DOI: 10.31534/engmod.2019.2-4.ri.01d
Original scientific paper
Received: 27.10.2018.

Towards Provenance Cloud Security Auditing Based on Association Rule Mining

Shanshan Tu, Xinyi Huang

Faculty of Information Technology, Beijing University of Technology, Beijing, CHINA
e-mails: sstu@bjut.edu.cn; junzilanfang@126.com

SUMMARY

Cloud storage provides external data storage services by combining and coordinating different types of devices in a network to work collectively. However, there is always a trust relationship between users and service providers, therefore, an effective security auditing of cloud data and operational processes is necessary. We propose a trusted cloud framework based on a Cloud Accountability Life Cycle (CALC). We suggest that auditing provenance data in cloud servers is a practical and efficient method to log data, being relatively stable and easy to collect type of provenance data. Furthermore, we suggest a scheme based on user behaviour (UB) by analysing the log data from cloud servers. We present a description of rules for a UB operating system log, and put forward an association rule mining algorithm based on the Long Sequence Frequent Pattern (LSFP) to extract the UB. Finally, the results of our experiment prove that our solution can be implemented to track and forensically inspect the data leakage in an efficient manner for cloud security auditing.

KEY WORDS: *cloud security auditing; provenance data; log analysis; user behaviour; association rule mining algorithm.*

1. INTRODUCTION

Lately, cloud computing has become an active research area. Since cloud computing requires of its users to share data and resources with service providers, it brings several security problems, which could potentially limit the development of cloud computing [1-3]. The Hewlett Packard security research centre proposed the "Trusted Cloud" concept and model of cloud security – the Cloud Accountability Life Cycle (CALC) [4]. In this model, the general solution to cloud data security is described as a correlate cycle structure. It divides the security process into seven stages: Policy, Trace, Logging, Store, Reporting, Auditing and Optimizing, and the whole security process is progressive [5]. On this basis, Ryan et al. proposed Flogger [6], S2Logger [7] and other schemes for collecting log data. The basic idea is to collect operating system log data in a whole-cloud data environment. These schemes are different from previous ones, which detected input and output data [8-9], in such a way that they are

able to ascertain the state of the data, and can solve the problems of credibility and security in the cloud environment [10-11]. Thus, it is improving the feasibility of the trusted cloud. However, with the huge increases in data traffic, limiting log analysis to the Logging stage alone, cannot satisfy the basic requirements for the trusted cloud. Therefore, an inductive method is needed which can analyse log data. Alternatively, more log data analysis in the CALC process are needed. To solve the above-mentioned problem, we compared the different cloud security controlling methods and proposed our own trusted cloud framework based on Cloud Accountability Life Cycle [12]. In this paper, we analyse the log data in this cloud framework and propose a new algorithm based on our previous research.

Comparativly, this paper introduces a trusted cloud framework and an association rule mining algorithm already used in Knowledge Discovery in Database (KDD) [13], in Data Mining as well as in Data Analyse. The original algorithm was used to solve the classic "Basket" issue by analysing the product bought by customers, and determining the relationship between different products (based on a statistical, not logical relationship). This paper focuses on the Frequency Pattern growth (FP-growth) algorithm, and proposes a practical, improved algorithm based on the real need for log analysis in cloud security auditing. The contribution of our work is presented as follows:

1. In the Store stage, having the introduction of log analysis, the original operating system log should be converted into data that can describe user behaviour (UB).
2. In the Reporting and Auditing stages, our solution provides a more readable and practical description, generating data the users are more familiar with, offering "Cloud Credibility" with more direct support and proof.
3. In the Optimizing and Policy stages, our process improves the Optimizing stage from the level of technology to the constraint of actual behaviours, and also guarantees simpler and more accurate accountability.

The rest of the paper is organized as follows. In Section 2, related work is discussed, and the provenance of the cloud security auditing system framework is demonstrated. In Section 3, our proposed approach is described. The experimental procedure, results and dataset explanation are given in Section 5, followed by the conclusion and suggestions for the future work in Section 6.

2. RELATED WORK

2.1 CALC MODEL

The development of the cloud environment has led to the complexity of auditing, including the need to perserve data, and of determining any redundant information. Therefore, a security model that includes all key steps becomes especially important, for it can simplify many complex problems and make them clearer for data security auditors. It can also help auditors to focus on a particular step to carry out more research. Therefore, the CALC is designed for this purpose as illustrated in Figure 1.

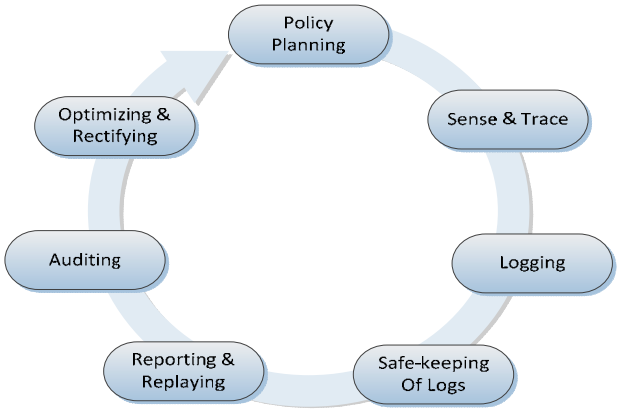

Fig. 1 *CALC Model*

1) Policy Planning

   Cloud service suppliers must decide which data items need to be collected and which events should be recorded in logs. There are four important points to consider:

   a) Event Data: a series of actions and related information;

   b) Action Data: the operator who deals with events (such as a human or an internet crawler);

   c) Timestamp Data: times and dates of events;

   d) Location Data: physical and virtual addresses of events.

2) Sense and Trace

   At this stage, the purpose is to recognize all operations in the cloud. Auditing schemes need to be able to track the data. The tracking is from the lowest-level system read/write calls, all the way up to irregularities in high-level workflows hosted in virtual machines in disparate physical servers and locations. It is also necessery to trace data packets transformed among network routes within the cloud [14].

3) Logging

   Log records should work in both virtual and physical layers in the cloud if documents are considered the centre. In addition, a log's active time, conservation place in the cloud, and other related factors should be taken into consideration.

4) Safe-keeping of Logs

   After log collection, the integrity of the log should be protected by preventing unauthorized users to illegally obtain or revise log data, while data encryption as well as some reasonable backup mechanism should be introduced.

5) Reporting and Replaying

   Reporting tools consist of a log summary that considers the documents as the centre, reports auditing clues, documents records obtained with operational cycle time in the cloud. Therefore, the report may include many factors, such as historical records of virtual machine, physical services and OS-level read/write files.

6) Auditing

   Auditors or stakeholders audit logs and reports, and imply possible illegal actions. The auditing products can be assigned if automation is possible. Automatic and mandatory auditing is very practical for a cloud environment since it has a large data storage capability. It can also be very effective for identifying any illegal action.

7) Optimizing and Rectifying

   In this part of the process, any problem areas in the cloud, and any security gaps, can be removed or revised; regarding the controls and supervision that need to be further improved.

2.2 ASSOCIATION RULES ALGORITHMS

In 1993, Agrawal [15] first proposed an association rule mining algorithm for the purpose of finding associations between a product and a customer transaction. The basic idea was to find the datasets that satisfied the requirements of frequency and analysed any hidden associations between them. Since then, another research has been carried out that led to the improvements and extensions in the performance of this algorithm.

Two particular algorithms, Apriori and Frequent Pattern (FP)-growth [16], have been proposed. The Apriori algorithm gradually produces multiple sets of association analysis based on iterative thought in order to obtain sets with more items, all leading to a better analysis result. In 2000, Jiawei Han proposed the FP-growth algorithm. To overcome the problems of time and space complexity caused by iterations in Apriori, FP-growth had a "tree" concept introduced into algorithm. Consequently, it changed the iteration process to one that scans a dataset only once to produce an FP-tree. The structure of the tree is able to describe the frequency of the dataset and to gain the frequent multiple dataset, using the tree's growth algorithm. As a result of its good performance, FP-growth entered the mainstream of association rule mining algorithms.

As for the objective aspect, the AprioriAll algorithm [17] adds a time constraint to item sets based on the Apriori algorithm in order to analyse any associations between short sequences. AprioriAll produces candidate sequences based on the last scanned database, while at the same time it can calculate the minimum support degree needed and select suitable sequences for next scanning.

The Generalized Sequential Pattern (GSP) algorithm [18] is an extension of the AprioriAll algorithm. The GSP algorithm introduces time constraint, sliding window, and classified level technology, and also adds the constraint of a scan. It decreases the number of candidate sequences efficiently and overcomes the limitations of the basic sequence model. It also decreases the production of excessive useless sequences. Furthermore, GSP uses a hash tree to store candidate sequences and decreases the amount of sequences to be scanned. In addition, the GSP algorithm converts the representations of data sequences, so that it is easier to detect whether a candidate is the consequence of the data sequence. Nevertheless, Apriori and its improved algorithms must still produce a large number of intermediate sets (candidate sets) in the cloud environment. However, it causes the complexity of time and space to become larger, and thus the efficiency of these algorithms is suboptimal for dealing with big data.

FP-growth introduces a graphic structure to process the innovative algorithm, calculative framework and database structure. Improved algorithms based on FP-growth focus on active

areas such as the distributed system [19] and the project database [20]. However, these algorithms rely too heavily on a "tree data structure".

2.3 CONSTRAINT PATTERNS

In order to improve the association rule of the algorithms, it is necessary to introduce one constraint condition [21]. The Apriori algorithm is based on a simple logical constraint. This means that any subset of a frequent set must also be recurrent. The AprioriAll and GSP algorithms filter useless multiple sets introducing a time constraint to the dataset. Moreover, the FP-growth algorithm introduces a "tree" to more obvious claim constraints. Hence, this kind of algorithm can improve the efficiency of mining technique.

Based on the types of constraints in mining, association rule mining can be divided into the following:

1) Monotone Constraint: The constraint C_m is called a monotone constraint, because set S can satisfy C_m, and any superset of S can also satisfy C_m.
2) Anti-monotone Constraint: The constraint C_a is called "anti-monotone", because if there is any item of set S which does not satisfy C_a, there are no supersets of S that can satisfy C_a.
3) Convertible Constraint: A convertible Constraint is that one where the constraints can be transferred between the item set and its subset. What makes it difficult to use is if one constraint cannot be given in the form of monotone or anti monotone. Thence, in order to solve it, the problem can be converted by the special organization of data.
4) Succinct Constraint: The succinct constraint is that a subset of item I_s can be represented as a selection operation $\delta_p(I)$ by using a selective predicate p. Obviously, if a constraint is succinct, then it can directly use an SQL query to achieve this set so as to meet the conditions. In the different stages of data mining, it can be used to assess succinct constraints to avoid unnecessary testing. Depending on the source of constraint conditions, the constraints can be classified as follows:
    1) Knowledge type constraint: Background knowledge can be used to filter irrelevant information.
    2) Data constraint: Using some properties of the data, this type of constraint can isolate unrelated information from the current problem.
    3) Dimension, layer, degree constraint: Using the structure and the scale of data or a database, data mining can be limited to a certain range to filter out any additional information.

3. PROVENANCE CLOUD SECURITY AUDITING SYSTEM

Aiming at one or more steps in the whole cloud auditing framework, this paper outlines a "provenance" cloud security auditing approach. It records the actions in the system (including documents, the internet, and programs) to describe clearly the status of data in a cloud environment. In different cloud environments, a provenance system can decide on the data content of security auditing. This means that it can obtain information about users' data, actions, etc., over a certain period of time, and place this information as a data source for a cloud security auditing system. In this paper, the provenance data refers to information that

can describe the operational status and the source of any data across all sequences of an operation.

3.1 TRUSTED CLOUD FRAMEWORK

CALC describes a whole process of log auditing at every step, but it focuses more on the theoretical integrity of the model. It is possible to set up a cloud data security system with specific CALC, making it more operationally useful. Therefore, a plan is proposed that is based on the trusted cloud framework proposed by Ko and colleagues [22], with an auditing framework based on provenance cloud data as shown in Figure 2.

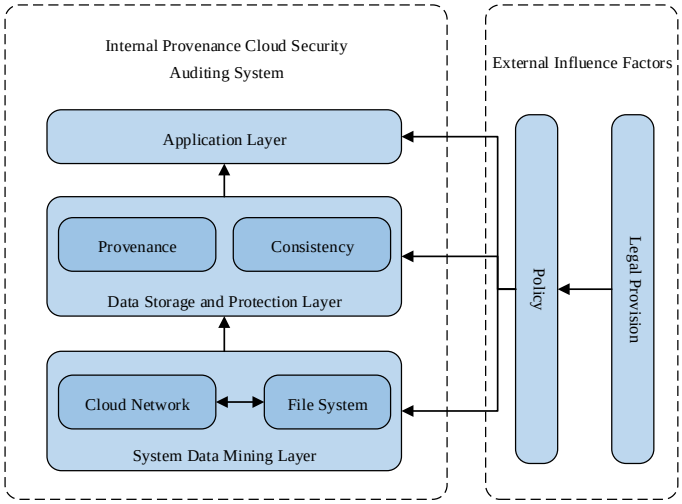

Fig. 2 *Provenance cloud security auditing system framework*

3.2 FRAMEWORK INTRODUCTION

This framework has two external and three internal parts. The external part consists of legal provision and strategy, and the internal part contains data mining layer, data storage and protection layer, and an application analysis layer. Each of these five parts will be discussed as follows.

1) Legal Provision

With the development of e-commerce and cloud computing, people started paying more attention to the protection of private data. Relevant organizations in all countries have realized the necessity to improve relevant laws and regulations. These laws relate to the collection, management, storage, and transformation of data and are also, undoubtedly, suitable for log data which should be protected as well.

Relevant laws stipulate some important issues related to data security, such as the content, purpose, time, methods, authorization, and surface of data collection. All organizations and individuals must develop strategies for data collection and analysis that are subjected to these laws.

2) Policy

A complete implementation strategy is necessary in order to achieve cloud security auditing based on the log collection. In the first part of the framework, laws and regulations limit the approach to data collection and analysis. Any organization or individual involved in the

formulation of strategies should fully consider the constraints of these laws. These constraints penetrate all three levels of the log audit system. Each organization and individual can choose their own strategies under the constraints of such a law for the data to be collected, as well as the technique of storing the data.

3) System Data Mining Layer

The lowest abstract layer of the provenance system is the data mining layer. In this layer, one needs to know where to get useful system data for analysis and auditing in the next step. Data collection centred on files mainly occurs in the system and internal network.

a) File System

Cloud computing technology makes people no longer focus only on a physical server within a company, but it draws their attention more to a file-centred system. Log data collection mainly happens in this location, and it will record any sensitive behaviour related to the files such as opening, modifying, and closing them. At the same time, core data from such system information, like the virtual and physical storage addresses, operators, and times, can be obtained.

b) Cloud Network

Due to the characteristics of a cloud data storage system, a large amount of network data communications between the hosts is produced across the whole system. This kind of network information contains such items as the storage address of data, file details, and system call information. Therefore, the secure collection and management of network information within the cloud is also necessary.

4) Data Storage and Protection Layer

This layer gathers basic data obtained via a log collector such as the data from a file system and the cloud network. It also guarantees the security of the stored data and ensures back up. This paper focuses on two important properties of the log collector i.e., the provenance and consistency.

a) Provenance

History tracking allows users to control the real operating conditions of their own data. Provenance data is considered to be the basis for privacy and trust models. It improves the reliability of cloud computing relying on the provenance of data. It can help people distinguish between legal and illegal operations. The features of the Provenance data are discussed in the next section.

b) Consistency

It is almost impossible to guarantee data consistency for cloud service providers. However, it is very important for cloud computing services to respond quickly, restore data to their original state, recover data, and perform valid backup procedures. Cloud service providers should implement these features for data consistency.

5) Application Layer

After collecting and storing data, the application layer needs to be analysed in order to obtain valid data. There are three points to focus on in the application layer i.e., auto-audit, cloud fragments audit, and service audit.

a) Auto-audit

There are many possibilities for implementing auto-audits based on the excellent computing power of cloud computing and the reliability of data provided by a trusted cloud framework. This is a prerequisite for the application in financial and commercial areas of cloud computing. However, the virtualized environments continuous auditing is a very complex process, and requires both business logic and financial auditing software.

b) Cloud Fragments Audit

There are many file fragments within the cloud and their fixing and updating also needs to be audited. Especially, the interrelated documents need to be audited more strictly to ensure the data validity.

c) Service Audit

Cloud computing is a service-oriented hierarchy, each of which is easy to copy. Therefore, a combination of services is highly feasible. Once services are integrated among different levels, credibility will be tested, and service integration audited.

4. LONG SEQUENCE FREQUENT PATTERN ALGORITHM

With the development of network technology, association rules about streaming data, a dataset with time constraints, draw more attention. Based on the analyses of our previous research, we realized that the object of association rule mining is still used for discrete data. Even though the GSP algorithm introduces time constraints into transactions, its oriented dataset still remains discrete. Hence, aiming at a dataset of long sequence structure it cannot meet the necessary functional requirements. Our scheme is based on the FP-growth algorithm, introducing reasonable constraint conditions. These include a strong time constraint, as well as the comparable "Directed Graph" structure, to express the constraint conditions directly. Through the transformation of the algorithm structure and the introduction of a new model, we designed an association rule mining algorithm based on a Long Sequence Frequent Pattern (LSFP), adapted for a cloud security auditing system.

4.1 LSFP ALGORITHM

The frequent pattern tree structure of the FP-growth algorithm is not effective when describing the ordering of a dataset that aim at the characteristic of strong time constraints of a dataset. For sequence-type datasets, the nature of the data itself may make the frequent pattern tree size too large to be applied. The main reason for this is that the tree structure can more easily express the relationship between nodes, but the order of the data makes the "edges" between the nodes more meaningful when describing the data. A kind of graph structure that can directly express the relationship among the "edges" is needed for ordered data. Therefore, our scheme introduced a new graphical structure called "directed graph". This structure has two main advantages. First, a directed graph is not the same as the tree structure. The "edge" of a directed graph can describe the transfer relationship from node-to-node, which could effectively describe the "order information" of strong time constraints in the data. Second, in comparison with the tree structure, a directed graph structure can guarantee no redundancy node in the graph providing the improvements in space complexity. Based on this theory, the algorithm structure and process design are outlined below.

The algorithm is divided into two steps. The first step is the algorithm training phase which uses the training set to get the frequency of the entire dataset. The second step is the analysis of the data mining stage performed through the obtained data frequency to acquire UB which meets the frequency in the dataset. Related parameter definitions are given in Table 1.

Table 1 *Parameter definition*

Parameter	***Designation***	***Interpretation***
$TS=\{i_1,i_2,...,i_n\}$	*Training sequence*	i *is known as the item, in the actual data, represents a process level of system operation.*
$I=\{i_1,i_2,...,i_m\}$	*Item set*	I *is a proper subset of the training set, including all items in* TS *. Each of them has properties:* $begin$, end .
D	*Incidence matrix*	*It is an* $m\times m$ *matrix, the item* (i,j) *in a matrix represents the number of times directed edge appears from node* i_i *to node* i_j *. That is, the number of times node* i_j *appears after the node* i_i *in* TS *.*
$\tilde{D}$	*Frequency incidence matrix*	$\tilde{D}$ *is correlation matrix after optimizing (pruning)* D *. When* $D(i,j)<s$ *(* s *is threshold),* $\tilde{D}(i,j)=0$ *; when* $D(i,j)>s$, $\tilde{D}(i,j)=D(i,j)$.
$S=\{i_1,i_2,...,i_n\}$	*Operation sequence*	*Pending set formed by process level log.*
$B=\{b_1,b_2,...,b_n\}$	*User behaviour set*	*Result set obtained by algorithm process.*
$b=\{i_1,i_2,...,i_n\}$	*User behaviour*	*Formed by ordered items, representing an approved system operation combination.*
min_sup	*Minimum support*	*The threshold of frequency,* i *smaller than minimum support degree is not frequent.*

In the item set, the node contains attributes $i_m.begin$ and $i_m.end$, utilized to describe whether the node can be used as a starting node and terminating node for UB.

4.2 ALGORITHM STEPS

1) Algorithm first stage:

   Step 1: Scan training set, generate set *I*

   Step 2: Generate the directed graph and describe *D*

   Step 3: Select the appropriate threshold *min_sup* to prune the directed graph

   Step 4: Introduce priority rules to optimize *D*, and obtain the complete $\tilde{D}$

   Step 5: Obtain the possible *b* of the short sequence format, through the directed graph search

2) Algorithm second stage:

   Step.1: According to $\tilde{D}$, divide the long sequence *S* into several short sequences, and produce the sequence set to be extracted;

   Step.2: Extract *b* from this kind of sequence set, and generate *B*.

5. ALGORITHM IMPLEMENTATION AND PERFORMANCE ANALYSIS

5.1 EXPERIMENTAL ENVIRONMENT

1) Algorithm environment

For algorithm design reasons and practical issues, we selected a Linux operating system log as a dataset, because it possesses good time properties and suitable representative. Then the appropriate hardware and software environments based on the complexity of the algorithm were selected.

Virtual machine	*VMware WorkStation*
CPU	*Intel i7 2.4GHz*
RAM	*4G*
Operation system	*CentOS 6.4, 64bit*
Programming language	*C language*

2) Algorithm parameters

The impact of key parameters on the performance of the algorithm mainly includes two aspects, i.e., the training set size and minimum support (*min_sup*).

The size of the training set: The proposed algorithm selects part of the dataset as a training set to obtain frequency, mainly because the dataset and the training set have same source. Since they have similar characteristics, the frequency obtained from analysing the training set is also suitable for the dataset.

When using the entire dataset as a training set, both sets must have the same frequency. However, an oversize training set reduces the overall efficiency of the algorithm. Thus, selecting an appropriately sized training set can have a great impact on the performance of the algorithm.

Minimum support: Minimum support determines the operation combinations "*b*" can consider. It will inevitably obtain more operation combinations by picking a smaller support. However, whether these combinations occur frequently enough to be treated as *b, this* cannot be guaranteed. However, the accuracy of gaining *b* can be guaranteed if a larger support is picked. Contrariwise, it will also mean that the algorithm will get less operation combinations.

5.2 ALGORITHM ANALYSIS

Before analysing the algorithm's performance, we need an intuitive perception of the results of the algorithm. Here, several common user behaviours corresponding to log sequences (rules) from an operating system are shown:

File Creation

Create,wzq,3103,2406,2406,2387,nautilus,/home/wzq/Desktop/test/new file,/home/wzq/,193,438,30
Close,wzq,3103,2406,2406,2387,30

File Copy

Open,wzq,3354,3054,3054,2752,test,log,/home/wzq/Desktop/test/,577,438,4
Close,wzq,3355,3354,3054,2752,3
Duplicate2,wzq,3355,3354,3054,2752,4,1
Close,wzq,3355,3354,3054,2752,4

File Edit

Move,wzq,2956,2928,2928,2856,vim,aaa.txt,aaa.txt~,/home/wzq/Desktop/test/
Create,wzq,2956,2928,2928,2856,vim,aaa.txt,/home/wzq/Desktop/test/,577,436,3
Write,wzq,2956,2928,2928,2856,3,0,6F6E650A0A
Read,wzq,2856,1,2424,2405,21,0,0D1B5B3F32356C226161612E747B7422
Close,wzq,2956,2928,2928,2856,3
Chmod,wzq,2956,2928,vim,aaa.txt,33204

In addition, we need the theoretical analysis of the factors that may affect the proposed algorithm. Here, experimental results may directly reflect the performance of the algorithm.

1) Analysis of the algorithm performance for the training set

In this section, the size of the dataset and the training set means the size of the log through "pre-processing". This is because the log collection process will produce large amounts of system log information which is not concerned with data operations, that are more than *80%* of the original log data. In order to analyse the effect of training set size on the performance of the algorithm, the same dataset and minimum support should be selected. However, these should be the main indicators of algorithm performance evaluation, the number of rules, the number of behaviours, and the running times.

a) Rule number and behaviour number

From the experiment shown in Figure 3, with the scale of the training set gradually increasing from *1M* to *10M*, more complete rule information in the training phase can be achieved. For the same dataset, the number of behaviours increasing from *400ms* to *900ms*. These two results show that increasing the scale of the training set gives substantially a non-linear logarithmic relationship.

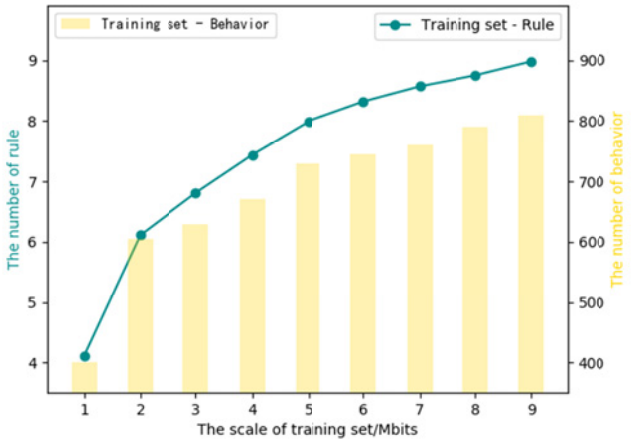

Fig. 3 *The number of rules and behaviours when increasing the training set*

b) Algorithm running time

Figure 4 illustrates that the running time is directly related to the scale of the training set. When the scale of training set is increased, the time required for the training phase also increase the relationship between them which is essentially linear. The number of rules obtained increase with an increase in the training set, and both show a logarithmic relationship. Increasing the rules will also lead to a more time consuming extraction stage. During the experiment, along with the scale of the training set gradually increasing from *1M* to *10M*, the algorithm running time also increases from *400ms* to *900ms*. The algorithm running time and training set size show a substantially linear relationship.

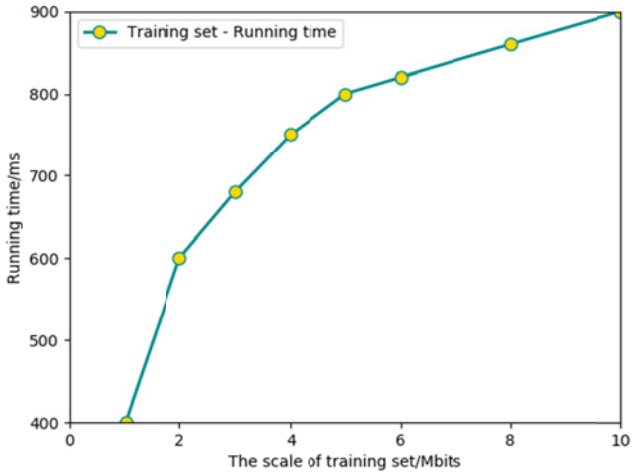

Fig. 4 *Running time with increased training set size*

2) Performance analysis of minimum support

The minimum support of an association rule algorithm can be obtained through the use of multiple methods, which is an important aspect of our research. In this paper, the influence of different minimum supports on the performance of the algorithm is analysed.

In order to analyse the effect of minimum support on the algorithm's performance, the same dataset and training set is selected. The main indicators for evaluating the performance of the algorithm are the number of rules, the number of behaviours, and the running times.

a) Rule number and behaviour number

A decrease in the minimum support is bound to increase the number of rules, and more behaviours in the extract phase can be obtained. As illustrated in Figure 3, as the minimum support increase from *20* to *100*, the number of behaviours is reduced from *1200* to *400*. The numbers of behaviours, rules, and the minimum support show a non-linear exponential relationship. However, by reducing the minimum support, the original infrequent operation combination can be identified as a frequent rule which makes the accuracy of actual behaviour extractions low, as shown in Figure 5.

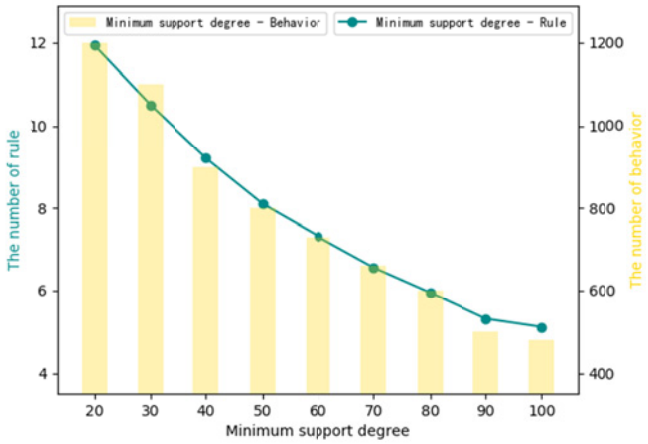

Fig. 5 *The number of rules and behaviours when increasing the degree of minimum support.*

b) Algorithm running time

As the training set size is constant, the running time required for the algorithm training phase remaines unchanged. However, if minimum support is decreased, leading to an increase in the number of rules obtained, more time consuming algorithm in the extraction phase is brought. Figure 6 shows that as minimum support increases from *20* to *100*, running time decreases from *1200ms* to *400ms*. The running time and the minimum support show a non-linear exponential relationship.

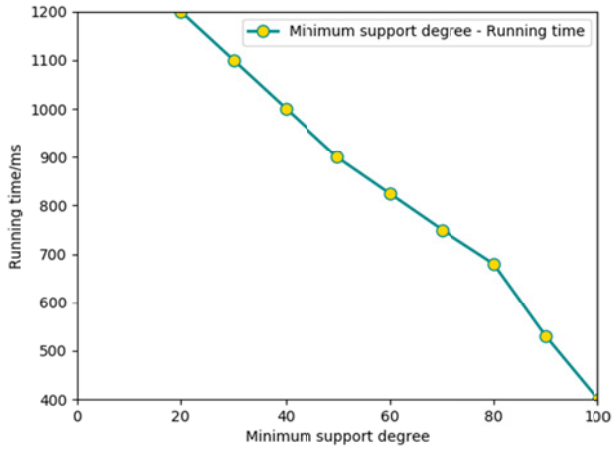

Fig. 6 *Running time with increased minimum support degree*

6. CONCLUSION

In terms of practical problems related to cloud security auditing, this paper described the advantages and disadvantages of the existing solutions. It analysed the association rule mining algorithms that are currently main-stream in the data mining field. A provenance cloud security auditing system framework was introduced, and an association rule mining algorithm based on Long Sequence Frequent Patterns (LSFP) was proposed. That converted a process of log in to an operating system into a description of user behaviour. A directed graphical structure for extracting user behaviour was designed reducing effectively both the space complexity and the time consumption. The experiments evaluations showed that the minimum

support had a direct influence on the results and the efficiency of the proposed algorithm. In future work, our focus will be on designing and implementing a more suitable method for calculating the minimum support and improving the performance of the algorithm. Such improvements can track and forensically investigate the data leakage during the development of cloud data privacy.

7. ACKNOWLEDGEMENTS

This work is supported in part by the National Natural Science Foundation of China (No. 61801008), National Key R&D Program of China (No. 2018YFB0803600), Beijing Natural Science Foundation National (No. L172049), Beijing Science and Technology Planning Project (NO. Z171100004717001).

8. REFERENCES

[1] S. Tu and Y. Huang, Towards efficient and secure access control system for mobile cloud computing, *China Communications*, Vol. 12, No. 12, pp. 43-52, December 2015.

https://doi.org/10.1109/CC.2015.7385527

[2] S. Tu, S. Niu and H. Li, A fine-grained access control and revocation scheme on clouds, *Concurrency & Computation Practice & Experience*, Vol. 28, pp. 1697–1714, May 2016.

https://doi.org/10.1002/cpe.2956

[3] S. Niu, S. Tu and Y. Huang, An Effective and Secure Access Control System Scheme in the Cloud, *Chinese Journal of Electronics*, Vol. 24, No. 3, pp. 524-528, Jul 2015.

https://doi.org/10.1049/cje.2015.07.015

[4] R.K.L. Ko et al., TrustCloud: A Framework for Accountability and Trust in Cloud Computing, *2011 IEEE World Congress on Services*, Washington DC, pp. 584-588, 2011.

[5] R.K.L. Ko, M. Kirchberg and B.S. Lee, From System-centric to Data-centric logging - Accountability, Trust and Security in Cloud Computing, in *Defence Science Research Conference and Expo*, 2011.

[6] R.K.L. Ko, P. Jagadpramana and B.S. Lee, Flogger: A File-Centric Logger for Monitoring File Access and Transfers within Cloud Computing Environments, *IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications*, Changsha, pp. 765-771, 2011.

[7] C.H. Suen, R.K.L. Ko, Y.S. Tan, P. Jagadpramana and B.S. Lee, S2Logger: End-to-End Data Tracking Mechanism for Cloud Data Provenance, *2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*, Melbourne, VIC, pp. 594-602, 2013. https://doi.org/10.1109/TrustCom.2013.73

[8] Z. Fu, K. Ren, J. Shu, X. Sun and F. Huang, Enabling Personalized Search over Encrypted Outsourced Data with Efficiency Improvement, in *IEEE Transactions on Parallel and Distributed Systems*, Vol. 27, No. 9, pp. 2546-2559, Sept. 2016.

https://doi.org/10.1109/TPDS.2015.2506573

[9] Z. Fu, X. Sun, Q. Liu, L. Zhou and J. Shu, Achieving Efficient Cloud Search Services-Multi-Keyword Ranked Search over Encrypted Cloud Data Supporting Parallel Computing, *IEICE Transactions on Communications*, Vol. E98.B, No. 1, pp. 190-200, Jan 2015.

https://doi.org/10.1587/transcom.E98.B.190

[10] Z. Xia, X. Wang, X. Sun and Q. Wang, A Secure and Dynamic Multi-Keyword Ranked Search Scheme over Encrypted Cloud Data, in *IEEE Transactions on Parallel and Distributed Systems*, Vol. 27, No. 2, pp. 340-352, Feb. 2016.

https://doi.org/10.1109/TPDS.2015.2401003

[11] R.K.L. Ko, Data Accountability in Cloud Systems, in *Security, Privacy and Trust in Cloud Systems*, Springer-Verlag, pp. 211-238, 2014.

https://doi.org/10.1007/978-3-642-38586-5_7

[12] H. Chen, S. Tu, C. Zhao and Y. Huang, Provenance cloud security auditing system based on log analysis, *2016 IEEE International Conference of Online Analysis and Computing Science* (ICOACS), pp. 155-159, 2016. https://doi.org/10.1109/ICOACS.2016.7563069

[13] U.M. Fayyad, G.P. Shapiro and P. Smith, *Advances in Knowledge Discovery and Data Mining*, AAAI/MIT Press, pp. 154-161, 1996.

[14] W. Zhou, M. Sherr, T. Tao, X. Li, B.T. Loo and Y. Mao, Efficient querying and maintenance of network provenance at internet-scale, *Proc. International Conference on Management of Data* (SIGMOD 2010), ACM, pp. 615-626, 2010.

https://doi.org/10.1145/1807167.1807234

[15] R. Agrawal, T. Imielinski and A. Swami, Mining Associations between Sets of Items in Massive Databases, *Proc. of the ACM-SIGMOD 1993 Int'l Conference on Management of Data*, Washington DC, 1993. https://doi.org/10.1145/170035.170072

[16] J.W. Han, J. Pei, and Y.W. Yin et al. Mining frequent patterns without candidate generation, *Data Mining and Knowledge Discovery*, Vol. 8, No. 1, pp. 53-87, 2004.

https://doi.org/10.1023/B:DAMI.0000005258.31418.83

[17] R. Agrawal and R. Srikant, Mining Sequential Patterns, *Proc. Int. Conf. Data Engineering* (ICDE' 95), Taipei: *IEEE Computer Society*, pp. 3-14, 1995.

[18] R. Agrawal and R. Srikant, Mining Sequential Patterns: Generalizations and Performance Improvements, *Proc. 5th Int. Conf. Extending Database Technology* (EDBT), Avignon: *Lecture Notes in Computer Science*, pp. 3-17, 1996.

[19] M. El-Hajj, Osmar. R and Za iane, Parallel leap: large-scale maximal pattern mining in a distributed environment, *Proceedings of the 12th International Conference on Parallel and Distributed Systems*, Washington DC, USA: IEEE, pp. 135-142, 2006.

https://doi.org/10.1109/ICPADS.2006.77

[20] J. Pei, J.W. Han and Mortazavi-Asl B et al. PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-ProjectedPattern Growth, *Proc. of 2001 Int. Conf. on Data Engineering*, Heidelberg, Germany, pp. 215–224, 2001.

[21] R. Agrawal et al. Fast similarity search in the presence of noise, scaling, and translation in time series databases, in *Proc. 21st Int. Conf. Very Large Data Bases*, pp. 490-501, 1995.

[22] R.K.L. Ko, P. Jagadpramana, M. Mowbray, S. Pearson, M. Kirchberg, Q.H. Liang and B.S. Lee, TrustCloud: A Framework for Accountability and Trust in Cloud Computing, *2nd IEEE Cloud Forum for Practitioners* (IEEE ICFP 2011), Washington DC, USA, July 7-8, 2011.

[23] Z. Xia, X. Wang, L. Zhang, Z. Qin, X. Sun and K. Ren, A Privacy-preserving and Copy-deterrence Content-based Image Retrieval Scheme in Cloud Computing, *IEEE Transactions on Information Forensics and Security*, 2016.

https://doi.org/10.1109/TIFS.2016.2590944

[24] Y. Ren, J. Shen, J. Wang, J. Han and S. Lee, Mutual Verifiable Provable Data Auditing in Public Cloud Storage, *Journal of Internet Technology*, Vol. 16, No. 2, pp. 317-323, 2015.