Mitigating Insider Threats on the Edge: A Knowledgebase Approach

Qutaibah Althebyan[1,2]
[1]College of Engineering, Al Ain University, UAE
[2]Software Engineering Department, Jordan University of Science and Technology, Jordan

Abstract: *Insider Threats, who are cloud internal users, cause very serious problems, which in terns, leads to devastating attacks for both individuals and organizations. Although, most of the attentions, in the real world, is for the outsider attacks, however, the most damaging attacks come from the Insiders. In cloud computing, the problem becomes worst in which the number of insiders are maximized and hence, the amount of data that can be breached and disclosed is also maximized. Consequently, insiders' threats in the cloud ought to be one of the top most issues that should be handled and settled. Classical solutions to defend against insiders' threats might fail short as it is not easy to track both activities of the insiders as well as the amount of knowledge an insider can accumulate through his/her privileged accesses. Such accumulated knowledge can be used to disclose critical information –which the insider is not privileged to- through expected dependencies that exist among different data items that reside in one or more nodes of the cloud. This paper provides a solution that suits well the specialized nature of the above mentioned problem. This solution takes advantage of knowledge bases by tracking accumulated knowledge of insiders through building Knowledge Graphs (KGs) for each insider. It also takes advantage of Mobile Edge Computing (MEC) by building a fog layer where a mitigation unit -resides on the edge- takes care of the insiders threats in a place that is as close as possible to the place where insiders reside. As a consequence, this gives continuous reactions to the insiders' threats in real-time, and at the same time, lessens the overhead in the cloud. The MEC model to be presented in this paper utilizes a knowledgebase approach where insiders' knowledge is tracked and modeled. In case an insider knowledge accumulates to a level that is expected to cause some potential disclosure of private data, an alarm will be raised so that expected actions should be taken to mitigate this risk. The knowledgebase approach involves generating Knowledge Graphs (KGs), Dependency Graphs (DGs) where a Threat Prediction Value (TPV) is evaluated to estimate the risk upon which alarms for potential disclosure are raised. Experimental analysis has been conducted using CloudExp simulator where the results have shown the ability of the proposed model to raise alarms for potential risks from insiders in a real time fashion with accurate precision.*

Keywords: *Insider Threats, Fog, Mobile Edge, Cloud, Knowledge Graph, Dependency Graph, Database.*

Received February 29, 2020; accepted June 9, 2020
https://doi.org/10.34028/iajit/17/4A/6

1. Introduction

Cloud computing has an open environment that is distributed among different areas. In cloud computing high quality services are offered. These services are guaranteed to be efficient and, at the same time, have a considerably reduced cost. Many resources are shared within the cloud computing architecture via a resource repository with heterogeneous and diverse formats and shapes. These heterogeneous resources are utilized within the cloud model in an exceedingly decentralized fashion to produce services for customers supporting their needs and specifications [1, 10, 11, 12]. Some of these services are applications' services while others are hardware services among others [4]. These Software and Hardware resources are governed by cloud suppliers and leased to customers based on different customers' requests. Observance, provisioning, de-provisioning, are some examples of the governance and management of the cloud resources [6]. Among the several issues that should be governed and managed is the cloud security and privacy issues. Cloud services provided are usually categorized as Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS) as highlighted in the following Figure 1.

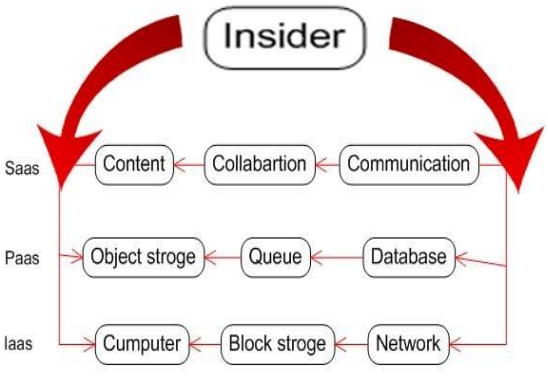

Figure 1. An Insider at several cloud computing components.

Security of cloud computing [13] has become a critical issue that needs to be handled and maintained.

In fact, it is a critical issue for organizations as well as for individual users. Organizations should concentrate on the security of their resources to maintain their survivability as well as users' trust. On the other side, individual users of the cloud want to maintain their private data uncovered and secure. In fact, disclosure of cloud users' data can happen either from cloud outsiders or it can be from (probably malicious) insiders (cloud agents) of the cloud. Hence, insiders' threats of the cloud need to be of great importance for the whole cloud community. Cloud properties such as Multi-tenancy where more than one version of cloud resources or even cloud resources might be partitioned and run at different tenants could be a vulnerability that might be exploited either by insiders or by outsiders.

It is important mention that securing the cloud data and the users' privacy is one of the most important issues that should not be ignored. Controlling activities of Cloud users (who are considered as Insiders to the cloud) should be governed and monitored. Several rules and procedures should be presented to ensure that no malicious actions and requested will be granted. This in fact impose that all activities of individuals especially the insiders should be recorded, monitored and evaluated so that all malicious activities are denied. For example, an insider of a cloud might maliciously request shared private data of another insider [15, 16]. This might lead to breach the second insider security in case this request is granted. In literature, many ways are being used to ensure the security of cloud resources. Examples of these are not limited to Crypt DB, Homomorphic Encryption, and Encryption Deterministic [7]. Although the mentioned methods are used to limit cloud providers' workers from breaching customer' private data, however, they provide no help in case of breaches that come from insiders with valid cloud privileges [5].

This paper proposes a Mobile Edge Computing (MEC) [10, 11,13] solution. This MEC solution needs to monitor insiders' activities at his/her site. This means that this monitoring facility needs to be as close to him/her as possible where a MEC monitoring agent will be in the insider site. A higher MEC agent that coordinates all MEC agents will be used for higher kind of decisions that are usually for work organization and management. Moreover, it is important to mention that this paper is an extended version of the paper published in [2].

2. Related Work

Insider Threat is considered as a crucial security issue that needs to be carefully handled because of its tremendous sever consequences and defects. Many researchers defined insiders at the system level. For example, Althebyan and Panda [3] described the insider as "an individual who has the knowledge of the organization's information system structure to which he/she has authorized access and who knows the underlying network topologies of the organization's information systems". Yaseen and Panda [17] defined the insider as "an individual who has right to utilize benefits, is acquainted with conditions and their requirements and is acquainted with the framework under thought".

Researchers used existing frameworks for recognizing outside hazard. For example, Duncan *et al*. [8] used honey pots to detect insiders' dangers. Such frameworks do not gain accurate results since insiders utilize various ways or strategies to ambush their system resources utilizing their definite aggregated knowledge of their underlying systems structure and details. In various terms, insiders use their privileged knowledge to find out details about structures to their systems and hence, the sensitive data using ways that are hard to be recognized by security frameworks. Other researchers, for instance, Althebyan and Panda [3] presented new frameworks to deal with this insider threat issue by building a model that is solely worked to deal with the insider threat issue at the system level. In their model they presented a knowledgebase model to manage notifications that leads to prediction of and foreseeing the insiders' misuse and consequently forestall any malignant exercises that can be presented by any insider of the underlying system.

A few Researchers contemplated the insider threat problem issue at the cloud level, in particular, the authors of [8, 14] featured insider ambushes inside a distributed cloud environment. They re-described insiders "malignant insiders" through the setting of the cloud environment and the outfitting samples to raise an alarm of the severity of malicious insiders. They tended to a specific burden named as APT "created unending dangers", they likewise plotted the bother like "Adept" as an uncommon essential issue. If the assailant can persuade option to use the host operating system or the hypervisor, they would induce all through most of the underlying machines on the server and immediately pass the abnormality to the hypervisor.

3. MEC Prposed Model

Insiders can use their accumulated knowledge through their valid privileges to get access to cloud resources that they are not privileged to. Combining this accumulated knowledge with the fact that cloud resources might have several versions at several tenants of the cloud pose a significant risk of cloud resources. This risk should be dealt in a specialized manner that suites the nature of the cloud and the insider threats in a timely manner and at the same time reduce the overhead on the cloud. Therefore, an enhanced solution to mitigate the insider threat problem in the cloud is introduced. Such solution is guaranteed to be very close to the place where insiders reside, gives real-time responses to attack, and reduces the overhead on the

cloud. In fact, the MEC solution is guaranteed to provide the best solution as it has the following advantages over classical cloud-based solution:

1. Provides a better solution than cloud solutions since it provides real time responses for attacks as the MEC layer resides in the insider side
2. Reduces the overhead on different cloud layers
3. Provides good and reliable solutions for short distance travels.

In this paper, we use a very well-known networks architecture for cloud data center which is the Fat Tree Network (FAT) architecture as it is illustrated in Figure 2.

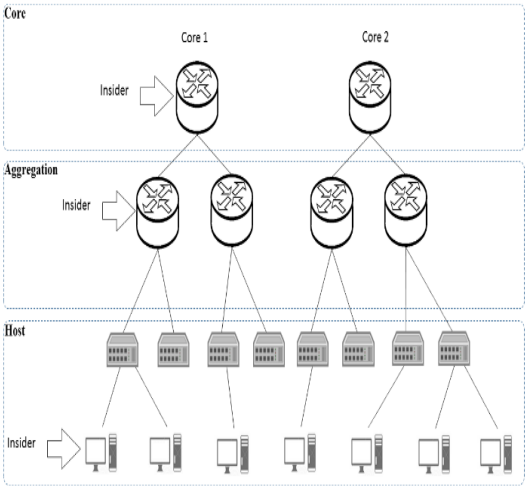

Figure 2. Cloud architecture as FAT architecture.

In the insider threat context, we adapted the FAT model to produce an updated version of the cloud data center that fits our MEC model specification. In fact, we can differentiate three different layers as shown and labeled on the Figure 3:

1. Cloud Layer: This layer contains a directory of all insiders. This layer also enforces implementation of Insider Threats Policies on the fog nodes in case a request is received from any fog nodes in the Fog Layer.
2. Fog Layer: In this layer, several fog nodes exist to enforce Insider Threats Policies. Each fog node contains an insider threat decision unit, a knowledgebase of its associated insiders, and a RDBMS.
3. Insider Layer: In this layer different cloud insiders exist. These insider performs their daily requests which will be handled either locally or will be forwarded to the Fog layer in case requests to data exist in other clouds are requested.

As it is just mentioned in the fog layer, insiders' knowledge should be maintained and monitored. This in fact, will be monitored through a knowledgebase component for each insider. The following section shows how the insiders' knowledge is accumulated and maintained.

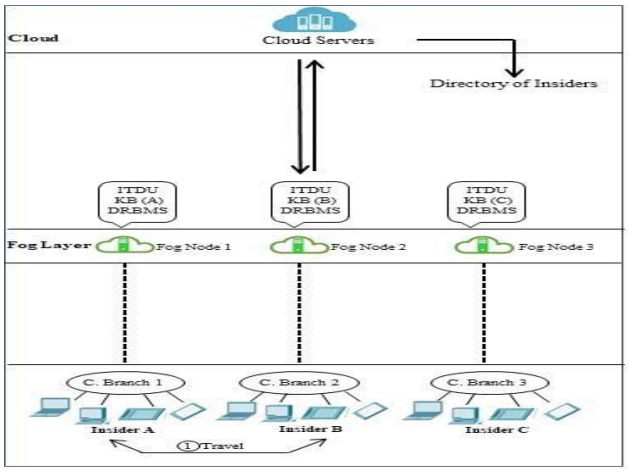

Figure 3. MEC model as fat.

3.1. Insiders' Knowledge

Insiders accumulate their knowledge through their privileged accesses suing their valid and privileged accounts. This insider's accumulated knowledge is represented in a knowledge graph. In our sake of building such knowledge graphs for each insider, we distinguish the following types of accumulated knowledge [17, 18]:

- Inferred Knowledge: This kind of knowledge can be defined as: "Given a dependency relevance A→B in Cloud database system D such that A and B are objects in D, then the information that an insider deduces about B via getting access to A, is referred to as inferred knowledge".
- Computed knowledge: This kind of knowledge can be defined as: "Given a dependency relevance A→B in a Cloud database D such that A and B are items in D, then the information an insider gains about B via computation the usage of A, which he/she has accessed, is referred to as computed Knowledge".
- Aggregated knowledge: This kind of knowledge can be defined as: "Given two related data items A and B in a Cloud database D, the knowledge accomplished with the aid of combing A and B collectively is known as aggregated Knowledge".

In Figure 4 a knowledge graph of the accumulated knowledge of an insider is presented. For more details about the knowledge graph and all kinds of knowledge units of an insider readers can refer to [3, 18].

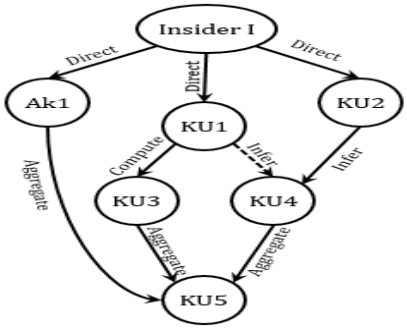

Figure 4. Insider knowledge graph.

In fact, algorithm 1 used in building insiders' knowledge graphs can be relaxed to reflect each insider level. Hence, one algorithm can be extracted for building knowledge graph for a Host insider, a second algorithm can be extracted for building knowledge graph for an aggregate insider and a third algorithm can be extracted for building knowledge graph for a Core insider. Although the differences are very slight among them, however, we only list the algorithm for building the insiders' knowledge at the host level as it is the one of concentration.

Algorithm 1: Insiders Knowledge Graph KG

Abbreviations: V: Vertex; E: Edge; H: Host Insider; A: Attribute; T: Table
Input: H, Dependency Graph,
Neural Dependency Graph, Hosts.
Output: Knowledge graph of the H.
Knowledge graph is initialized KG = (V, E),
Where V = H, E = { }.
for each H, T in H do{
V = v ∪ T
E = e ∪ e(H, T)
}
for each A ϵ T do{
V = v ∪ A
E = e ∪ e (T, A)
}
for each A ϵ T do{
V = v ∪ A
E = e ∪ e (T, A)
}
for each T in H is transitive dependent for all A{
V = v ∪ T
E = e ∪ e (H, T)
}
for each A ϵ T with transitive dependent{
V = v ∪ A
E = e ∪ e (T, A)
}
for each T in H is aggregated dependent in T{
V = v ∪ T
E = e ∪ e (H, T)
}
for each A ϵ T is aggregated dependent in A{
V = v ∪ A
E = e ∪ e (T, A)
}
for each E E(T, A) do{
Weight E (H, T)
Weight E (T, A)/n
}

3.2. Dependency Graph

When an insider in requests a read to any data item in the cloud our model starts evaluating the requesting as it will be discussed later. This in fact will estimate the risk of granting the requested data item and hence subsequently will decide whether granting this request might impose any risk on the system by this insider. In case that granting this request might impose any risk on the system, the request will either be grated or highlighted as risky. In case it is highlighted as risky, more monitoring and investigation will be performed to the insider's future actions. In this evaluation process, a new graph called Dependency Graph (DG) is built. The DG graph shows dependencies among objects in the system, the amount of knowledge that one object might infer about other objects. The DG also shows the amount of knowledge a group of objects might infer about other objects [17, 18]. Figure 5 shows a snapshot of a DG graph taken at a specific time after an insider makes an access request.

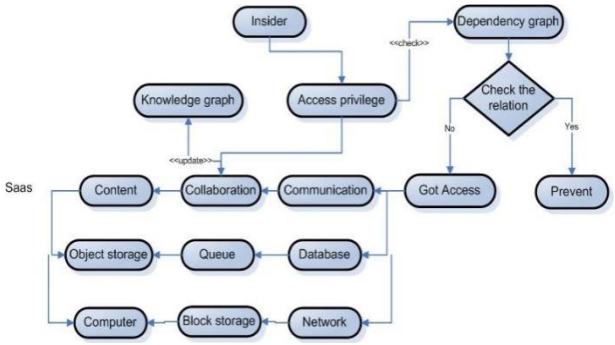

Figure 5. KG unit and DG unit of an insider.

3.3. Mitigating Insiders' Malicious Activities at Host Level

When an insider requests a read to any data item in the cloud our MEC model starts evaluating the requesting insider knowledge level. This in fact will estimate the risk of granting the requested data item and hence subsequently will decide whether granting this request might impose any risk on the system by this insider. In case that granting this request might impose any risk on the system, the request will either be grated or highlighted as risky. In case it is highlighted as risky, more monitoring and investigation will be performed to the insider's future actions after informing the insider with these actions so that this insider, if has some bad intension, will be very careful in doing his/her action.

As an illustration of our MEC model, suppose that the following scenario is implemented, where it can be traced on Figure 3:

1. An insider A existing in cloud branch 1 travels to another place that is covered by cloud branch 2.
2. Subsequently, fog node 2 asks the cloud server for the hosting fog node of insider A.
3. The cloud server (has a directory of all insiders) replies back to fog node 2 with the hosting fog node of insider A as it is fog node 1.
4. Immediately fog node 2 contacts fog node 1 and requests for knowledge graph of insider A.
5. Fog node 1 sends the knowledge graph (KG) of A, KG(A) to fog node 2.
6. Fog node 2 starts evaluating insider's A new request by starting the Insider Threat Decision Unit (ITDU) and categorizes his/her requests as safe or risky based on the threat prediction value to be calculated.
7. Fog node 2 sends back updated KG(A) to fog node 1 in a timely basis.

A Threat Prediction Value (TPV) obtained from the Threat Prediction Graph (TPG) graph is calculated to measure the risk of an insider requests. The TPV is calculated as follows:

$$TPV(k) = F(k) / T(k) \quad (1)$$

Where $F(k)$ is the amount of information an insider has accumulated about k, and $T(k)$ represents the total amount of information that an insider is authorized to accumulate about k. For more details about the TPG and TPV, readers are encouraged to read the papers [3, 17, 18].

Figure 6 shows a sample instance of a TPG graph for a specific insider. New insider's actions are considered and taken. If the TPV value indicates that a risk equal or greater than the risk threshold value, then the insider's request is either denied or it will be granted, however, with more monitoring for the Host insider. In the case of granting the requested access, subsequent actions of updating both the KG of the insider and the DG (if needed) will be performed to reflect the updates implemented. Algorithm 2 describes the details of actions performed in the mitigation process. In summary, the algorithm summarizes our model where all insiders' activities and requests are monitored, evaluated and either fulfilled or denied. If any insider's request might pose a threat to the underlying cloud data item, then the requested action will be highlighted and an alarm will be raised. Consequently, more evaluation will be performed to this request. If this request is judged to pose a threat, then the request will either be denied and the knowledgebase of the insider will not be updated as well or it was judged to be granted with subsequent monitoring and evaluation.

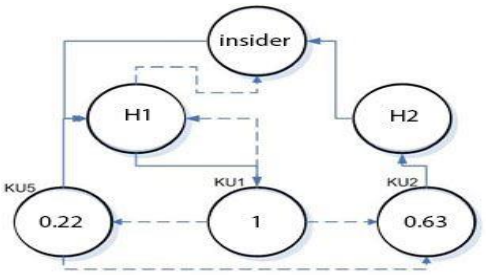

Figure 6. An Instance of a TPG.

Algorithm 2: Preventing Insiders Threats

Abbreviations: V: Vertex; H: Host Insider; E: Edge; TPG: Threat Prediction Graph; T: Threshold Value, NDIG: Neural Dependency and Inference Graph; KU: Knowledge Unit
Input: H, T, NDIG, KG.
Output: TPG of H.
T1 = (KU(1),TKU(1)) is initialized
Initialize KG for H and NDIG
and TPG.
for each KU ∈ V(TPG) do{
TPV1(KU(1)) = F(KU(1))/T(KU(1))
}
for each request KU(1) By H do{
if TPV1(KU(1)) > 1 then{
Deny request
else
V = V ∪ H
E =E ∪ E(H, Host)
}
V =V ∪ (KU(1))
}
TPV1 = f(KU(1))/T(KU(1))
for each KU(1) has dependency
with RKU(1) do{
if TPV1(KU(1)) > 1 then{
Deny KU(1) and remove
}
}

4. Simulation and Experimental Results

To insure the performance of the proposed MEC model, simulation experiments are conducted. In the experiments CloudExp simulator [9] is used. In this simulation, each host has an allocated storage with the same id. Tables 1 and 2 show the specifications of both the physical devices as well as each host where every host has one virtual machine.

Table 1. Physical devices specifications.

ISA	**X86**
Operating system	Linux
Virtual Machine Monitor	Xen
Storage capacity	1 Tera
Memory capacity	8 GB
Network Bandwidth	10 Mbps

Table 2. Simultation specifications.

Host ID	**ID (0- number of hosts)**
Storage Capacity	1 Terabyte
MIPS for Each CPU	1024
Memory Capacity	2 GB
Virtual Machine Scheduler	Space shared
Network Bandwidth	10 Mbps

In the conducted experiments, the proposed model is allowed to run with different number of randomly generated insiders. Each insider is allowed to send random requests to objects that the insider is eligible to access. These requests are also randomly generated. A TPV value for each insider is evaluated based on the accumulated insider knowledge considering the different dependencies that exist among different objects as indicated in the DG graph. Based on TPV values; insiders are categorized as either Normal insiders (where their TPV values are less than the threshold value), or Malicious insiders (where their TPV values exceeds the threshold value).

Figure 7 shows the performance of the proposed model. In this figure different variations for the number of insiders (both normal and malicious) have been considered where the number of malicious insiders is random. As it can be noticed in the figure; when increasing the number of insiders, the system evaluated their behavior and based on their calculated TPV values, alarms were raised. The system denied some of the insiders' activities. As it is expected, the system detected the activities of the malicious insiders and denied them. Normal activities of non-malicious insiders continue without any intervention. As the number of experiments increased, it was noticed that the proposed model saturated and the number of malicious insiders stayed the same without any increase. Hence, increasing the number of experiments do not change the behavior of the system.

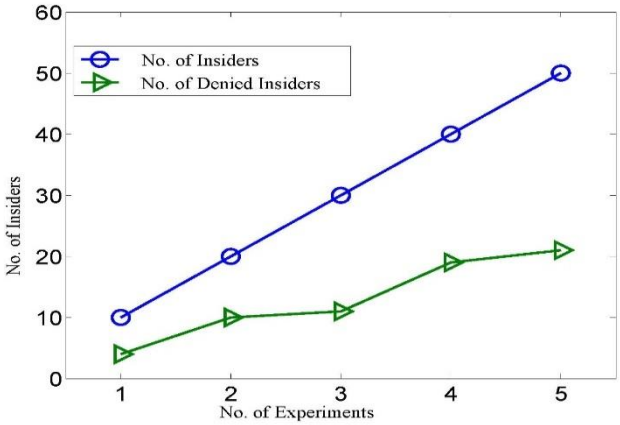

Figure 7. Number of insiders vs. number of blocked malicious insiders.

Figure 8 shows the behavior of the proposed system at a host level. The figure shows that the model detects more insider threats as the number of insiders increases. (detection of more than 95% of the insiders). The detection was immediate. This indicates that most of the malicious insiders do not cause any risk to the system. Figure 8 shows also that the number of false alarms raised is minimized. This gives an indication that our proposed model works with good accuracy. It is important to mention that the number of false alarms will never be completely avoided as the proposed system is a probabilistic system that gives potential risk that, sometimes, might not be a real risk. This small percentage of false alarms is very small and about to be negligible which does not affect the availability of the system from one hand. From another hand, it will not cause a problem at all as the security of system will not be breached since this false negative is for one request for accessing a knowledge unit that constitute only part of the knowledge that can obtained from a critical data item.

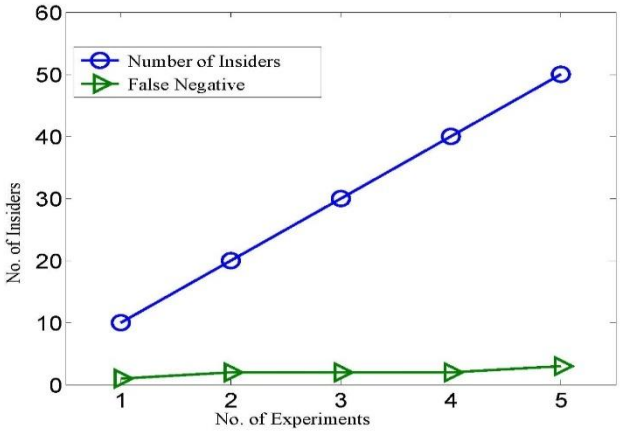

Figure 8. Number of Insiders VS. False Negatives.

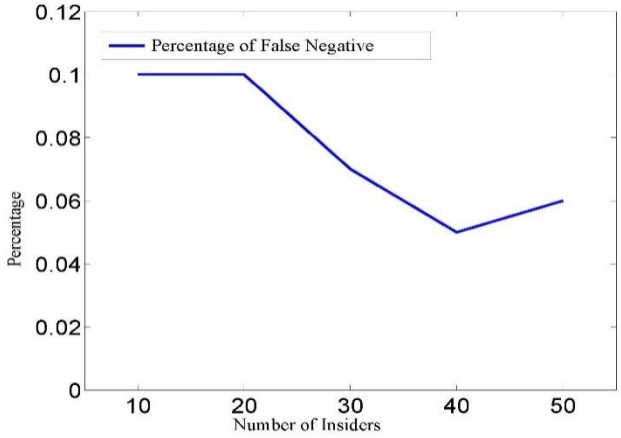

Figure 9. Number of insiders vs. percentage of false negatives.

Figure 9 shows the percentage of false negatives when compared to the number of insiders involved. The figure shows very low percentage of false negatives with small number of insiders. Moreover, the percentage of false negatives tend to be very close to zero as the number of insiders increases. This decrease is due to the fact that with more insiders accessing the system, their requests to access data is increased and hence, their accumulated knowledge reveilles more information about their malicious intensions. This, in fact, is reflected by higher TPV values, and hence, more accurate raised alarms. The figure also shows that the although the system was very effective in catching the insiders' threats, however, the overhead to the system was minimal. Hence, this proves the performance of the proposed model where the performance of the system will not be affected at all.

5. Conclusions and Future Work

This paper presented a model to mitigate insider threats in the cloud. The model utilized a MEC and fog solution. The MEC solution provided promising results for mitigating the insiders' malicious activities. The proposed model is built in a way that suits the special characteristics of both the insiders as well as the cloud environment. In fact, it is built to ensure that the MEC model resides in a place that is very close to the insider. This property gives real time handling of insiders requests and hence assures fast risk evaluation of malicious insider's requests. In the MEC model a knowledgebase approach is used where a knowledge graph is constructed for each insider. A dependency graph DG for different objects of the underlying system is built. The insider's knowledge is evaluated by calculating a threat prediction value TPV that utilized the insider's knowledge graph and the underlying system dependency graph. The calculated TPV values indicates the risk that future insider's value might impose on the system. Hence, the TPV value is evaluated and upon its value consequent decisions were made to insure that the insiders' knowledge will not increase to a point that might cause harm on the system. Several experiments were conducted and results were evaluated which showed that the proposed system gave promising results. For example, after applying the proposed model, the number of blocked insiders was reduced to the minimum which ensured the full accuracy and availability of the system to all insiders with real time responses.

As a future direction, simulation models will be built to compare the proposed MEC knowledgebase model with other cloud computing models to prove the superiority of our proposed model in terms of the accuracy of the results, the better performance, as well as the less overhead of our proposed model on the cloud.

References

[1] Alsaffar A., Hung P., and Huh E., "An Architecture of Thin Client-Edge Computing Collaboration for Data Distribution and Resource Allocation in Cloud," *The International Arab Journal of Information Technology*, vol. 14, no. 6, pp. 842-850, 2017.

[2] Althebyan Q., "A Mobile Edge Mitigation Model for Insider Threats: A Knowledgebase Approach," *in Proceeding of the 20th International Arab Conference on Information Technology*, Al Ain, pp. 188-192, 2019.

[3] Althebyan Q. and Panda B., "A Knowledge-Base Model for Insider Threat Prediction," *in Proceedings of the IEEE Workshop on Information Assurance (IAWâŁ™07)*, New York, pp. 239-246, 2007.

[4] Armbrust M., Fox A., Griffith R., Joseph A., Katz R., Konwinski A., Lee G., Patterson D. Rabkin, A. Stoica I. and Zaharia M., "Above the Clouds: A Berkeley View of Cloud Computing," *Technical Report*, University of California at Berkeley, 2009.

[5] Bertino E., Paci F., Ferrini R., and Shang N., "Privacy-Preserving Digital Identity Management for Cloud Computing," *IEEE Data Eng. Bull*, vol. 32, no. 1, pp. 21-27, 2009.

[6] Boss G., Malladi P., Quan D., Legregni L., and Hall H. "Cloud Computing," *IBM White Paper*, Internet: http://download.boulder.ibm.com/ibmdl/pub/software/dw/wes/hipods/Cloud_computing_wp_final_8Oct.pdf, Last Visited, 2020.

[7] Curino C., Jones E., Popa R., Malviya N., Wu E., Madden S., Balakrishnan H., and Zeldovich N., "Relational Cloud: a Database Service for the Cloud," *in Proceeding of the 5th The Biennial Conference on Innovative Data Systems Research*, USA, pp. 235-240, 2011.

[8] Duncan A., Creese S., and Goldsmith M., "Insider Attacks In Cloud Computing," *in Proceeding of the IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, Liverpool, pp. 857-862, 2012.

[9] Jararweh Y., Jarrah M., kharbutli M., Alshara Z., Alsaleh M., and Al-Ayyoub M., "CloudExp: A Comprehensive Cloud Computing Experimental Framework," *Simulation Modeling Practice and Theory*, vol. 49, pp. 180-192, 2014.

[10] Kashif B., Osman K., Erbada A., and Khan S., "Potentials, Trends, and Prospects in Edge Technologies: Fog, Cloudlet, Mobile Edge, And Micro Data Centers," *Comuter Networks Journal*, vol. 130, pp. 94-120, 2018.

[11] Khan W., Ahmed E., Hakak S., Yaqoob I., and Ahmed A., "Edge Computing: A Survey," *Future Generation Computer Systems Journal*, vol. 97, pp. 219-235, 2019.

[12] Rindos A., Vouk M., and Jararweh Y., "The Virtual Computing Lab (VCL): An Open Source Cloud Computing Solution Designed Specifically for Education and Research," *International Journal of Service Science, Management, Engineering, and Technology*, vol. 5, no. 2, pp. 51-63, 2014.

[13] Roman R., Lopez J., and Mambo M., "Mobile Edge Computing, Fog Et Al.: A Survey and Analysis of Security Threats and Challenges", *Future Generation Computer Sceicne Journal*, vol. 78, pp. 682-698, 2018.

[14] Spitzner L., "Honeypots: Catching the Insider Threat," *in Proceedings of the 19th Annual Computer Security Applications Conference*, Las Vegas, pp. 170, 2003.

[15] Takabi H., Joshi J., and Ahn G., "Security and Privacy Challenges in Cloud Computing Environments," *IEEE Security and Privacy*, vol. 8, no. 6, pp. 24-31, 2010.

[16] Yaseen Q., Althebyan Q., Panda B., and Jararweh Y., "Mitigating Insider Threat in Cloud Relational Databases," *Security and Communication Networks Journal*, vol. 9, no. 10, pp. 1132-1145, 2015.

[17] Yaseen Q. and Panda B., "Predicting and Preventing Insider Threat in Relational Database Systems," *in Prococeedings of the Information Security Theory and Practices. Security and Privacy of Pervasive Systems and Smart Devices*, Passau, pp. 368-383, 2010.

[18] Yaseen Q. and Panda B., "Knowledge Acquisition and Insider Threat Prediction in Relational Database Systems," *in Proceedings of the International Conference on Computational Science and Engineering*, Vancouver, pp. 450-455, 2009.

Qutaibah Althebyan is an associate professor and Dean of College of Engineering at Al Ain University, UAE. He has been there since January 2018. Prior to joining Al Ain University, he was an associate professor in the department of Software Engineering at Jordan University of Science and Technology (JUST) since August of 2008.Dr. Qutaibah Althebyan finished his Ph.D. degree in 2008 in Computer Science from University of Arkansas - Fayetteville and his Master degree in 2004 in Computer Information Systems from the University of Michigan – Dearborn. Dr. Althebyan published several papers in high ranked journals and conferences.He is also a reviewer for many journals and conferences.Dr. Althebyan main research interests are, but not limited to, in information security, database security, security in the cloud, big data management, health information systems, information assurance, software metrics and quality of open-source systems. Lately, he has been working in different security, e-health and software engineering projects, namely; Large Scale Insider Threat Assessments and damage assessment in the cloud in the area of cloud security. Also, studies of Power laws and their effects in object oriented metrics in the area of software engineering.