Data Sci. Eng. (2016) 1(2):73–85
DOI 10.1007/s41019-016-0009-x

INVITED PAPER

A Data-Driven Evaluation for Insider Threats

Yuqing Sun[1] · **Haoran Xu**[1] · **Elisa Bertino**[2] · **Chao Sun**[1]

Received: 13 April 2016 / Accepted: 10 June 2016 / Published online: 18 July 2016

Abstract Insiders are often legal users who are authorized to access system and data. If they misuse their privileges, it would bring great threat to system security. In practice, we could not have any knowledge about fraud pattern in advance, and most malicious behaviors are often in accordance with security rules; thus, it is difficult to predefine regulations for preventing all kinds of frauds. In this paper, we propose a data-driven evaluation model to detect malicious insiders, which audits user behaviors from both parallel and incremental aspects. Users are grouped together according to their positions and responsibilities, based on which the normal pattern is learned. For each user, a routine behavior pattern is also learned for historical assessment. Then, users are evaluated against both group patterns and routine patterns by probabilistic methods. The deviation degree is adopted as an evidence to justify an anomaly. We also recognize the abnormal activities that often make a user behavior much deviate, which can help an administrator revisit security policies or update activity weights in assessment. At last, experiments are performed on several real dataset.

Keywords Insider threat · Audit · Behavior analysis

✉ Yuqing Sun
sun_yuqing@sdu.edu.cn

Haoran Xu
hr_xu1990@163.com

Elisa Bertino
bertino@purdue.edu

Chao Sun
sunchao1207@126.com

[1] School of Computer Science and Technology, Shandong University, Jinan, China

[2] Department of Computer Science, Purdue University, West Lafayette, IN, USA

1 Introduction

A malicious insider refers to an employee, a contractor or a business partner who has or had been authorized access to an organization information system and intentionally exceeds or misuses his/her privilege in a manner that negatively affects the confidentiality, integrity or availability of an information system [1]. It may involve fraud, theft of commercial secrets or intellectual property, sabotage of information and system, etc. [2,3]. Insider threats often result in economic loss and organizational reputation damage. According to Computer Crime and Security survey sponsored by CSI and FBI, 87.1 % of respondents which include major organizations in the USA said that 20 % of their losses should be attributed to malicious insiders [4]. Such loss is often in hundreds of millions magnitude and is increasing year by year. Hence, insider threat is inherently a major problem to address.

However, malicious insiders are often legal users and are authorized to access information system or data. It is a challenge to detect whether their motivations and behaviors are benign or malicious. In order to solve this problem, a lot of efforts have been made to detect and prevent insider threats. From a managerial perspective, regulations are enforced on both an individual action and group activities according to security requirements and business responsibilities. For example, a database user is allowed only to access the permitted cells of database or to perform some statistic queries rather than to query a concrete cell [5,6]. Another example on group regulation is the separation of duty (SoD) policy, which distributes the authorizations of performing sensitive tasks to multiple persons so as to reduce the fraud risk [7,8].

From the technical perspective, the main purpose is to ensure the effective enforcement of security regulations. Audit is an important technique of examining whether user behaviors in a system are in conformance with security poli-

cies [9,10]. Many methods audit a database processing by comparing a user SQL query expression against some predefined patterns so as to find out an anomaly [11,12]. But a malicious query may be made up as *good looking* so as to evade such syntactic detection. To overcome this shortcoming, the data-centric method further audits whether the data a user query actually accessed has involved any banned information [13,14]. However, such audit concerns a concrete policy rather than the overall view of multiple security policies. It requires clear audit commands that are articulated by experienced professionals and much interactive analysis. Since in practice an anomaly pattern cannot be articulated in advance, it is difficult to detect such fraud by the current audit method.

The anomaly detection technology is used to identify abnormal behaviors that are statistical outliers [15]. Some probabilistic methods learned normal patterns, against which they detected an anomaly [16–18]. But these methods assume very few users are deviated from normal patterns. In case there are a number of anomalous users, the normal pattern would be diverged. These works do not examine user behavior from either a historical or an incremental view, which may overlook some malicious behaviors. Furthermore, if a group of people collude together, it is difficult to find them by the current methods.

In this paper, we tackle the insider threat problem from a data-driven systemic view. User actions are recorded as historical log data in a system, and our evaluation investigates the date that users actually process. From the horizontal view, users are grouped together according to their responsibilities and a normal pattern is learned from the group behaviors. This is motivated by the fact that users associated with the same position should behave similarly or their processed data should satisfy the similar distribution over different aspects. Users who deviate from the normal patterns should be suspected. We further identify which activity results in an anomaly and its impact. If some activity is unusual in audit results, it is an important indicator for detecting anomaly. Such evaluation provides a useful reference for security administrators to set up a weight to each activity on anomaly detection.

We investigate a suspected user also from the diachronic view by comparing his/her historical behaviors with the historical average of the same group. The greater this distance, the more suspicious. This evaluation can overcome the false-positive case of *overactive*, namely some user may be active than others. It also solves the $gradually\ malicious$ threat since the historical statics measures the accumulative results.

Since many organizations enforce security policies, such as Separation of Duty, to distributed sensitive authorizations to multiple people, it is difficult to detect collusion by investigating each user individually. There are many practical examples that are applicable to this context. For example, in a financial company, a group of clerks are authorized the rights of loan. A complete process of granting a loan is restricted to be performed by two different persons. This security rule has been embedded into system, and any grant definitely satisfies this separation of duty regulation. But in fact, if two persons colluded and together loaned a large amount of money to a discredit company for many times, it would bring high fraud risk. So, the collusion problem should be solved from a system view. We investigate user relationships in the context of sensitive tasks to assistant fraud detection.

The remainder of this paper is organized as follows. In Sect. 2, we present prior research related to this work. Section 3 introduces the data-driven insider threat evaluation framework. Section 4 investigates the collusion problem. In Sect. 5, we discuss how to apply our method into practical system. Experimental evaluation is discussed in Sect. 6. Finally, we summarize the work and offer future research direction.

2 Related Work

2.1 Database Audit

The most related work is about database audit. It audits a database query log file against some predefined regulations. Audit requirements are formalized as query commands on logs. The queries referred to as sensitive data are detected as suspicious queries [19]. This method can be classified into two categories: syntax-based query auditing and data-based query auditing. In syntax-based query auditing [20], normal syntax patterns are interactively specified. The queries different from the normal patterns are regarded as anomaly [11,12]. Shebaro et al. [21] propose an anomalous query detection system and integrate it into a relational database management system. For each query, it extracts relevant features from syntax tree of SQL commands and compares with normal query patterns so as to determine whether SQL command is an anomaly.

Although this kind of audit can find out most unexpected expressions, it cannot discover some malicious queries which are made up as normals. To make audit results more accurate, the data-based query audit is proposed, which focuses on actual data involved in query results [13]. It creates a feature vector on each query result and compares it with the vectors of normal queries so as to find malicious queries. Thong et al. propose the misuseability weight to analyze the sensitive level of data involved in query results so as to evaluate the query risk, separately [5,6]. Although database audit can protect sensitive data from malicious query effectively, it requires some specialist to interactively specify audit commands, which is time-consuming. Furthermore, it focuses on concrete policies and cannot cover all security requirements

of an organization. Another shortcoming is not applicable to unexpected anomaly pattern.

2.2 Anomaly Detection

Our work is also related with the anomaly detection. Most of these works mine user behavior patterns and determine whether any particular user is sufficiently different from normalities. The supervised anomaly detection approaches require the labeled dataset so as to establish anomaly detection models. New actions are classified by Bayesian networks [17,22] and hidden Markov models [23]. Such methods are suitable for detecting anomalies against the preciously known normal patterns, which cannot solve the anomaly detection problem without any knowledge about context in advance.

Unsupervised approaches on anomaly detection do not rely on training dataset. Generally, abnormal behavior detection is based on identifying behaviors that are statistical outliers [15]. A density-based anomaly detection is presented in [18], and it measures how a user is deviated from his/her surrounding neighbors. Authors in [16,24] present a community-based anomaly detection system. It is based on the assumption that similar users tend to attend or form the same community. A statistic model is used for measuring the distance of users from involved communities to predict anomalies. Another similar community-based method detects the anomaly by checking whether the presence of a user in a community will decrease the cohesion of community [25]. However, these methods mainly focus on individual view of anomaly detection and can not detect a group fraud. A number of anomalous users can make normal behavior pattern inclined. And these methods do not identify concrete activities which lead to an outlier. Furthermore, these methods only consider user behaviors from a single period without evaluation on anomalies from a diachronic view, which is desired in practice. Different from them, we would tackle the insider threat problem by auditing user behaviors from both parallel and historical views.

2.3 Collusion Analysis

Our work is also related to the collusion prevention and detection. The most popular way to prevent a collusion is authorization management in the context of access control. It assesses the sensitive authorizations and distributes them to multiple people so as to reduce the convenience of insider collusion [7]. Game theory is also used for collusion analysis [18]. It finds the optimal strategies of inside attack by searching the game equilibriums. The challenge of this method is the strategies of insiders are not easy to be obtained in an organization due to its dynamic and evolving environments.

Collusion detection usually appears in specific situation, such as P2P systems [26], online rating systems [12,20,23], and so on. Online rating systems are subject to collusion attack mainly by posting unfair rating score collaboratively. It may use a frequent itemset mining algorithm to find candidate groups, and then, several indicators are used for identifying collusion groups [23]. Zimniak et al. [20] identify suspicious collusion on rating systems by leveraging social network. It finds that user behaviors are greatly affected by the social distance and interest similarity in social network. Then, it identifies collusion behavior patterns. However, these methods cannot detect the data-centric semantic fraud. Also, they are not suitable for detecting unknown patterns in a dynamic context.

3 Data-Driven Anomaly Detection

3.1 Basic Notions

Generally, in a large organization, there are often many business positions. Each position is associated with a set of responsibilities and authorizations of accessing system. A user in a system refers to a person in real life. Each user is assigned one or more business positions to take on the corresponding responsibilities and perform certain tasks via systems. For example, in the role-based access control model (RBAC) which is widely adopted in formation applications and database systems [27], the notion of role maps to a business position. A user is assigned to one or multiple roles to take on business tasks. Users taking on different roles collaborate together for a whole business mission. Consider an information system does not apply RBAC, users are grouped together according to their responsibilities. For convenience, we represent such business position as the concept of group so as to discuss the problem in a consistent way.

A business task completion generally requires the coordination of multiple users to meet a common goal. Each task consists of multiple activities, where an activity refers to an atom operation work in a system.

Definition 1 *(Task and Sensitive Level)* A task is a high-level work with a certain business purpose. Each task is associated with a sensitive level according to an organization requirement. A task is considered sensitive if its sensitive level is greater than a given threshold.

Definition 2 *(Activity)* An activity is used for completing a job function. A specific activity is a part of a task representing a set of certain authorizations of performing this task. Let A be the set of all activities in a system.

Whether a task is regarded sensitive depends on the justification of damage in case an anomaly occurs on this task. For example, a loan of 200 million is more sensitive than a

loan of 5 million for a bank. So, risk evaluation mainly concerns sensitive tasks and user behaviors on these tasks in an organization.

To detect insider threats from a data-driven view, we abstract the data semantics from practical transactions. The data involved in transactions can be classified into sensitive clusters against sensitivity and transaction types. Taking into account all aspects involved in transactions, the multiple semantic dimensions are created, called *transaction dimensions*. Each transaction is then represented as the coordinates against these dimensions. The data that a user actually processes can be described as the statistics against the dimensions, which is regarded as the set of user behavior features.

Definition 3 *(Transaction Dimensions)* Given a specific kind of transactions and the data attributes involved in transactions, each attribute value domain is partitioned into a finite set of value intervals according to practical data. The transaction dimensions Ψ are specified as the union of these sets, denoted by $\Psi = \{a_1, a_2, \ldots, a_{|\Psi|}\}$.

Definition 4 *(User behavior vector)* Given a user $u \in U$, an audit period τ and transaction dimensions Ψ, a user behavior represents all the activities performed by u in τ. A user behavior vector is the statistics on the occurrences against Ψ and each coordinate maps to one dimension of Ψ. Formally,

$$B_u^\tau = \left(c_u^\tau(a_1), c_u^\tau(a_2), \ldots, c_u^\tau(a_{|\Psi|})\right) \tag{1}$$

where $c_u^\tau(a_i)$ is the occurrence of u's behavior on dimension $a_i \in \Psi$. It also can be in a normalized form,

$$v_u^\tau = \left(p_u^\tau(a_1), p_u^\tau(a_2), \ldots, p_u^\tau(a_{|\Psi|})\right) \tag{2}$$

where

$$p_u^\tau(a_i) = \frac{c_u^\tau(a_i)}{\sum_{a_j \in |\Psi|} c_u^\tau(a_j)}, \quad i \in [1..|\Psi|] \tag{3}$$

An audit period τ refers to a period of time such as one year, a time widow $\tau = [t_1, t_2]$ or the recent 10 months. In the following discussion, we would adopt the normalized form of user behavior vector. Having user behavior vectors within an audit period, the anomaly analysis is performed from both parallel and diachronic views. We present the detection methods in the following two sections, respectively. For reference, Table 1 summarizes the variables and notation used throughout the paper.

3.2 Problem and Evaluation Framework

In this section, we present the data-driven insider threats evaluation framework. Since we do not have any background on insider threat patterns in advance, a normal pattern is learned from data processed by the people in the same position. Based on the fact that users in the same position should take on similar responsibilities and perform closely in a system, an anomaly is justified against this standard.

Table 1 A summary of symbols

Symbol	Description
τ	Audit period
$c_u^\tau(a)$	Occurrences of user u perform activity a at τ
v_u^τ	User $u's$ behavior vector at τ
$\bar{v}^\tau$	Normal pattern of a group of users at τ
$\kappa_o, C_{\kappa_o}^\tau$	Parallel overall metric and anomaly set
$\kappa_l, C_{\kappa_l}^\tau$	Parallel local metric and anomaly set
$\kappa_h, C_{\kappa_h}^\tau$	Diachronic metric and anomaly set

Definition 5 *(User Group and Standard Vector)* Given an audit period τ and a group of users U, the standard vector $\bar{v}^\tau$ is defined as the average of user behaviors:

$$\bar{v}^\tau = \left(\bar{p_1}, \ldots, \bar{p_i}, \ldots, \bar{p_{|\Psi|}}\right) \tag{4}$$

where

$$\bar{p_i} = \frac{\sum_{u \in U} c_u^\tau(a_i)}{\sum_{a_j \in |\Psi|} \sum_{u \in U} c_u^\tau(a_j)}, \quad i \in [1..|\Psi|] \tag{5}$$

Definition 6 Given an audit period τ, a set of users U, a metric κ and a positive real number as a threshold $\gamma \in R^+$, a user $u \in U$ is called an *anomaly* if $\kappa^\tau(u, U) > \gamma$.

The anomaly evaluation considers both parallel and diachronic aspects. From the parallel view, the metric κ can be chosen as a specified audit period for a group of users. Users who deviate from normal patterns are evaluated malicious and the deviation degree is regarded as an evidence with respect to a suspicious level. From the diachronic view, the metric κ is set as the incremental pattern against the same user group. The historical detection compares a user behavior with one's historical behaviors, as well as with other users' historical deviation. Such diachronic analysis can avoid the false-positive case on *active* persons.

To be mentioned here, our purpose is to figure out potentially malicious insiders for future examination rather than to make a definite judgment.

In this paper, we also detect the collusion of multiple persons by learning user relationship from log data. This justification is motivated by the actual risk highly relying on data sensitivity. The gang who tightly cooperate on many sensitive tasks should be kept eyes on. We would discusses the details in the following subsections.

3.3 Parallel Overview Metric

In parallel analysis, for each group U, a standard behavior vector $\bar{\nu}$ is computed against formula 4. For each user $u \in U$, its behavior vector ν_u is computed. There are many methods of measuring the difference between two probability distribution vectors, such as Euclidean distance, Manhattan distance and Kullback–Leibler divergence, etc. Since the initial purpose of KL divergence [28] is to measure the uncertainty of one probability distribution against a standard, it is consistent with our measurement. Although KL divergence does provide a measure of discrepancy between two distributions, it is not a true metric, for example it is not symmetric. So we make some modification on KL divergence as our evaluation tool. Given two distribution vectors with the same size $P = \{p_1, p_2, \ldots, p_n\}$ and $Q = \{q_1, q_2, \ldots, q_n\}$, where n is a positive integer denoting the length of the vector, the modified KL distance is computed as follows:

$$\hat{D}(P||Q) = \sum_{i=1}^{n} p_i \times \Lambda \tag{6}$$

where

$$\Lambda = \begin{cases} \Lambda_{\max} & \text{if } p_i = 0 \, or \, q_i = 0 \\ \min\{\Lambda_{\max}, |\ln \frac{p_i}{q_i}|\} & \text{if } p_i < q_i \\ \min\{\Lambda_{\max}, |\ln \frac{q_i}{p_i}|\} & \text{if } p_i > q_i \end{cases} \tag{7}$$

Since $|\ln \frac{p_i}{q_i}|$ and $|\ln \frac{q_i}{p_i}|$ range from 0 to infinity, we set a maximum $\Lambda_{\max}$ to bound them. In practice, the verification of a real number in each dimension being equal 0 is difficult. So a substitutive judgement can be selected in a very small range. The experiments (in Sect. 6) show that the modified KL distance enlarges the difference between two behavior distributions, which makes the abnormal users more obvious than others.

By this modified KL distance, we calculate the deviation of a user behavior vector $\nu_u^\tau = \{p_1, p_2, \ldots, p_{|\Psi|}\}$. Here, we introduce the symbol $\bar{\nu}_{-u}^\tau$ to represent the standard user vector $\bar{\nu}_{-u}^\tau = \{\bar{p_1}, \bar{p_2}, \ldots, \bar{p_{|\Psi|}}\}$ computed against user set U except user u, namely $U - \{u\}$. So, the distance of user u against the normal pattern is given below:

$$\text{Dist}^\tau(u, U^{-u}) = \hat{D}(\nu_u^\tau, \bar{\nu}_{-u}^\tau) \tag{8}$$

The reason of this removal of u is to eliminate the influence of a being evaluated person. The greater the distance, the larger deviation of u's behaviors against the normal pattern.

The coming question is how to choose an appropriate threshold $\gamma_{\kappa_o}^\tau$ to justify an anomaly. A nature choice is the expectation $\mu_{\kappa_o}^\tau$ of all user distances under metric κ_o, say

$$\mu_{\kappa_o}^\tau = \frac{\sum_{u \in U} \text{Dist}^\tau(u, U^{-u})}{|U|} \tag{9}$$

However, such setting may cause many normal users evaluated as anomaly. In practice, not every benign user behaves exactly the same with the normal behavior pattern. To overcome the overstrict justification, we introduce a threshold to select users who obviously deviate from a normal behavior pattern. This metric is based on the principle of *Chebyshev*'s inequality. That is to say for a random variable with finite expected value exception μ and finite nonzero variance σ^2, $Pr(|X - \mu| \geq \gamma\sigma) \leq \frac{1}{\gamma^2}$ holds for any real number $\gamma > 0$. So, for a predicated percentage p, γ can be chosen as $\sqrt{\frac{1}{p}}$. And p percent out of U would be regarded as malicious candidates. Here, $\mu_{\kappa_o}^\tau$ is computed against Eq. 9 and the variance $\sigma_{\kappa_o}^\tau$ is computed against the same set of user behavior deviations. Then, $\gamma_{\kappa_o}^\tau = \sqrt{\frac{1}{p}} * \sigma_{\kappa_o}^\tau$. This method is based on the assumption that there is an average percentage of a population often act differently. Although the percentage value does not always hold for all cases, it does provide a metric for choosing candidates.

This threshold can be adjusted according to different roles and audit periods. Any user whose behavior distance is greater than the threshold is suspicious. Formally, the overview parallel metric κ_o at audit period τ is given as below:

$$\kappa_o^\tau(u, U) = \text{Dist}^\tau(u, U^{-u}) - \mu_{\kappa_o}^\tau \tag{10}$$

We adopt $C_{\kappa_o}^\tau = \{u | u \in U \cap \kappa_o^\tau(u, U) > \gamma_{\kappa_o}^\tau\}$ to denote the set of malicious candidates in this audit period τ.

3.4 Parallel Local Metric

In this subsection, we perform a deeper measurement on anomalies from the local view. We use the local outlier factor (LOF for short), denoted by κ_l, to measure the anomaly degree of malicious candidates in $C_{\kappa_o}^\tau$.

Local outlier factor is used for measuring how much a user is being an outlier [18]. It is based on the concept of a local density, which is estimated by the typical distance at which a point can be *reached* from its neighbors. The locality is given by k nearest neighbors [26]. The degree of an outlier depends on how much a user is isolated from surrounding neighbors.

Given two user behavior vectors ν_u^τ and $\nu_{u'}^\tau$, their distance $D^*(u, u')$ can be calculated by the Euclidean distance D^U or the modified KL divergence $\hat{D}$. For a user $u \in U$ and a setting k, the set of *k-nearest* neighbors is computed to $N_k(u)$. The distance of $N_k(u)$ to u is defined as the max distance between u and each nearest neighbor, namely $k\text{-Dist}^\tau(u) = \max_{v \in N_k(u)} D^*(\nu_u^\tau, \nu_v^\tau)$. The reachability distance

from u is defined as:

$$\mathrm{RD}_k^\tau(u, u') = \max\left\{k\text{-Dist}^\tau(u), D^*(v_u^\tau, v_{u'}^\tau)\right\} \tag{11}$$

The local reachability density of u is the inverse of the average of reachability distance to one's neighbors:

$$\mathrm{lrd}_k^\tau(u) = \frac{|N_k(u)|}{\sum_{u' \in N_k(u)} RD_k^\tau(u, u')} \tag{12}$$

For a setting k, the anomaly degree $LOF_k(u)$ of user u is computed as the local reachability densities of $u's$ neighbors.

$$\mathrm{LOF}_k^\tau(u) = \frac{\sum_{u' \in N_k(u)} \mathrm{lrd}_k^\tau(u')}{|N_k(u)| \cdot \mathrm{lrd}_k^\tau(u)} \tag{13}$$

So, a higher $\mathrm{LOF}_k(u)$ indicates a user acts quite differently with others. If the local reachability density of a user u is quite lower than neighbors, u should be further suspected abnormal. From this definition, we can see that the LOF evaluation is very sensitive to the choice of k. We would verify how k influences the results in the experiment section.

Choosing an appropriate threshold $\gamma_{\kappa_l}^\tau$ for the parallel local metric κ_l can be similar with the solution on the overall parallel metric in last section. For a group of users U, a user $u \in U$, the set of outliers $\mathrm{LOF}_k^\tau(u)$ are calculated. Then, $\mu_{\kappa_l}^\tau$ and $\sigma_{\kappa_l}^\tau$ can be obtained. Under a predicated percentage p of anomalies, the threshold is set $\gamma_{\kappa_l}^\tau = \sqrt{\frac{1}{p}} * \sigma_{\kappa_o}^\tau$. For example, $p = k/|U|$. So, justifying an anomaly at audit period τ is against the following rule:

$$C_{\kappa_l}^\tau = \{u | u \in U \cap \kappa_l^\tau(u, U) > \gamma_{\kappa_l}^\tau\} \tag{14}$$

where $\kappa_l^\tau(u, U) = LOF_k^\tau(u) - \mu_{\kappa_l}^\tau$.

3.5 Diachronic Metric

Although the parallel metrics provide an unbiased justification of abnormal users on a user group in an audit period, it may evaluate an active user as malicious. For example, if user u's job function is flexible, then her anomaly degree is greater than others against parallel analysis metrics. But in fact, $u's$ behavior is normal and is in consistency with historical pattern. To avoid misjudge of a benign activist, a diachronic analysis is needed. The goal of diachronic behavior analysis is to compare a user behavior with one's historical pattern. If one's behaviors are obviously different from his/her historical pattern, this user should be suspected more. Such evaluation also provides a reference for setting importance weights to activities in anomaly detection. If the probability of an activity evaluated anomalous is high in historical periods, this activity is sensitive. So the weight of this activity should be set as a larger one in order to make anomaly detection more effectively.

To make diachronic analysis on a malicious candidate $u \in C_{\kappa_o}^\tau \cup C_{\kappa_l}^\tau$, we need to choose several historical periods as comparative references, denoted by Γ. Comparing with the current audit period $\tau = [t_1, t_2]$, each chosen historical period $\tau' = [t_1', t_2'] \in \Gamma$ should satisfy $t_2' < t_1'$. The time window in audit process can be chosen from the view of desired sensitive requirements. All chosen historical periods should be relative to the current audit period and can represent the overall historical pattern of a target user.

For a suspected user u and two audit periods τ and τ', the behavior vectors are recorded as v_u^τ and $v_u^{\tau'}$. Then, their difference is reversely related with their similarity, denoted by

$$\Delta_u(\tau, \tau') = 1 - \mathrm{sim}\left(v_u^\tau, v_u^{\tau'}\right) \tag{15}$$

There are many functions to calculate the similarity between two probability distributions, such as *cosine similarity*. $\mathrm{sim}(v_u^\tau, v_u^{\tau'}) = \frac{v_u^\tau \cdot v_u^{\tau'}}{\|v_u^\tau\| \cdot \|v_u^{\tau'}\|}$

To justify whether the behavior variance of a user u is normal, we need to get rid of the negative influence of the anomaly candidates. That is to say the normal users are chosen as $\hat{U} = \{u | u \in U \cap u \notin C_{\kappa_o}^\tau \cup C_{\kappa_l}^\tau\}$. We can obtain the behavior vectors of $\hat{U}$ by Definition 4 on both audit periods τ and τ', denoted as $\bar{v^\tau}$ and $\bar{v^{\tau'}}$. Then, standard difference is defined against them as follow:

$$\Delta_{\hat{U}}(\tau, \tau') = 1 - \mathrm{sim}\left(\bar{v}^\tau, \bar{v}^{\tau'}\right) \tag{16}$$

If the behavior of a suspected user is similar with her historical pattern, an auditor may consider his/her practical job functions, especially for those flexible job positions. For a user $u \in C_{\kappa_o}^\tau \cup C_{\kappa_l}^\tau$ and a threshold γ_h, the historical metric is defined

$$\kappa_h^\Gamma(u, U) = \max_{\tau' \in \Gamma}\left\{\Delta_u(\tau, \tau') - \Delta_{\hat{U}}(\tau, \tau')\right\} \tag{17}$$

$\kappa_h^\Gamma(u, U) > \gamma_h$ indicates u being different with others from the historical aspect.

3.6 Justification of Activity

Different with the above section on how to evaluate an abnormal person, this subsection discusses how to find the concrete activity making a user seem abnormal and how much influences it may cause.

For convenience of discussion, we adopt C_κ^τ to denote the union of all suspected user set, say $C_\kappa = C_{\kappa_o}^\tau \cup C_{\kappa_l}^\tau \cup C_{\kappa_f^\Gamma}$.

The core of our method is to assess how much each activity influences the anomaly of a user behavior. Let v_u^τ and $v_u^{\tau'}$ denote the behavior vectors of u at audit period τ with activity a and without a. We evaluate an activity a by comparing the anomaly degree of a user $u \in C_\kappa^\tau$ performing a with the anomaly degree under suppression of a. Let $\kappa_*^\tau(u, U)$ denote an alternative evaluation metric from κ_o, κ_l or κ_h. The difference of u's anomaly degree is denoted as $\Delta_u^\tau(a)$ and is computed as follows:

$$\Delta_U^\tau(u, a) = \kappa_*^\tau(u, U) - \kappa_*^\tau(u, U_{-a}) \tag{18}$$

The larger the value $\Delta_U^\tau(u, a)$, the greater the likelihood that activity a is abnormal. A larger difference indicates a larger influence by a for u being abnormal.

An auditor needs to carefully justify whether such difference is either an inherent character or an important indicator of anomaly. Such evaluation also provides a reference for setting important weights to activities in anomaly detection. If the probability of an activity evaluated anomalously is high in historical periods, this activity is sensitive and a larger weight should be set to this activity in afterward anomaly detection.

Let τ denote an audit period and τ' be its successor period. The weight of activity a_j at τ in anomaly detection is denoted by w_j^τ. The number of users who are detected anomalously on a_j at τ can be used as the reference of anomaly probability of a_j. Formally, for a given threshold δ,

$$\chi_\tau(a_j) = \left|\{u | \Delta_U^\tau(u, a) > \delta \cap u \in C_\kappa^\tau\}\right| \tag{19}$$

So the anomaly probability of a_j is computed as $p_j^\tau = \chi_\tau(a_j)/|U|$. The importance weight of a_j in anomaly detection is updated as:

$$w_j^{\tau'} = w_j^\tau * (1 + \log p_j^\tau) \tag{20}$$

After normalization, an activity weights is:

$$w_j^{\tau'} = \frac{w_j^{\tau'}}{\sum_{a_j \in DT} w_j^{\tau'}} \tag{21}$$

So, a user behavior vector under a weighted anomaly detection in next round of audit is denoted as Eq. 22. This update supports a context-aware learning process for malicious insider detection.

$$v_u^{\tau'} = (w_1^{\tau'} * p_u^{\tau'}(a_1), \ldots, w_{|\Psi|}^{\tau'} * p_u^{\tau'}(a_{|\Psi|})) \tag{22}$$

4 Data-Driven Collusion Detection

The above model detects each user anomaly. In this section, we consider multiple people together collusion. Since in practice, with the improvement of security management, more organizations enforce security constraints in systems by distributing privileges of fulfilling sensitive tasks to multiple users. For example, the Separation of Duty (SoD) enforces that single user only owns reasonable authority. As a consequence, it is difficult for a single user to launch a fraud, and he/she needs to adopt social engineering to collude with others. Hence, the collusion risk highly increases, which relates to the relationship between critical users who actually cooperate on sensitive data. The current assessment of insider threat mostly focus on each user independently and less consideration of band.

4.1 Sensitive Task Relationship

Without loss of generality, an SOD policy can be specified in the form of $SOD(a_1, \ldots a_k)$, which requires k users together complete a sensitive task, where $k \in N^+$ and activities $a_i, i \in [1 \ldots k]$ constitute the task [8]. Users who are assigned the authorizations to perform the activities are called critical users. From the business aspect, users only involved in the same sensitive task have chances to commit fraud. So, our collusion analysis focuses on sensitive-tasks-driven user relationship.

Let T_s be the set of sensitive tasks in an audit period τ, which are defined against organization context. For each sensitive task $t \in T_s$, let U_t denote the set of users involved in t. Let $t.a_i$ denote the person who performed activity a_i. For example, to loan larger than 100 million is a sensitive task. For security purpose, two persons are required to complete this task, say SOD(a_1, a_2). Suppose three users, $Alice$, Bob and $Carol$, are authorized to perform a_1, and two users Dan and $Frank$, are authorized to perform a_2. In τ, there are a set of sensitive loan tasks $T_s = \{t_1, t_2, t_3\}$, which are completed by $U_{t_1} = \{Alice, Dan\}$,$U_{t_2} = \{Carol, Dan\}$ and $U_{t_1} = \{Alice, Frank\}$. We first quantify the relationships between user and sensitive activities and then justify user relationship of sensitivity.

Definition 7 (ι-Sensitive Relationship) Given an audit period τ, a sensitive level $\iota \in N^+$, a security policy $SOD = (a_1, \ldots a_k)$ and a set of sensitive tasks T_s^τ at τ, the relationship between user u and activity $a_i \in$ SOD is defined as follow:

$$s_\iota^\tau(u, a_i) = \sum_{t \in T_s^\tau} t.\text{level} * 1(t.\text{level} > \iota \wedge u = t.a_i)) \tag{23}$$

where t.level is the sensitivity level of task t, $1(*)$ is the index function which equals 1 when the expression holds and 0 otherwise.

ι-sensitive relationship quantifies how many times a user participates in sensitive tasks by performing a specific activity. So the cumulative risk relates to task sensitivity. Having this relationship, we can create a weighted bipartite graph, which provides a reference to justify the chance for multiple users to commit fraud.

Definition 8 (ι-Sensitive Graph) Given an audit period τ, an ι-Sensitive Graph $G_\iota^\tau =< A \cup B, E, W >$ is defined as a bipartite graph, where the nodes in A represent users and B is the set of sensitive activities in SOD. Each weighted edge represents the ι-sensitive relationship $s_\iota^\tau(u, v)$ between u and v.

Definition 9 (γ-SOD Graph) Given a positive real number $\gamma \in R^+$ as a risk threshold, an ι-sensitive graph G_ι^τ and a policy $SOD(a_1, \ldots a_k)$ (SOD $\subset B$), the γ-SOD Graph γ-$G_\iota^\tau =< A \cup \text{SOD}, E^\gamma, W^\gamma >$ is a subgraph of G_ι^τ such that $\forall(u, v) \in E^\gamma, s_\iota^\tau(u, v) > \gamma$.

A γ-SOD graph remains the user activity relationship that is higher than γ, based on which, an auditor detects a high risky band by verifying whether there is a Vertex Cover on SOD nodes. The threshold γ can be set by auditors against the importance of business, that evaluates the potentially damage in case an attack occurs. The higher γ, the higher fraud risk.

4.2 Critical User Relationship

Definition 10 *(User Closeness)* Given an audit period τ, a sensitive task set T_s^τ and two users $\forall u_i, u_j \in U$ involved in T_s^τ, user closeness is defined as their accumulative sensitive tasks,

$$r_{ij}^\tau = \sum_{t \in T_s^\tau} t.\text{level} * 1(u_i \in U_t \wedge u_j \in U_t) \tag{24}$$

where t.level is the sensitivity level of task t, $1(*)$ is the index function which equals 1 when the expression holds and 0 otherwise.

User closeness quantifies how many times two users together participate in sensitive tasks. So for each SOD policy, the set of users involved in related sensitive tasks consist of a weighted graph $G^\tau = <V, E, W>$, called user relationship graph, where each node in V represents a user and the weight of each edge denotes user closeness. Since a security policy requires a set of users together to complete a sensitive task, this graph also provides a reference to justify the chance for multiple users collusion.

Definition 11 (γ-User Graph) Given an audit period τ, a user relationship graph $G^\tau =< V, E, W >$ at τ and a positive real number $\gamma \in R^+$ as a threshold, the γ-user graph $G_\gamma^\tau =< V, E_\gamma, W_\gamma >$ is a subgraph of G^τ, where the weight of each edge $(u, v) \in E^\gamma$ satisfies $w_{uv}^\tau > \gamma$.

For each security policy SOD $= (a_1, \ldots a_k)$, the user relationship graph reflects user collaboration on sensitive tasks. So an auditor can set a risk threshold γ and create a γ-user graph G_γ^τ. Then, a high risky band can be detected by verifying whether there is a k clique on G_γ^τ. If we together consider user relationship in real life, the collusion probability increases, such as family relatives, previous colleagues, graduates from the same university, common friends, etc. Auditors should pay more attention on such users or reallocate their responsibilities.

5 Discussion

In this section, we discuss the adaptability of the proposed model. Firstly, we adopt a role-based access control (RBAC) system to illustrate how to enforce the proposed model into practical applications.

In an RBAC system, users complete their responsibilities via assigned roles. User behaviors in system are recorded in log files, denoted as $Operation\ Record$. Users with the same role should behave similarly. For user u, each operation on activity a for task ts via role r is recorded in the form of rec $= <u, r, a, ts, t>$ in the system, where t is the timestamp. An auditor can justify the sensitive tasks for anomaly audit, and a behavior vector is created against the sensitive tasks for each user. Audit commands are used to select the behavior records of audited users from log files.

Definition 12 *(Audit Command)* An audit command is denoted as a tuple AC=<$user, role, a, task, t_1, t_2$>, where $user \in U$, $role \in R$, $a \in A$, and $task \in T$. t_1, t_2 are the time stamps satisfying $t_1 < t_2$.

The selected behaviors by an audit command are denoted as REC, satisfying $REC = \{rec | rec.u = user \wedge rec.r = role \wedge rec.a = activity \wedge rec.ts = task \wedge t_1 \leq rec.time \leq t_2, rec \in LogFile\}$. Concerning different purposes, the above audit command can be in some special forms. Each entry in the audit command can be specified as '$*$' if we want to select all corresponding records. The following are some examples.

Example 1 (Select the records about a user)

$AC_1 =$ <Alice, $*$, $*$, $*$, 20140101, 20140105> selects all operation records performed by $Alice$ between 20140101 and 20140105.

AC_2 = <Alice, casher, $*$, $*$, 20140101, 20140105> restricts this selection to the role $casher$.

Example 2 (Select the records about a specific role)

AC_3 = <$*$, casher, $*$, $*$, 20140101, 20141212> selects all user action records on role $casher$ during the period of [20140101, 20141212].

Example 3 (Select the records about a specific task)

AC_4 = <$*$, $*$, $*$, loan, 20140101, 20140110> selects all users' operations about the task $loan$ during the period of [20140101, 20140110].

The basic idea of the proposed model is to quantify insider threat via data that a user actually processed in a system. It resides on two aspects: the probability of fraud and the potential loss it may cause. The former depends on the occurrences of performing sensitive tasks, and the later depends on a task sensitivity. As previously discussed, we adopt the sensitivity level to represent the importance of a task. For example, to loan 100 million is more risky than to loan 100 thousand. Also, to loan 100 million to a company with 5 star credibility is less risky than to a 1 star credibility company. It is easy to understand that a manager can set up a set of rules to specify such assessment on business risk. Each rule maps to one transaction dimension as Defined in 3.

Example 4 Given a set of business rules on risk level RS = {([0, 10^6], Low), ([10^6, 10^8], Mid), ([10^8, $-$], High)} and a set of credibilities of company CRED = {(A, l_1), (B, l_2), (C, l_3), (D, l_5)}, we can specify a set of transaction dimension as Ψ = RS $\times$ CRED = {(Low, l_1), (Low, l_2),..., (High, l_5)}. The size of Ψ is 3*5 = 15. A highly sensitive activity can be associated with a lager weight. For example, (High, l_1) = 0.8 denotes the risk weight is 0.8 for a loan larger than 10^8 to a l_1 low-credibility company. Having these rules, user behavior vectors are created against transaction dimension that they actually performed. So, the proposed model can quantify highly risky tasks and find critical users.

Although the specification of transaction dimension may vary in different situation such as organization and audit periods, it does provide a quantification on business risk. In this context, using the number of performing sensitive activities can reflect the probability of fraud.

Another aspect needed justification is the normal pattern. Although it is learned from group users, it does not mean that only the $average$ user is benign. We have proposed a set of metrics to evaluate a user behavior from several aspects, such as parallel overall metric, parallel local metric and historical metric, etc., which should be taken into account so as to find those users who deviate much from several normal patterns. To be mentioned here, many departments materialize well-defined separation of duty policies. But it does not influence our justification since we evaluate users on the same position or same role in a system. Considering the case of multiple people collusion, we adopt the second model to detect them.

Compared with previous works, our model has three contributions: it integrates business context into risk quantification; it proposes several metrics to quantify user behavior risk from different aspects; and it presents the collusion detection.

6 Experiments

In this section, we evaluate the performance of the proposed method. We firstly introduce the real dataset and how to inject anomalies. Then, we analyze the effectiveness and efficiency of our method from two aspects.

6.1 Datasets

This paper is about anomaly detection of user behaviors. An ideal dataset should be obtained from a practical system with concrete job functions. But in fact, it is very sensitive for almost every organization or company. We adopt two real datasets.

The first dataset $SALE$ adopts the transaction records from a provincial department of a large energy retail enterprise. It contains 37 millions data items collected from Jun 2014 to April 2015. Each record maps to a transaction. There are total 6886 concrete persons in the dataset. We first analyze each attribute of the transaction data and finally create the transaction dimensions DT with the size of 60. Then, all users are clustered into 30 categories against DT. Each cluster is regarded as a business position in our evaluation. We choose four representative clusters and the details are as follows: cluster 13 with size 1279, cluster 11 with size 764, cluster 30 with size 417 and cluster 23 with size 218.

The second dataset STU is the collected network access record from a group of students in our university. We adopt Wireshark, a free and open-source packet analyzer. The dataset contains 2,655,223 records, 221 students, which were collected in January, 2015. We focus on student access of network during 24 hours of each day, which are regarded as activities. Each user behavior vector is computed against these activities. The normal behavior patterns are created by learning the behaviors of all students. In the whole process, students did not know they were monitored and the collection of user behaviors are without any bias.

Although there are unusual students in real dataset, we also inject some anomalies for detection. The injected anomalies are set differently with the normal behavior pattern from

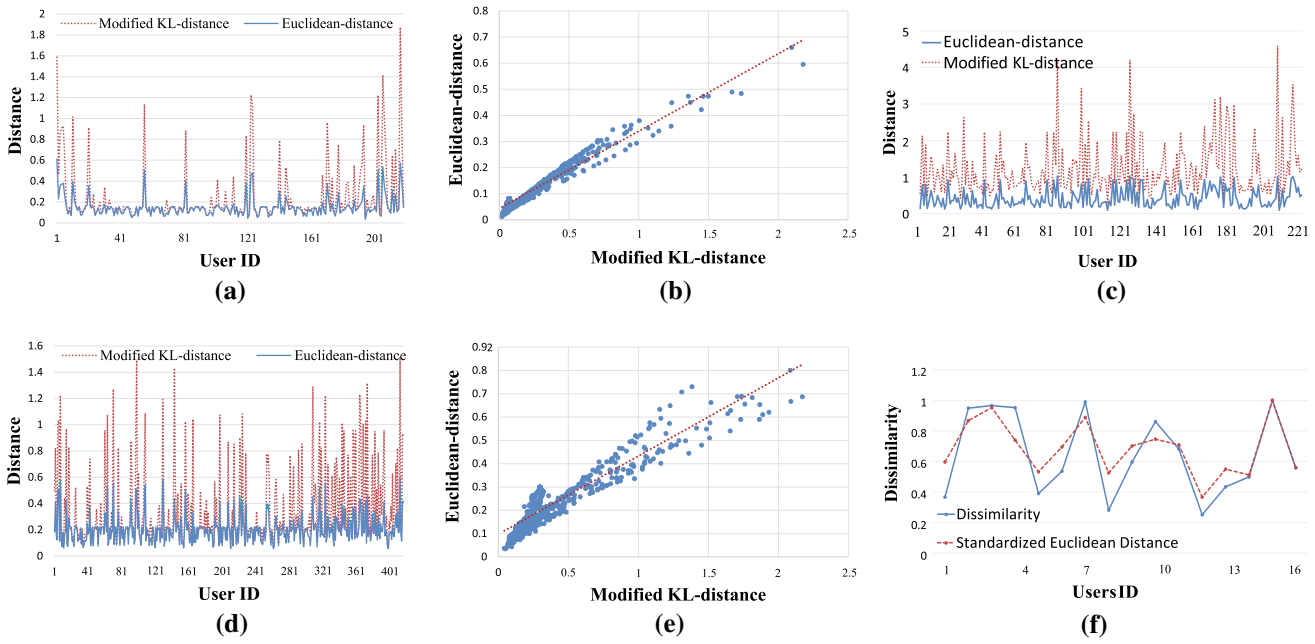

Fig. 1 Evaluation against different metrics. **a** Cluster 23, **b** Cluster 13, **c** Parallel analysis, **d** Cluster 30, **e** Cluster 11, **f** Historical analysis

several aspects. We set an anomaly factor p to indicate the percentage of abnormal users out of the whole user set, say $|U| \times p$ outliers are injected. For each outlier u, the injection is achieved via three random processes. The first random process is to generate a percentage p, which means there are $|U| * p$ users abnormal. The second random process is the percentage α of abnormal activities such that $|A| * \alpha$ activities are randomly chosen as abnormal. The third random process is to generate the value $p_u^{a_j}$ for different activity j, which means the execution of activity is abnormal. By this injection rule, we generate a variety of datasets with different parameters p and α, etc. For each setting, we perform 10 experiments and report the average values.

6.2 Parallel Analysis

We first perform the parallel analysis. Figure 1a, d shows the anomaly evaluation on dataset $SALE$ by both the modified KL-distance and the Euclidean distance. The larger the each point values, the more abnormal behaviors are. From the results, we can see that the peaks on different distance metrics follow the same rules. This indicates that users obviously deviate from normal pattern and two metrics are consistent in detecting anomalies. Differently, the modified KL-distance amplifies the deviation than Euclidean distance.

Figure 1b, e present the abnormal results on dataset $SALE$ in another way. The x-axis gives the modified KL-distance, while the y-axis gives the Euclidean distance. From the results, we can see that for a given k, the top-k abnormal users are slightly different. In cluster 13, the top-k users overlap more against two metrics than in cluster 11. After having a look at the concrete data, we find many zero values in abnormal behavior vectors of cluster 11, which cause the distance computation different. Overall, the more the points reside along the approximation line, the more consistent they are on the behavior detection against two distance metrics.

The results of parallel analysis on dataset STU are shown in Figure 1c, f. From the results, we can see that there are 9 distinct peaks on the modified KL-distance, which means these users obviously deviate from normal pattern. This follows the similar trend as results in $SALE$.

Figure 2 shows the results of parallel local outlier detection based on LOF metric on dataset *SALE*. The Fig. 2a, b, d, e refers to results on Cluster 11, 13, 23 and 30. In each subfigure, the x-axis refers to the number of abnormal users and the y-axis is the predicated percentage p. Since the computation of LOF is based on the k-nearest neighbors, we adopt different colors to represent the result of different k. In these figures, the number of abnormal users increases with p under a certain k. This is because an increasing p indicates lower anomaly criterion to identify users anomaly. Given a fixed p, the results vary under different ks. For example, when $p = 10\,\%$, the number of abnormal users is the highest under $k = 4$ in Fig. 2a, while the highest number occurs under $k = 3$.

6.3 Historical Analysis

We perform a historical anomaly detection for each suspected user by comparing his/her behavior in a period to historical

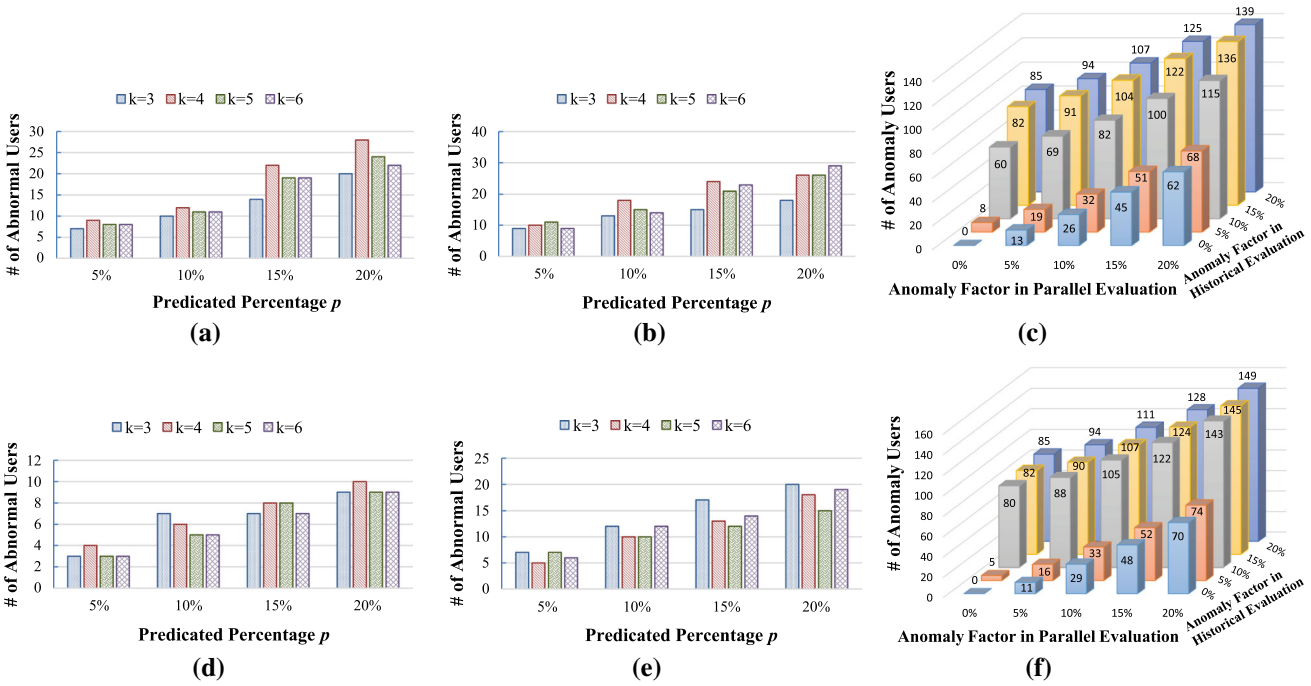

Fig. 2 Local outlier detection and consistence analysis. **a** Cluster 11, **b** Cluster 13, **c** Modified KL-distance, **d** Cluster 23, **e** Cluster 30, **f** Euclidean distance

records so as to find an abnormal change. Figure 1f shows the historical variance of potential anomalies in STU. The blue solid line is the variance by Eq. 15, and the red dotted line is the normalized Euclidean distance. In the figure, the 6 high points in blue solid line are the injected outliers, which are obviously different from historical behaviors. Comparatively, the 9 low points in blue line are only slightly different with their historical behaviors, who are actually real users in dataset. Although their behaviors are deviated from others, they conform to the similar regular routines, respectively. This proves that our diachronic analysis is effective for anomaly detection and can avoid false detection on normal users.

To make a comparison between parallel and historical evaluation, we evaluate the concrete anomaly persons on $SALE$ dataset. The results are shown in Fig. 2c, f, where the x-axis is the parallel evaluation against different chosen percentages p, the y-axis is historical evaluation against different percentages p and the settings of p are 5 %, 10, 15 and 20 %. The z-axis is the number of detected anomalous users on the corresponding x and y settings. The left figure adopts the modified KL-distance, and the right figure shows the results against the Euclidean distance. For example, on the point ($x = 5\,\%$, $y = 5\,\%$) in Fig. 2c, $z = 19$ means 19 users are detected as anomaly in both parallel and historical evaluation against $p = 5\,\%$. If the point resides on x-axis, namely ($y = 0$), the anomalous users are detected only by the parallel evaluation, who are evaluated normal in the historical detection. Such intersect examination can help an auditor better understand user behaviors.

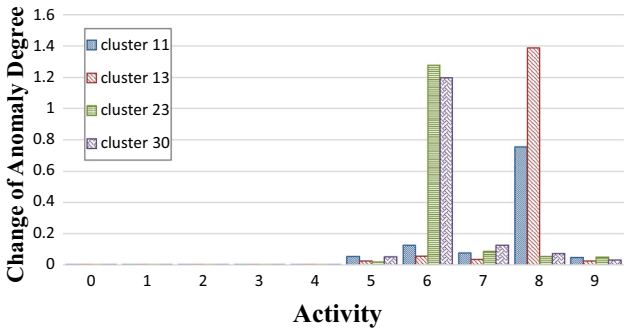

Fig. 3 Anomaly activity detection

6.4 Anomaly Activity Detection

Figure 3 shows the results of anomaly activity detection. The x-axis refers to 10 different activities, and the y-axis gives the change of anomaly degree without a certain activity, which is computed according to Eq. 18. From this picture, we can see that the 6th activity, which is the amount of diesel purchase in the real dataset, is abnormal in cluster 23 and 30. While for cluster 11 and 13, the anomaly activity is the 8th activity, which is the amount of No. 92 gasoline purchase.

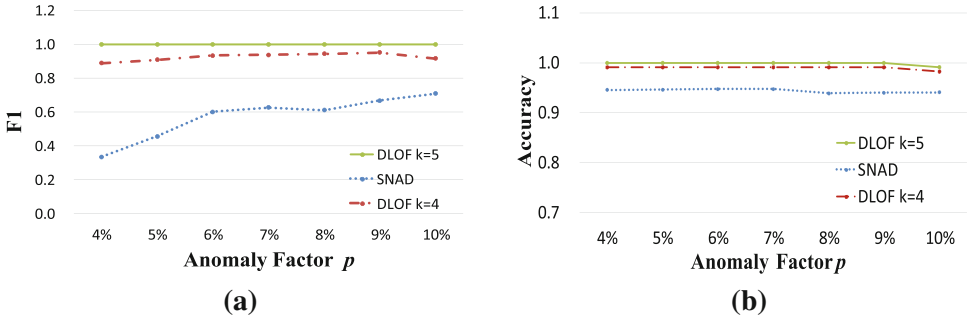

Fig. 4 Comparison with related methods. **a** F1, **b** Accuracy

Table 2 $F1$ score

k	Anomaly Factor p						
	4 (%)	5 (%)	6 (%)	7 (%)	8 (%)	9 (%)	10 (%)
3	33.3	45.5	60.0	62.5	61.1	61.9	58.0
4	88.9	90.9	93.3	93.8	94.4	95.2	91.7
5	100	100	100	100	100	100	100
6	100	100	100	100	100	100	100

Table 3 *Accuracy* measurement

k	Anomaly factor p						
	4 (%)	5 (%)	6 (%)	7 (%)	8 (%)	9 (%)	10 (%)
3	94.6	94.6	94.7	94.7	93.9	93.1	91.5
4	99.1	99.1	99.1	99.1	99.1	99.1	98.3
5	100	100	100	100	100	100	99.2
6	100	100	100	100	100	100	100

6.5 Comparison with Related Method

We compare our method with the most related work, which is the specialized network anomalous insider actions detection method (SNAD for short) proposed in [25]. We measured the performance by the well-known metrics $F1$ measure and $Accuracy$, which are defined as follows,

$$F1 = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}}$$
$$\text{Accuracy} = \frac{|TP| + |TN|}{|U|} \tag{25}$$

where $|TP|$ means the number of true positives and $|TN|$ means the number of true negatives.

The parameters in our method are $k = 4$ and $k = 5$, respectively. Figure 4a shows the comparison of $F1$ measure and Fig. 4b compares the $Accuracy$ of two methods. Both results show that our method outperforms SNAD on different k settings.

6.6 Evaluation of Parameter Setting

In the anomaly measurement, the results highly rely on selection of the parameter k. Malicious users are identified by anomaly degree in our model, so the performance is very sensitive to k. We evaluate our method by $F1$ and $Accuracy$ measures. The results are shown in Tables 2 and 3. Overall, both $F1$ and $Accuracy$ are high under different anomaly factor p settings and a larger k brings a better result. For example, although $F1$ is relatively small in the case $k = 3$, it increases well after $k > 4$, almost 100 %. As to $Accuracy$, our method is more effective, and the accuracies are greater than 90 % with different anomaly factor p in all different k settings.

7 Conclusion

In this paper, we tackle the insider threat problem by auditing user behaviors from both parallel and incremental views with probabilistic methods. The basic idea is that users associated with the same responsibilities should behave similarly in an information system. Users who deviate from normal patterns are regarded as malicious candidates and the anomaly degree is evaluated as an evidence in conformity with malicious probability. To avoid false justification on benign active users, we analyze the anomalies from a diachronic view by comparing a user behavior with one's historical data. We also justify the negative influence on abnormal degree by activities, which provides a reference for setting importance weights to activities in anomaly detection. At last, we perform experiments to verify our methods and the results show it is effective in detection. In the future, we would investigate more sophisticated patterns on insider threat and use real dataset to improve the metrics.

Acknowledgments This work is supported by the National Natural Science Foundation of China (61173140), SAICT Experts Program, Special Program on Independent Innovation & Achievements Transformation of Shandong Province (2014ZZCX03301) and Sci-

ence & Technology Development Program of Shandong Province (2014GGX101046).

References

1. Cappelli DM, Moore AP, Trzeciak RF (2012) The CERT guide to insider threats: how to prevent, detect, and respond to information technology crimes (Theft, Sabotage, Fraud). Addison-Wesley Professional
2. Bertino E (2012) Data protection from insider threats. Synthesis Lectures on Data Management 4.4, pp 1–91
3. Mayhew M, Atighetchi M, Adler A et al (2015) Use of machine learning in big data analytics for insider threat detection. In: IEEE Proceedings of military communications conference. pp 915–922
4. Richardson R (2011) 15th annual 2010/2011 computer crime and security survey. Computer Security Institute, New York
5. Harel A, Shabtai A, Rokach L et al (2012) M-score: a misuseability weight measure. IEEE Trans Dependable Secure Comput 9(3):414–428
6. Thong TV, Buttyán L (2012) Query auditing for protecting max/min values of sensitive attributes in statistical databases. Springer, Berlin, Heidelberg
7. Sun Y, Li N, Bertino E (2011) Proactive defense of insider threats through authorization management. In: Proceedings of 2011 international workshop on Ubiquitous affective awareness and intelligent interaction, ACM
8. Li N, Tripunitara MV, Bizri Z (2007) On mutually exclusive roles and separation-of-duty. ACM Trans Inform Syst Secur (TISSEC) 10.2:5
9. Rafael A, Thomas S (2008) Automated privacy audits based on pruning of log data. In: Proceedings of the 12th enterprise distributed object computing conference workshops, Washington, USA, pp 175–182
10. Accorsi R (2008) Automated privacy audits to complement the notion of control for identity management.Policies and Research in Identity Management. Springer, Newyork
11. Kamra A, Terzi E, Bertino E (2008) Detecting anomalous access patterns in relational databases. VLDB J 17(5):1063–1077
12. Lee SY, Low WL, Wong PY (2002) Learning fingerprints for a database intrusion detection system. Computer security?aESORICS. Springer, Berlin, Heidelberg
13. Mathew S, Petropoulos M, Ngo HQ et al (2010) A data-centric approach to insider attack detection in database systems. Recent advances in intrusion detection. Springer, Berlin, Heidelberg
14. Gavai G, Sricharan K, Gunning D et al (2015) Detecting insider threat from enterprise social and online activity data. In: Proceedings of the 7th ACM CCS international workshop on managing insider security threats. pp 13–20
15. Greitzer FL, Ferryman TA (2013) Methods and metrics for evaluating analytic insider threat tools. In: IEEE of security and privacy workshops (SPW), pp 90–97
16. You C, Nyemba S, Malin B (2012) Detecting anomalous insiders in collaborative information systems. IEEE Trans Dependable Secure Comput 9(3):332–344
17. Kittler J, Christmas W, de Campos T et al (2014) Domain anomaly detection in machine perception: a system architecture and taxonomy. IEEE Trans Pattern Anal Mach Intel 36(5):845–859
18. Breunig MM, Kriegel HP, Ng RT et al (2000) LOF: identifying density-based local outliers. ACM Sigmod Record, ACM, vol 29.2, pp 93–104
19. Agrawal R, Bayardo R, Faloutsos C et al (2004) Auditing compliance with a hippocratic database. In: Proceedings of the thirtieth international conference on very large data bases, VLDB endowment, vol 30, pp 516–527
20. Zimniak M, Getta JR, Benn W (2014) Deriving composite periodic patterns from database audit trails. Intelligent information and database systems. Springer International Publishing, Berlin
21. Shebaro B, Sallam A, Kamra A et al (2013) PostgreSQL anomalous query detector. In: Proceedings of the 16th international conference on extending database technology, ACM, pp 741–744
22. Mascaro S, Nicholso AE, Korb KB (2014) Anomaly detection in vessel tracks using Bayesian networks. Int J Approx Reason 55(1):84–98
23. Du H, Wang C, Zhang T et al (2015) Cyber insider mission detection for situation awareness. Intelligent methods for cyber warfare. Springer International Publishing, Berlin
24. Chen Y, Malin B (2011) Detection of anomalous insiders in collaborative environments via relational analysis of access logs. In: Proceedings of the first ACM conference on data and application security and privacy, ACM
25. Chen Y, Nyemba S, Zhang W et al (2012) Specializing network analysis to detect anomalous insider actions. Secur Inform 1(1):1–24
26. Kriegel H P, Kroger P, Schubert E et al (2009) LoOP: local outlier probabilities. In: Proceedings of the 18th ACM conference on information and knowledge management, ACM, pp 1649–1652
27. Sun Y, Wang Q, Li N, Bertino E (2011) On the complexity of authorization in RBAC under qualification and security constraints. IEEE Trans Dependable Secure Comput 8.6:883–897
28. Kullback S (1987) Letter to the editor: the Kullback-Leibler distance. Am Stat 41.4:340–341