Method and System for Detecting Anomalous User Behaviors: An Ensemble Approach

Xiangyu Xi*†, Tong Zhang*†, Guoliang Zhao§, Dongdong Du†‡, Qing Gao‡, Wen Zhao † and Shikun Zhang†
*School of Software and Microelectronics, Peking University
†National Engineering Research Center for Software Engineering, Peking University
‡School of Electronics Engineering and Computer Science, Peking University
§CASIC-CQC Software Testing and Assessment Technology(Beijing) Corporation
email:{xixy,zhangtong17,dudong,gaoqing,zhaowen,zhangsk}@pku.edu.cn

***Abstract*—Malicious user behavior that does not trigger access violation or data leak alert is difficult to detect. Using the stolen login credentials, the intruder doing espionage will first try to stay undetected, silently collect data that he is authorized to access from the company network. This paper presents an overview of User Behavior Analytics Platform built to collect logs, extract features and detect anomalous users which may contain potential insider threats. Besides, a multi-algorithms ensemble, combining OCSVM, RNN and Isolation Forest, is introduced. The experiment showed that the system with an ensemble of unsupervised anomaly detection algorithms can detect abnormal user behavior patterns. The experiment results indicate that OCSVM and RNN suffer from anomalies in the training set, and $iForest$ gives more false positives and false negatives, while the ensemble of three algorithms has great performance and achieves recall 96.55% and accuracy 91.24% on average.**

***Index Terms*—anomaly detection, insider threat, user behavior, unsupervised learning, ensemble**

I. Introduction

Insider threat has emerged in enterprise security and received increasing attention over last several years. A survey [1] by Haystack shows 56% of respondents feel that insider attacks have become more frequent. Privileged IT users such as administrators with access to sensitive information, pose the biggest insider threat. IT assets such as databases, file servers and mobile devices are top assets at risk.

Insider threat is defined as any activity by military, government, or private company employees whose actions or inactions, by intent or negligence, result (or could result) in the loss of critical information or valued assets [2]. Two types of insider threats are distinguished: malicious insider threats and unintentional insider threats [3]. The first threat is a current or former employee, contractor, or business partner who has or had authorized access to an organizations network, system, or data and intentionally exceeded or misused that access in a manner that negatively affected the confidentiality, integrity, or availability of the organizations information or information systems. The attempted attack by a Fannie Mae employee after being dismissed is a typical example of an insider threat likely motivated by revenge [4]. The second form is from insiders without malicious intent [5] such as human mistakes, errors.

This work was partially supported by National Key Research and Development Program of China (No. 2017YFB0802900).
DOI reference number: 10.18293/SEKE2018-036

A key problem discussed frequently is to detect compromised user accounts and insiders within the company, which does not induce enormous data flow and/or any access violation [6]. For example, an attacker may steal user credentials using social engineering and access sensitive information or copy it to untrusted storage. In this scenario, security systems such as firewalls, IDS [7] or Security Information and Event Management(SIEM), and Data Leak Prevention System(DLP) [8] cannot detect effectively. Relying on analysts to investigate attacks is costly and time-consuming, as they have to deal with millions of logs.

User Behavior Analytics, which has been used in online social media analysis [9] and improving web search ranking [10] , is emerging in security area. User behavior analytics is a cyber security process about detection of insider threats, targeted attacks, and financial fraud. They look at patterns of human behavior, and then apply algorithms and statistical analysis to detect meaningful anomalies from those patterns [11]. UBA collects various types of data such as organization structure, user roles and job responsibilities, user activity trace and geographical location. The analysis algorithms consider factors including contextual information, continuous activities, duration of sessions, and peer group activity to compare anomaly behavior. UBA determines the baseline of normal behavior of individual user or peer group according to history data. The deviation of ongoing user activities compared with past normal behavior is significant if the user acts abnormally [12].

This paper introduces an User Behavior Analytics Platform built to detect potential insider threats. Specifically, the platform can 1) collect and preprocess logs from systems and applications; 2) extract each user's activity records from logs; 3) aggregate activity records and generate feature vector for each user; 4) detect anomalous user access. Besides, an ensemble by multiple unsupervised anomaly detection algorithms is proposed and shows great performance in detecting user's anomalous access and operation within enterprise.

This paper is organized as follows. Section 2 introduces related work. User behavior analytics architecture and platform which contains four components is presented in Section 3. Section 4 introduces experiment scenario, data characteristics and feature selection. In Section 5, anomaly detection algo-

rithms for user behavior analytics are demonstrated. Section 6 gives dataset, experiment and results. Besides, discussion and comparison of algorithms are demonstrated. Finally, section 7 concludes the paper and provides future work.

II. Related Work

Anomaly detection is an important problem that has been researched within diverse research areas and application domains including information security [13]. Research of applying anomaly detection is popular in intrusion detection [7], fraud detection [14] [15], medical and public health anomaly detection and industrial damage detection. Many anomaly detection techniques have been specifically developed for certain application domains, while others are more generic.

Anomaly detection techniques being applied to user behavior analytics is increasingly popular. Veeramachaneni K et al. [16] put forward AI^2 that combines analyst intelligence with an ensemble of three outlier detection methods to detect account takeover, new account fraud and service abuse.

Madhu Shashanka et al. [17] presented the User and Entity Behavior Analytics(UEBA) module of the Niara Security Analytics Platform which uses a SVD-based algorithm to detect anomalies in user accessing server within an enterprise. Both users historical baseline and peer baseline are applied with same algorithm.

Sapegin A et al. [18] proposed a poisson-based two-step algorithm to identify anomaly user access to workstation within Windows domain. However, the dataset is from simulation scenario and of limited features. The algorithms are not persuasive enough and of limited extensibility.

Wei Ma et al. [19] defined a user behavior pattern and proposed a knowledge-driven user pattern discovery approach which can extract users behavior patterns from audit logs from distributed medical imaging systems. The work is emphasized on extracting user behavior patterns and there is a long way to go before administrator directly use it.

Li at al. [20] proposed a kind of security audit technology based on one-class support vector machine detect the abnormal behavior of database operations.

III. User Behavior Analytics Architecture and Platform

In this section, an architecture of user behavior analytics is presented. Based on that, the implementation of our UBA platform is described.

Relying on analysts to investigate attacks is costly and time-consuming, as they have to deal with millions of logs and alerts. Our UBA platform collects logs about user-related events and user session activity in real-time or near real-time, and compares each and every action to the corresponding baseline of users to spot anomalies in their behavior. Based on detection results, a risk label or score that reveals human risk will be assigned to every user, which is helpful and meaningful for security analysts especially when analysts investigate or monitor employees for suspicious behaviors or attacks. Fig.1 provides the architecture composed of four components. Each of them is described in the following.

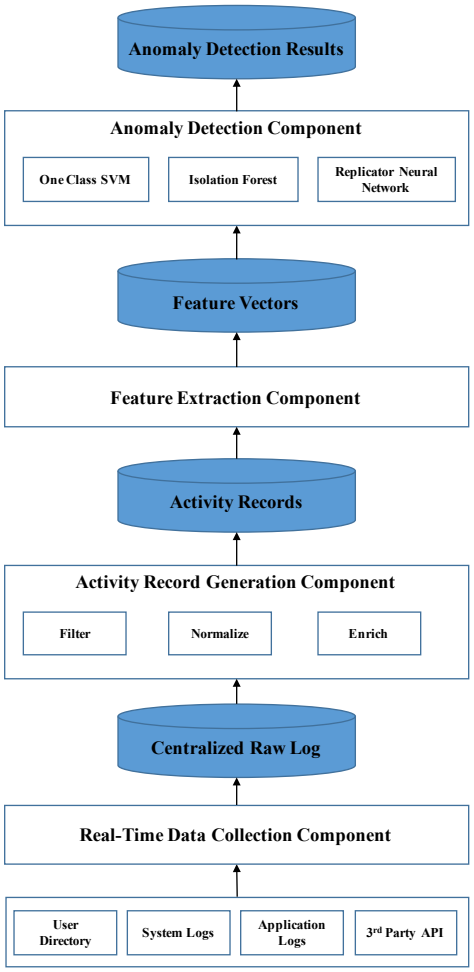


Fig. 1. Architecture of User Behavior Analytics Platform

A. Data Collection Component

Data collection component stores raw logs generated by systems and applications for further extraction and analysis. The collected raw logs are stored in ElasticSearch [21] , which is a distributed, JSON-based search and analytics engine designed for horizontal scalability, maximum reliability, and easy management. Logs of users accessing ftp server within enterprise and operations such as downloading, uploading, deleting files or directories are collected. The user information can be gathered from activity directory of the enterprise.

Data source our platform can process includes:

1) system logs,
2) application logs such as web access logs and DLP logs,
3) user directory logs, etc.

B. Activity Record Generation Component

Centralized raw logs are of respective unique formats from which feature cannot be extracted directly. For example, apache server logs and windows security logs consist of different items. Due to the lack of normalized formats, activity record generation component

1) builds a general schema for activity records,

2) generates regular expressions as filters for each type of log to extract useful information,
3) fills the schema with extracted information.

Then the activity records with user information are generated, which will be processed in the following Feature Extraction Component.

C. Feature Extraction Component

After generating normalized activity records with user information attached, we compute user behavioral features over an interval of time such as 24 hours. For performance consideration, the strategy from [16] is applied. Each hour we retrieve activity records within last hour and compute the features labeled with last hour. In midnight, we only need to retrieve the 23 feature sets and activity records within last hour rather than activity records within last 24 hours.

D. Anomaly Detection Component

With features extracted for each user, anomaly detection component detects anomalous users on a daily basis. The component is designed losely coupled, flexible and independent from other components. The details of algorithms are demonstrated in V.

IV. Data Characteristics

In this section, a scenario within a typical software company is introduced. The behavior of employees accessing ftp files and data within work groups are monitored and audit logs are generated and collected by UBA platform presented before. Then dataset and feature selection is introduced.

A. Experiment Scenario

Consider a file server within an enterprise, authorized employees can access the server for files and data with different authorization, which normally is configured by the administrator. For example, one can read, write, upload, download or delete files or directories. As documents and data are important information, accessing and operations need to be monitored for possible actions from compromised accounts or rogue users. UBA platform monitors the access patterns and operation patterns of each user while accessing the server and files.

B. Dataset

The ftp server logs are collected by the data collection component presented before. In total 8 kinds of logs are collected and each corresponds to 1 or more types of events. For example, Download Log only represents Download Event and the log carries information including timestamp, user name, SUCCESS/FAIL flag and client IP. An UPLOAD LOG may represent an upload file/directory event, create file event or remote copy event and cannot be distinguished by content as different events share same format. Table I shows the representation map between logs and events.

We collected operation logs within a software company for 3 months and selected four employees with explicit different behavior patterns. Activity records generated were checked and all can be considered as normal behaviors so we simulate several abnormal operations for each user as testing dataset. Based on investigation in the enterprise, abnormal behaviors mainly include four categories shown in Table II.

TABLE I
Logs and corresponding events

LOG	EVENT
CONNECT LOG	CONNECT Event
LOGIN LOG	Login Event
DOWNLOAD LOG	Download Event
UPLOAD LOG	Upload Event Create File Event Remote Copy Event
DELETE LOG	Delete File Event Delete Directory Event
MKDIR LOG	Make Directory Event
RMDIR LOG	Remove Directory Event
RENAME LOG	Rename Event Remotely Move Files Event

TABLE II
Categories of abnormal behavior

anomaly ID	Description
anomaly 1	multiple login attempts and failures
anomaly 2	anomalous downloads operations
anomaly 3	anomalous delete operations
anomaly 4	operations at non-working hours

C. Feature Selection

With activity records generated by activity record generation component, feature extraction component produces a feature vector for each user daily, which characterize the pattern of users' access to ftp server and operations. The features are shown in Table III. The features for a user daily is denoted by 21-dimention vector $x = (x_1, x_2, ..., x_{21})$.

TABLE III
The list of Features. 21 features are extracted and used in total.

Feature ID	Description
1	number of total connections of the day
2	timestamp of first login attempt of the day
3	timestamp of last login attempt of the day
4	number of login success of the day
5	number of login fail of the day
6	total download bytes
7	number of download success of the day
8	number of download fail of the day
9	largest download bytes of the day
10	total upload bytes
11	number of upload success of the day
12	number of upload fail of the day
13	largest upload bytes of the day
14	number of delete success of the day
15	number of delete fail of the day
16	number of mkdir success of the day
17	number of mkdir fail of the day
18	number of rmdir success of the day
19	number of rmdir fail of the day
20	number of rename success of the day
21	number of rename fail of the day

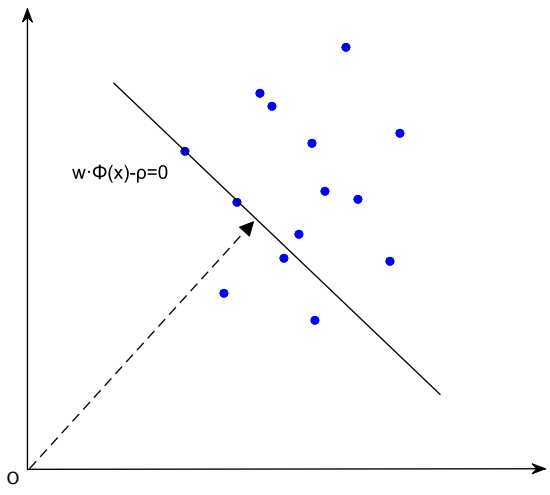


Fig. 2. diagram of OCSVM hyperplane

V. Algorithm

UBA platform practically is fed with data without label, which motivates us to use unsupervised anomaly detection techniques. In practice, it's unknown whether training set contains abnormal data points and the proportion, different algorithms are of better performance under different conditions. For example, when training set contains normal instances only, Replicator Neural Network and OCSVM work better, but Isolation Forest might suffer a small reduction. As a result, an ensemble of three unsupervised anomaly detection algorithms is used to improve the robustness and performance.

A. One Class SVM

OCSVM, proposed by Scholkopf [22], has been applied to anomaly detection. As Fig.2 shows, the OCSVM algorithm maps input data into a high dimensional feature space via a kernel and iteratively finds the maximal margin hyperplane which best separates the training data from the origin.

$$\begin{aligned} \min_{w,\zeta_i,\rho} \quad & \tfrac{1}{2}||w||^2 + \tfrac{1}{\nu n}\textstyle\sum_{i=1}^{n}\zeta_i - \rho \\ s.t. \quad & (w^T\phi(x_i)) > \rho - \zeta_i, i = 1, ..., n \\ & \zeta_i > 0, i = 1, ..., n \end{aligned} \tag{1}$$

The decision function presented is $f(x) = \text{sgn}(w^T\phi(x) - \rho)$. After solving the dual problem below:

$$\begin{aligned} \min_{\alpha} \quad & \tfrac{1}{2}\textstyle\sum_{ij}\alpha_i\alpha_j k(x_i, x_j) \\ s.t. \quad & 0 \le \alpha_i \le \tfrac{1}{\nu l}, i = 1, ..., n \\ & \textstyle\sum_{i=1}^{n}\alpha_i = 1 \end{aligned} \tag{2}$$

The decision function is given by :

$$f(x) = \text{sgn}(\sum_{i=1}^{n}(\alpha_i K(x_i, x) - \rho)). \tag{3}$$

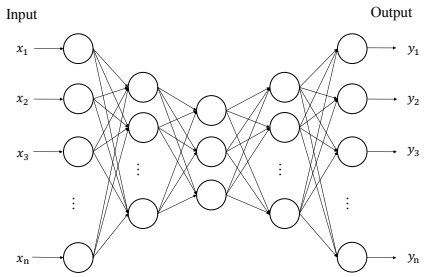


Fig. 3. Replicator Neural Network with three hidden layers

B. Replicator Neural Network

Replicator Neural Network is an artificial feed-forward multi-layer neural network with an output layer having the same number of nodes as the input layer. The purpose of Replicator Neural Network is to produce the output data which as is similar as the input data. Fig.4 presents the structure of the fully connected RNN with three hidden layers.

Replicator Neural Network is effective in anomaly detection as an unsupervised machine learning algorithm because anomalies are few and there exist some common patterns in normal data. By the trained RNN, the common patterns representing bulk of the data can be well reproduced, while anomalies will have a much higher reconstruction error. The reconstruction error for a d-dimensional instance $x = (x_1, x_2, ..., x_d)$ is computed as follow:

$$e = \sum_{i=1}^{d}(x_i - y_i)^2 \tag{4}$$

in which d is the dimension of input vector x and $y = (y_1, y_2, ..., y_d)$ is the reconstructed output.

C. Isolation Forest

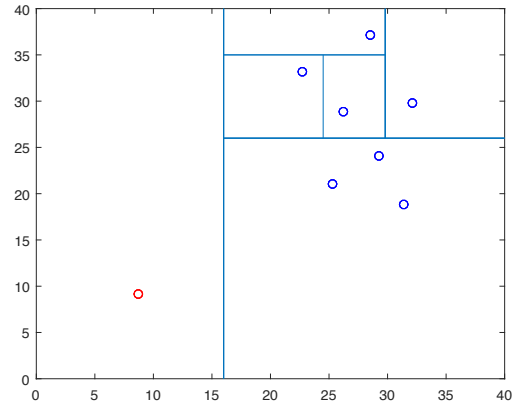
Since anomalies are few and different and therefore they are more susceptible to isolation. Based on the concept of isolation, Isolation Forest [23] builds a set of $iTrees$ for a given data set, then anomalies are those instances which have short average path lengths on the $iTrees$. For example, in Fig.4, the red outlier $(8.7, 9.2)$ is isolated at first split, while normal points marked as blue need more than 3 splits in the isolation tree.

To be specific, for a given dataset, iTrees are constructed by recursively partitioning the given training set until instances are isolated or a specific tree height is reached. There are only two variables in this method:

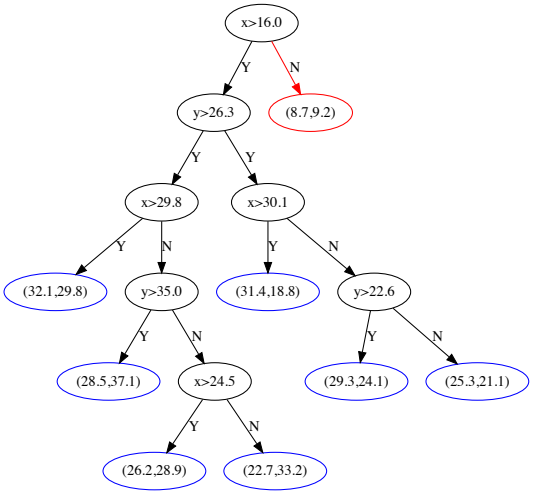
1) the number of trees to build t, and
2) the sub-sampling size ψ

Path length $h(x)$ of a point x is measured by the number of edges x traverses an $iTree$ from the root node until the traversal is terminated at an external node. The anomaly score s of an instance x is defined as:

$$s(x, n) = 2^{-\frac{E(h(x))}{c(n)}}, \tag{5}$$



(a) Training Data



(b) Generated Isolation Tree

Fig. 4. Isolation Tree Generated with data set

where $E(h(x))$ is the average of $h(x)$ from the trained collection of isolation trees.

D. Ensemble and Strictly Filtering

We combine the predictions of three algorithms introduced before and apply the strictly filtering strategy to predict whether a user is anomalous or not. OCSVM directly produces label $y \in \{0, 1\}$. Output of RNN is reconstruction error $err \in \mathbb{R}$ while $iForest$ generates anomaly score $s(x, n) = 2^{-\frac{E(h(x))}{c(n)}} \in (0, 1)$. Labels are given by comparing output with corresponding threshold. Output 0 represents abnormal while 1 represents normal.

$$f_{OCSVM}(x; X) = \begin{cases} 0, \\ 1 \end{cases} \tag{6}$$

$$f_{iForest}(x; X) = \begin{cases} 0, & s(x, n) > \varepsilon_1 \\ 1, & s(x, n) \leq \varepsilon_1 \end{cases} \tag{7}$$

$$f_{RNN}(x; X) = \begin{cases} 0, & err(x) > \varepsilon_2 \\ 1, & err(x) \leq \varepsilon_2 \end{cases} \tag{8}$$

where "; X" indicates the model is trained with X as training set. As recall is an important measure metric in security, we apply strictly filtering strategy and regard data point as abnormal as long as any of the three algorithms outputs 0, shown by formula 9 and 10.

$$f(x; X) = f_{OCSVM}(x; X) + f_{iForest}(x; X) + f_{RNN}(x; X) \tag{9}$$

$$s(x; X) = \begin{cases} 0, & f(x; X) < 3 \\ 1, & f(x; X) = 3 \end{cases} \tag{10}$$

Given the k_{th} user's historical behavior $X_k = [x_k^1, x_k^2, ..., x_k^i ..., x_k^m]$, in which $i \in \{1, 2, ..., m\}$ denotes the index of days and $x_k^i = (x_k^{i,1}, x_k^{i,2} ..., x_k^{i,21})^T$ is the feature vector of k_{th} user in i_{th}day.

For k_{th} user with feature vector to be detected and denoted by $\hat{x}_k = (\hat{x}_k^1, \hat{x}_k^2 ..., \hat{x}_k^{21})^T$, the prediction is given by:

$$score_k = s(\hat{x}_k, X_k) \tag{11}$$

VI. Experiment and Results

A. Experiment Setup

Data preserved on ftp server is not enough so we perform a simulation after fitting the data collected with a polynomial distribution. Besides, a small proportion 2.06% and 4.03% of anomalous user behaviors is mixed into training set to find out performance when training sets mixed with different purities. Hence three training sets are used as Table IV shows and the data sets are composed of five categories.

In OCSVM, we exploit LIBSVM [24] and RBF kernel $K(x_i, x_j) = \exp(-\gamma||x_i - x_j||^2), \gamma > 0$, where $\gamma = 0.01$ and $\nu = 0.05$ is selected as parameters.

In $iTree$,number of trees $t = 200$ and the sub-sampling size $\psi = 200$. The threshold $\varepsilon_1 = 0.45$.

A RNN with 3 hidden layers is applied and the number of neurons in each layer is $[20, 8, 4, 8, 20]$. Activation function $\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$ is selected. The threshold selected is $\varepsilon_2 = 1.20$. Based on stochastic gradient descent, the training contains 60,000 train epochs and batch-size is set 20.

B. Results and Discussion

The detection rate of single algorithm in different category of testing data is shown in Table V. Assuming abnormal data points is our focus and marked as positive, the overall accuracy, precision and recall is shown in Table VI.

With anomaly-free training set, OCSVM has best performance with recall=100% and accuracy=96.72%. All of the anomalies can be detected. With more anomaly points in the training set, performance of OCSVM gets worse. With 4.03% anomaly points, the recall is 75.86% and accuracy is 81.39%.

TABLE IV
COMPOSITION OF TRAINING SETS AND TESTING SET. TRAINING SETS WITH 0%, 2.06% AND 4.03% OF ANOMALIES MIXED INTO ARE USED.

dataset	normal	anomaly 1	anomaly 2	anomaly 3	anomaly 4	total	anomalies proportion
training set 1	1000	0	0	0	0	1000	0.00%
training set 2	1000	6	6	3	6	1021	2.06%
training set 3	1000	12	12	6	12	1042	4.03%
testing set	100	45	47	32	50	274	63.50%

TABLE V
DETECTION RATE OF OCSVM, RNN AND $iForest$ WITH DIFFERENT TRAINING SETS

category	training set 1(0.00%)			training set 2(2.06%)			training set 3(4.03%)		
	OCSVM	RNN	$iForest$	OCSVM	RNN	$iForest$	OCSVM	RNN	$iForest$
normal	91.00%	92.00%	51.00%	91.00%	88.00 %	87.00%	91.00%	92.00%	87.00%
anomaly 1	100.00%	100.00%	100.00%	93.33%	100.00%	91.11%	82.22%	62.22%	77.78%
anomaly 2	100.00%	100.00%	100.00%	79.59%	97.87%	68.09%	89.36%	55.32%	65.96%
anomaly 3	100.00%	100.00%	100.00%	100.00%	100.00%	68.75%	100.00%	81.25%	100.00%
anomaly 4	100.00%	92.00%	78.00%	54.00%	90.00%	38.00%	42.00%	92.00%	36.00%

TABLE VI
ACCURACY, PRECISION AND RECALL OF OCSVM, RNN AND $iForest$ WITH DIFFERENT TRAINING SETS

category	training set 1(0.00%)			training set 2(2.06%)			training set 3(4.03%)		
	accuracy	precision	recall	accuracy	precision	recall	accuracy	precision	recall
OCSVM	96.72%	95.08%	100.00%	83.58%	93.88%	79.31%	81.39%	94.29%	75.86%
RNN	95.62%	95.51%	97.70%	93.43%	93.33%	96.55%	79.56%	94.03%	72.41%
$iForest$	78.10%	76.89%	93.68%	73.58%	89.76%	65.52%	74.09%	89.92%	66.67%
ensemble	**91.60%**	**88.32%**	**100.00%**	**90.88%**	**89.84%**	**96.55%**	**91.24%**	**93.10%**	**93.10%**

RNN has the similar performance and trend to OCSVM. Fig.5 shows the mean reconstruction error $e = \sum_{i=1}^{d}(x_i - y_i)^2$ of 5 categories of test data during RNN training with 2.06% anomalies in training set. The training process converged and abnormal data has higher reconstruction error which makes separating possible. The mean reconstruction error of normal data is 0,539, much lower than abnormal data in testing dataset (4.947, 3.887, 8.627, 2.409). However, the time cost of training is much higher than other algorithms.

Isolation Forest has the worst performance of the three algorithms as Table V and Table VI shows. With training set 2, the anomaly score of test data from Isolation Forest is presented in Fig.6. The normal data has a lower anomaly score 0.390 than anomaly data (0.490,0.487,0.475,0.442). However, scores of some data are pretty close as Fig.6 shows, especially the data of operations at non-working hour. In each category of anomaly data we simulated, data is anomaly in only few dimensions. At training stage, attribute and split point is randomly selected. Statistically it's hard to split data with anomalous attributes before the tree goes deep. However, if the training set has more complicated and real anomalies, $iForest$ can be of better performance. Besides, the threshold ε_1 can be adjusted flexibly, which is a valuable characteristic. In addition, $iForest$ didn't suffer an obvious reduction with more anomalies mixed in the training set.

Table VI shows that the ensemble and strictly filtering strategy improve robustness and performance especially recall.

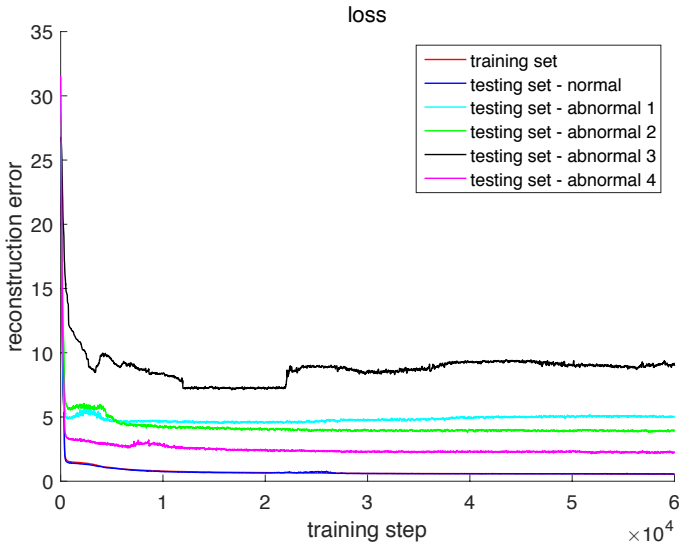


Fig. 5. Mean reconstruction error during training with Replicator Neural Network.

When anomalies in training set increase, algorithms alone are less reliable. With 2.06% anomalies in training set, the ensemble gives recall=96.55% and accuracy=90.88%. With 4.03% anomalies in the training set, RNN has recall=72.41% and accuracy=79.56%, while the ensemble has great performance recall=93.10% and accuracy=91.24%. It can be a good and

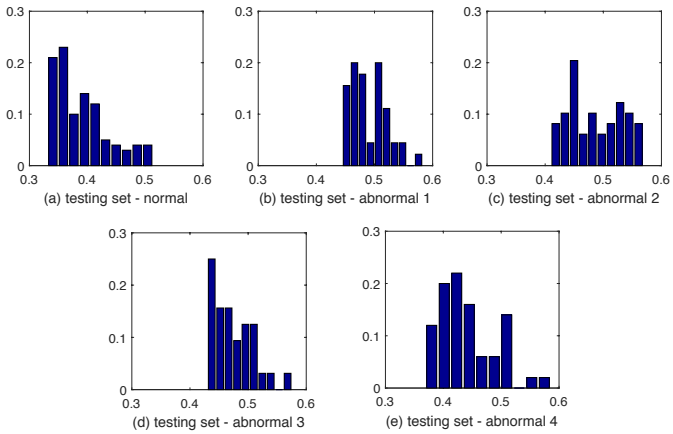


Fig. 6. Anomaly Score of Isolation Forest for test data of different categories

optional strategy especially when security analysts focus on recall.

VII. Conclusion and Future Work

This paper presents an overview of UBA architecture and platform for detecting anomalous user behaviors within enterprise. The platform, composed of four components working independently, is suitable for running on distributed platforms. The anomaly detection component contains an ensemble of OCSVM, RNN and Isolation Forest. Strictly filtering strategy is applied and can improve the performance and robustness no matter whether there exist anomalies in the training set.

The sequences of events contain valuable information about users and we will focus on anomaly detection of sequence data. Besides, the peer group analysis, which may play an important role in practice, can be introduced into the UBA platform in the future.

References

[1] New haystax technology survey shows most organizations ill-prepared for insider threats. https://haystax.com/blog/2017/03/29/new-haystax-technology-survey-shows-most-organizations-ill-prepared-for-insider-threats/ (accessed December, 2017).

[2] A. P. Moore, K. A. Kennedy, and T. J. Dover, "Introduction to the special issue on insider threat modeling and simulation," *Computational and Mathematical Organization Theory*, vol. 22, no. 3, pp. 261–272, 2016.

[3] D. M. Cappelli, A. P. Moore, and R. F. Trzeciak, *The CERT guide to insider threats: how to prevent, detect, and respond to information technology crimes (Theft, Sabotage, Fraud)*. Addison-Wesley, 2012.

[4] U. A. Office. Fannie mae corporate intruder sentenced to over three years in prison for attempting to wipe out fannie mae financial data. https://archives.fbi.gov/archives/baltimore/press-releases/2010/ba121710.htm/ (accessed December, 2017).

[5] F. I. P. Bureau, "Unintentional insider threats: A foundational study," 2013.

[6] M. Uma and G. Padmavathi, "A survey on various cyber attacks and their classification." *IJ Network Security*, vol. 15, no. 5, pp. 390–396, 2013.

[7] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2016.

[8] A. Shabtai, Y. Elovici, and L. Rokach, *A survey of data leakage detection and prevention solutions*. Springer Science & Business Media, 2012.

[9] Y. Amichai-Hamburger and G. Vinitzky, "Social network use and personality," *Computers in human behavior*, vol. 26, no. 6, pp. 1289–1295, 2010.

[10] E. Agichtein, E. Brill, and S. Dumais, "Improving web search ranking by incorporating user behavior information," in *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2006, pp. 19–26.

[11] T. Bussa, A. Litan, and T. Phillips, "Market guide for user and entity behavior analytics," *URL: https://www. gartner. com/doc/3538217/market-guide-user-entity-behavior (29.07. 2017)*, 2016.

[12] W. Ma, "User behavior pattern based security provisioning for distributed systems," Ph.D. dissertation, 2016.

[13] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, p. 15, 2009.

[14] D. J. Weston, D. J. Hand, N. M. Adams, C. Whitrow, and P. Juszczak, "Plastic card fraud detection using peer group analysis," *Advances in Data Analysis and Classification*, vol. 2, no. 1, pp. 45–62, 2008.

[15] M. Ahmed, A. N. Mahmood, and M. R. Islam, "A survey of anomaly detection techniques in financial domain," *Future Generation Computer Systems*, vol. 55, pp. 278–288, 2016.

[16] K. Veeramachaneni, I. Arnaldo, V. Korrapati, C. Bassias, and K. Li, "Aiˆ 2: training a big data machine to defend," in *Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS), 2016 IEEE 2nd International Conference on*. IEEE, 2016, pp. 49–54.

[17] M. Shashanka, M.-Y. Shen, and J. Wang, "User and entity behavior analytics for enterprise security," in *Big Data (Big Data), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1867–1874.

[18] A. Sapegin, A. Amirkhanyan, M. Gawron, F. Cheng, and C. Meinel, "Poisson-based anomaly detection for identifying malicious user behaviour," in *International Conference on Mobile, Secure and Programmable Networking*. Springer, 2015, pp. 134–150.

[19] W. Ma, K. Sartipi, and D. Bender, "Knowledge-driven user behavior pattern discovery for system security enhancement," *International Journal of Software Engineering and Knowledge Engineering*, vol. 26, no. 03, pp. 379–404, 2016.

[20] Y. Li, T. Zhang, Y. Y. Ma, and C. Zhou, "Anomaly detection of user behavior for database security audit based on ocsvm," in *Information Science and Control Engineering (ICISCE), 2016 3rd International Conference on*. IEEE, 2016, pp. 214–219.

[21] elasticsearch.io. https://www.elastic.co/products/elasticsearch (accessed December, 2017).

[22] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, and J. C. Platt, "Support vector method for novelty detection," in *Advances in neural information processing systems*, 2000, pp. 582–588.

[23] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. IEEE, 2008, pp. 413–422.

[24] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011, software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.