Dynamic and semantic-aware access-control model for privacy preservation in multiple data center environments

Aiguo Chen, Guoming Lu, Hanwen Xing, Yuan Xie and Shunwei Yuan

Abstract
With the rapid development of intelligent perception and other data acquisition technologies in the Internet of things, large-scale scientific workflows have been widely used in geographically distributed multiple data centers to realize high performance in business model construction and computational processing. However, insider threats pose very significant privacy and security risks to systems. Traditional access-control models can no longer satisfy the reasonable authorization of resources in these new cross-domain environments. Therefore, a dynamic and semantic-aware access-control model is proposed for privacy preservation in multiple data center environments, which implements a semantic dynamic authorization strategy based on an anomaly assessment of users’ behavior sequences. The experimental results demonstrate that this dynamic and semantic-aware access-control model is highly dynamic and flexible and can improve the security of the application system.

Keywords
Privacy preserving, access control, anomaly assessment, behavior sequence, dynamic authorization, scientific workflow

Date received: 20 February 2020; accepted: 2 April 2020

Handling Editor: Yan Huang

Introduction
With the development and application of data acquisition equipment and technology in the Internet of things, the joint use of multiple data centers is now regarded as essential for many online services. Simultaneously, data center privacy has also become a focus of attention. For example, Europe’s General Data Protection Regulation (GDPR) requires data center users to focus on privacy.

Due to the dynamic and heterogeneous nature of multiple data centers, privacy and security are regarded as the most difficult challenges; the misuse of legitimate access to data is a severe information security concern for both organizations and individuals. For example, after the 9/11 attacks, the US government allowed a greater sharing of information among government, public security, military and other departments as a defense procedure against future terrorist strikes. However, this plan leads to a massive leak of sensitive data by low-ranked personnel. From a security engineering viewpoint, this problem is partially due to abuse of privileges that have been granted by...
authentication or authorization services, both of which may be regarded as functions of access control.

Access control constitutes a fundamental aspect of security and privacy protection. Typically, access control is used to prevent unauthorized users from gaining access to system resources, to prevent authorized users from accessing resources in an unauthorized manner, and to allow authorized users to access resources in an authorized manner. Due to the complex and dynamic nature of the multi-center big data application field, some of these factors cannot be addressed or phrased in terms of traditional access control. In Crampton and Huth,4 a new access-control architecture was formulated, the realization of which might form part of an overall strategy for addressing the insider problem. In this architecture, trustworthiness and risk-assessment methodologies were combined and extended in traditional role-based access control. Here, a risk-assessment methodology is used to make decisions based on the context of the resource, and the requestor’s trustworthiness is defined as the probability of the requestor not abusing the authorization, which could be derived via statistical analysis of the requestor’s behavior.

Moreover, cross-center data processing applications are typically implemented as workflows,5 which renders the access-control-based privacy protection more complicated. A work flow could be represented by a task graph G = (V, E) that consists of a set of tasks that are represented by vertices V. In mass workflow environment systems, tasks are executed sequentially according to the execution order. Therefore, it is impossible to identify unusual entities from a running system that deviate from its normal pattern of behaviors. The use of user access sequences for dynamic access control provides a new approach.6 However, the current sequence anomaly analysis is still used in the application of system security detection, and little research has been conducted on the application of user access control. In-depth research on abnormal user behavior in access requests for resources or services is lacking. Therefore, a dynamic and semantic-aware access-control (DSAAC) model that is based on sequence anomaly evaluation is proposed by considering the characteristics of the workflow in a multiple data center environment. Through the sequence anomaly evaluation method and by introducing semantic constraints, users and resources in the access-control process can be restricted, and simultaneously, user resource associations can be defined; hence, this process is suitable for dynamic access control in a massive-resource environment.

The rest of this article is organized as follows. Section “Related work” presents the concept of the access control and different access-control models. Section “Our DSAAC model” presents the scheme and specific algorithm steps of the dynamic access-control model. Section “Experiments and analysis” presents experiment and analyzing. Finally, section “Conclusion” concludes the article.

Related work

From privacy and security perspectives, access control is one of the most fundamental security methods. Access control allows authorized users to obtain physical or logical access to various resources and ensures the confidentiality and integrity of system resources.7,8 Earlier, most information systems were small in scale and number of users, simple in business, and corresponding access-control systems mainly adopted the method based on access-control list. Along with the increasing complexity and scale of systems, in order to improve the flexibility of authorization management, role-based access-control (RBAC) model was proposed in combination with the management model of user authorization and role authorization.

Based on the proposed concepts of tasks and workflows, the task-based access-control (TBAC) model has begun to receive research attention, which focuses on the process of user execution and authorizes each requested resource according to the execution status of the current task.9–11 Many scholars have expanded TBAC according to their needs and have proposed such models as the task-role-based access-control (TRBAC) model12–14 and the extended TRBAC model.15,16 These models are expanded versions of the TBAC model and have realized satisfactory application results in various scenarios. However, extended access-control models of this type still use static policy rules, do not effectively use historical execution information, and cannot dynamically adjust the allocation of permissions according to the historical execution of tasks and other issues. Due to the relative complexity of TBAC in formulating static authorization policies, less research has been conducted on TBAC.

In a massive-resource environment, dynamic authorization is an effective technique for providing dynamic resource authorization by combining user behavior, context, task status, and other attribute information. Dynamic authorization approaches are characterized by the use of not only the policies but also environment features that are estimated in real time to determine access decisions. Among them, the risk-assessment method is an effective dynamic solution for assessing the uncertainty of a user’s behaviors in a complex environment to control the insecurity from a requestor. In Zheng and Cai17 and Diep et al.,18 a model for risk assessment has been proposed, which provides a useful reference for the consideration of context in risk assessment. An access-control model that is based on user behavior, context vulnerabilities, and resource
properties is proposed in Bouchami et al., but it focuses only on defining the risk level between collaborative environments. Lakshmi et al. proposed a model for the identification of insider attackers by adjusting the session based on risk assessment. Fall et al. also proposed a risk-adaptive authorization mechanism to satisfy the dynamicity in the cloud environment.

In addition, several preliminary studies combined risk and access control in using trust to assign users to roles. In Shaikh et al., a risk-trust authorization mechanism is proposed for assessing user credentials to adjust the access rights dynamically. In Ni et al., risk-based access-control systems that are based on fuzzy inferences are proposed. This study showed that fuzzy inference is a satisfactory approach for implementing a risk-based access-control system.

The current research on dynamic access control mainly focuses on the relationship between the user behavior and the context. In the access-control model that is based on user behavior and context, effective evaluation of user historical data is lacking, and it focuses on the analysis of the current states of users while paying less attention to the attributes of resources.

In workflow systems, a service is a task that is executed over a period and can be formalized into a task sequence. In security research in web and computing environments, anomaly detection in a user behavior sequence has become a hot topic, and remarkable results have been obtained. Sendi et al. uses a hidden Markov model to analyze user commands, to establish a normal behavior archive for the user’s command sequence, and to compare the current sequence with normal behavior sequences to determine the similarity, which is used to distinguish legal users from intruders. Sequence mining is often used to mine hidden user behaviors to realize better personalized recommendation and resource allocation. Sequence mining is also used to detect abnormal behaviors of users. As discussed in Xie and Yu, a hidden Markov model is used to describe the browsing behaviors of web users and to detect abnormal behaviors. Zhou et al. proposed a method for user behavior anomaly detection that is based on data stream sequence mining. By studying the sequence relationships between subsequences, a user behavior anomaly was discovered, and the common problems of low delay and low accuracy in sequence anomaly detection algorithms were overcome. Sequence anomaly detection can provide low-latency and high-efficiency detection if the amount of data is large.

However, in access-control technology, few studies that are based on sequence anomaly detection have been conducted. The user’s behavior is a chronologically ordered set of observation records. A method that is based on sequence anomaly detection is introduced into access control, which can well realize dynamic and flexible authorization in a multiple data center environment. Thus, we study how to introduce the method of sequence anomaly detection into the access-control process and how to improve the available sequence pattern mining and detection algorithm so that it can provide dynamic authorization management in the TBAC scenario and protect the privacy.

Our DSAAC model

In this DSAAC model, users must pass the identity authentication of the system first. Then, the request in the workflow process is formally modeled as a behavior sequence, and the user’s permission for each step of the task execution is decided via the method of sequence anomaly detection. The scheme is illustrated in Figure 1.

![Figure 1. Dynamic and semantic-aware access-control model scheme.](image-url)
subject includes all attributes that are related to the request; include various hardware and software resources; and

object typically a user or a program that is executed by a user; active entity that initiates access to resources, which is

termining, and sequence anomaly detection in detail.

introduce behavior sequence modeling, sequence pat-

tvide normal behavior sequence patterns for the

mining, update the sequence pattern library, and pro-

vide normal behavior sequence patterns for the
sequence anomaly detection module. Next, we will
introduce behavior sequence modeling, sequence pat-
tern mining, and sequence anomaly detection in detail.

Behavior sequence modeling

The modeling of a behavior sequence is the representa-
tion of the user’s request in a standard behavior
sequence format. We represent a request as a behavior
sequence via formalization, pruning, and merging
(Figure 2). Then, we can identify and use this standard
behavior sequence in the pattern mining and anomaly
detection process.

Basic unit definition. The most basic unit in the behavior
sequence is requests. Requests are also associated with
subjects, resources, and tasks, among others. We divide
each request into a triple

where request $R_i$ denotes the user’s request, which
includes all attributes that are related to the request;
subject $S$ denotes the originator of the request and is an
active entity that initiates access to resources, which is
typically a user or a program that is executed by a user;
object $O$ denotes the resources to be protected, which
include various hardware and software resources; and

operation $P$ denotes the operations of the subject on
the object, such as reading and writing.

Standardizing the sequence is equivalent to standard-
dizing the sequence $R$ to $\text{sig}(R)$. As expressed in for-
rmula (1), $f(s)$ represents the standardized function that
is defined on the subject $S$, $g(P)$ is the standardized
function that is defined on the operation $P$, $d(O)$ is the
standardized function that is defined on the object
resource $O$, and $f(s)$, $g(P)$, and $d(O)$ are formulated by
relevant authority

$$\text{sig}(R_i) = \sum_{i=1}^{m} f(s) + g(P) + \sum_{i=1}^{n} d(O) \quad (1)$$

A task is composed of a standard $R_i$ and a semantic
c, and a set of tasks includes the formal description of
the request and the semantic set. A single task is defined
as follows

$$t = \{\text{sig}(R), c\} \quad Fu = (t_1, t_2, \ldots, t_n)$$

where behavior sequence $Fu$ is composed of a series of
tasks $t$ and $n$ represents the length of the behavior
sequence.

We establish a sequence pattern library according to
the behavior sequence. The sequence pattern library is a
database that contains many normal sequence patterns

$$MFu = \{Fu_1, Fu_2, \ldots, Fu_i, \ldots, Fu_m\}$$

Semantic-related definition. Due to the diversity needs
among the types of businesses in various systems, the
modeling of basic units alone cannot satisfy the expres-
sions regarding the tasks, behaviors, and resources. For
example, in a public security system, the staff must spe-
cify a file number to query a case. The file number is
distributed according to the level and grouping of the
staff, and it is private attribute of the staff; it has no
relation with the forward step. This file number is
semantic information. Therefore, we introduce semantic-related definitions. When matching sequence
patterns, semantic constraints are used to check the leg-
ality of current behaviors with resource operations. We
also use semantic information to guide risk assessment
when mining sequence patterns. The following describes
the establishment of a semantic-related model.

Semantic recognition. The main objective of semantic
recognition is to identify semantics. In special tasks, we
must attach semantic information to the tasks. We
typically use a URL to represent a task; thus, specified
semantic information can be appended through URL
parameters. In a user request log, a request record is of
the following format:
GET/http://market.scau.edu.cn/goods.php?id=1407246132sjn1j88b&iaction=view&ast=0f

A request record can be sliced to obtain recognizable information according to the preset regular rule “/id=(\[0-9a-z\]+)&iaction=(\w+)&ast=(\w+)”, and we can obtain triple\{S, O, P\} as [id:1407246132sjn1j88b,"action:view","ast:0f"]

Semantic variables. The semantic variable is defined as a quad (X, S(X), U, G), where X represents the variable name; S(X) is a set of items of X, where each item is a fuzzy variable represented by s and each item is in the range of the field U of the basic item u; and G is the syntax rule for generating the item s of X, as shown in formula (2)

\[ s F(s, \delta), \quad \delta \in u \]  

where s denotes the meaning of the item, \(\delta\) denotes the formalized representation of the item after blurring, and the value of \(\delta\) is in the range \(u(u \subset U)\). These grammar rules can be used, for example, to model, divide classes, define attributes, and define sub-attributes.

Semantic rules. A semantic rule is a regular sentence. A sentence is composed of elements of an alphabet \(\Sigma\), namely, \(x \in \Sigma^*\), where \(x\) is called a sentence on \(\Sigma\). A sentence contains \(<\text{noun phrase}>\), \(<\text{verb phrase}>\), \(<\text{noun phrase}>\), and \([<\text{noun phrase}>\), \(<\text{value phrase}>\)] elements. Formally, a semantic rule is a process of dividing a noun phrase and a verb phrase in a sentence and expressing their relationship using symbols.

For example, consider sentence “User A accesses resource B with semantic constraint person_job_family 1.” The process is as follows.

Use the relevant semantic rules to check the semantics of the statements that are processed above, as expressed in formula (3)

\[ X_\mu \rightarrow X_r \rightarrow X_b \]  

Semantic constraints. Semantic constraints are related constraints from the perspective of resources, such as inclusive relationships, functional relationships, mutually exclusive relationships, self-reflexive relationships, and self-checking constraints. These constraints are combinatorial conditions that are based on common logical relationships, such as \((A \text{ and } B), (A \text{ or } B),\) and \((\text{not } A)\). For example, semantic value A is PERSON_JOB_FAMILY and equals 1; B is PERSON_LOCATION and equals 2. We posit that only when A and B are satisfied simultaneously can they be satisfied normally. Corresponding semantic constraints are defined on each step of the task to constrain the operation of resources. A standard semantic constraint is of the following form:

\(<\text{PERSON_JOB_FAMILY} = = 1 \text{ and } \text{PERSON_LOCATION} = = 2>\)

Sequence modeling process. Based on the above definition, the process of modeling a user behavior sequence will be described below. Each step of the user’s request involves the corresponding subject and object and semantic information. Behavior sequence modeling is the extraction of information for anomaly detection from the user’s request. The steps are illustrated in Figure 3.

In the figure, a request \(R_i\) specifies the requested object resource and related information regarding the request and is expressed as \(\text{sig}(R_i)\). We can obtain standardized semantic environment set \(c_i\) by consulting relevant standards. The current task and semantic

Figure 3. Sequence modeling process.
environment information are regarded as a task \( t \). Collect historical tasks of this request using a data search engine and construct the task sequence (also named the behavior sequence). The behavior sequence can be expressed as

\[
Fu(R_i) = t_0, t_1, t_2, \ldots, t_i, \ldots, t_n
\]

Session information maintenance. Data collection is a prerequisite step for sequence modeling. In this step, we will complete session information maintenance to obtain the forward task. We collect basic subject information and object information that are related to the current request \( R_i \), first, and we collect historical task data that are related to the current request. There are three main sources of historical data for users: server data, customer data, and intermediate data (proxy server data and packet detection).

Semantic environment extraction. Process the basic information of the request to obtain semantic environment information. For example, a standard network request log format is presented in Table 1. Through semantic constraints, useful subject environment semantic information and semantic annotation of object resources can be extracted from basic information, such as the IP address, date, file type, resource domain, and user agent.

Sequence construction. Sequence construction is the process of merging a single request into a request sequence. In the process of sequence construction, it is necessary to conduct data preprocessing, clean up dirty/noisy data, extract and merge data from various sources, and convert the data into a suitable format. We identify tasks by semantics, and we filter out repeated tasks by pruning and setting time windows. Finally, we merge the tasks to construct the task sequence \( Fu \).

Sequential pattern mining

A sequential pattern mining algorithm is used to realize risk assessment. The advantage of using sequential pattern mining is that it can mine more instructive behavior patterns without requiring security administrators to formulate many complicated policy rules. Moreover, the pattern library will not store behavior sequential patterns that have the same meaning, which can reduce the burden of database storage. We utilize a closed sequence mining algorithm that is based on behavior.

Core strategy in algorithm design. The core strategy of the closed pattern mining algorithm is to expand the \( s \) pattern and the \( I \) pattern of the newly added sequence, recursively mine the closed sequence pattern with the expanded sequence, extract the frequent set and remove the closed pattern in the closed pattern mining, and, finally, obtain the frequent sequence.

Due to the particularity of TBAC scenarios, the mining of behavior sequence patterns differs substantially from the traditional mining of sequence patterns. In mining behavior sequence patterns in TBAC scenarios, the following three requirements must be satisfied:

1. **The closure of patterns.** All sequences in the pattern database \( MFu \) must satisfy the closed sequence pattern, namely, there are no patterns that have the same meaning. In the behavior-based closed sequence mining algorithm, the closed sequence pattern must be filtered, and simultaneously, the patterns of the same task must be merged;
2. **The first fixed principle.** In the process of user execution and access, the first operation is always fixed. Hence, only the suffix is extended in the mode extension;
3. **The particularity of behavioral items.** In the process of mining, we define tasks as sequential items. The tasks include requester information, requested actions, and requested resources. Information of this type is difficult to handle in traditional sequential pattern mining.

To overcome the problem that the result set is too large and contains many sequences that have the same meaning, the behavior sequence pattern mining algorithm adopts the closure sequence mining algorithm. For example, only closed sequence patterns are mining and identical are filtered out. We use first fixed frequent sequence for expansion, which can increase the mining efficiency. We also use semantic constraints to extend the pattern.

Basic definition. The relevant definitions for this behavior-based closure mining algorithm are as follows:

**Definition 1.** Task set \( I \) contains all tasks: \( I = \{t_1, t_2, \ldots, t_m\} \).

**Definition 2.** The behavior sequence \( T \) (transactions) is the set of tasks \( t \)

| Table 1. Semantic description. |
|-------------------------------|
| `<ip_addr>` | `<base_url>` |
| `<date>` | `<method>` |
| `<file>` | `<protocol>` |
| `<code>` | `<bytes>` |
| `<referrer>` | `<user_agent>` |
\[ T = \{ t_i, t_i + 1, \ldots, t_j \}, \quad 1 \leq i \leq m \]

**Definition 3.** Behavior sequence set \( D \) (the transaction set) is expressed as \( D = \{ T_1, T_2, \ldots, T_n \} \).

**Definition 4.** Behavior sequence pattern set \( A \) is a set of tasks, and task set \( T \) contains pattern set \( A \), \( A \subseteq T \). K-mode indicates that \( A \) is of length \( k \), namely, it contains \( k \) tasks (items). \( I_1 \) is closed, namely, \( I_1 \) is a frequent sequence in the MFu of the pattern database, there is no sequence \( I_2 \) that satisfies \( I_1 \) as the parent sequence of \( I_2 \), and \( I_1 \) and \( I_2 \) have the same support degree.

**Definition 5.** The degree of support for pattern \( A \), namely, \( A.\text{count} \), refers to the number of transactions in the transaction set that contain the pattern, and \( |D| \) represents the total number of transactions in the transaction set

\[
\text{sup}(A) = \frac{A.\text{count}}{|D|}
\]

**Definition 6.** Frequent patterns and frequent items—given a minimum support \( m_{in\_sup} \), if mode \( A \) satisfies \( A.\text{count} \geq m_{in\_sup} \), we call \( A \) a frequent mode. A single item in task set \( I \) is called a frequent item if it occurs more frequently than \( m_{in\_sup} \) in \( D \)'s transactions.

**Implementation of the sequential pattern mining algorithm.** The flow of the sequential pattern mining algorithm is illustrated in Figure 4.

To overcome the particularity of behavior items, semantic constraints are used for guidance in the mining process. Semantic constraints are defined on each task and are constraints on resources. In this algorithm, we separate \( S \)-extension and \( I \)-extension and check the validity of the extended sequence pattern; the steps are as follows:

**Step 1.** First, we look for frequent items with length \( k \) (starting from 1). We focus on the tasks that are related to the request \( \text{query}_i = \{ S, O, P \} \) in the process of sequence mining, while other information that is specified by the request is used when the pattern expanded.

**Step 2.** For each 1-frequent sequence \( \beta \), establish a suffix map database \( MFu \). Set \( \alpha \) as a sequence pattern in the sequence database \( S \), and map sequence \( \beta' \) of \( \beta \) with \( \alpha \) as a prefix, namely, \( \beta' = \alpha \cup \beta \).

**Step 3.** Call the schema extension to the schema database:

**Step 3.1.** First, check the ending condition to determine if there is a backtracking pattern, namely, if the currently expanded pattern is a subset of the existing pattern or the values of the two patterns are the same. If the backtracking condition is satisfied, end the mode extension, return to step 1, and searching for 2-frequent sequences.

**Step 3.2.** \( S \)-mode expansion, such as \(<(a), (b), (c)> \) to \(<(a), (b), (c), (d)> \). When expanding, semantic check is used to check whether it is the same task, for example, for a task \( t_1 \{ \text{get, } S_1 \} \) and another task \( t_2 \{ \text{get, } S_2 \} \), since \( S_1 \) and \( S_2 \) differ, the traditional mode expansion will regard these tasks as different tasks. \( t_1 \) and \( t_2 \) can be determined to be the same task via semantic checking; hence, only \( t_1 \) items must be expanded.

**Step 3.3.** \( I \)-mode expansion under semantic constraints, such as \(<(a), (b), (c)> \) to \(<(a), (b), (c), (d)> \). Unlike the traditional sequence, the sub-item in the behavior sequence item corresponds to the description of the request resource and the description of the request attribute; hence, we must check the semantic constraints. For example, for behaviors \( t_1 \) and \( t'_1 \), assuming that all operations are the same but the resource fields and types of the operations differ, we can filter this unreasonable pattern extension based on semantic constraint checking.

**Step 3.4.** Add the extended schema to the schema database.

**Step 4.** Continue mining frequent sequences that are of the next length, namely, \( k + 1 \).

Simultaneously, to reduce the storage space, we use a pseudo projection database in our algorithm.
Projected $MFu(P)$ is $P$’s map database, and $MFu$ is a pointer instead of a physical copy.

**Sequence anomaly detection**

The normal sequence patterns that are obtained via sequence pattern mining can be used as a pattern library in sequence anomaly detection. Sequence anomaly detection includes pattern matching of sequences and semantic checking of current behaviors. Pattern matching of the behavior sequence is the comparison of the normal behavior sequence of the user with the current behavior sequence pattern to judge whether the current user request is abnormal. Define $sup_{MFu}(Fu)$ as the support degree of the normal behavior sequence to the current behavior sequence, which has a range of $[0, 1]$; the larger the $sup_{MFu}(Fu)$ value, the greater the coincidence between the current behavior pattern sequence and the normal behavior pattern. Define each sequence in the behavior pattern library as a directed tree $T(Vp, Ep)$ in which each node $v_i \in Vp$ represents a request log $q_i$, which is formally expressed as $\langle \text{sig}(q_i), c_i \rangle$, where $\text{sig}(q_i)$ denotes request $q_i$ and $c_i$ denotes semantic information that is related to this request. The edges $e_{ij} \in Ep$ represent the execution sequence of $q_i, q_j$, namely, $q_j$ is executed after $q_i$.

The detection process of several request sequences in the pattern library and table is illustrated in Figure 5. The patterns in the pattern library are stored as trees. Figure 5(a) shows the stored partial normal sequence patterns in the pattern library. Each task sequence starts at $t_0$. Figure 5(b) shows two normal traffic flow requests, namely, $t_0\rightarrow t_1\rightarrow t_2\rightarrow t_3$ and $t_0\rightarrow t_2\rightarrow t_3$, and Figure 5(c) shows two abnormal traffic requests, namely, $t_0\rightarrow t_2\rightarrow t_3$ and $t_3\rightarrow t_6$. Sequence pattern matching uses the sequence similarity and support for calculation.

The support degree of the normal sequence to the current behavior sequence, namely, $sup_{MFu}(Fu)$, is calculated as the similarity degree between the normal sequence and the current sequence, and the similarity degree depends on the length.

Sequence patterns of equal length, as shown in task sequence $t_0\rightarrow t_1\rightarrow t_2\rightarrow t_3$ in Figure 5(b), have a similarity of 1. If the length is not equal but the prefix of the normal sequence in the pattern library exactly matches the current sequence, the similarity is also 1; for example, the task sequence $t_0\rightarrow t_2\rightarrow t_3$ in Figure 5(b) also has a similarity of 1. Define the similarity between the two cases as expressed in formula (4)

$$Sim(Fu, Fu_i) = 1$$ (4)

The lengths differ, and the prefixes of the current sequence pattern are normal behavior sequences in the matching pattern library, for example, $t_0\rightarrow t_2\rightarrow t_5$. We define the similarity formula in formula (5)

$$Sim(Fu, Fu_i) = \frac{\text{length}(Fu_i)}{\text{length}(Fu)}$$ (5)

Select a set of similar behavior patterns for calculating the support degree of the normal behavior pattern with respect to the current behavior, such as in formula (6), wherein $\text{distance}(Fu_i[i] - Fu[i])$ is the relevant semantic attribute distance on each behavior node, such as the level of resources or the time distance

$$sup_{MFu}(Fu) = \sum_{i} distance(Fu_i[i] - Fu[i])$$ (6)

By combining the length similarity and the semantic attribute distance, we can obtain the support degree for the current mode, as expressed in formula (7), where $m$ is the number of similar behavior modes

$$sup_{MFu}(Fu) = \sum_{i} sup_{MFu}(Fu) * Sim(Fu, Fu_i)/m$$ (7)
Finally, we can grant authorization or not according to the support degree of the normal sequence in the pattern library with respect to the current sequence and the threshold value.

Experiments and analysis

In our experiments, the training and test data sets are obtained from the Amazon access sample data set (https://archive.ics.uci.edu/ml/datasets/), which consists of the visit records for the Amazon website for a week in October 2016. The access is divided into two parts: one part contains the access records of all users, and the other part is the sequence data set after sampling and processing. There are 9,022,000 sequences, the users of these data are identified by serial numbers, and each user ID has a corresponding attribute tag. There are 38,000 normal authorized access sequence records for anonymous users.

This experimental environment is the host of a core i5 processor, with 8GB memory, 256 GB, and Windows 10 operating system.

Performance comparison of mining algorithms

The sequence pattern data-mining model is used to train the legitimate request sequence to obtain the training behavior pattern set. Since the selected training set size and the setting of the minimum support (\textit{minsup}) threshold affect the accuracy of the training data set, we adjust the minimum support to yield the optimal result during the experiment. We select 20,000 legitimate requests as the training set, and we compare the available pattern mining algorithms (the pattern extension-based mining algorithm \textit{prefix} and the closure-based mining algorithm \textit{clospan}) and the closure-based mining algorithm \textit{clospan}). We use various values of \textit{minsup} (5%, 10%, 15%, 20%, 25%, and 30%) to mine frequent sequence patterns, and we compare the three algorithms in terms of efficiency. As presented in Table 2, the sizes of the training sets that are generated by the three algorithms differ among the support levels, and the training set of the prefix algorithm for mining frequent patterns is very large because there are patterns that have the same meaning in the prefix and there are no restrictions. However, the training sets of \textit{clospan} and \textit{DSAAC} are much smaller because the existence of closure patterns is considered in both algorithms, which substantially reduces the sizes of the training sets. Under the previous premises, we compare the running times of the three algorithms, as presented in Figure 6.

The running time of \textit{DSAAC} is substantially shorter than those of \textit{clospan} and \textit{prefix} because we consider the behavior sequence in this \textit{DSAAC} algorithm, and we identify the same task. By introducing semantic rules for guidance, the algorithm can identify closure sequences quickly; thus, it can reduce the time that is required for pattern expansion and mining of each sequence.

The running time is related to the complexity of the algorithm. Since \textit{prefix} will conduct repeated mining on each sequence in the mining process, the time complexity is high, whereas \textit{clospan} only mines frequent sequences under closures; hence, compared with prefix, the running time is reduced by nearly 10 times. Fortunately, our pattern mining algorithm, which is based on the user access request sequence, considers the characteristics of the user sequence and reduces the generation of repeated semantic sequences; hence, it is more efficient than the \textit{prefix} and \textit{clospan} algorithms. Therefore, our algorithm realizes higher efficiency in the pattern mining of user access request sequences. In the following experiment, we will evaluate the correctness of this dynamic authorization scheme.

Experimental analysis of the authorization strategy

First, select several training sets that are generated by the three considered algorithms for testing. To distinguish these training sets, we numbered them; for example, we numbered the five sets of training sets that were generated by \textit{prefix} as 5-prefix, 10-prefix, 15-prefix, 20-
prefix, 25-prefix, and 30-prefix, which are training sets that were generated under supports of 5%, 10%, 15%, 20%, 25%, and 30%, respectively. The other numbers are 5-clospan, 10-clospan, 15-clospan, 20-clospan, 25-clospan, 30-clospan, 5-DSAAC, and 10-DSAAC, and so on.

We design sequence rules that are based on the existing sequence and the pattern that is generated by the DSAAC algorithm, and we design these rules as the access-control policy library in the standard language of the TBAC model for evaluation of the traditional policy-based access-control model. The abnormal sequence in the experiment is constructed by inserting items, deleting items, and modifying item information. The semantic information in the experiment is based on the rule base that is provided by the data platform. Its main function is to identify tasks. Since no relevant professional knowledge is available for reference, this experiment lacks the formulation of semantic constraints. Therefore, in practical applications, security managers can formulate suitable semantic constraints that are based on their professional knowledge. In a perfect semantic library system, the actual application performance of this model should be higher than that in this experiment. The experimental results are presented and analyzed below.

Performance test. We chose the response time as a reference index in the performance test of access control. This experiment compares the three algorithms in terms of the average response time, namely, the time between when each request is sent and the authorization result is obtained.

According to Figure 7, the times for anomaly detection that are realized using the generated pattern databases that are based on the clospan algorithm and the DSAAC algorithm are similar and much shorter than that of the data set that is obtained using the prefix algorithm. This is because prefix generate a large pattern database; thus, the response time is longer than those of the other two algorithms.

Accuracy analysis. We use the statistical false-positive rate and the accuracy rate to evaluate our model.

False-positive rate test. The false-positive rate refers to the proportion of the number of requests that are regarded as abnormal behaviors in the test data. First, we compare the false-positive rates of the training sets that are generated using the three algorithms. The experimental results are presented in Figure 8.

Algorithm DSAAC realizes the lowest false-positive rate. Using a suitable training set, the false-positive rate of this DSAAC algorithm is approximately 5%. Then, we compare the algorithm before versus after the introduction of the semantic constraint (DSAAC with the semantic constraint) and the policy-based access-control model. The experimental results are presented in Figure 9.

As seen from the above figure, after using semantic constraints, the false-positive rate of this algorithm decreased from 7%–5% to 2%–3%, which is close to the policy-based access-control model. Because the static policy rules of our model are generated according to legal sequences, the false-positive rate is low.
Accuracy test. The correct rate refers to the proportion of abnormal behaviors that are regarded as abnormal in the test results relative to the number of abnormal requests. We compare the accuracies of the training sets that were generated by the three algorithms, and the experimental results are presented in Figure 10.

According to the experimental results in Figure 10, the correct rate of the training set that was produced by DSAAC is higher than those by clospan and prefix. Thus, the training set that was generated by this algorithm is relatively accurate and is suitable for the identification of user behaviors. Next, we evaluate the correctness of this algorithm after the introduction of the semantic constraint (DSAAC with semantic constraint) and the policy-based access-control model. The experimental results are presented in Figure 11.

According to the experimental data in Figure 11, when using DSAAC algorithm in combination with semantics, the accuracy rate reaches approximately 93%, while the accuracy rate of the traditional policy-based access-control model is lower. Based on the experimental results, it is concluded that our model realizes a high accuracy and a low false-positive rate for dynamic permission control under mass services.

Conclusion
In this article, we studied the privacy protection issues of joint analysis across multi-data centers from the perspective of access control. In traditional service-based authorization models, static rules are used, which render the authorization process inflexible and unable to support the authorization requirements in multiple data center scenarios. According to the characteristics of scientific computing workflows across data centers, we incorporated the method of semantic verification and sequence anomaly detection into the access-control process and provided dynamic authorization management in the TBAC model and protected data privacy and security in multiple data center environments.

Declaration of conflicting interests
The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding
The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This research was funded by the National Development and Reform Commission 2018 Digital Economy Pilot Project (Grant No. 2018FGW005), the Key Research Plan for State Commission of Science Technology of China (Grant No. 2018YFC0807501), the Foundation of Science & Technology Department of Sichuan province (Grant Nos 2018HH00075, 2018JY0605, 2018JY0073, 2017KP035, and 2017JZ0031).

ORCID iD
Guoming Lu https://orcid.org/0000-0001-7477-5800

References
1. Kraska T, Pang G, Franklin MJ, et al. MDCC: multi-data center consistency. In: EuroSys’13: proceedings of the 8th ACM European conference on computer systems, Prague, 15–17 April 2013, pp.113–126. New York: ACM.
2. Tene O, Evans K, Gencarelli B, et al. GDPR at year one: enter the designers and engineers. IEEE Sec Priv 2019; 17(6): 7–9.
3. Desmedt Y and Shaghaghi A. Function-based access control (FBAC). In: Proceedings of the 2016 international workshop on managing insider security threats—MIST 16, Vienna, 28 October 2016. New York: ACM.
4. Crampton J and Huth M. Towards an access-control framework for countering insider threats. In: Probst CW, Hunker J, Gollmann D, et al. (eds) Insider threats in...
cyber security advances in information security. Cham: Springer, 2010, pp.173–195.
5. Lin B, Guo W, Xiong N, et al. A pretreatment workflow scheduling approach for big data applications in multi-cloud environments. *IEEE T Netw Serv Man* 2016; 13(3): 581–594.
6. Sendi AS, Dagenais M, Jabbarifar M, et al. Real time intrusion prediction based on optimized alerts with hidden Markov model. *J Netw* 2012; 7(2): 311–321.
7. He Z, Cai Z and Yu J. Latent-data privacy preserving with customized data utility for social network data. *IEEE T Veh Tech* 2018; 67(1): 665–673.
8. Boughrous M and Bakkali HE. A comparative study on access control models and security requirements in workflow systems. In: Abraham A, Haqiq A, Muda A, et al. (eds) *Innovations in bio-inspired computing and applications*. *IBICA* 2017. Advances in intelligent systems and computing, vol. 735. Cham: Springer, 2017, pp.361–373.
9. Cai Z and Zheng X. A private and efficient mechanism for data uploading in smart cyber-physical systems. *IEEE T Netw Sci Eng*. Epub ahead of print 24 April 2018. DOI: 10.1109/TNSE.2018.2830307.
10. Thomas RK and Sandhu RS. Task-based authorization controls (TBAC): a family of models for active and enterprise-oriented authorization management. In: Lin TY and Qian S. (eds) *Database security XI*. Boston, MA: Springer, 1998, pp.166–181.
11. Strembeck M and Mendling J. Modeling process-related RBAC models with extended UML activity models. *Inf Softw Technol* 2011; 53(5): 456–483.
12. Cai Z and He Z. Trading private range counting over big IoT data. In: *The 39th IEEE international conference on distributed computing systems (ICDCS2019)*, Dallas, TX, 7–9 July, New York: IEEE.
13. Cai Z, He Z, Guan X, et al. Collective data-sanitization for preventing sensitive information inference attacks in social networks. *IEEE T Depend Sec Comput* 2018; 15(4): 577–590.
14. Xiong D and Chen L. T-RBAC based multi-domain access control method in cloud. *Netw Prot Algor* 2017; 8(4): 29.
15. An J, Yao SZ and Zhang L. Research on delegation authorization model based on TRBAC and attribute. *Adv Mater Res* 2013; 601: 307–311.
16. Uddin M, Islam S and Al-Nemrat A. A dynamic access control model using authorising workflow and task-role-based access control. *IEEE Access* 2019; 7: 166676–166689.
17. Zheng X and Cai Z. Privacy-preserved data sharing towards multiple parties in industrial IoTs. *IEEE J Select Areas Commun*. Epub ahead of print 16 March 2020. DOI: 10.1109/JSAC.2020.2980802.
18. Diep NN, Lee S, Lee YK, et al. Contextual risk-based access control. In: *International conference on security & management, SAM 2007*, Las Vegas, NV, 25–28 June 2007, pp.406–412. Las Vegas: CSREA.
19. Bouchami A, Goettelmann E, Perrin O, et al. Enhancing access control with risk metrics for collaboration on social cloud-platforms. In: *IEEE trustcom/BigDataSE/ISPA*, Helsinki, 20–22 August 2015, pp.864–871. New York: IEEE.
20. Lakshmi H, Namitha S, Seemannthi, et al. Risk based access control in cloud computing. In: *International conference on green computing and internet of things (IGC-CIoT)*, Noida, India, 8–10 October 2015, pp.1502–1505. New York: IEEE.
21. Cai Z, Zheng X and Yu J. A differential-private framework for urban traffic flows estimation via taxi companies. *IEEE T Ind Infor* 2019; 15: 6492–6499.
22. Fall D, Okuda T, Kadobayashi Y, et al. Risk adaptive authorization mechanism (RadAM) for cloud computing. *J Inf Process* 2016; 24(2): 371–380.
23. Shaikh RA, Adi K and Logrippo L. Dynamic risk-based decision methods for access control systems. *Comput Secur* 2012; 31(4): 447–464.
24. Ni Q, Bertino E and Lobo J. Risk-based access control systems built on fuzzy inferences. In: *Proceedings of the 5th ACM symposium on information, computer and communications security*, Beijing, China, 13–16 April 2010, pp.250–260. New York: ACM.
25. Xie Y and Yu SZ. A large-scale hidden semi-Markov model for anomaly detection on user browsing behaviors. *IEEE/ACM T Netw* 2009; 17(1): 54–65.
26. Zhou Y, Wang Y and Ma X. A user behavior anomaly detection approach based on sequence mining over data streams. In: *International conference on parallel and distributed computing, applications and technologies*, Guangzhou, China, 16–18 December 2016, pp.376–381. New York: IEEE.