Tackling Insider Threat in Cloud Relational Databases

Qussai Yaseen
Department of Computer Science
Yarmouk University
Irbid, Jordan
qussai.yaseen@gmail.com

Brajendra Panda
Department of Computer Science and Engineering
University of Arkansas
Fayetteville, AR, USA 72701
bpanda@uark.edu

Abstract—Cloud security is one of the major issues that worry individuals and organizations about cloud computing. Therefore, defending cloud systems against attacks such as insiders’ attacks has become a key demand. This paper investigates insider threat in cloud relational database systems (cloud RDBMS). It discusses some vulnerabilities in cloud computing structures that may enable insiders to launch attacks, and shows how load balancing across multiple availability zones may facilitate insider threat. To prevent such a threat, the paper suggests three models, which are Peer-to-Peer model, Centralized model and Mobile-Knowledgebases model, and addresses the conditions under which they work well.

Keywords—cloud computing; databases; insider threat; security;

I. INTRODUCTION

Security is one of the major concerns when moving to cloud. Proving the security of data in cloud is crucial to achieve users’ trust on cloud providers. Insider threat is one of the problems that worry organizations and individuals about cloud computing. Moving data to cloud increases the number of insiders, which may increase insider threat. Moreover, preventing data in the cloud from organizations’ insiders may require new methodologies.

According to different surveys, such as Forrester Research [1] and E-Crime watch survey [2], insider threat poses more harm than that posed by outsiders’ threat. Nonetheless, little research has been performed on insider threat. Knowledgebase is a serious source for insider threat. Researchers in [3][4] have addressed this problem and proposed solutions to mitigate it. However, cloud databases technology has new vulnerabilities that may enable insiders to breach existing solutions and launch attacks using their knowledgebases. The contribution of the paper is as follows. It shows how existing insider threat preventing methodologies, which prevent insiders from exploiting their knowledgebases to pose threat, can be avoided by insiders in cloud RDBMS. Moreover, the paper proposes three models that can be used to prevent the threat of knowledgebases in cloud RDBMS, and it addresses the conditions under which they can be used effectively.

The rest of the paper is organized as follows. The next section introduces some related work. Section 3 discusses the problem. Section 4 introduces the proposed models to solve the problem. Finally, section 6 concludes the work.

II. BACKGROUND AND RELATED WORK

Cloud computing is a promising technology that offers large-scale and on-demand computing infrastructure. Achieving low cost and high availability live migration, which are discussed in [5][6], are main goals of the research on cloud computing. Assuring cloud security is necessary to achieve the trust of customers since there are many security concerns arising with this new technology. Some security concerns in cloud computing are discussed in [7][8]. Research in cloud databases is still in its early stages. Few papers have been published in this field such as Relational Cloud [9] and CloudDB [10]. The cloud relational database service has been introduced by some cloud providers such as Amazon (Amazon RDS) [11] and Microsoft (Microsoft SQL Azure) [12]. The guarantee of protecting data that resides on the cloud from the threat of cloud providers’ employees is a major requirement by customers. Encryption is one of the methods suggested to protect data. For instance, CryptDB, Homomorphic Encryption (HOM) and Encryption Deterministic (DET) are encryption methods that can execute the operations of relational databases queries on encrypted data [9].

Most research that has been performed on insider threat focused on system level such as the work by Spitzner [13]. Very little research has dealt with insider threat in relational databases such as [3]. To the best of our knowledge, there is no concrete research that has been performed on insider threat in cloud relational databases (cloud RDBMS).

III. INSIDER THREAT

The knowledgebases of insiders and dependencies among data items play a major role in insider threat in relational databases. The knowledgebase of an insider represents the history of accesses to data items (and their values) by the insider. These data items might be risky since they can be combined with other data items using dependencies to infer sensitive information [4]. For instance, suppose that an insider has the (Name, Rank) of an academic staff (say Alice) in his/her knowledgebase. In this case, if the insider gets the information (Rank, Salary), s/he can infer Alice’s salary. This information (Alice, Salary) could be unauthorized information and a threat. We should mention here that relationships among Name, Rank, and Salary attributes are common knowledge. However, there exist many such dependencies that are not this obvious. For simplicity, we have used (Name, Rank, Salary) example in this paper.

Building and checking knowledgebase of an insider is not a hard problem in relational databases that are local in nature. However, building knowledgebase in cloud relational databases and using it needs new methodologies since cloud databases are replicated at different sites to assure availability and reliability. Cloud providers store data
in multiple datacenters that are both geographically and logically separated. To achieve greater performance and fault tolerance, an application’s traffic may be distributed across multiple availability zones and data centers. The former technique is called elastic load balancing. Cloud relational databases are fragmented and replicated to increase availability and reliability of the databases. Workloads on replicas’ nodes are balanced using live migration, where tenants (applications) are migrated from overloaded nodes to idle (or low-loaded) nodes to achieve load balancing. Users have no control on choosing the location or the instance that they prefer. Thus, different user’s requests may be executed on different instances in the same availability zone or in different availability zones or data centers.

Replication and load balancing may increase the probability of insider threat, which arises when a cloud relational database system fails to use the knowledgebase of insiders to detect threats. In other words, an insider may combine data items s/he gets from database instances in different availability zones to launch attacks, which is shown in Fig. 1. The insider accesses the data item \(D_1\) in the availability zone \(I(AZ_1)\) (that may have the knowledgebase of the insider) and then accesses \(D_2\) in the availability zone \(n (AZ_n)\) (that may not have the insider’s knowledgebase). In this case, the system on the availability zone \(n\) fails to detect this threat and allows the insider to access \(D_2\). Thus, the insider combines the two data items and gets the sensitive information \(S_1\), which is a threat. In addition, this problem may occur when insiders travel and their transactions are sent to different availability zones. We should mention here that to the best of our knowledge no research has discussed the threat of knowledgebase in cloud environment, and no research has discussed how to manage knowledgebases in this new environment.

![Figure 1. Insider Threat in Cloud RDBMS](Image)

In light of the previous discussion, an up-to-date knowledgebase of an insider should be checked at each access request by the insider to prevent threats. Furthermore, the knowledgebase should be updated after each access the insider performs. Thus, cloud RDBMS need new methodologies to build, store and synchronize knowledgebase in cloud environments since local knowledgebase are no longer suitable.

IV. MODELS FOR MITIGATING INSIDER THREAT

Securing cloud RDBMS against insider threat needs a methodology that monitors the activities of insiders in different instances and locations of cloud relational databases. The knowledgebases of insiders should be monitored and synchronized to achieve this purpose. In traditional RDBMS, building, maintaining, and checking knowledgebases are the responsibilities of organizations (owners). Nonetheless, when moving to the cloud, these operations are transferred to cloud providers (Cloud RDBMS). Keeping these responsibilities for local systems when moving to a cloud violates the concept of cloud computing. Moreover, keeping the knowledgebase of an insider in local storage needs transferring it with every access by the insider, which is infeasible due to the network overhead that it requires especially when knowledgebases get large. This section introduces three frameworks to maintain knowledgebases in a Cloud RDBMS, and demonstrates the features and limitations of each one.

A. Peer-to-Peer Model

In this model, the knowledgebase of each insider is built and stored in all availability zones. At each access of an insider to a data item in an availability zone, the knowledgebase of the insider in the availability zone is updated. Next, the updates are sent to all other availability zones and data centers simultaneously to keep knowledgebases consistent. Transactions are monitored locally at each availability zone or database instance. Thus, insider threat monitoring is performed locally without a need to communicate with other nodes. Fig. 2 shows the proposed framework, where \(AZ\) denotes an availability zone, \(LB\) denotes load balancing and \(U(KBs)\) denotes updating of knowledgebases.

![Figure 2. Peer-to-Peer Model](Image)

As shown in figure 2, an insider sends his/her query to a cloud RDBMS. The cloud system sends the query to the closest availability zone. If the availability zone has a high load, the query is transferred to another availability zone. In both cases, the insider’s knowledgebase is checked to ensure that there is no threat. Once the query is executed, the knowledgebase of the insider is updated to reflect the access and the knowledgebases in all other availability zones are updated as well.

A key benefit of this model is that there is no single point of failure. Moreover, transactions and threat detection procedure are executed fast since all processing are performed within a single availability zone and no communications are needed with other parts of the cloud system. Furthermore, the processing needed for manipulating knowledgebases are distributed among all availability zones, which balances the load on them.
The challenge that arises when using this model is the profiling of activities (building knowledgebases) for each insider. Local profiling processes transactions faster, but it introduces synchronization problems. Knowledgebases at all database servers should be updated simultaneously. Keeping knowledgebases updated needs a lot of immediate processing, which is time and resources consuming, and it causes delays in processing transactions. In summary, the use of this model results in high network traffic and delays processing of transactions especially in the case of large number of replicas. Therefore, this approach is suitable when the number of instances is small.

B. Centralized Model

This model uses a coordinator site that builds, stores, and manages the knowledgebases of all insiders. Moreover, an insider’s query is sent to the coordinator first. Then, the coordinator checks the query against insider threat using the insider’s knowledgebase that it has. If no threat exists, the coordinator sends the query to one of the cloud RDBMS nodes with taking into account the load balancing. After executing the query successfully, the cloud RDBMS sends back an acknowledgement to the coordinator so that it updates the knowledgebase of the insider. The model in this state has a bottleneck. That is, failure of the coordinator turns off the entire system of insider threat prediction and prevention. Fig. 3 shows a modified model. The modified model uses a secondary coordinator to mitigate the bottleneck problem, which is similar to the idea used in damage recovery in distributed systems by Panda and Zue [14]. The secondary coordinator is used only in case of failure. However, the secondary knowledgebase should be updated to keep both knowledgebases consistent as shown in Fig. 3, where \( U(KBs) \) indicates updating the knowledgebases, \( LB \) indicates load balancing, and “Ackn.” denotes an acknowledgement.

Advantages of this model include the relatively small amount of network traffic in comparison to the previous model. Thus, this model is more scalable than Peer-to-Peer model. Moreover, the synchronization is performed between the instances of knowledgebases only (the primary and secondary sites). That is, there is no delay occurring due to the synchronizing process between knowledgebases instances. However, the delay may happen because all requests are inspected and filtered at the central unit. Therefore, the central unit should be equipped with high performance capabilities to serve this job.

C. Mobile-Knowledgebases Model

This model has the advantages of Peer-to-Peer model, and mitigates the associated disadvantages. In this model, an availability zone at a datacenter stores the knowledgebases of insiders who are geographically close to it, instead of storing the knowledgebases of all insiders. For example, Fig. 4 shows how knowledgebases of insiders in the USA may be stored, where Arkansas insiders’ knowledgebases can be stored in availability zone 4 and Washington insiders’ knowledgebases can be stored in availability zone 1. Hence, the availability zones may belong to different data centers. This model depends on the assumption that insiders are highly probably performing most of their work in one location. However, an insider may perform his/her work from different (geographically) locations, which is a key-advantage of cloud computing. In this case, the cloud system should send a copy of the knowledgebase of the insider to the new location so that it can check his/her queries against insider threat. In the figure, \( Send KBs \) stands for sending a copy of a knowledgebase of an insider, which may be needed when balancing a load or when an insider accesses an availability zone that does not have his/her knowledgebase.

To show how the model works, suppose that an insider, say Bob, works for a company in Arkansas, which belongs to availability zone 4 (AZ4). Assume that Bob travelled to Washington, which belongs to availability zone 1 (AZ1), and s/he wants to access the database in the cloud. Fig. 5 shows how the model works in this case. Bob sends his query to the cloud system, which forwards his request to AZ1. The cloud system in this availability zone checks whether Bob’s knowledgebase exists or not. Since the knowledgebase is not available, the cloud system in AZ1 contacts other availability zones asking for the
knowledgebase of Bob. Availability zone 4, which has the knowledgebase, sends a copy of Bob’s knowledgebase to AZ1. Then, the cloud system in AZ1 checks whether there is a threat posed by Bob. If there is no threat, AZ1 executes Bob’s request and sends the updates on the Bob’s knowledgebase to AZ4.

Mobile-Knowledgebases model eliminates the need to store knowledgebases of all insiders at every availability zone as in Peer-to-Peer model. Moreover, it causes less traffic than Peer-to-Peer model since updates of knowledgebases are sent to host availability zones only in case of "moving" insiders. Thus, Mobile-Knowledgebases model is more scalable than Peer-to-Peer model. Furthermore, a failure of an availability zone only affects the tasks of other insiders at other availability zones, which means it does not cause a bottleneck as in the Centralized model (more reliable). Furthermore, in most cases, the model needs to process the transactions and manage the knowledgebases of a few insiders only, which means it has less processing overhead than other models.

This model can be optimized more in order to eliminate the need to send messages to all availability zones (AZs) searching for the knowledgebase of a “moving insider”. This can be achieved by storing a directory for all insiders at an organization and their hosting AZs. Thus, when an insider’s request is sent to an AZ other than his/her hosting one, the cloud system at the new AZ looks up the directory to retrieve the insider’s hosting AZ. Then, a message is sent to the latter AZ only to retrieve the knowledgebase of the insider. Storing the directory of all insiders needs more storage space, but it greatly reduces the traffic overhead, especially when the number of AZs and the number of datacenters become larger.

V. CONCLUSION

This paper has discussed insider threat in cloud relational databases. It has shown how load balancing across availability zones and data centers may enable insiders to launch attacks. To prevent such threat, the paper has demonstrated new insider threat prediction and prevention models that are suitable for the cloud environment, which are Peer-to-Peer model, Centralized model and Mobile-Knowledgebases model. Furthermore, it has addressed the conditions under which the models can work with optimal performance, and has presented the advantages and disadvantages of each model with regard to processing time, network traffic and overall cloud RDBMS performance. In our future work, we plan to conduct experiments to establish the effectiveness of the proposed models, and measure and compare the overhead (processing time and network traffic) that the models may add to the cloud RDBMS.

ACKNOWLEDGMENT

This work has been supported in part by US AFOSR under grant FA 9550-08-1-0255. We are thankful to Dr. Robert L. HerKlotz for his support, which made this work possible.

REFERENCES