Experiment

We use the r6.2 dataset provided by CMU-CERT, which covers 135,117,169 behavior events for 4,000 users over 516 days. The organization activity log consists of five different files, which correspond to five different executable activities: Login, USB device, email, web access, and file access. Each record contains a timestamp, a user ID, a device ID (that is, what device recorded the action), and a specific activity name (for example, logon/logoff, file upload/download). The dataset includes malicious behavior that the expert manually injected.

The dataset also contains LDAP (Lightweight Directory Access Protocol) files, which record all user metadata for each month in the organization, such as their role in the organization, the projects assigned to them, and the team they are working on.

As the log files have a large amount of data, we use Spark, a big data computing framework, to preprocess it. First, all five log files are read and stored as a spark dataframe, and then all the log data is merged. As the dataset has injected malicious internal attacks, we mark the ' if_insider ' tag of all attack events as ' 1 ' and the normal events are marked '-1 '. Each row represents an event, including the user name, the user's role, event ID, date, the ID of the PC used, the type of activity, the specific operation of the activity, the specific attribute information for the activity (such as the recipient of the sending mail activity, the sender, and the content of the message, etc.).

All event streams are sorted in chronological order, then the first 6906662 events which do not contain malicious behavior are selected as baseline.

4.1 Evaluation of algorithms

We first compare the performance metrics of different algorithms. We aggregate the event flow in T(T=3600s) and get 22,083,308 normal samples and 1292 anomaly samples. In the following experiment, we select user behavior data conducted by those whose role is salesman. We use the Python language to implement our model. First, we use the smote algorithm to synthesize the abnormal samples proportionally, so that the ratio of normal sample to anomaly sample is 1:1. In this paper, the grid search method is used to realize the systematic traversal of multiple parameter combinations, and the best parameters are determined by Cross-validation as shown in table 3.

Table 3. *XGBoost parameters*

child_weigh	***max_depth***	***gamma***	***subsample***	***colsample_bytree***
1	6	0.2	0.9	0.8

In order to avoid excessive learning and the lack of learning, we have k-fold cross-validation, we use k-fold cross-validation[24] which divide the original data into k independent subsets, each time a subset is used as a validation set, the rest of the k-1 subset of data as a training set. Then we get k models, and the average accuracy rate of the k models is used as the performance index of the classifier under this K-CV. In this paper, the value of k is 5. Table 4 shows the detection performance of three algorithms.

Table 4. *Performance of three algorithms*

Detection Method	**Accuracy**	**Precision**	**Recall**	**F1**
RandomForest	77.10%	86.82%	90.41%	88.58%
mlpc	83.11%	92.41%	94.12%	93.26%
XGBoost	99.13%	98.34%	98.21%	98.27%

As can be seen from the table above, the performance of XGBoost algorithm is better than random forest algorithm and multilayer perceptual classifier algorithm.

We have randomly selected a small portion of events conducted by salesman as the training set. Then we use the rest of the data as a validation set to validate the performance of the XGBoost algorithm. We also validate the algorithm using behavior

data of the users whose roles are Electrical engineer and IT Admin, respectively. The test performance can be shown in table 5.

Table 5. *Detection performance of different roles*

Role	Accuracy	Precision	Recall	F1
Salesman	99.13%	98.34%	98.21%	98.27%
Electrical engineer	95.54%	95.54%	100%	99.96%
IT Admin	97.70%	97.73%	99.97%	98.84%

4.2 Evaluation of different aggregation interval

We aggregate the event flows in different intervals t (10minutes, half an hour, one hour and one day), and then use the XGBoost algorithm to compare the performance of different aggregation intervals. We take Salesman for example. Table 6 shows performance of different aggregation intervals.

Interval	Accuracy	Precision	Recall	F1
Ten minutes	93.37%	93.75%	92.31%	93.02%
Half an hour	96.65%	96.08%	94.12%	96.45%
one hour	99.13%	98.34%	98.21%	98.27%
one day	90.89%	91.77%	89.10%	90.41%

Table 6. *Performance of different aggregation interval*

As can be seen from the results in table 6, the aggregation of eigenvectors in hours can achieve the best results, which means that all activities within one hours can fully represent the behavioral characteristics of the employee. Aggregations per half hours are slightly less effective. The accuracy and recall rates of aggregation in days are the lowest because users may contain both normal and abnormal behavior within one day, which may cause false positives.

5 Conclusion

In our paper, we use the smote algorithm to improve the ratio of abnormal events so as to obtain a more balanced dataset. And we propose a user behavior analysis model, by aggregating user behavior in a period of time. In order to improve the detection rate of abnormal events in unbalanced training data. Our model can achieve a high recall of 100% and F1 of 99.96% which is much better than the compared methods. The results show that the model can achieve good detection effect. In the future, we will use real company traffic flow to achieve real-time traffic collection and detection, thus strengthening our model.

6 Acknowledgements

The authors would like to thank the anonymous reviewers for their detailed reviews and constructive comments, which help improve the quality of this paper. Supported by Beijing Natural Science Foundation under Grant No.4172006, General Program of Science and Technology Development Project of Beijing Municipal Education Commission of China under Grant No.km201410005012, the Key Lab of Information Network Security, Ministry of Public Security, Humanity and Social Science Youth foundation of Ministry of Education of China under Grant No. 13YJCZH065; General Program of Science and Technology Development Project of Beijing Municipal Education Commission of China under Grant No. km201410005012; Open Research Fund of Beijing Key Laboratory of Trusted Computing, Open Research Fund of Key Laboratory of Trustworthy Distributed Computing and Service (BUPT), Ministry of Education.

References

1. Grant Pannell, Helen Ashman. "Anomaly detection over user profiles for intrusion detection", University of South Australia, 2012
2. Xuan liu, Fengli Zhang, Li Ye. "User behavior mining algorithm design based on NetFlow", computer application Research, 2009, 26 (2):319-321
3. Yifeng Lian, Yingxia Dai, Hang Wang. "User behavior anomaly detection based on pattern Mining", Journal of Computer Science, 2002 , 25 (3) :325-330
4. Liping Wang, Na an, Xiaonan Wu, Dingyi Fang. "Behavior pattern mining in Intrusion Detection system", Journal of Communications, 2004, 25 (7):168-175
5. J. B. Camina, C. Hernandez-Gracidas, R. Monroy, and L. Trejo, "The Windows-Users and-Intruder simulations Logs dataset (WUIL): An experimental framework for masquerade detection mechanisms," Expert Systems with Applications, vol. 41, no. 3, pp. 919–930, 2014
6. A Gamachchi, S Boztas, "Insider Threat Detection Through Attributed Graph Clustering", Trustcom/bigdatase/icess, 2017 :112-119
7. M. Kandias, V. Stavrou, N. Bozovic, L. Mitrou, and D. Gritzalis, "Can we trust this user? predicting insider's attitude via youtube usage profiling", in Ubiquitous Intelligence and Computing, 2013 IEEE 10th International Conference on and 10th International Conference on Autonomic and Trusted Computing (UIC/ATC). IEEE, 2013, pp. 347–354.
8. B.M.Bowen,M.Ben Salem, S.Hershkop, A.D.Keromytis, and S. J. Stolfo, "Designing host and network sensors to mitigate the insider threat," IEEE Secur. Privacy, vol. 7, no. 6, pp. 22–29, Dec. 2009.
9. M. A. Maloof and G. D. Stephens, "ELICIT: A system for detecting insiders who violate need-to-know," in Recent Advances in Intrusion Detection. New York, NY, USA: Springer, 2007, pp. 146–166
10. W. Eberle, J. Graves, and L. Holder, "Insider threat detection using a graph-based approach," Journal of Applied Security Research, 2010, 6 (1) :32-81
11. J. Myers, M. R. Grimaila, and R. F. Mills, "Towards insider threat detection using web server logs," in Proceedings of the 5th Annual Workshop on Cyber Security and Information Intelligence Research: Cyber Security and Information Intelligence Challenges and Strategies. New York, NY, USA: ACM, 2009, pp. 54:1–54:4.

12. H. Eldardiry, E. Bart, J. Liu, J. Hanley, B. Price, and O. Brdiczka, “Multi-domain information fusion for insider threat detection,” Security and Privacy Workshops (SPW), 2013 IEEE, 2013.
13. S Andropov, A Guirik, M Budko, M BudkoNetwork. "Anomaly detection using artificial neural networks", Open Innovations Association , 2017 :26-31
14. T Rashid , I Agrafiotis, JRC Nurse, "A New Take on Detecting Insider Threats: Exploring the use of Hidden Markov Models", International Workshop , 2016 :47-56