Find My Trustworthy Fogs: A Fuzzy-based Trust Evaluation Framework

Fatin Hamadah Rahman[a], Thien-Wan Au[a], S.H. Shah Newaz[a,*], Wida Susanty Suhaili[a], Gyu Myoung Lee[b]

[a]*School of Computing and Informatics, Universiti Teknologi Brunei, Jalan Tungku Link, Gadong, BE1410, Brunei Darussalam*
[b]*Faculty of Engineering and Technology Liverpool John Moores University, Liverpool*

Abstract

The growth of IoT is proven with the massive amount of data generated in 2015, and expected to be even more in the years to come. Relying on the cloud to meet the expanding volume, variety, and velocity of data that the IoT generates may not be feasible. In the last two years, fog computing has become a considerably important research topic in an attempt to reduce the burden on cloud and solve cloud's inability to meet the IoT latency requirement. However, fog environment is different than in cloud since fog environment is far more distributed. Due to the dynamic nature of fog, backups such as redundant power supply would deem unnecessary, and relying on just one Internet Service Provider for their fog device would be sufficient. If obstacles arise in this fog environment, factors such as latency, availability or reliability would in turn be unstable. Fogs become harder to trust, and this issue is more complicated and challenging in comparison to the conventional cloud. This implies that trustworthiness in fog is an imperative issue that needs to be addressed. With the help of a broker, managing trust in a distributive environment can be tackled. Acting as an intermediary, a broker helps in facilitating negotiation between two parties. Although the brokering concept has been around for a long time and is widely used in the cloud, it is a new concept in fog computing. As of late, there are several research studies that incorporates broker in fog where these brokers focus towards pricing management. However to the best of our knowledge there is no literature on broker-based trust evaluation in fog service allocation. This is the first work that proposes broker-based trust evaluation framework that focuses on identifying a trustworthy fog to fulfill the user requests. In this paper, fuzzy logic is used as the basis for the evaluation while considering the availability and cost of fog. We propose Request Matching algorithm to identify a user request, and Fuzzy-based Filtering algorithm to match the request with one of the predefined sets created and managed by the broker. In this paper, we present a use case that illustrates how fuzzy logic works in determining the trustworthiness of a fog. Our findings suggest that the algorithms can successfully provide users a trustworthy fog that matches their requirement.

Keywords: Fog computing, Broker, Trust evaluation, Fuzzy logic

1. Introduction

Today's technology advancement allows computer chips to be embedded in everyday things, enabling them to communicate. These things make up the Internet of Things (IoT) where end-users are becoming more sophisticated as seen in the appearance of smart meters, smart watches, and sensor-enabled light bulbs to name just a few. The growth of IoT is proven with the massive 145GB of data generated in 2015, and is expected to increase even more by 2020 with 600GB of data [1]. Relying on the cloud to meet the IoT's expanding volume, variety, and velocity of data [2] may not be feasible. This is where fog computing plays a role in reducing such burden faced in cloud. Fog allows processing to be done locally, therefore all the data generated by IoT does not have to be processed by the cloud.

Fog infrastructure can be dedicated if the fog resource is deployed alongside the existing cloud infrastructure to help content providers expand their services to the edge of the network [3]. On the other hand, a fog can be leased in instances where resources are intentionally deployed for other purposes other than serving as fog. This can be road-side units (RSU) for intelligent traffic system (ITS) [4], home gateway, or smart refrigerators. Aside from that, fogs can be advantageous to mission-critical applications such as in augmented reality (AR) and virtual reality (VR) [5] as illustrated in Fig. 1. Critical delay requirement and the need of 5.2Gbps of network throughput to support VR would indeed be a challenge. Although the fogs in these instances have different objective in using the resources, they contribute in computing some of the tasks in the local area, concurrently reducing the load from the cloud. Moreover, in contrast to cloud that is operated and maintained by technical expert teams, fog can be managed by anyone and requires little to no human intervention.

*Dr. S.H. Shah Newaz is the corresponding author

Email addresses: p20171005@student.utb.edu.bn (Fatin Hamadah Rahman), twan.au@utb.edu.bn (Thien-Wan Au), shah.newaz@utb.edu.bn (S.H. Shah Newaz), wida.suhaili@utb.edu.bn (Wida Susanty Suhaili), g.m.lee@ljmu.ac.uk (Gyu-Myoung Lee)

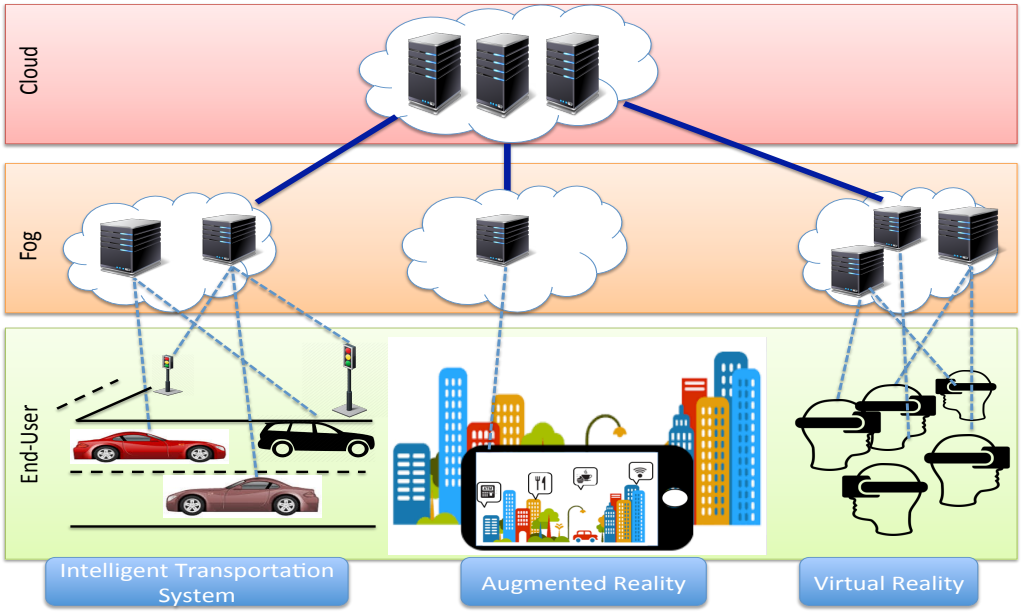


Figure 1: Fog scenarios.

Unlike cloud that can achieve high availability due to its redundancy capability, redundancy is not highly feasible in fog. Relying on just one Internet Service Provider (ISP) for a fog would be sufficient such as the use of fog in home environment, and investing heavily on additional resources in fog such as having extra power supply would deem unnecessary and in fact incur extra cost. This leaves fog at a vulnerable and uncertain state despite its heterogeneous and flexible environment. Fog may not be highly available, considering that they may also be mobile. At any given time, unforeseen circumstances would cause a fog to be jeopardized. For instance, a car serving as fog can be involved in an accident, and drones performing edge services can be malfunctioned. On the other hand, fog may also operate perfectly without disruptions. Hence performance indicators such as throughput, availability or reliability would be difficult to foresee. As this issue becomes more complicated and challenging as compared to cloud, how can we ensure trust in fog? It is difficult to predict and trust fogs with their sporadic characteristic. This implies that a fog's trustworthiness or lack thereof is an imperative issue that needs to be addressed.

While trust has been a research focus in distributed systems as of late, however, managing trust in computing of distributive and ubiquitous nature can be complicated especially without a central entity. Growing research has highlighted the importance of intermediate entities such as reseller, auditor, carrier and broker. We believe that the existence of broker in particular, can help facilitate the trust evaluation process and eradicate the problem in fog. Since trust is very subjective, a fog may be perceived trustworthy for a user but untrustworthy for another user. Often times, the fog that is selected for a user may not satisfy their requests at all. Hence, trust-based broker needs a simple yet effective solution to handle the circumstances and to be able to work dynamically in identifying user priorities in order to cater to their requests. Although the use of brokers in fog are growing, the brokers

are used to facilitate pricing management [6] [7]. To the best of our knowledge, no broker-based trust evaluation system exists in the fog computing literature. In this paper, we propose a trust evaluation framework to evaluate the trustworthiness of fog while taking the users' specific preferences into consideration. We list down our contributions as follows:

- Facilitating trust evaluation by means of a fog broker that takes into consideration availability, quality of service (QoS), security, user feedback, and cost.
- Identifying users sequence of request and their preferred value of the request.
- Filtering approach in selecting fogs by means of fuzzy logic.

The rest of the paper is arranged as follows: Section 2 presents the literature studies on the notion of trust, broker-based trust, and trust evaluation using fuzzy logic. Section 3 elaborates more on the components and workflow of the framework. RM and FTF algorithms are described in this section as well. Our use case will be presented in Section 4, followed by discussions in Section 5, and the conclusion and future works in Section 6.

2. Literature Review

This section discusses the notion of trust in existing literature in Section 2.1. The use of broker in fog domain is studied in Section 2.2. Due to the lack of trust-related literature in fog computing, existing broker-based trust management in cloud is elaborated Section 2.3. Finally the research efforts on trust evaluation using fuzzy logic are discussed in Section 2.4.

2.1. Definition of Trust

Different field has different definitions of trust. Generally, the definition of trust is a willingness of a party to be vulnerable to the actions of another party based on the expectation that the other will perform a particular action important to the trustor, irrespective of the ability to monitor or control that party" [8] and "a psychological state comprising the intention to accept vulnerability based upon positive expectations of the intentions or behavior of another" [9]. The definition is not far aligned in the computer science field as well, as it is defined as the "expectation that a device or system will faithfully behave in a particular manner to fulfil its intended purpose" [10].

2.2. Broker in Fog

The model in [6] has covered the issues of resource prediction, customer type based resource estimation and reservation, advance reservation, and pricing for new and existing IoT customers, on the basis of their characteristics. Meanwhile [7] proposed a system model in fog computing paradigm that endorses an efficient device communication to utilize the full potential of the resourceful IoT nodes. Although both studies have incorporated brokers in their research, the fog-based brokers in their studies are focused towards pricing management.

Both [11] and [12] have included the publish-subscribe pattern in their brokers. A clone brokering system, FogMQ is proposed in [12] to provide a device cloning service where clones are able to self-discover and autonomously migrate to potential cloud hosting platforms to achieve low latency. On the other hand, the fog broker in [11] is responsible for enriching the messages they receive from the lower layers, as well as for task management and allocation. This is achieved by means of the Workload Balancer. Work in [13] proposed an architecture of IoT service delegation and resource allocation based on collaboration between fog and cloud computing. The fog broker is responsible for receiving user request/services and delegating service to other fog environments. It is obvious from these literature that the fog brokers that were used to ease the management were not trust-based.

2.3. Load Balancing

A few studies in the domain of cloud computing have incorporated the use of load balancing in their studies to reduce high utilization and minimize the waiting time. Authors in [14] have studied the effect of different load balancing techniques on data center capability where they stated that higher utilization results in high energy consumption and delay. Study in [15] has proposed a new load balancing algorithm for Infrastructure as a Service (IaaS) cloud with the aim to minimize the waiting time and completion time of the tasks. Meanwhile the work in [16] was inspired by the foraging behavior of honey bee where tasks from a heavy-loaded Virtual Machine (VM) to ones with lesser load.

2.4. Broker-based Trust Management

While there are no studies of trust-based broker in the fog environment, there are such brokers in the cloud domain. The studies of trust in cloud mainly looked at trust between two entities i.e. provider and consumer. The authors in [17] proposed a multipurpose cloud broker system to search cloud resources, at the same time incorporating trust as part of the evaluation, we believe that such two-level broker system would not be suitable in a fog computing environment since having local broker for each of the fogs would increase cost, computation and introduce more latency. In [18], the authors proposed a Compliance-based Multi-dimensional Trust Evaluation System (CMTES) that enables cloud clients to determine the trustworthiness of a cloud service provider from different perspectives. Their study introduced other entities - auditor and agent, apart from the broker in handling trust. However, the existence of multiple entities might complicate the management of trust. Since trust is computed based on these entities' perspective, the failure of one entity in providing data would render inaccurate trust value. In [19], the authors formulated a hybrid model to calculate the trustworthiness of service providers where the broker is located on the client side to ensure the efficiency of the system. Work in [20] proposed a generic architecture for a cloud service broker operating in an Intercloud environment by using the latest cloud standards. The broker aims to find the most suitable cloud provider while satisfying the users service requirements in terms of functional and non-functional Service Level Agreement (SLA) parameters. They have developed the testbed based on the CloudSim simulation toolkit for the broker. However, their study did not look into any particular order of the user's preference.

OPTIMIS [21] is a cloud broker-based mediation layer to deal with complex decision of selecting a trustworthy cloud provider, that fulfills the service requirements, create agreements and also provisions security. The OPTIMIS cloud broker has the capability to operate as cloud service recommendation, intermediation, aggregation, or arbitrage. Although all four modes have different methods in evaluating trust, the authors failed to show how the broker decides which mode it should operate in. The authors only conducted simulation of cloud broker as a cloud service recommendation. In [22], they have proposed a trust management framework for the calculation of both the objective and the subjective trust of a cloud service provider (CSP). The framework is based on a set of trust service providers (TSP), distributed over the clouds, which can elicit raw trust evidence from different sources and in different formats. T-broker [23], a trust-aware service-brokering scheme for efficient matching cloud services (or resources) is proposed to satisfy various user requests. It uses a hybrid and adaptive trust model to compute the overall trust degree of service resources. Although it is a good solution in evaluating trust, the lack of incorporating availability is one of the limitation in their approach.

2.5. Trust Evaluation using Fuzzy Logic

Fuzzy logic is adopted in our study as it is the most widely adopted approach in evaluating trust as applied in several studies [24] [25] [26]. In [26] QoS, reputation, security aspects and social relationships are taken into account in their fuzzy control system. The authors applied a threshold concept that only grants a permit to a subject if it exceeds the trust threshold value. However, the study is done in the IoT concept. Work in [27] have used fuzzy logic to estimate trust value for CSP based on the CSPs performance to aid users in choosing a CSP. However, trust is only evaluated based mostly from the CSPs physical aspect such as number of processors, processors' speed, and memory size. They did not include other relevant metrics that could be used in evaluating trust. Moreover, the combination of Mamdani and Sugeno fuzzy inference systems in their study would deem inconsistent and no apparent advantage is elaborated with the use of both. On the other hand, for work in [28], the trust value for each CSP is based on four basic parameters namely security, availability, cost and performance. However, since the metrics are evaluated separately, it cannot be confirmed that the trust values would correctly represent the exact condition of the CSP compared to if these metrics are evaluated together. Similarly using fuzzy logic, [29] has evaluated trust from the following aspects: performance, reliability, security, price, and reputation. To develop an overall measure of trust value of the cloud providers, [30] uses Seguno fuzzy technique to process four criteria; scalability, availability, security, usability. Unlike the works in [27] and [28], study in [30] has evaluated all the metrics together but explanation of how each of the criteria are quantified is lacking in all of the works mentioned above.

It is evident from the existing literature that many studies have been adopting the brokering concept as a centralized entity in managing trust in the cloud which shows that it is also applicable in fog computing. Similarly, the widespread use of fuzzy logic as means to evaluate trust strengthens our reason to use it in this study. However, the literature from the combined use of both broker and fuzzy logic in managing trust is inadequate.

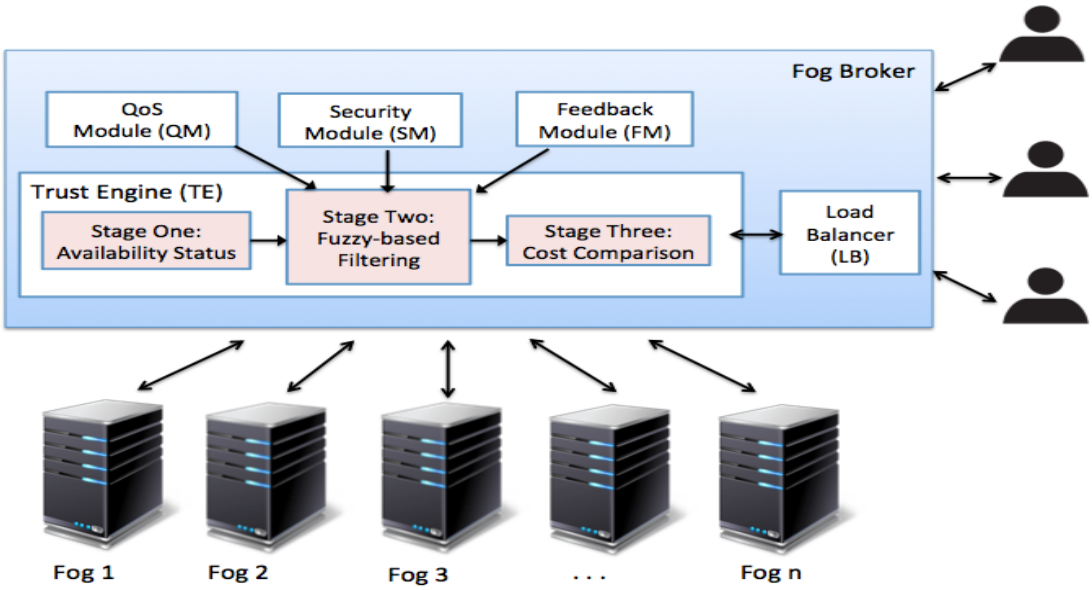


Figure 2: Fog broker.

3. Fuzzy-based Trust Evaluation Filter (FTEF) Framework

This section presents the proposed broker-based trust evaluation framework in fog. Section 3.1 elaborates more on each of the components of the broker. Section 3.2 will explain the working procedure of the framework along with RM and FTF algorithms respectively.

3.1. Components

The broker consists of five components namely Quality of Service Module (QM), Security Module (SM), Feedback Module (FM), Trust Engine (TE) and Load Balancer (LB). Security is the main component of SM. Although trust is beyond the need to ensure security, security is nonetheless chosen as part of our trust evaluation because end users (people) are involved in our study. In such instances, knowing that a fog's security level can provide the assurance or extra push that these people need in order to pursue a fog's service, especially in dealing with transaction of confidential information. In QM, Quality of Service (QoS) can be used to measure the service performance a fog can provide. It is used for our trust evaluation as it helps to justify the reasoning of the broker's decision. The better the performance of the fog, the higher the expectation of the broker towards it. To see whether an expectation is met, feedback of the fogs from the users is crucial. The use of feedback to evaluate trust is also seen in [23], [31], [32]. Therefore, FM is included as part of the evaluation as trust evaluation solely based on the QoS and security is insufficient. In our proposal, the QM, SM and FM collect the updates from the fogs periodically. Meanwhile, TE is the core of the broker that collects information from QM, SM and FM. Fig. 2 illustrates the framework of the broker.

3.1.1. QoS Module (QM)

All QoS-related updates from the fogs are recorded in QM. QoS can be measured in terms of transmission rate, response time, usability or availability. Following the work in[33], we adopt their meaning and formulations of reliability and response time since both metrics were used as QoS parameters as well. Reliability can be interpreted as how a service can operate without failure during a given time and condition. To show how reliability is calculated we adopt Equation 1,

$$Rel = \left(1 - \frac{numfailure}{n}\right) * P_{mttf}, \tag{1}$$

where Rel represents reliability, $numfailure$ is the number of users who experienced a failure in a time interval less than promised by the provider, n is number of users, and P_{mttf} is the promised mean time to failure. On the other hand, the response time indicates the time taken for the provider in this case, fog, to serve the request. We adopt Equation 2 to calculate response time.

$$R_t = \frac{\sum_{i}^{n} T_i}{n}, \tag{2}$$

where R_t is the average response time, T_i is time between when user i requested for a service and when it is actually available, and n is the total number of service requests.

3.1.2. Security Module (SM)

The security module looks into confidentiality and authentication. Authentication is ensured whereby the fog and IoT user only need to be authenticated once instead of every interaction. This helps reduce the processing time and is especially beneficial in order to achieve a low latency in fog. Alongside providing confidentiality and authentication to user, this module also manages history of compromised servers and performs intrusion detection and prevention as defense mechanism of the broker. Confidentiality is quantified in terms of the fog's capability in providing data encryption and transmission encryption. To quantify authentication, two of the most commonly used protocols for centralized authentication i.e. Remote Authentication Dial-In User Service (RADIUS) and Terminal Access Controller Access-Control System (TACACS) are considered. Hence, we propose Equation 3 to quantify confidentiality, where C, e_d and e_t represent confidentiality, data encryption and transmission encryption respectively. On the other hand, we propose Equation 4 to quantify authentication, from the addition of both the RADIUS and TACACS, where A_t, *rd*, and *tc* represent authentication, RADIUS, and TACACS respectively.

$$C = e_d + e_t. \tag{3}$$

$$A_t = RADIUS + TACACS. \tag{4}$$

3.1.3. Feedback Module (FM)

User feedback upon completion of service are stored in the FM. Each fog has a feedback database collected from each of the users that have used its services. Any incomplete feedback will be discarded and not used for future evaluation. The fog will compute the average right after the feedback is submitted. The overall feedback will be based on user experience and recommendation from the user. Both the user feedback and recommendation are ranged from high, medium to low.

3.1.4. Trust Engine (TE)

The trust engine is the core of the broker as it performs all the trust evaluation. It consists of three components that operates sequentially in stages namely Availability Status (AS) for Stage 1, Fuzzy-based Filtering (FF) for Stage 2, and Cost Comparison (CC) for Stage 3. In Stage 2, TE would identify the values gathered from QM, SM and FM for trust evaluation by means of fuzzy logic where it also performs both the RM and FTF algorithms based from the gathered values.

3.1.5. Load Balancer (LB)

Once TE has calculated the trust values of the fogs that meet the requirements, the load balancer (LB) will begin assigning the task to the fogs. In this paper, the utilization, ρ, of the fogs are also calculated. Ideally, the more trustworthy fogs would be utilized more than the others, causing the fog to be overloaded. To prevent such situation, a threshold value, T_h is applied to balance the load. Once a fog's ρ has exceeded the T_h^i, then LB will distribute the load to other qualified fogs.

3.2. Working Procedures

Before the evaluation can commence, it is imperative to gather related information from all the fog servers, particularly the QoS status, security levels and service feedback. The fogs notify this information to the broker. We assume that the users and fogs have registered with the broker, where QoS is defined prior to the user sending request. Rather than relying solely on the fog broker to get the best service, a user is able to set the order of attributes that they require in their request. Their request contains two parts. The first part is the particular order of three attributes; QoS, security and feedback. The second part is the preferred values for each of the attributes. Upon receiving the user's request, TE performs RM algorithm to process the first part of the user request. Since there are 3 attributes, there are 6 possible sets where S1 is $\{QoS, Security, Feedback\}$, S2 is $\{QoS, Feedback, Security\}$, S3 is $\{Security, QoS, Feedback\}$, and S4 is $\{Security, Feedback, QoS\}$. Meanwhile S5 and S6 are $\{Feedback, QoS, Security\}$ and $\{Feedback, Security, QoS\}$ respectively. S1 through S6 have three fuzzy sets, each for the attributes. However, an additional set, S7 $\{no - preference\}$ is defined for users with no specific preference where it has only one fuzzy set. Ideally, for S1 through S6, each fuzzy set has small number of rules as compared to S7. The RM algorithm runs in TE before the start of the evaluation stages.

Algorithm 1 shows the RM algorithm. The first entry will be checked. If it is QoS, the possible matching sets would be either S1 or S2. When the second entry is feedback,

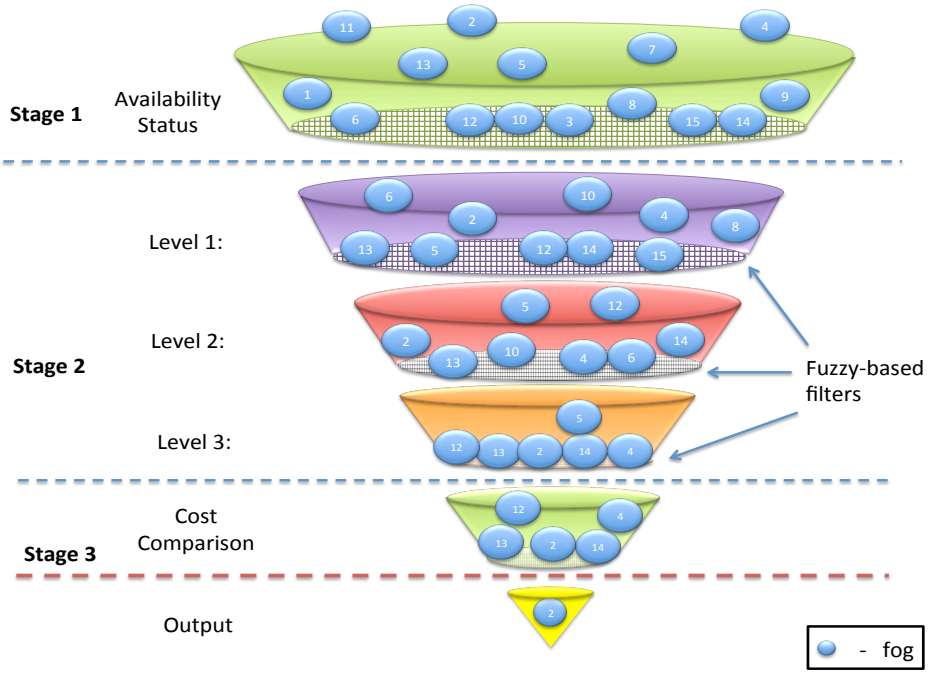


Figure 3: Trust evaluation process.

thus automatically the third entry would be security. Hence the user's request matched S1. The same process will be repeated for each user request until a match is made. If the first entry is no-preference, and the subsequent entries are left blank, thus the request is set to S7 by default. However, if the entries does not match any of the attributes, the request will be discarded. After the right set has been identified and invoked, all fogs have to go through three sequential stages of evaluation as shown in Fig. 3. Stage 1 is the checking of availability status, Stage 2 is fuzzy-based filtering and Stage 3 is the cost comparison of fog. The end result will give the best fog that have met all the user requirement.

3.2.1. Stage 1

The first stage of the evaluation is to see the availability of the fog servers. Availability can be looked at in terms of a fog's connectivity, computing resources, and even power capability. In this study, It is important to ensure availability before calculating the trust as less computation is needed to calculate the trust of known available fogs, rather than calculating the trust values for all fogs while the availability status of these fogs are still unknown. The most common way to measure availability is based on mean time to failure and mean time to repair [34] [35]. We use Eq. (5) to measure availability in fog computing as follows:

$$Availability = MTTF/(MTTF + MTTR), \tag{5}$$

where MTTF (mean time to failure) is the average time before a failure, and MTTR (mean time to repair) is used to determine the mean time to repair a failed component.

Algorithm 1: Request-Matching Algorithm:

```
Data: QoS, Feedback, Security;
Result: S1, S2, S3, S4, S5, S6, S7;
if entry-1 is QoS then
    /* If entry-1 is QoS, then automatically check entry-2 */
    if entry-2 is feedback then
        /* If entry-1 is QoS, entry-2 is feedback, it is certain entry-3 is security */
        entry-3 is security
        then invoke S1
    else if entry-2 is security then
        /* If entry-1 is QoS, entry-2 is security, it is certain entry-3 is feedback */
        entry-3 is feedback
        then invoke S2
else if entry-1 is feedback then
    /* If entry-1 is feedback, then automatically check entry-2 */
    if entry-2 is QoS then
        /* If entry-1 is feedback, entry-2 is QoS, it is certain entry-3 is security */
        entry-3 is security
        then invoke S3
    else if entry-2 is security then
        /* If entry-1 is feedback, entry-2 is security, it is certain entry-3 is QoS */
        entry-3 is QoS
        then invoke S4
else if entry-1 is security then
    /* If entry-1 is security, then automatically check entry-2 */
    if entry-2 is QoS then
        /* If entry-1 is security, entry-2 is QoS, it is certain entry-3 is feedback */
        entry-3 is feedback
        then invoke S5
    else if entry-2 is feedback then
        /* If entry-1 is security, entry-2 is feedback, it is certain entry-3 is QoS */
        entry-3 is QoS
        then invoke S6
else if entry-1 is no-preference then
    invoke S7
else
    discard request
```

Assuming that both broker and fog have agreed to have 99.9999 availability level in the Service Level Agreement prior to receiving the request, it is necessary for this evaluation stage to select fogs with that value. If fogs availability is greater than 99.9999, they may proceed to next stage for trust evaluation.

3.2.2. Stage 2

Fogs that meet the availability requirement can then go through Algortithm 2 which is the FTF algorithm in Stage 2 where the second part of the user request is processed. In the second part of the user request, users are able to specify the values, v with range $[0, 1]$ for each attribute. This stage is dynamic in a sense that instead of running a large block of rule set of QoS, security and feedback, the rule set is separated into three parts, i.e. one smaller rule set for each of the QoS, security and feedback. Consequently, this separation reduces the computation time the TE needs to process. These three rule sets are identical for each of the three levels. When there is a user request, it only runs the rule set that matched the user request for each level. Algorithm 2 shows the FTF algorithm for S3. The FTF algorithm for the remaining sets follow the same structure as the one shown for S3, except for S7. In this algorithm, firstly it checks the first entry of the user request i.e. security. We assume the second part of the user request is a minimum value of 0.8. If a fog's security value is $\leqslant 0.8$, it is considered satisfactory and these fogs can proceed to Level 2 of filtering. Unsatisfactory fogs are filtered out from entering the subsequent processes. TE then proceed to check the next entry i.e. QoS with minimum required value of 0.7. If a fog's QoS value surpassed that of the minimum value specified by the user, it is considered satisfactory and these fogs can proceed to Level 3 of filtering. Finally, in Level 3, the feedback values of the fogs are checked. Similarly, fogs that do not meet the minimum user requirement are filtered out. Fuzzy rules are used to calculate the security, QoS, and feedback values of the successful fogs, where at the end, are then used to calculate trust. The fuzzy rules and graphs are presented in Fig. 4, 5 and 6.

3.2.3. Stage 3

At this stage, the total operational cost of these fog servers are then compared where the one with the lowest cost is chosen to be provisioned to the user. As mentioned in the previous section, we assume that the details of the cost is made readily available from the fogs' respective ISPs. It is from this information that we are able to compare and identify the fog server with the lowest cost. Once trust computation is completed, TE summons a notification to the selected fog to start its service provisioning to the user. After service is

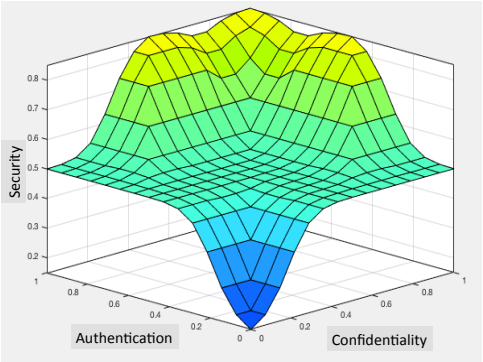
completed, the user will send a feedback back to the broker to be processed in the FM.

Algorithm 2: Fuzzy-based Trust Filtering Algorithm:

```
while entry-1 is Security do
    calculate Security
    if Security == satisfactory then
        /* Satisfactory if value exceeds user's minimum required value */
        proceed checking entry-2
    else
        eliminate fog
while entry-2 is QoS do
    calculate QoS
    if QoS == satisfactory then
        /* Satisfactory if value exceeds user's minimum required value */
        proceed checking entry-3
    else
        eliminate fog
while entry-3 is Feedback do
    calculate Feedback
    if Feedback == satisfactory then
        /* Satisfactory if value exceeds user's minimum required value */
        proceed Level-3
    else
        eliminate fog
```

Rule	Confidentiality	Authentication	Security
1	High	High	High
2	High	Medium	High
3	High	Low	Medium
4	Medium	High	High
5	Medium	Medium	Medium
6	Medium	Low	Medium
7	Low	High	Medium
8	Low	Medium	Medium
9	Low	Low	Low

(a) Security fuzzy rules.

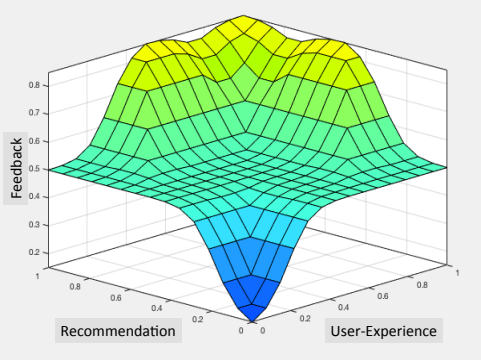


(b) Security graph.

Figure 4: Security rules and graph.

Rule	User Experience	Recommendation	Feedback
1	High	High	High
2	High	Medium	High
3	High	Low	Medium
4	Medium	High	Medium
5	Medium	Medium	Medium
6	Medium	Low	Low
7	Low	High	Medium
8	Low	Medium	Low
9	Low	Low	Low

(a) Feedback fuzzy rules.

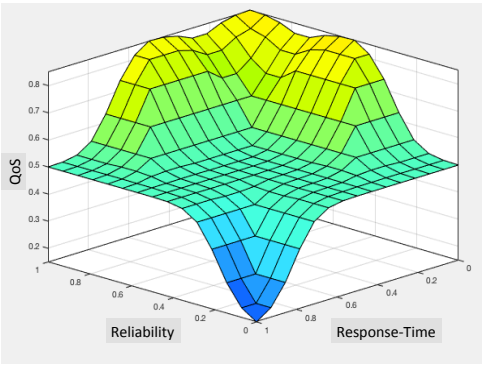


(b) Feedback graph.

Figure 5: Feedback fuzzy rules and graph.

Rule	Reliability	Response Time	QoS
1	High	High	Medium
2	High	Medium	High
3	High	Low	High
4	Medium	High	Medium
5	Medium	Medium	Medium
6	Medium	Low	High
7	Low	High	Low
8	Low	Medium	Medium
9	Low	Low	Medium

(a) QoS fuzzy rules



(b) QoS graph

Figure 6: QoS fuzzy rules and graph.

4. Performance Evaluation

In this performance evaluation, the objective is to show how our solution dynamically adapts to the given situation by using Matlab Fuzzy logic toolbox. We consider two tasks that matches S6 i.e. in the order of feedback, security, and QoS with minimum requirement of $\{0.5, 0.8, 0.7\}$ and $\{0.5, 0.8, 0.6\}$ respectively. We assume that there are five fogs involved in this evaluation. The filtering evaluation process runs on repeat from the time the task is received until it is completed to find the optimal fog that can carry out the given task. For simplicity, the feedback and security values are kept constant which means that for every evaluation at time t_n, these fogs will qualify until Level 2. It is the varying QoS values that determines whether the fogs will proceed or disqualified at Level 3. After the trust evaluation process, the load balancer will compute the utilization, ρ, of the fogs where a T_h is set at 80%. By doing so, the request can be distributed to other qualified fogs if the current processing fog is overloaded. Utilization is calculated by $\frac{\lambda}{\mu} * 100$ where λ and μ are

arrival rate and service rate accordingly. It is assumed that all fogs have the same service rate of 10 requests/sec throughout the time.

4.1. Results

Fig.5 presents the overall workflow of the tasks and the Fig. 6 shows the fogs utilization performance. At t_1, Task 1 can be potentially processed by Fog 1 and Fog 3. However, as Fog 3s utilization is beyond the 80% threshold, Fog 1 will process the task. Meanwhile, Fog 1 will also process Task 2 simultaneously. At t_2, Task 1 at Fog 1 migrated to Fog 2 as it is the only fog that has a QoS value beyond 0.7 and utilization within threshold. Task 2 is migrated to Fog 5. Task 1 migrated to Fog 3 at t_3 and on the other hand, Task 2 is processed by Fog 1 again from t_3 to t_5. At t_4, Fog 3 still continues to process Task 1. However, to balance its utilization that has exceeded 80%, some the task have be mitigated to Fog 5 that will continue to process the task until t_6. At t_6, Fog 4 have processed Task 2. After that, Fog 1 and Fog 5 resumed Task 1 and Task 2 at t_7 respectively. Since Fog 2s QoS is above 0.7 and has utilization value below 80%, it took charge of Task 1 at the subsequent t_8 and t_9. The evaluation went on until t_{10}, Task 1 is migrated to Fog 5 and Task 2 is migrated to Fog 2. This result shows that by having different values of a single attribute over a period of time can change the course of action of the process. Overall, this evaluation shows how our approach is dynamically changing with the given situation while still being able to meet the user requirements.

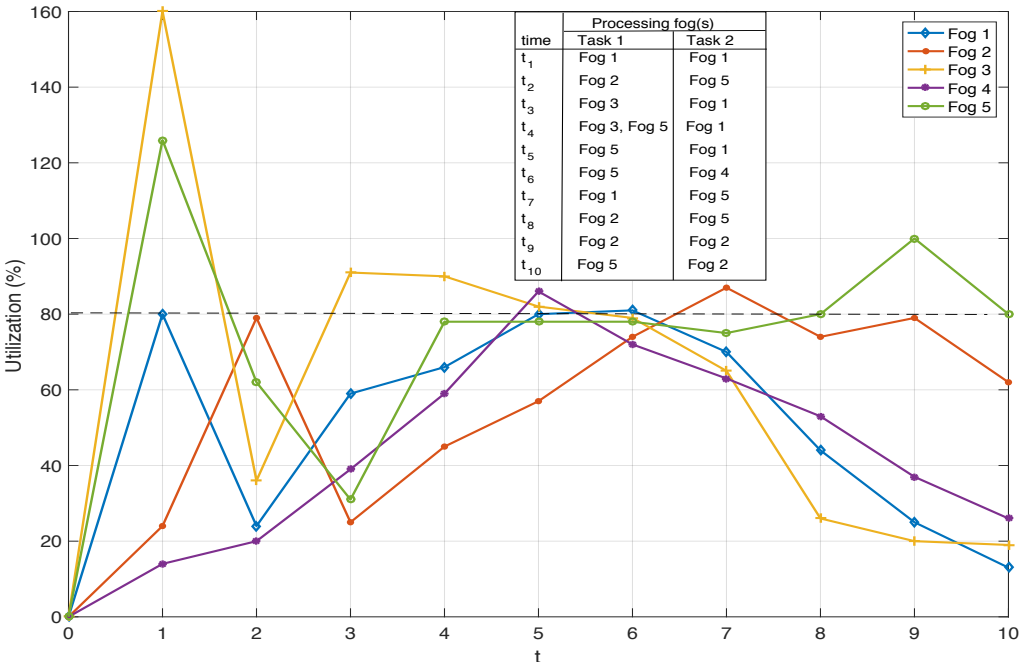


Figure 8: Fogs utilization performance.

Fig. 7 shows the effect of increasing λ towards the number of fogs needed to process a task. T_h values of 70% and 90% are taken for comparison. With μ remaining constant, the

		t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}
Fog 1	QoS	0.9	0.3	0.6	0.6	0.8	0.8	0.7	0.5	0.3	0.1
	Trust	0.686	0.5	0.517	0.517	0.686	0.686	0.561	0.5	0.5	0.5
	λ (rps)	8.0	2.4	5.9	6.6	8.0	8.1	7.0	4.4	2.5	1.3
	ρ (%)	80	24	59	66	80	81	70	44	25	13
	Task 1										
	Task 2										
	Other tasks										
Fog 2	QoS	0.3	0.8	0.3	0.6	0.5	0.7	0.8	0.7	0.8	0.6
	Trust	0.5	0.686	0.5	0.517	0.5	0.561	0.686	0.561	0.686	0.517
	λ (rps)	2.4	7.9	2.5	4.5	5.7	7.4	8.7	7.4	7.9	6.2
	ρ (%)	24	79	25	45	57	74	87	74	7.9	62
	Task 1										
	Task 2										
	Other tasks										
Fog 3	QoS	0.9	0.3	0.8	0.9	0.7	0.6	0.5	0.3	0.2	0.2
	Trust	0.686	0.5	0 .686	0.686	0.561	0.517	0.5	0.5	0.5	0.5
	λ (rps)	16	3.6	9.1	9.0	8.2	7.9	6.5	2.6	2.0	1.9
	ρ (%)	160	36	91	90	82	79	65	26	20	19
	Task 1										
	Task 2										
	Other tasks										
Fog 4	QoS	0.1	0.2	0.3	0.4	0.7	0.6	0.5	0.4	0.3	0.2
	Trust	0.5	0.5	0.5	0.5	0.561	0.517	0.5	0.5	0.5	0.5
	λ (rps)	1.4	2.0	3.9	5.9	8.6	7.2	6.3	5.3	3.7	2.6
	ρ (%)	14	20.0	39	59	86	72	63	53	37	26.0
	Task 1										
	Task 2										
	Other tasks										
Fog 5	QoS	0.9	0.6	0.3	0.7	0.7	0.7	0.6	0.6	0.9	0.7
	Trust	0.686	0.517	0.5	0.561	0.561	0.561	0.517	0.517	0.686	0.561
	λ (rps)	12.6	6.2	3.1	7.8	7.8	7.8	7.5	8.0	10	7.7
	ρ (%)	126	62	31	78	78	78	75	80	100	77
	Task 1										
	Task 2										
	Other tasks										

Figure 7: Flow of tasks.

increasing λ also increases ρ. Thus the ρ would closely reach T_h, and eventually exceed the T_h. This would provoke the LB to mitigate the task to other qualified fogs.It shows that the lower the T_h, the greater the number of fogs needed to process a task.

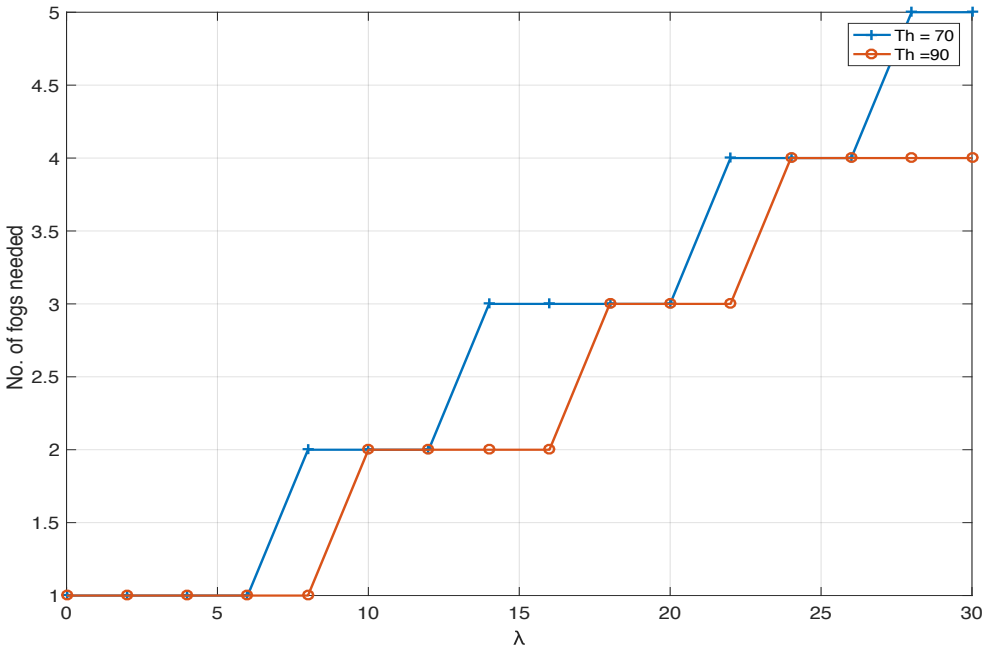
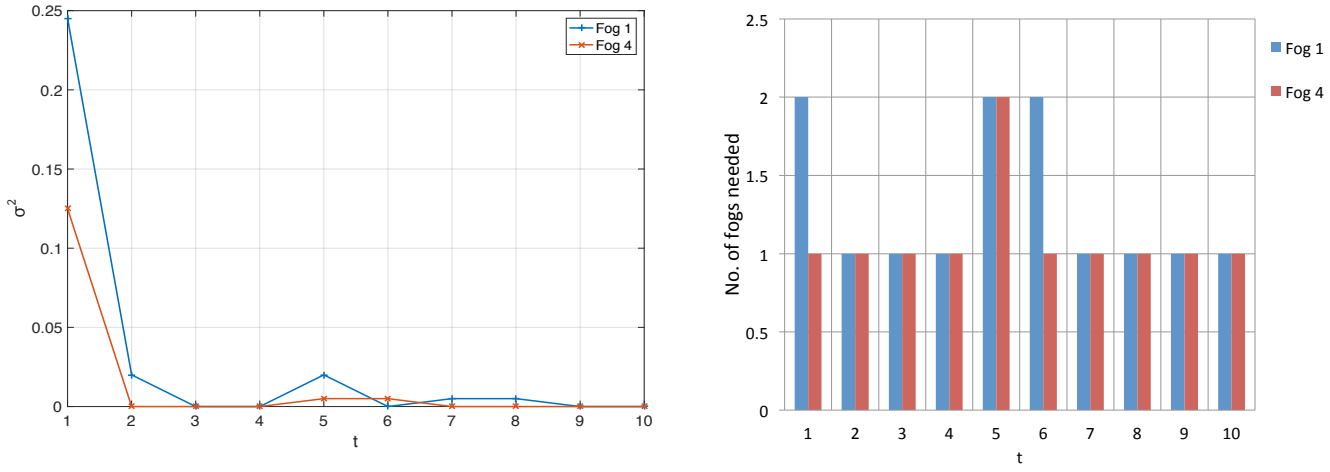


Figure 9: Number of fogs needed against arrival rate.

5. Discussions

When a fog's attribute value change abruptly, it causes the trust value to change as well resulting in a large variance, denoted by σ^2. The higher the trust value, the higher the ρ and vice versa. Consequently, higher utilization will trigger the LB to distribute the tasks to other fogs. Hence, the number of fogs needed to process a task will also increase. Despite that, this is only true if two conditions are met. Inevitably, a large σ^2 would also result from the rapid attribute value decline. Thus the first condition to be met would be the increase of σ^2 due to the increase of attribute value as time progresses by. Secondly, the number of needed fogs will only rise if the trustworthy fog that is processing the task has exceeded T_h. In order to observe the effect of σ^2 and the number of fogs needed, the σ^2 for Fog 1 and Fog 4 are taken. Instances of migration when conditions are met are seen at the spike of σ^2 at t_1 and t_6 of Fog 1 in Figure 8. In both cases, two fogs are needed to process the task. No task migration have occurred at t_6 to t_{10} of Fog 4 as its $\rho \leqslant T_h$. On the other hand, Fog 4's high σ^2 at t_1 did not trigger the increase number of fogs. This is due to the low ρ value of 14% that is significantly lower than the T_h. Let x be the fogs, $x = \{Fog_1, Fog_2, Fog_3, Fog_4, Fog_5\}$. Fogs that meet the user requirement is denoted as y. If migration M is to occur, $M = \{y|y \epsilon x, y > 1, \rho > T_h\}$. This indicates that frequent task migration should be avoided as the network would suffer several disadvantages, for instance,

extra consumption of various resource types such as bandwidth of the path between the source and destination [36]. Furthermore, additional delay would similarly incur, resulting in lower QoS instead [37]. Hence this suggest that fogs with abrupt behavior should be given the least consideration in the evaluation.



(a) Trust σ^2

(b) Number of fogs needed

Figure 10: Relationship between trust variance and the number of fogs needed.

6. Conclusions and Future Works

This paper has proposed a fuzzy-based broker trust evaluation framework (FTEF) that provides as user the service that they have specified with the help of a load-balancer. It works dynamically in a way that the whole process is tailored to meet a user's request. The broker which consists of QM, SM, FM and TE, performs the core operations namely RM and FTF algorithms. The outcome of the algorithms result in identifying the fog server that best fits the criterion defined by the user while simultaneously maintaining the utilization beneath the threshold. Our approach suggests that the broker is able to work dynamically in processing user requests, and the broker can successfully provide the most trustworthy fog. This paper has shown that without user request specification, a typical broker might simply choose the best fog from the broker's perspective without considering a user's preference. Furthermore, trust value should differ in the perspective of different users. Thus, the addition of fuzzy logic performed in multiple levels has allowed us to define they grey area in evaluating the trustworthiness value of a fog. In the future, we will look at ways to solve cases of malicious feedback given by users.

References

[1] Cisco, Cisco Global Cloud Index : Forecast and Methodology, 2015-2020, Tech. rep. (2016).

[2] Cisco, Fog Computing and the Internet of Things: Extend the Cloud to Where the Things Are, Tech. rep., Cisco (2015).
URL http://www.cisco.com/c/dam/en{_}us/solutions/trends/iot/docs/computing-overview.pdf

[3] S. Yi, Z. Qin, Q. Li, Security and privacy issues of fog computing: A survey, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 9204 (2015) 685–695. doi:10.1007/978-3-319-21837-3_67.

[4] C. A. R. L. Brennand, F. Cunha, G. Maia, E. Cerqueira, A. Loureiro, L. Villas, FOX : A traffic management system of computer- based vehicles FOG FOX : A Traffic Management System of Computer-Based Vehicles FOG, IEEE Symposium on Computers and Communications (ISCC) (June) (2016) 982–987. doi:10.1109/ISCC.2016.7543864.

[5] C. Westphal, Challenges in networking to support augmented reality and virtual reality, in: ICNC, 2017.

[6] M. Aazam, E. N. Huh, Fog computing micro datacenter based dynamic resource estimation and pricing model for IoT, Proceedings - International Conference on Advanced Information Networking and Applications, AINA 2015-April (2015) 687–694. doi:10.1109/AINA.2015.254.

[7] S. F. Abedin, M. G. R. Alam, N. H. Tran, C. S. Hong, A Fog based system model for cooperative IoT node pairing using matching theory, 17th Asia-Pacific Network Operations and Management Symposium: Managing a Very Connected World, APNOMS 2015 (2015) 309–314doi:10.1109/APNOMS.2015.7275445.

[8] F. D. Schoorman, R. C. Mayer, J. H. Davis, An integrative model of organizational trust: Past, present, and future, Academy of Management Review 32 (2) (2007) 344–354. doi:10.5465/AMR.2007.24348410.

[9] D. Rousseau, S. Sitkin, R. Burt, C. Camerer, Not so different after all: A cross-discipline view of trust, Academy of Management Review 23 (June) (1998) 393–404.

[10] H. Oh, T.-w. Um, J. K. Choi, Trust Provisioning for future ICT infrastructures and services, Tech. rep., ITU-T (2016).

[11] A. Kapsalis, P. Kasnesis, I. S. Venieris, I. Dimitra, CONNECTING FOG AND CLOUD COMPUTING Fog Approach for Effective Workload Balancing, IEEE Cloud Computing (2017) 36–45.

[12] S. Abdelwahab, B. Hamdaoui, FogMQ: {A} Message Broker System for Enabling Distributed, Internet-Scale IoT Applications over Heterogeneous Cloud Platforms, eprint arXiv:1610.00620 (2016) 1–7arXiv:arXiv:1610.00620v1.
URL http://arxiv.org/abs/1610.00620

[13] A. A. Alsaffar, H. P. Pham, C. S. Hong, E. N. Huh, M. Aazam, An Architecture of IoT Service Delegation and Resource Allocation Based on Collaboration between Fog and Cloud Computing, Mobile Information Systems (2016) 1–15doi:10.1155/2016/6123234.

[14] D. Wong, Peak Efficiency Aware Scheduling for Highly Energy Proportional Servers, Proceedings - 2016 43rd International Symposium on Computer Architecture, ISCA 2016 (2016) 481–492doi:10.1109/ISCA.2016.49.

[15] M. Adhikari, T. Amgoth, Heuristic-based load-balancing algorithm for IaaS cloud, Future Generation Computer Systems 81 (2017) 156–165. doi:10.1016/j.future.2017.10.035.
URL https://doi.org/10.1016/j.future.2017.10.035

[16] L. D. Dhinesh Babu, P. Venkata Krishna, Honey bee behavior inspired load balancing of tasks in cloud computing environments, Applied Soft Computing Journal 13 (5) (2013) 2292–2303. doi:10.1016/j.asoc.2013.01.025.
URL http://dx.doi.org/10.1016/j.asoc.2013.01.025

[17] H. K. Mehta, P. Pawar, P. Kanungo, A Two Level Broker System for Infrastructure as a Service Cloud, Wireless Personal Communications 90 (3) (2016) 1135–1147. doi:10.1007/s11277-016-3382-x.

[18] S. Singh, J. Sidhu, Compliance-based Multi-dimensional Trust Evaluation System for determining trustworthiness of Cloud Service Providers, Future Generation Computer Systems 67 (2017) 109–132.

[19] V. Viji Rajendran, S. Swamynathan, Hybrid model for dynamic evaluation of trust in cloud services,

Wireless Networks 22 (2016) 1807–1818. doi:10.1007/s11276-015-1069-y.
[20] F. Jrad, J. Tao, A. Streit, SLA based service brokering in intercloud environments, in: CLOSER 2012 - Proceedings of the 2nd International Conference on Cloud Computing and Services Science, 2012, pp. 76–81.
[21] P. Pawar, M. Rajarajan, T. Dimitrakos, A. Zaisman, Trust Assessment Using Cloud Broker, in: J. Zhou, N. Gal-Oz, J. Zhang, E. Gudes (Eds.), Trust Management VIII. IFIP Advances in Information and Communication Technology, Vol. 430, Springer, 2014, pp. 237–244. doi:10.1016/j.cpr.2010.02.004.
[22] W. Fan, H. Perros, A novel trust management framework for multi-cloud environments based on trust service providers, Knowledge-Based Systems 70 (2014) 392–406. doi:10.1016/j.knosys.2014.07.018.
[23] Y. Li, An Overview of the DSRC/WAVE Technology, in: X. Zhang, D. Qiao (Eds.), Quality, Reliability, Security and Robustness in Heterogeneous Networks, Springer, 2015, pp. 9–21.
[24] S. Pandey, A. K. Daniel, Fuzzy Logic Based Cloud Service Trustworthiness Model, IEEE International Conference on Engineering and Technology (ICETECH) (2016) 1–6.
[25] R. Falcone, G. Pezzulo, C. Castelfranchi, A fuzzy approach to a belief-based trust computation, Trust reputation and security 2005 (11 July) (2003) 1–15.
[26] J. Bernal Bernabe, J. L. Hernandez Ramos, A. F. Skarmeta Gomez, TACIoT: multidimensional trust-aware access control system for the Internet of Things, Soft Computing (2015) 1763–1779doi:10.1007/s00500-015-1705-6.
[27] M. Supriya, V. L.J, K. Sangeeta, G. K. Patra, Estimating Trust Value for Cloud Service Providers using Fuzzy Logic, International Journal of Computer Applications 48 (19) (2012) 28–34. doi:10.5120/7457-0491.
[28] S. Kumar, S. Mittal, M. Singh, Fuzzy based trust management system for cloud environment, Advances in Science and Technology Research Journal 10 (30) (2016) 32–37. doi:10.12913/22998624/62703.
[29] Y. Huo, Y. Zhuang, S. Ni, Fuzzy trust evaluation based on consistency intensity for cloud services, Kybernetes 44 (1) (2015) 7–24. doi:10.1108/K-03-2014-0058.
[30] M. Alhamad, T. Dillon, E. Chang, A Trust-Evaluation Metric for Cloud Applications, International Journal of Machine Learning and Computing 1 (4) (2011) 416–421. doi:10.7763/IJMLC.2011.V1.62.
URL http://www.ijmlc.org/index.php?m=content{&}c=index{&}a=show{&}catid=25{&}id=248
[31] N. Ghosh, S. K. Ghosh, S. K. Das, SelCSP: A framework to facilitate selection of cloud service providers, IEEE Transactions on Cloud Computing 3 (1) (2015) 66–79. doi:10.1109/TCC.2014.2328578.
[32] P. Varalakshmi, T. Judgi, Multifaceted trust management framework based on a trust level agreement in a collaborative cloud, Computers and Electrical Engineering 0 (2016) 1–16. doi:http://dx.doi.org/10.1016/j.compeleceng.2016.10.002.
[33] S. Kumar, S. Versteeg, R. Buyya, A framework for ranking of cloud computing services, Future Generation Computer Systems 29 (4) (2013) 1012–1023. doi:10.1016/j.future.2012.06.006.
URL http://dx.doi.org/10.1016/j.future.2012.06.006
[34] M. Almashor, I. Khalil, Z. Tari, A. Zomaya, S. Sahni, Enhancing Availability in Content Delivery Networks for Mobile Platforms, IEEE Transactions on Parallel and Distributed Systems 26 (8). doi:10.1109/TPDS.2013.2297927.
[35] O. Osanaiye, S. Chen, Z. Yan, R. Lu, K. Choo, M. Dlodlo, From cloud to fog computing: A review and a conceptual live VM migration framework, IEEE Access 5 (2017) 8284–8300. doi:10.1109/ACCESS.2017.2692960.
URL http://ieeexplore.ieee.org/document/7896564/
[36] R. Boutaba, Q. Zhang, M. F. Zhani, Virtual Machine Migration in Cloud Computing Environments: Benefits, Challenges and Approaches, in: H. T. Mouftah, B. Kantarci (Eds.), Communication Infrastructures for Cloud Computing, Vol. 8, Information Science Reference, 2014, Ch. 17, pp. 383–408. doi:10.14257/ijgdc.2015.8.5.33.
[37] G. S. Aujla, N. Kumar, MEnSuS: An efficient scheme for energy management with sustainability of cloud data centers in edge-cloud environment, Future Generation Computer Systemsdoi:10.1016/j.future.2017.09.066.