**Research Article**

**RTBAC: A Risk-Aware Topic-Based Access Control Model for Text Data with Paragraph-Level Authorization**

Ke Ma\(^1\) and Geng Yang\(^{1,2}\)

\(^1\)School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing, China
\(^2\)Jiangsu Key Laboratory of Big Data Security and Intelligent Processing, Nanjing, China

Correspondence should be addressed to Geng Yang; yangg@njupt.edu.cn

Received 11 December 2021; Revised 27 February 2022; Accepted 11 April 2022; Published 5 May 2022

Academic Editor: Chien Ming Chen

Copyright © 2022 Ke Ma and Geng Yang. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Granting users precise access rights is one of the purposes of access control technologies. With the increasing requirements of fine-grained authorization, too strict or too loose access control models may cause many problems. In this paper, aiming at insufficient authorizations in text databases, we propose a risk-aware topic-based access control (RTBAC) model, which uses topics to represent the content relationships between users and data. The RTBAC model also uses risk technologies to grant users corresponding access rights based on their historical behaviours and their access requests. The RTBAC model is a fine-grained access control model, and the authorization of RTBAC can reach the paragraph level. Experimental results show that RTBAC is an efficient access control model and the access control granularity of the RTBAC model is more than 3 times that of the existing content-based access control models.

1. Introduction

In the information age, data has become an asset with economic importance. To use data more efficiently, sharing data safely becomes an important requirement. As a key technology for ensuring safe data sharing, access control plays an important role in the big data era. Traditional access control models include discretionary access control (DAC)\(^1\), mandatory access control (MAC)\(^2\), and role-based access control (RBAC)\(^3\). However, in the era of big data, due to the inability to effectively obtain the required patterns, these traditional access control techniques may not be able to perform authorization well. Attribute-based access control (ABAC)\(^4\) can obtain patterns according to the attributes of the access subjects and objects. Therefore, compared with traditional access control technology, ABAC can better meet the authorization requirements in the big data environment. However, unstructured data (most are text data) have become a vital part of data, and it is difficult to extract internal attributes among the data. Therefore, for unstructured data, ABAC does not have effective authorizations. To further motivate our research, we consider the following examples:

**Example 1.** In a law enforcement agency, a supervisor assigns a case to agent Alice in Department A for investigation. Naturally, Alice also needs to access related or similar cases. As an agent in Department A, Alice has the authority to access related files in Department A’s database. However, Alice cannot access related cases that belong to other departments. When Alice needs to access the data that belong to other departments, it is possible that: (i) because Alice does not have the corresponding department attributes, Alice cannot obtain any access rights to the corresponding file (insufficient authorization); (ii) for Alice’s investigation to run smoothly, Alice can obtain the file directly (over authorization).

Each of these situations has drawbacks. In Situation (i), the important file is inaccessible to Alice, which can affect Alice’s investigation procedure, and even prevent Alice’s investigation from continuing. In Situation (ii), Alice can easily access files that exceed her security level, which could...
cause infinite risk, and confidential documents may also be leaked. We hope that Alice can only access the data related to her work, and Alice’s ability to access the sensitive data should be reduced as much as possible. If accessible data and sensitive data are in the same file, they should be divided into different parts, and the accessible part should be given to Alice for her access. To explain this requirement visually, see example 2.

Example 2. We assume that a person’s age, height, weight, and address are their private information, which should not be obtained by others as sensitive information. The following are four paragraphs in a text.

“Bob is a student at New York University. XXXXX. He is 20 years old, 180 cm tall, and weighs 65 kg. XXXX. He lives in the XX community of Brooklyn, with a zip code of 11200. XXXX. His hobbies are swimming, fitness, and travel. XXXXX.”

Obviously, the second and third sentences are sensitive personal data. If Alice applies for accessing this passage, all sensitive data that are not suitable for her access should be deleted. Then, the text she can access is as follows.

“Bob is a student at New York University. XXXXX. His hobbies are swimming, fitness, and travel. XXXXX.”

In the above example, it seems that the problem can be solved by fine-grained attribute-based access control algorithms, as long as the sentences can be represented by different attributes. In applications for structured data, ABAC can perform very well because attributes can be well identified. However, for applications based on unstructured text data, it is almost impossible to extract attributes, and it is difficult to identify the internal relationships between the paragraphs or files. It is also difficult for ABAC to recognize which text data have the same attributes. ABAC is not applicable in the above example.

Content-based access control (CBAC) [5] is a fine-grained access control model for content-centric databases that uses the internal relationship between different text files to grant users file-level access rights. However, to complete the requirements described in the above examples, CBAC has two obvious drawbacks: (i) CBAC is a file-level access control model that cannot divide the paragraphs of one file into “accessible” and “inaccessible” parts and grant access rights to the user; (ii) CBAC aims at “safe” situations, where all text files are accessible to users, while in example 1, the files belonging to other departments are theoretically inaccessible to Alice. Thus, CBAC cannot be used to solve the problem.

The key issue in the above example is to find which part of the data can be accessed in a text file. Natural language processing (NLP) [6] algorithms can help solve the problem. The latent Dirichlet allocation (LDA) [7] is a document-topic generation model, also known as a three-layer Bayesian probability model, which has a three-layer structure of words, topics, and documents. In the LDA model, each word of a document selects a certain topic with a certain probability; at the same time, the topic also selects the word with a certain probability. Each document represents a probability distribution composed of some topics, and each theme represents a probability distribution composed of some words. “Documents to topics” follows a polynomial distribution, and so does “topics to words.”

In this paper, we use the LDA model to represent the paragraphs in the text file as topics. Then, we use the generated topics to propose a risk-aware topic-based access control model, RTBAC. The model has fine-grained access authorization, and can detect user access risks in time, as well as adjust user access rights. Experimental results show that the RTBAC model has better access control performance than existing content-based access control models.

Our contributions in this paper are as follows:

1. We propose a fine-grained topic-based access control model. We use topics to produce paragraph-level authorization. The topics are extracted from files and paragraphs according to the natural language processing model, the latent Dirichlet allocation (LDA). Users will be assigned fine-grained access rights according to their corresponding topics.

2. We propose a risk-aware access control model. We use information entropy and sliding window technology to monitor users’ access behaviour. If a user is marked as a “curious user,” his access rights will be considerably reduced.

3. We add RTBAC enforcements on different datasets. We develop two RTBAC enforcements on different datasets (20-newsgroups dataset and ICD-10 dataset). In both cases, the experimental results show that the RTBAC model can provide finer-grained access authorizations than existing content-based access control models. Meanwhile, in both cases, the RTBAC model can easily distinguish curious users from normal users.

The remainder of this paper is organized as follows: Section 2 presents related works. Section 3 describes the overall framework of the RTBAC model. Section 4 and Section 5 present the specific RTBAC model, including paragraph-level authorization and risk management. Section 6 discusses the performances of RTBAC. Section 7 uses the 20-newsgroups dataset and the ICD-10 dataset to conduct comparative experiments. The paper is concluded in Section 8.

2. Related Works

2.1. Access Controls with Semantics and Contents.

Comprehensive data protection requires related mechanisms to enforce access control policies based on data contents. In recent years, many access control policies that were related to “semantics” have been proposed. Based on XACML with the application of semantic interoperation, Zhao et al. [8] realized the semantic interoperation between the attributes of the service requester and the service provider, which increased the security of semantic web services. Elahi et al. [9] and Liu et al. [10] performed semantic analysis on users’ access requests and developed access control rules. Wang et al. considered semantic spatial trajectories in the
Security and Communication Networks

access control of social networks and presented a secure semantic model for spatial trajectories in social networks [11]. Ma et al. combined the concept of content into an attribute access control model and proposed a content-driven attribute-based access control model CABAC [12]. Chen et al. proposed semantic-aware access control that extended the RBAC with semantic specifications for grid applications [13]. To enable fine-grained access control definition and enforcement in cross-company data exchanges, Fabian et al. designed an architecture for controlling access to semantic repositories and presented an infrastructure for controlled sharing of semantic data between cooperating business partners [14]. For applications with the Internet of Things (IoT), although the raw IoT data have no deeper meaning, when a semantic abstraction is added, it becomes suitable for reasoning, fusing, and actuation. Stojanov et al. evaluated the Linked Data Authorization platform for semantic data access control in the IoT context, which could provide contextual protections for semantic data [15]. Aiming at the data and secure communication in wind power systems, Nagarajan et al. proposed a generic role-based access control model [16]. Aiming at Android systems, to equip the user with the ability to effectively manage Android permissions, Talegaon et al. introduced three models for the administration of RBAC in Android, which supported the principle of least privilege to reduce unwanted permission exposure [17]. Aiming at the initial development of access control policies (ACPs), Narouei et al. proposed a new framework toward extracting ACPs from unrestricted natural language documents using semantic role labelling, which could correctly identify ACP elements with a high F1 score [18]. Based on health information systems, Lima et al. presented a security mechanism that allowed data extraction from a regional health information system using semantics. In the mechanism, data were tagged semantically and mapped individually to several access levels [19]. Aiming at the data in the Mobile Healthcare system (mHealth), Li et al. proposed a fine-grained policy-hiding and traceable access control scheme HTAC, which achieved a large attribute universe, access policy privacy, and white-box traceability [20].

Aiming at security and privacy solutions in big data, El Haourani et al. proposed a knowledge-based access control model (KBAC). The model added a semantic access control layer to the original access controls (such as “RBAC” roles and “ABAC” attributes) and provided thinner access control tailored to big data [21]. Based on BigData platforms, content-based access control approaches targeting MapReduce systems and HDFS resources have also been proposed. Vigiles [22] and GuardMR [23] were two fine-grained access control models for MapReduce systems that used access control filters (ACFs) to filter unauthorized pairs and clean up sensitive information in the data content. Aiming at cloud data in the fifth-generation mobile communication (5G) environment, Ma et al. proposed a novel secure data deletion and verification scheme based on CP-ABE to achieve fine-grained secure data deletion and deletion verification [24].

More relevant to our work, CBAC [5] is a fine-grained content-based access control policy (accurate to the file level for each user) for content-centric databases, which also uses content relevance to grant authority to users. However, compared to our work, the ways in which content similarity is used differs. CBAC aims at “safe” scenarios, and all the files in the database are accessible to users, while our RTBAC considered constraints, and some specific data are inaccessible to some users. Aiming at “unsafe” scenarios, RCBAC [25] is proposed to combine risk awareness to the content-based access control model, which uses risk quota to grant access rights to users. At the same time, RCBAC can distinguish curious users from normal users. However, RCBAC is a content-based access control model with file-level authorization, while the authorization of RTBAC can reach paragraph level.

2.2. Risk-Based Access Controls. Recently, risk-based access control policies have attracted increasing attention. M C. JASON Program Office first introduced the concepts of risk quantification and risk quotas into access control (the risk can be used to determine whether users have authority to access data or not) in 2004. Subsequently, many risk-based access control models have been proposed. Most of them are access control models that combine the concept of risk with traditional access control models.

To integrate risk into traditional access control models, based on role-based access control (RBAC), Celike and others combined the access request and the access history to calculate the risk [26]. Nissanke et al. used the risk level to define a partially ordered set and reconstitution role levels for RBAC [27]. Bijon et al. analysed the differences between the traditional method and the risk quantification method of RBAC and proposed a risk-aware RBAC model [28]. For research on multilevel security, Cheng et al. proposed a fuzzy MLS model that defines acceptable risk sections and divided these sections into several risk bands [29]. Based on the user’s risk band, it grants the user corresponding authority. Aiming at mandatory access control (MAC), Lu et al. used the similarity between text records to determine whether their tags were correct. If the user accesses the falsely tagged records, he would cause different risks [30].

The above methods integrated risk into various traditional access control methods to expand the access capacity for users and users could access resources in exceptional cases. However, these methods could not adjust the risk of users’ access behaviours over time, which resulted in poor self-adaptation performance and the necessity for substantial manual management.

To address the above problem, Chen et al. designed a dynamic risk-based access control model by adding risk engines, risk authorization services, and risk policies to XACML [31]. Dos Santos et al. designed a risk matrix for managing users’ access to the cloud and proposed a corresponding risk-based access control model [32]. To adaptively compute trust and risk values in dynamic occasions, Shaikh et al. proposed two dynamic risk-based decision methods for access control systems, which could
not only allow broader authorities under certain controlled conditions (e.g., if users showed a positive record of use toward the resources they acquired in the past, they can be allowed broader authorities) but also restrict legitimate access of bad-authorized users [33]. Mcgraw proposed a self-adaptive access control mechanism (RAdAC), which compares the degree of demand and the risk, before deciding whether to grant the user corresponding authority [34]. However, RAdAC provided only a risk self-adaptive mechanism, not a specific risk quantification method. Based on the medical treatment system, Wang et al. proposed a risk-based access control model that was based on the concept of RAdAC [35]. In this model, doctors generated high risk if they accessed data that were of little relevance to their work. Hui et al. improved the access control model in [35], which applied the EM algorithm and the information entropy technique to quantify the risk [36]. Using the quantified risk, the model could detect and control the over authorization and exceptional access of patient data. Taking into consideration the relationships between data and access behaviours, Zhang et al. proposed a dynamic risk-adaptive access control model for health IT systems. The model trained topic models to portray individual and group-level access behaviours, which could quantify the risk for each user over a certain period of time [37]. Based on health platforms, Nakamura et al. proposed a risk-based access control modelling that integrates risk assessment elements in the attribute-based model to organize the identification, authentication, and authorization rules, which can be used to help produce more efficient and effective decisions in terms of granting access to specific objects [38]. To provide the required flexibility to access system resources and works well in unexpected conditions and situations of the IoT system, Atlam et al. proposed a risk estimation technique which integrates the fuzzy inference system with expert judgment to assess security risks of access control operations in the IoT system [39]. To provide an intelligent environment for users to conduct their day-to-day activities in cyber-physical spaces, Cao et al. proposed a topology-aware access control (TAAC) model and a risk assessment approach to evaluate the user behaviour and ensure that the suspicious behaviours executed by authorized users can be handled correctly [40]. Aiming at insider threats, Liu et al. used risk budget and magnitude price to propose a budget-based access control model [41].

Other risk-based access control models have also been proposed in recent years. Aluvalu et al. proposed a dynamic attribute-based risk-aware access control model, which could be hybridized with static access control models with various attribute encryptions, such as KP-ABE, CP-ABE, and HASBE [42]. Based on an extension of XACML, Dos Santos et al. proposed a framework for enforcing risk-based policies [43]. Aiming at grid virtual organizations, Nogoorani et al. proposed a TIRIAC framework, which was a trust-driven risk-aware access control framework that used obligations to seamlessly monitor users and mitigate risks [44].

Of the above risk-based access control models, only [30, 37] combine data content with risk. However, the method of using content was different from our RTBAC model, and our RTBAC model has finer-grained authorization than these access control models.

2.3. Topic Models. The topic model is a statistical model for clustering latent semantic structures of documents in an unsupervised learning manner [45]. Topic models are mainly used for semantic analysis and text mining in natural language processing (NLP), such as collecting, classifying, and reducing the dimensionality of the text by subject. It is also used for research on biological information [46]. The traditional topic model, probabilistic latent semantic analysis (PLSA), was proposed by Hofmann in 1999 [47]. In 2000, Papadimitriou et al. proposed another traditional topic model, latent semantic indexing (LSI) [48]. In 2003, Blei et al. proposed latent Dirichlet allocation (LDA) [7]. The LDA is one of the most famous topic models, which has been widely used and has been derived from many improved versions, such as the latent Dirichlet allocation with wordnet model LDAWN [48]. LDWAN is a multi-labelled classification model based on the tags of documents labelled LDA [49], and other extended models of LDA [50–52]. The concept of topic models has also been combined with access control models in recent years. For health care information systems, Zhang et al. use topic models to propose a privacy-aware risk-adaptive access control model, where the topics were used to determine doctors’ behaviours, and to distinguish malicious doctors from honest doctors [37]. In our topic-based access control models, data in the data space are mapped to topic features, and we use the topics to grant users fine-grained permissions.

3. Overall Framework of the RTBAC Model

To solve the insufficient authorization and over authorization problems in access control systems, we propose a risk-aware topic-based access control RTBAC model. Figure 1 shows the overall framework of the RTBAC model.

In the framework, we could see that there are four main layers in the RTBAC model: the input layer, the topic layer, the risk layer, and the output layer. The functions of the four layers include the following:

1. The function of the input layer enables the user to send an access request to the specified data with the access category (the access category is used to determine whether the user needs other data with semantic or content relevance).
2. The function of the topic layer uses the LDA algorithm in this layer to determine the topics of each document and their paragraphs. At the same time, the basic attributes of the user will also be associated with the topics. The content relevance of different files and the user’s access request is computed based on these topics.
3. The function of the risk layer enables the content risk of the user’s access request to be computed in this layer. The risk level of the request is computed based on the content risk and the user’s historical risk. If
the access request is allowed by the RTBAC model, the user consumes his risk quota to obtain the corresponding access rights.

(4) The function of the output layer enables the RTBAC model to assign different levels of access rights (entire access rights of target data and related data, entire access rights of target data, partial access rights of target data, and no access right) to users based on the results of the risk layer.

4. Access Request and Topic System

In this section, we explain how users request accessing target data in the RTBAC model. Then, we introduce the way the topic system works based on access requests.

4.1. Sending Access Requests in RTBAC. In the RTBAC model, an access request indicates “who” wants to access “what” by “which mode.” The access request \( ar \) can be specified as a 4-tuple:

\[
\text{ar} = \left[ \text{subject}, \text{object}, \text{base set}, \text{access mode} \right].
\]

where the \text{subject} denotes the user who wants to access some text data, and the \text{data object} is the target data that the user requested for access. In the RTBAC model, the \text{object} is a content-rich document or a set of text data from the same property, which has enough semantic information, such as a teaching reference book, an academic paper, or an operation guide.

The \text{base set} is a unique initial dataset assigned to each user. The RTBAC model will add typical documents to the user’s base set according to the user’s basic attributes (such as occupation, access location, and access time). These files could be manually selected by the data administrator or selected based on clustering algorithms. Furthermore, users can also request adding base set items to their base sets and ask the administrator to approve the request. The \text{base set} is used as the basis for computing the relationship between users and data contents.

The \text{access mode} denotes the type of access request, which has two values. \text{Access mode} = 0 means that the user only requests the access right of the target data; \text{Access mode} = 1 means that the user not only requests the access right of the target data, but also requests access to other data related to the content of the target data.

Algorithm 1 shows the overall execution process from sending the user’s access request to the authorization.

Remark 1. \( DP_{tmp} \) is a collection of accessible documents and their corresponding accessible paragraphs. The items in \( DP_{tmp} \) are \(< \text{document_key}, \text{paragraph_nums} > \) pair, which records the key of document \( d \) and the number of accessible paragraphs in \( d \).

In Algorithm 1, \( r(u, d_{tar}, bs(u), am) \) is the paragraph-level access rights granted to user \( u \), which is a collection of paragraphs accessible to \( u \). Steps 1 to 9 use a topic system to compute content similarities and generate temporarily accessible document-paragraph collection \( DP_{tmp} \), which will be illustrated in Section 4.2. Steps 10 to 13 use a risk management system to grant corresponding access rights to \( u \) based on the \( DP_{tmp} \) and content similarities, which will be explained in Section 5.

4.2. Topic System in the RTBAC Model. Equations should be provided in a text format, rather than as an image. Microsoft Word’s equation tool is acceptable. Equations should be numbered consecutively, in round brackets, on the right-hand side of the page. They should be referred to as (1), etc. in the main text.

4.2.1. Extracting Topics. In the RTBAC model, the access control decision for each user against each file is totally content-driven, and topics are used to determine users’
In LDA, we consider that each word in a paragraph is obtained through a process of “selecting a certain topic with a certain probability, and selecting a certain word from this topic with another certain probability.” Figure 2 shows the relationships between paragraphs, topics, and words. A topic is the latent semantic structure of a text, which is often used for document modelling and information retrieval. In the RTBAC model, topics can be modelled as an infinite mixture over an underlying set of topic probabilities, a set of keywords, or even realistic topics. In one RTBAC system, the complete topic set can be represented as \( T = \{t_1, t_2, \ldots, t_m\} \), where each element \( t_i \) is a specific and independent topic and \( m \) is the number of all the topics.

The set of all the paragraphs in the database is represented as \( \text{PARA} \). Each paragraph \( \text{para} \) in \( \text{PARA} \) is regarded as a word sequence \( [w_1, w_2, \ldots, w_n] \), where \( n \) is the number of words and \( w_i \) represents the \( i \)-th word. All the different words involved in \( \text{PARA} \) constitute a large set \( \text{VOC} \). We use the LDA to train two result matrices \( \theta \) and \( \phi \). For each \( \text{para} \in \text{PARA}, \theta_{\text{para}} = [p_{t_1}, p_{t_2}, \ldots, p_{t_k}] \) represents the probability of \( p \) corresponding to \( T \), where \( p_{t_i} = n_{t_i}/n \), and \( n_i \) indicates the number of words corresponding to the \( i \)-th topic in \( \text{para} \).

For each \( t \in T \), \( \varphi_t = [p_{w_1}, p_{w_2}, \ldots, p_{w_N}] \) represents the probability of \( t \) corresponding to the \( \text{VOC} \), where \( p_{w_i} = N_{w_i}/N \), where \( N_{w_i} \) represents the number of the \( i \)-th word in the \( \text{VOC} \) corresponding to \( t \), and \( N \) represents the total number of words corresponding to \( t \).

When the LDA algorithm starts, first, we randomly assign values to \( \theta_{\text{para}} \) and \( \varphi_t \) (for all \( \text{para} \) and \( t \)). Then, we iterate continuously, and the final result of convergence (\( \theta \) and \( \varphi \)) is the output of the LDA. Ultimately, they are used for computing the topics of different entities.

LDA is the most typical topic generation model. In fact, as long as the topics can be effectively generated for the paragraph, other topic generation algorithms can also be applied to our model.
The topic of document d can be represented as \( d.topic = d.primTopic \cup d.normalTopic \).

User’s Basic Topics. According to the accessible documents in the user’s basic set, user u’s basic topics are generated by the topics of the files in \( bs(u) \). The basic topic of user u is represented as \( u.topic \).

User’s Constraint Topics. Constraints in the RTBAC model are the restrictions on topics that the subject can access according to its own attributes. For example, in an MLS (multilevel security) system, topics about “asthma” only appear in the files belonging to the Department of Respiratory, and a user belonging to the paediatrics department should not access such topics. In an RBAC system, the topics accessible to each role are fixed to ensure that some topics are not accessible for a specific role, and a user belonging to the role should also not access the files with these topics. The set of topics inaccessible to user u can be represented as \( u.conTopic \).

Topics of Newly Added Files. In some cases, a new file is added to the system. According to the word distribution of each paragraph of the file and the \( \Phi \) matrix computed by the LDA, the topic distributions corresponding to each paragraph of the file can be calculated. The topics of the entire file and each paragraph are generated by the topic distributions.

**Topic Update.** When the number of newly added files exceeds 10% of the original number of files, the LDA is reused to compute all the topics.

**4.2.3. Content Similarity Computing.** The RTBAC model uses the topic system to compute the intermediate result of authorization, i.e., temporarily accessible document-paragraph collection \( DP_{tmp} \). The temporarily accessible document-paragraph collection \( DP_{tmp} \) is extended to be a topic-based access control function.

If the primary topic of document d intersects with user u’s constraint topics, it means that user u conflicts with document d. Their content similarity \( SIM(u, d) \) should be 0. In other cases, the content similarity should be the proportion of topics owned by u to topics of d, which is calculated as follows:

\[
SIM(u, d) = \begin{cases} 0, & u.conTopic \cap d.primTopic \neq \emptyset, \\ \frac{|u.topic \cap d.topic|}{d.topic}, & \text{others.} \end{cases}
\]

The content similarity between different documents \( d_i \) and \( d_j \) is as follows:

\[
SIM(d_i, d_j) = \frac{|d_i.topic \cap d_j.topic|}{|d_i.topic \cup d_j.topic|}.
\]

If \( SIM(u, d) \) reaches a threshold \( T_{SIM} \) (\( T_{SIM} \) is the threshold to judge whether u has enough similarity with d, we set the default \( T_{SIM} \) as 0.25), u will have a chance to access the paragraphs in d. The accessible paragraph of d is as follows:

\[
\text{para}(u, d) = \{ p_i | d_i.topic \cap u.topic \neq \emptyset \land d_i.topic \cap u.conTopic = \emptyset \}.
\]

If user u only wants to access the target data \( d_{tar} \), \( Para_{tar} \) should be the number of accessible paragraphs of \( d_{tar} \) (\( Para(u, d_{tar}) \)).

If u also wants to access other related data, the paragraphs of other related documents should also be added to \( Para_{tmp} \). First, we need to compute the collection of the related documents \( D_{SIM} \) in document database D as follows:

\[
D_{SIM} = \{ d_i | SIM(d_i, d_{tar}) \geq T_{DOCSIM} \}.
\]

where \( T_{DOCSIM} \) is the threshold to determine whether two documents have enough similarities. For access request \( ar = [u, t_{tar}, am, bs(u)] \), the temporary document collection \( D_{tmp} \) is as follows:

\[
D_{tmp} = \begin{cases} \emptyset, & \text{if } SIM(u, d_{tar}) < T_{DOCSIM}, \\ d_{tar}, & \text{else if } am = 0, \\ d_{tar} \cup D_{SIM}, & \text{others}, \end{cases}
\]

and the temporary document-paragraph collection \( DP_{tmp} \) for user u is as follows:

\[
DP_{tmp} = \{ <d_i, para(u, d_i)> | d_i \in D_{tmp} \}.
\]

**5. Risk Management and Authorization**

In this section, we introduce how to compute access risks based on content similarities and sliding windows. Then, we use the historical risks to determine whether the user is a “curious user” or a “normal user.” Finally, we will introduce the risk quota and the way to consume the risk quota to acquire corresponding access rights.

**5.1. Computing Access Risks.** The RTBAC model is a completely content-driven access control model, and the computing of access risk is also based on data contents. We define the access risk between user u and target document d as \( \text{Risk}_u(u, d) \in [0, 1] \). \( \text{Risk}_u(u, d) = 0 \) proves that the access...
has no risk, and $\text{Risk}_u(u,d) = 1$ shows that access is extremely risky. Algorithm 2 shows the process of computing an access risk.

Remark 2. No matter what the access mode $am$ is, the access risk of user $u$ requesting accessing target document $d$ is only based on the content of $u$ and $d$. However, the value of $am$ affects user $u$’s subsequent consumption of risk quota and acquiring access rights.

To store and describe the users’ access histories with limited storage space, the RTBAC model defines a sliding window, which is denoted as $SW(u)$ for each user $u$. Each sliding window is divided into three sub-windows of various granularities (near, mid, and far) for storing the access messages from various periods. Furthermore, three sub-sliding windows are used in $SW(u): SW_S(u)$, $SW_M(u)$, and $SW_L(u)$. $SW_S(u)$ stores user $u$’s access history from the nearest period (short term), $SW_M(u)$ stores user $u$’s access history from the nearest period and the middle period (mid term), and $SW_L(u)$ stores the histories from all the periods (long term). The RTBAC model computes the history risks in each sub-sliding window for each user. Figure 4 shows the sliding windows in the RTBAC model.

For each sub-sliding window for user $u$ ($SW_S(u)$, $SW_M(u)$, and $SW_L(u)$), $u$’s historical risk $\text{Risk}_{SW}(u)$ can be computed according to the access risk of all requests in $SW$ as follows:

\[
\text{Risk}_{SW}(u) = \frac{\sum_{d \in D_{SW}} \text{Risk}_{u}(u,d)}{|D_{SW}|},
\]

where $D_{SW}$ indicates all files that user $u$ requests for access in sliding window $SW$, and $|D_{SW}|$ is the number of files in $D_{SW}$.

To simplify the expression, let $R_S(u)$, $R_M(u)$, and $R_L(u)$ be $\text{Risk}_{SW}(u)$, $\text{Risk}_{SW_{-u}}(u)$, and $\text{Risk}_{SW}(u)$, respectively. We also set a parameter $T_{Risk}$ which is used to determine whether the access history has sufficiently high risk ($T_{Risk}$ can be changed by the administrator according to the application). Then, we compute $\text{Risk}_u(u)$ (the history risk of $u$) according to the following rules:

1. If $T_{Risk} < \max\{R_S(u), R_M(u), R_L(u)\}$, $\text{Risk}_u(u)$ should be $\max\{R_S(u), R_M(u), R_L(u)\}$. This means that the user’s access history has sufficiently high risk.

2. If $R_M(u) = \{R_S(u), R_M(u), R_L(u)\}$, $\text{Risk}_u(u)$ should be $\max\{R_S(u), R_M(u), R_L(u)\}$. The user may access many relevant files in the middle period to conceal his curious access behaviour.

3. If $R_L(u) < R_M(u) < R_S(u)$, $\text{Risk}_u(u)$ should be $(R_S(u) + R_M(u) + R_L(u))/3$. Because user access is becoming safer, it is also possible for the curious user to start accessing safe files.

4. If $R_L(u) < R_M(u) < R_S(u)$ or $R_S(u) = \min\{R_S(u), R_M(u), R_L(u)\}$, $\text{Risk}_u(u)$ should be $R_S(u)$. The user has totally different access behaviour in the short term.

5. If $R_L(u) < R_S(u) < R_M(u)$, $\text{Risk}_u(u)$ should be $(R_S(u) + R_L(u))/2$. This means that the user has accessed documents of high risk in the short and middle terms and that access in the middle period has the highest risk. Therefore, the access history risk should reflect the risks in these two periods.

6. For other situations, we set $\text{Risk}_u(u)$ as $(R_S(u) + R_M(u) + R_L(u))/3$ to reflect the average history risk of the user.

Then, we use the historical risk to determine the category of users. We divide the users into two categories: normal users and curious users.

Definition 1. Normal user. Normal users will always request to access data that has content relevance to their work under normal circumstances. However, in rare cases, they might try to access unrelated data. Normal users never or seldom try to access the data unrelated to them. We believe that the normal user’s frequency of accessing unrelated data should be less than 10%.

Definition 2. Curious user. Curious users will access data that are related to their duties to complete their work, while also accessing unrelated data often. Compared with normal users, curious users will access unrelated data with a higher frequency, which could be 20% or higher.
If a user’s access history involves conditions (1), (2), or (5) (suspected “unsafe” access), the system adds one “suspected risk” to his access history. Users’ access requests should be rejected under condition (1), and the users should pay a heavy price to obtain the authorization in conditions (2) and (5). If the ratio of one user’s “suspected risk” is sufficiently large, the user could be a curious user. Users’ high-risk access histories will be evaluated by our RTBAC model, and access requests from them will attract the administrator’s attention.

5.2. RTBAC Authorization. In this section, we will introduce how users consume risk quotas to acquire access rights in the RTBAC model.

The risk quota specifies the tolerance for the risk that is caused by a user. The RTBAC model quantifies every access request as a risk value, and the users can consume their risk quota to gain access authority. If a user’s risk quota is less than 0, he or she can only access the data that are access risk Risk\(_k\)(\(u,d\)) = 0. Other requests from the users should be rejected unless their risk quota is increased.

For a specified user \(u\) and one accessing document \(d\), we set Risk\(_k\)(\(u,d\)) to represent the total access risk, where Risk\(_k\)(\(u,d\)) = \([0,1]\). If \(u\)'s ban Topic \(\cap d\)'.prim Topic \(\neq \emptyset\), this indicates that \(d\)'s primary topics are in conflict with \(u\)'s constraint topics, and Risk\(_k\)(\(u,d\)) should be 1. In other cases, Risk\(_k\)(\(u,d\)) is defined by the user’s access risk Risk\(_k\)(\(u,d\)) and his historical risk Risk\(_h\)(\(u,d\)) as follows:

\[
\text{Risk}(u,d) = a \times \text{Risk}_h(u,d) + \beta \times \text{Risk}_k(u,d). \tag{9}
\]

where \(a\) and \(\beta\) are the proportions of Risk\(_h\)(\(u,d\)) and Risk\(_k\)(\(u,d\)), respectively. \(a \in [0,1]\), \(\beta \in [0,1]\), and \(a + \beta = 1\). Normally, for an access request \([u,d_{tar},bs(u),am]\), if computing Risk\(_k\)(\(u,d_{tar}\)), we set the default \(\alpha = 0.5\) and \(\beta = 0.5\). When computing Risk\(_h\)(\(u,d\), \(d \in D_{sim}(am = 1)\)), we set the default \(\alpha = 0.7\) and \(\beta = 0.3\) such that it focuses on the content risk of access without ignoring the historical risk of users.

Therefore, the total risk of one access request \([u,d_{tar},bs(u),am]\) (Risk\(_k\)(ar)) is as follows:

\[
\text{Risk}(ar) = \text{Risk}(u,d_{tar}) + \sum_{v \in D_{sim}} \text{Risk}(u,d_v). \tag{10}
\]

When a user sends an access request \(ar\), the RTBAC model records \(u\)'s remaining risk quota. After computing the total risk of \(ar\), the risk management module deducts it from \(u\)'s risk quota to grant the user access rights. Algorithm 3 shows the process of using risk quotas to grant access rights.

Remark 3. Since the risk quota is not negative, when the risk quota is 0 or very small, users can only access risk-free files and cannot access other content-related files.

To limit the accessibility of curious users, the RTBAC model automatically assigns risk quotas to each user according to their access histories.

Typically, the risk quota is distributed regularly and the distribution should satisfy these conditions: (i) normal users’ risk quotas are not exhausted before the next risk quota distribution, and (ii) curious users’ access capabilities are limited after several accesses.

We refer to the period between two adjoining distributions as one distribution cycle. Based on the definition in Section 5.1, the RTBAC model can compute user \(u\)'s ratio of suspected “unsafe” access (conditions (1), (2), and (5)) in the \(i\)-th distribution cycle, which is denoted as \(p_u(u)\). If \(p_u(u) > P\) (\(P\) is the threshold that has been set by the administrator for identifying curious users), the RTBAC model considers \(u\) as “suspected curious” in the \(i\)-th distribution cycle.

In the initial phase of the \(k\)-th distribution cycle, the RTBAC model (i) defines \(Q_k\) as an experience value that is set by the administrator or (ii) computes \(Q_k\) based on the risk values of users who are not “suspected curious users” in the \((k-1)\)-th distribution cycle. The mean of these risk values is denoted as \(\mu_{k-1}\), and the variance is denoted as \(\sigma_{k-1}\). We define \(Q_k = \mu_{k-1} + n \sigma_{k-1}\), where \(n\) can be dynamically set according to the scenario.

The RTBAC model retrieves the nearest \(m\) distribution cycles for each user \(u\) and obtains the number of “suspected curious” cycles, which is denoted as \(m'\). The final risk quota for \(u\) in the \(k\)-th distribution cycle \((Q_k(u))\) is as follows:

\[
\text{Algorithm 3: Using risk quota to grant access rights.}
\]

<table>
<thead>
<tr>
<th>Input: User u’s risk quota Q(u), access request [u, d_{tar}, bs(u), am] (ar)</th>
<th>Output: Access rights r(u, d_{tar}, bs(u), am)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) r(u, d_{tar}, bs(u), am) = ∅;</td>
<td>(2) if Q(u) &gt; Risk_h(u,d_{tar}) then</td>
</tr>
<tr>
<td>(3) if am = 0 or Q(u) &lt; Risk(ar) then</td>
<td></td>
</tr>
<tr>
<td>(4) Add (d_{tar}, Para(u,d_{tar})) into r(u, d_{tar}, bs(u), am);</td>
<td>(5) Q(u) = Q(u) - Risk_h(u,d_{tar});</td>
</tr>
<tr>
<td>(6) else</td>
<td>(7) add DP_{tmp} into r(u, d_{tar}, bs(u), am);</td>
</tr>
<tr>
<td>(8) Q(u) = Q(u) - Risk(ar);</td>
<td>(9) end</td>
</tr>
<tr>
<td>(10) end</td>
<td>(11) Return r(u, d_{tar}, bs(u), am);</td>
</tr>
</tbody>
</table>

\[
\text{Remark 3. Since the risk quota is not negative, when the risk quota is 0 or very small, users can only access risk-free files and cannot access other content-related files.}
\]

To limit the accessibility of curious users, the RTBAC model automatically assigns risk quotas to each user according to their access histories.

Typically, the risk quota is distributed regularly and the distribution should satisfy these conditions: (i) normal users’ risk quotas are not exhausted before the next risk quota distribution, and (ii) curious users’ access capabilities are limited after several accesses.

We refer to the period between two adjoining distributions as one distribution cycle. Based on the definition in Section 5.1, the RTBAC model can compute user u’s ratio of suspected “unsafe” access (conditions (1), (2), and (5)) in the i-th distribution cycle, which is denoted as p_i(u). If p_i(u) > P (P is the threshold that has been set by the administrator for identifying curious users), the RTBAC model considers u as “suspected curious” in the i-th distribution cycle.

In the initial phase of the k-th distribution cycle, the RTBAC model (i) defines Q_k as an experience value that is set by the administrator or (ii) computes Q_k based on the risk values of users who are not “suspected curious users” in the (k-1)-th distribution cycle. The mean of these risk values is denoted as \mu_{k-1}, and the variance is denoted as \sigma_{k-1}. We define Q_k = \mu_{k-1} + n \sigma_{k-1}, where n can be dynamically set according to the scenario.

The RTBAC model retrieves the nearest m distribution cycles for each user u and obtains the number of “suspected curious” cycles, which is denoted as m’. The final risk quota for u in the k-th distribution cycle (Q_k(u)) is as follows:
where $\eta$ is the cutting margin for reducing the user’s risk quota, which is set according to the application scenario. Therefore, if a user is extremely “curious,” he or she will only obtain a small risk quota after several distribution cycles. At the same time, this frequent reduction in risk quotas will also attract the attention of administrators to detect suspected risk behaviours as soon as possible.

6. Discussion

In this section, we will discuss the computational complexity, space complexity, and other performances of RTBAC.

6.1. Computational Complexity and Space Complexity

6.1.1. Computational Complexity. Efficiency is an important metric for evaluating access control models. In RTBAC, the steps of extracting topics from documents belong to offline computing, and other steps (using topics to grant access rights to users) belong to online computing.

For offline computing, we use LDA to extract topics from documents and paragraphs. LDA uses an iterative method to compute the relationship between paragraphs, topics, and words. Let $K$ be the number of topics, $N_D$ be the number of documents, $\bar{N}_p$ be the average number of paragraphs in each document, and $I$ be the average length (number of unique words in one paragraph) of each paragraph. In one iteration, LDA assigns all the $K$ topics for each paragraph based on their own words once. Therefore, in one iteration, the computational complexity of LDA is $\Theta(KN_D\bar{N}_pI)$. Let $N_{iter}$ be the number of iterations, and the total computational complexity of extracting topics is $\Theta(N_{iter}KN_D\bar{N}_pI)$.

For online computing, let $N_{tu}$ be the number of topics of the user, and $N_p$ be the number of topics of each paragraph. RTBAC uses the topics between the user and the target document to compute the access risk, and the computational complexity of computing the access risk is $\Theta(\bar{N}_p(N_{tu} + N_{tp}))$. In terms of computing the historical risk, RTBAC uses the access risks of the documents in the sliding window to compute the historical risk; the computational complexity of computing the historical risk should be $\Theta(N_{SW}\bar{N}_p(N_{tu} + N_{tp}))$, where $N_{SW}$ is the number of documents in the sliding window of the user. The computational complexities of other parts of online computing (computing the total risk and using the total risk to compute access rights) are $\Theta(1)$. Therefore, if the user only wants to get the access right of the target document, the total computational complexity of online computing will be $\Theta(\bar{N}_p((N_D + N_{SW})N_{tp} + N_{SW}N_{tu}))$.

6.1.2. Space Complexity. In offline computing, RTBAC needs to store two result matrices: $\theta$ and $\phi$. Matrix $\theta$ stores the weights of all paragraphs and all topics, while matrix $\phi$ stores the weights of all topics and words. Therefore, the space complexity of this part is $\Theta(K(N_D\bar{N}_p + N_{tu}))$, where $N_{tu}$ is the number of unique words in all the documents. For each document, RTBAC stores the topics of the document and its corresponding paragraphs, and the space complexity is $\Theta(N_{td} + \bar{N}_pN_{tp})$, where $N_{td}$ is the number of topics for one document. The total space complexity of all the documents is $\Theta(N_D(N_{td} + \bar{N}_pN_{tp}))$.

In online computing, RTBAC needs to store the topics of each user, and the space complexity is $\Theta(N_{tu})$. For each user, RTBAC uses a sliding window to store the historical accesses, and the space complexity for this part is $\Theta(S_{SW}\bar{N}_p\bar{N}_{tp})$, where $S_{SW}$ is the size of the sliding window, and $\bar{N}_{p_{sw}}$ is the average number of paragraphs in each sub-sliding window. The space complexity of storing risk quotas is $\Theta(1)$. Therefore, for all the users (the number of users is $N_U$), the total space complexity is $\Theta(N_U(N_{tu} + S_{SW}\bar{N}_p\bar{N}_{tp}))$.

6.2. Discussion on Risk Management, Dynamic, Content-Based, and Fine-Grained Access Control

6.2.1. Risk Management of RTBAC. RTBAC uses semantic contents to determine the risks of access requests. First, it uses the topics of the user and the target document to compute the access risk. Second, RTBAC uses the historical accesses in the sliding windows to compute the historical risk. Finally, it uses the access risk, historical risk, and different parameters ($\alpha$ and $\beta$) to compute the total risk (9). In (9), $\alpha$ denotes the ratio of the access risk and $\beta$ denotes the ratio of the historical risk. The greater the alpha, the greater the correlation between the total risk and document contents. In order to balance the content and user history access behaviour, RTBAC sets default $\alpha = 0.5$ and $\beta = 0.5$ for the target document. For similar documents ($am = 1$), since they are not directly requested by the user, the total risk of these documents should be more relevant to their own content, not the user’s access behaviour. RTBAC sets default $\alpha = 0.7$ and $\beta = 0.3$ for these documents. These parameters can also be changed according to the actual application requirements, and only need to satisfy $\alpha + \beta = 1$.

RTBAC also uses access risk and historical risk to determine whether the user’s access request is suspected an “unsafe” request. If the ratio of the user’s suspected unsafe requests is sufficiently large, RTBAC will identify him as a “curious user.” If a user is identified as a curious user, he will be assigned less risk quota. Meanwhile, the higher the risk of access requests, the more risk quota the user needs to consume. Compared with normal users, curious users will consume their risk quotas more quickly. If a user’s risk quota is low enough, he can only access the data completely related to his topics. RTBAC can limit the accessibility of curious users effectively.

6.2.2. Dynamic Access Control. In RTBAC, the topics of users can be changed according to their base sets. If the base...
set is changed, the authorization will also be changed dynamically. RTBAC designs sliding windows for each user to store historical access behaviours. If a user’s access behaviour is changed, RTBAC will recognize it and take countermeasures automatically (e.g., identifying the user as a curious user dynamically and distributing risk quota dynamically). While in the content-based access control model CBAC [5], the historical access behaviours are not stored, and it cannot grant authorization dynamically based on changing user behaviours.

### 6.2.3. Content-Based and Fine-Grained Access Control

In RTBAC, semantic content is used to enforce access control. RTBAC uses LDA to extract topics from documents and their paragraphs. Meanwhile, RTBAC assigns base sets to each user and determines topics for the users according to the documents in the base sets. When users send access requests to the system, RTBAC uses the topics of users and documents to compute risks and grant access rights to users. In RTBAC, documents and paragraphs are connected with users according to topics, and users can get paragraph-level access rights based on different topics. In other content-based access control models (e.g., CBAC [5] and RCBAC [25]), the enforcements are directly according to the semantic contents of the documents, not topics. Therefore, the other content-based access control models do not consider the different contents in different paragraphs of the same document. Therefore, these access control models can only provide file-level authorization. The differences between RTBAC and other access control models are shown in Table 1.

### 7. Experiments

In this section, we use the 20-newsgroups dataset and the ICD-10 medical dataset to perform two groups of comparative experiments on the RTBAC, CBAC, and RCBAC models. Table 2 shows the details of both datasets, and Table 3 shows the parameters and corresponding values used in our experiments.

We use the LDA Gibbs Sampling to extract topics from documents, and we set the number of topics to 100. For these two datasets, after 100 iterations, the two matrices (matrices $\theta$ and $\varphi$ in LDA) of the LDA completely converge, and thus we set the parameter iterations to 100. In Table 3, $T_{\text{DOCSIM}}$ is the threshold of authorization content similarity, which is set as 50% (normal threshold) and 80% (high threshold). $PUSReq$ is the percentage of “unsafe” requests, which refers to the probability that a user may access files that are of little relevance to his or her basic topics. In our experiments, normal users have 0% or 10% $PUSReq$, and curious users have a 30% $PUSReq$.

The experiment is run on a 64-bit Windows 10 system, with an Intel Core™ i7-3740QM @ 2.70 GHz CPU and 8.0 GB RAM. The sizes of the sub-sliding windows (near, middle, and far) are set as 2, 3, and 5.

### 7.1. Experiments on the 20-Newsgroups Dataset

In this section, the 20-newsgroups dataset is used to show different access control results from the RTBAC and RCBAC models. The 20-newsgroups dataset has rich semantic information, and all of the data are divided into different categories (newsgroups). The categories can be used to determine initial authorities.

We treat the files of four categories in the dataset (comp os ms-windows misc, comp sys ibm. pc hardware, comp sys mac hardware, and comp windows x) as unique files of 4 different departments. The staff members in the department have the initial right to access all the data from the department, but they do not have the initial right to access the data from other departments. In our experiment, we assume that the user is a member of department A (comp os ms-windows misc). Now, the user is assigned a task and needs to access the data from the other three departments. Our experiment separately authorizes the user with the RTBAC and RCBAC models, and evaluates the performance of these two access control models by the accessible number of paragraphs and the files of the user. We choose four different files in comp os ms-windows misc to set four different initial access rights of the user, and the experimental results show the average access control effects of the RTBAC and RCBAC models.

Figure 5 shows the average number of paragraphs and files that users of different $PUSReq$ can access under the four initial access permissions by the RTBAC and RCBAC models. The experimental results show that compared to users authorized by the RCBAC model, users authorized by the RTBAC model can access more files, but the total number of accessible paragraphs is extremely low. The reason for this is that the RTBAC model regards paragraphs

---

**Table 1: Comparison with different access control models.**

<table>
<thead>
<tr>
<th>Model</th>
<th>Dynamic</th>
<th>Risk-aware</th>
<th>Data content</th>
<th>Fine-grained authorization</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABAC</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>Up to file</td>
</tr>
<tr>
<td>RCBAC [25]</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>File</td>
</tr>
<tr>
<td>RTBAC</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>Paragraph</td>
</tr>
</tbody>
</table>

**Table 2: Details of the datasets.**

<table>
<thead>
<tr>
<th>Dataset</th>
<th>20-newsgroups</th>
<th>ICD-10 medical data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of texts</td>
<td>3908</td>
<td>1131</td>
</tr>
<tr>
<td>Number of paras</td>
<td>21867</td>
<td>6331</td>
</tr>
<tr>
<td>Average size of texts</td>
<td>2.08 KB</td>
<td>8.69 KB</td>
</tr>
</tbody>
</table>

**Table 3: Parameters and corresponding values in the experiments.**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_{\text{DOCSIM}}$</td>
<td>50%; 80%</td>
<td>Threshold of document content</td>
</tr>
<tr>
<td>$PUSReq$</td>
<td>0%; 10%; 30%</td>
<td>Percentage of unsafe requests</td>
</tr>
<tr>
<td>$\eta$</td>
<td>0.2</td>
<td>Cutting margin</td>
</tr>
<tr>
<td>$P$</td>
<td>10%; 15%; 20%</td>
<td>Threshold of normal/curious users</td>
</tr>
</tbody>
</table>
as the smallest authorized unit, and the RTBAC model can accurately assign the access rights of specific paragraphs in each document for users. However, once the RCBAC model determines to grant the access rights of one document to a user, it must grant all of the access rights of all the paragraphs of the document. Therefore, the RTBAC model can enforce paragraph-level authorization, while the RCBAC model can only enforce file-level authorization. Figures 5(a) and 5(b) show that for each authorized document, the number of paragraphs authorized by the RCBAC model is 6–8 times that of the RTBAC model. In other words, for the 20-newsgroups dataset, without considering the constraint topic, the RTBAC model has 6 times the authorization granularity compared to the RCBAC model.

Based on the above initial access rights, Figures 5(c) (1) and 5(d) show the experimental results of considering constraint topics. Compared with the experimental results shown in Figures 5(a) and 5(b), both the scopes of authorization of the RTBAC and RCBAC models are reduced. However, the reduction in the RCBAC model is much larger than that in the RTBAC model. The reason for this is that under the RCBAC authorization model, once a certain paragraph in a file contains constraint topics, all paragraphs of the entire file should be inaccessible to the user, but under the RTBAC authorization mode, only a certain paragraph cannot be accessed. When specific topics are inaccessible, the performance of the RTBAC model is also better than that of the RCBAC model. The RTBAC model can accurately grant users access rights that the RCBAC model cannot grant.

Figure 6 shows the performance of the RTBAC model. In Figure 6(a), the experimental results show the authorization changes of the RTBAC and RCBAC models after adding constraint topics. The RTBAC model cuts down approximately 50% of the user’s access rights with precision.
However, to prevent users from accessing content related to constraint topics, the RCBAC model prohibits users from accessing all documents with the content, resulting in the reduction of permissions reaching nearly 90%. Figure 6(b) shows the different access abilities between normal users and curious users. The ordinate represents the access ability multiples of normal users compared to curious users under different risk quota distribution parameters. The experimental results show that the RTBAC model can obviously grant more access rights to normal users. Even in the worst case, the RTBAC model can grant more than 3 times the access rights to normal users than curious users. The RTBAC model can clearly distinguish curious users from normal users. In order to test the efficiency of RTBAC, we log in as 1, 2, 4, 8, and 16 users, respectively. Figure 6(c) shows the average execution time for each user in RTBAC (access mode = 1), CBAC, and RCBAC. The experimental results show that the execution time of RTBAC is shorter than CBAC, but is slightly longer than RCBAC (about 100 milliseconds). The reason is that RCBAC only uses the keywords of each document to determine authorization, without considering the contents of different paragraphs. RTBAC needs to compute the topics of documents and their corresponding paragraphs such that it can grant paragraph-level authorization. In the experiment using the 20-newsgroups dataset, the average execution time for each user in RTBAC is less than 800 milliseconds, which is acceptable. Figure 6(d) shows the execution time with varied number of topics. We set the number of topics as 5, 10, 15, and 20; the experimental results show that the authorization time increases with an increase in topics, but the increase is not large (from 662 ms to 725 ms). Generally, the topics of a document will not exceed 20, and therefore the number of topics will not affect the authorization efficiency very much.
We assume that for a user belonging to Department A, the authorized paragraphs without constraint topics in the comp os ms-windows misc dataset are necessary paragraphs, the authorized paragraphs with constraint topics in other datasets are suspected to be risk paragraphs, and other paragraphs are indifferent. Table 4 shows the ratios of the three kinds of authorized paragraphs in the RTBAC and RCBAC models.

From Table 4, we can see that the RTBAC model has a higher necessary paragraph ratio and a lower suspected paragraph risk ratio than the RCBAC model. The RTBAC model has better authorization performance than the RCBAC model.

7.2. Experiments on the ICD-10 Dataset. In this section, we use Wikipedia’s entries on the ICD-10 disease classification to compare the RTBAC model and the RCBAC model. We selected four categories of ICD-10 disease entries: anemias, blood disorders, cancer, and nutritional deficiencies. We choose four different files from anemias, blood disorders, and cancer to set four different initial access rights of the model.

![Figure 7: Number of accessible paragraphs and files (ICD-10 dataset).](image)

(a) $T_{DOCSIM} = 50\%$, without constraint topics. (b) $T_{DOCSIM} = 80\%$, without constraint topics. (c) $T_{DOCSIM} = 50\%$, with constraint topics. (d) $T_{DOCSIM} = 80\%$, with constraint topics.

We assume that for a user belonging to Department A, the authorized paragraphs without constraint topics in the comp os ms-windows misc dataset are necessary paragraphs, the authorized paragraphs with constraint topics in other datasets are suspected to be risk paragraphs, and other paragraphs are indifferent. Table 4 shows the ratios of the three kinds of authorized paragraphs in the RTBAC and RCBAC models.

From Table 4, we can see that the RTBAC model has a higher necessary paragraph ratio and a lower suspected paragraph risk ratio than the RCBAC model. The RTBAC model has better authorization performance than the RCBAC model.

Table 4: Ratio of different paragraphs (20-newsgroups dataset).

<table>
<thead>
<tr>
<th>Model</th>
<th>Necessary (%)</th>
<th>Indifferent (%)</th>
<th>Suspected risk (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTBAC</td>
<td>40.1</td>
<td>59.9</td>
<td>0</td>
</tr>
<tr>
<td>RCBAC</td>
<td>13.4</td>
<td>85.2</td>
<td>1.4</td>
</tr>
</tbody>
</table>

![Figure 7: Number of accessible paragraphs and files (ICD-10 dataset).](image)

(a) $T_{DOCSIM} = 50\%$, without constraint topics. (b) $T_{DOCSIM} = 80\%$, without constraint topics. (c) $T_{DOCSIM} = 50\%$, with constraint topics. (d) $T_{DOCSIM} = 80\%$, with constraint topics.
user, and the experimental results show the average access control effects of the RTBAC model and the RCBAC model.

In content-related access control authorization, we hope to authorize as accurate content-related data as possible. Figures 7(a) and 7(b) show the number of accessible paragraphs and files between the RTBAC model and the RCBAC model (without constraint topics). From the experimental results of the ICD-10 dataset, we can see that the number of files authorized by the RTBAC model is always greater than that of the RCBAC model. Even in the worst case, the number of paragraphs authorized by the RTBAC model is more than 2 times that of the RCBAC model. For each authorized document, the number of paragraphs authorized by the RCBAC model is more than 3 times that of the RTBAC model. In other words, for the ICD-10 dataset, without considering the constraint topic, the RTBAC model has 3 times the authorization granularity compared to the RCBAC model.

Figures 7(c) and 7(d) show the experimental results after adding the constraint topics. Both the scopes of authorization of the RTBAC model and the RCBAC model are reduced. As PUSReq grows, the RTBAC model grants lower access rights to users, but the RCBAC model grants the same access rights. Therefore, in some extreme cases (e.g., the ICD-10 dataset with constraint topics added), the RCBAC model cannot distinguish curious users from normal users easily, but the RTBAC model can still work.

Figure 8(a) shows that in the ICD-10 dataset, the RTBAC model and the RCBAC model have similar reduced accessibilities. However, since in some extreme cases the RCBAC model cannot distinguish curious users from normal users, the RTBAC model has a better performance than the RCBAC model in the ICD-10 dataset. Figure 8(b) shows the different access abilities between normal users and curious users in the ICD-10 dataset. Even in the worst case, the

![Figure 8: Performance of RTBAC (ICD-10 dataset). (a) Reduced access abilities compared with RCBAC. (b) Magnification of access abilities. (c) Execution time of different models. (d) Execution time of different topics.](image-url)
The RCBAC model has an even higher ratio of suspected risk paragraphs compared to the ratio of necessary paragraphs, the RCBAC model does not have enough of a higher necessary paragraph ratio and less suspected paragraph risk ratio than the RCBAC model. Meanwhile, compared to the ratio of necessary paragraphs, the RCBAC model has an even higher ratio of suspected risk paragraphs.

In this case, the RCBAC model does not have enough of a good performance, but the RTBAC model has much better authorization performance than the RCBAC model.

RTBAC model can grant more than 2.4 times the access rights to normal users than curious users. We log in as 1, 2, 4, 8, and 16 users, respectively. Figure 8(c) shows the average execution time for each user in RTBAC (access mode = 1), CBAC, and RCBAC. The experimental results show that the execution time of RTBAC is shorter than CBAC, but is slightly longer than RCBAC (about 80 milliseconds). RTBAC use this time to compute topics of paragraphs to complete file-level authorization, and the extra time overhead is acceptable. Figure 8(d) shows the execution time with varied number of topics (5, 10, 15, and 20); the experimental results show that the authorization time increases with an increase in topics, but the increase is not large (from 557 ms to 576 ms). Compared with the experimental results based on the 20-newsgroups dataset, the increase is much smaller. The reason is that the number of documents and paragraphs in the ICD-10 dataset is smaller than that in the 20-newsgroups dataset, which is consistent with the conclusions of computational complexity in Section 6.1.

We assume that for a user belonging to Department A, the authorized paragraphs without constraint topics in the blood disorders dataset are necessary paragraphs, the authorized paragraphs with constraint topics in other datasets are suspected to be risk paragraphs, and other paragraphs are indifferent. Table 5 shows the ratios of the three kinds of authorized paragraphs in the RTBAC model and the RCBAC model.

From Table 5, we can see that the RTBAC model has a higher necessary paragraph ratio and less suspected paragraph risk ratio than the RCBAC model. Meanwhile, compared to the ratio of necessary paragraphs, the RCBAC model has an even higher ratio of suspected risk paragraphs. In this case, the RCBAC model does not have enough of a good performance, but the RTBAC model has much better authorization performance than the RCBAC model.

### 8. Conclusion and Future Work

In this paper, we propose a fine-grained content-based access control model RTBAC, which uses topics and risk management to realize dynamic paragraph-level authorization. We use two different datasets to conduct comparative experiments with CBAC and RCBAC. The experimental results show that RTBAC can execute file-level authorization efficiently. Meanwhile, RTBAC can distinguish “curious users” from “normal users” efficiently and restrict the accessibility of curious users.

To the best of our knowledge, the RTBAC model is the first risk-aware access control model with paragraph-level authorization, and we can conduct further researches based on RTBAC in the following two directions:

1. More efficient offline processing. In RTBAC, the topics are extracted based on latent Dirichlet allocation. LDA is one of the most classic topic models, but the execution efficiency of LDA is not very good. Although offline extracting topics does not affect the efficiency of online authorization, in order to apply RTBAC in super large-scale text data, it is also necessary to improve the efficiency of offline extracting topics.

2. RTBAC with encryption technology. In the information age, many data are stored in the clouds. RTBAC is now an access control model for the plaintext database. In order to apply RTBAC in the cloud computing platform, it is also very necessary to combine RTBAC with encryption technology.

### Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

### Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

### Acknowledgments

This work was supported by the National Natural Science Foundation of China (61972209, 61572263, and 61872197), the Postgraduate Research and Practice Innovation Program of Jiangsu Province (KYCX18_0891 and KYCX21_0789), the Natural Science Foundation of Jiangsu Province (BK20161516 and BK20160916), the Postdoctoral Science Foundation Project of China (2016M601859), the Natural Research Foundation of Nanjing University of Posts and Telecommunications (NY217119), and the Anhui Provincial Key Laboratory of Network and Information Security (AHNIS2020002).

### References


