This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2018.2886379, IEEE Transactions on Network and Service Management

Trust-based Service Management for Mobile Cloud IoT Systems

Ing-Ray Chen, Jia Guo
Computer Science
Virginia Tech
irchen@vt.edu
jiaguo@vt.edu

Ding-Chau Wang
Information Management
Southern Taiwan University of Science and Technology
dcwang@stust.edu.tw

Jeffrey J.P. Tsai
Bioinformatics and Biomedical Engineering
Asia University
jjptsai@gmail.com

Hamid Al-Hamadi
Computer Science
Kuwait University
hamid@cs.ku.edu.kw

Ilsun You
Information Security Engineering
Soonchunhyang University
ilsunu@gmail.com

***Abstract* — We propose and analyze a 3-tier cloud-cloudlet-device hierarchical trust-based service management protocol called IoT-HiTrust for large-scale mobile cloud IoT systems. Our mobile cloud hierarchical service management protocol allows an IoT customer to report its service experiences and query its subjective service trust score toward an IoT service provider following a scalable report-and-query design. We conduct a formal scalability analysis along with a ns-3 simulation performance analysis demonstrating that IoT-HiTrust not only achieves scalability without compromising accuracy, convergence, and resiliency properties against malicious attacks but also outperforms contemporary distributed and centralized IoT trust management protocols. We test the feasibility by applying IoT-HiTrust to two case studies: a smart city travel service composition and binding application and an air pollution detection and response application. The results demonstrate that IoT-HiTrust outperforms contemporary distributed and centralized trust-based IoT service management protocols in selecting trustworthy nodes to maximize application performance, while achieving scalability.**

Keywords— Internet of things; scalability; trust management; service management; mobile cloud computing; service composition; performance analysis.

I. INTRODUCTION

In recent years we have witnessed a proliferation of Internet of things (IoT) devices such as RFID tags, sensors, smartphones, smart appliances, environmental monitoring devices, etc. capable of providing services upon request. There will be a huge number of such IoT devices competing for service. Therefore, a central issue is whether the service provided by a selected IoT device is trustworthy. Further, most IoT devices are mobile and will connect to the Internet (cloud) on and off, depending on the location they roam into as well as the energy status of individual IoT devices. Hence, it calls for an effective and efficient trust-based IoT service management protocol that can scale to a large number of heterogeneous devices in IoT systems.

Trust-based service management is needed because not all IoT devices will be trustworthy and some IoT devices may behave maliciously to disrupt the cloud service (e.g., an adversary) or just for their own gain (e.g., for increasing their chances to be selected to provide requested services). Furthermore, users who own IoT devices are likely to be socially connected via social networks. Therefore, misbehaving nodes with close social ties can collude and monopoly a class of services. For example, for service-oriented IoT systems [5] it is important for an IoT device to select only trustworthy IoT service providers before service composition is performed. For participatory sensing IoT applications [11], it is critical to assess source trustworthiness of IoT devices which report sensing results, so untrustworthy data can be filtered out before data analysis is taken.

In the literature, trust-based service management protocols for IoT systems can be categorized into distributed [2, 3, 4, 5, 6, 10, 17, 20] and centralized (cloud-based) [7, 14, 15, 23]. The basic issue of distributed trust-based service management protocols is scalability, i.e., an IoT device's communication and storage cost cannot scale with a large number of IoT devices in the system. The basic issue of centralized trust-based service management protocols is that it is difficult if not impossible to maintain a consistent global view of social and interaction relationships for every pair of IoT devices dynamically in a large and rapidly changing IoT system. An inconsistent view of social and interaction relationships among IoT devices will make the trust prediction inaccurate and render trust-based service management ineffective. Since the cloud cannot physically collect social and interaction relationships itself, it needs to collect such information from individual IoT devices dynamically. The amount of traffic generated by a large number of IoT devices simultaneously to the cloud will not only consume IoT energy but also cripple the cloud communication network. Our research motivation is to address the scalability issues in existing distributed and centralized trust-based IoT service management protocols, without compromising desirable trust accuracy, convergence, and resiliency properties.

In this paper, we propose and analyze a mobile cloud hierarchical trust management protocol called IoT-HiTrust with the goal to support scalable trust-based service management in large mobile cloud IoT applications. The reason we focus on mobile cloud IoT systems is that mobile cloud IoT applications will have great social impacts to our everyday life [19]. One example mobile cloud IoT application is environmental monitoring [11] where IoT devices (e.g., smart phones carried by humans) collect environmental data (noise, air pollution, temper-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2018.2886379, IEEE Transactions on Network and Service Management

ature, humidity, light, etc.) and submit via wireless data communication links to a processing center located in the cloud for environmental data analysis. In return, a user (e.g., a user with health concerns) can send a query to the cloud to query a location's noise or air pollution level. Another example is road/traffic monitoring [11] by which traffic flows, potholes, bumps, braking, and honking information reported from IoT devices (smart phones carried by passengers/drivers in a car) are aggregated by a data processing center located in the cloud to unveil traffic patterns previously unobserved with existing monitoring infrastructure. A third example is service-oriented architecture (SOA) based service management [5] in which each IoT device is both a service provider (SP) and a service requestor (SR) as in a web service system. An SR would request the availability of SPs resided in the same location, compose a composite service, and select (or bind) trustworthy SPs for executing the composite service [18].

Our paper has the following unique contributions:

1. To the best of our knowledge, IoT-HiTrust is the only scalable trust protocol for mobile cloud IoT by means of a new "report-and-query" design in our proposed 3-tier cloud-cloudlet-device hierarchy (see Section III.A for detail) that allows an IoT device to report its service experience and social relationship with another IoT device dynamically and query the service trustworthiness of another IoT device locally through the local cloudlet to the cloud and the cost remains more or less constant. It achieves scalability because an IoT device's communication and storage cost will remain the same, regardless of the number of IoT devices in the system. An IoT device does not store trust and service experience data in its limited storage, acquire recommendations from other IoT devices it encounters, or compute the trust score of another IoT device. All trust and service experience data are stored in the cloud and all trust score computations are performed by the cloud.
2. We propose a new cloud-based "subjective trust" evaluation design that allows a target IoT device to obtain its subjective trust of another IoT device by incorporating the target IoT device's own observations and other IoT devices' recommendations weighted by the target IoT device's subjective social relationship and trust view toward the recommenders. This is very different from contemporary centralized IoT trust protocols (e.g., ObjectiveTrust [14]) where only the concept of "objective trust" (i.e., common belief or reputation) is maintained. We argue that "subjective trust" must be used for an IoT device to select an IoT service provider because service quality is inherently related to the social relationship of human owners who control the service behaviors of their IoT devices, so that two IoT devices with a close social relationship will likely provide/receive good service.
3. By a formal scalability analysis and a ns-3 simulation performance analysis, we demonstrate that IoT-HiTrust not only achieves scalability without compromising system desirable properties including accuracy, convergence, and resiliency against self-promotion, bad-mouthing, ballot-stuffing, discriminatory, and opportunistic service attacks, but also outperforms contemporary distributed IoT trust management protocols (e.g., Adaptive IoT Trust [6]) as well as centralized IoT trust management protocols (e.g., ObjectiveTrust [14]). Furthermore, we demonstrate that accuracy, convergence, and resiliency properties can still be achieved despite intermittent network disconnection.
4. We demonstrate the applicability of IoT-HiTrust with two real-world case studies: a smart city travel service composition application and an air pollution detection and response application. The results demonstrate that IoT-HiTrust outperforms existing trust-based service management protocols including Adaptive IoT Trust [6] and ObjectiveTrust [14] in selecting trustworthy nodes to maximize application performance.

This paper has been substantially extended from our previous work [22] as follows: (a) we conduct a thorough literature survey of the current state of the art in trust management for IoT systems, and compare and contrast our work against existing work (Section II); (b) we extend the previous trust model in [22] by explicitly considering actions taken by an IoT device, a cloudlet, and a cloud server in responses to events occurring to them during protocol execution, illustrated by action flowcharts (Section IV); (c) we validate our hierarchical trust protocol design with two real-world mobile cloud applications, i.e., a smart city travel service composition application and an air pollution detection and response application, using web service quality traces [21] (Sections VI and VII); and (d) we compare IoT-HiTrust with ObjectiveTrust [14] which is the only centralized IoT trust management protocol to-date that considers social standing and relationships for credibility rating.

The rest of the paper is organized as follows. Section II discusses related work. Section III discusses the system model. Section IV describes IoT-HiTrust in detail and explains our efficient and effective hierarchical trust protocol design for managing a huge number of IoT devices. Section V conducts a scalability analysis as well as a performance analysis of IoT-HiTrust and demonstrates IoT-HiTrust can achieve scalability without compromising trust accuracy, convergence, and resiliency properties, when compared with Adaptive IoT Trust [6] and ObjectiveTrust [14]. Sections VI and VII demonstrate the applicability by a smart city travel service composition application and an air pollution detection and response application, respectively, and also compare the performance of IoT-HiTrust with Adaptive IoT Trust [6] and ObjectiveTrust [14]. Finally, Section VIII concludes the paper and outlines some future research areas.

II. Related Work

Mobile IoT systems can be characterized as a hybrid of P2P and social networks [6, 14] because IoT devices are mostly mobile heterogeneous entities with limited capacity, yet are mostly human carried or human operated. IoT trust management must take into account social relationships among device owners in order to maximize protocol performance. Existing trust management for P2P or mobile ad hoc systems [10, 17, 20, 25, 26, 36] therefore cannot be applied directly to mobile IoT systems

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2018.2886379, IEEE Transactions on Network and Service Management

because they do not scale well with a huge number of heterogeneous IoT nodes and do not consider "social trust" among IoT device owners for service trustworthiness assessment.

Trust management protocols for IoT systems are still emerging [35]. There are only a handful of IoT trust protocols designed and evaluated to-date.

Bao, et al. [2, 3, 4, 5, 6] pioneered the concept of distributed IoT trust management where each IoT device evaluates other IoT devices using both direct service experiences and indirect recommendations. Adaptive IoT Trust, a distributed IoT trust management protocol, is the end product. Adaptive trust management is achieved by determining the best way to combine direct trust (from direct experiences) and indirect trust (from recommendations) dynamically to minimize convergence time and trust estimation bias in the presence of malicious nodes performing opportunistic service and collusion attacks. Direct service experiences are collected based on own service experiences, while recommendations are collected at the time nodes encounter each other through social contacts. They used social similarity to rate recommenders. A drawback is that a node may not encounter each other often to collect enough recommendations to make informed decisions. Our IoT-HiTrust protocol eliminates this problem, since service ratings toward a trustee node are sent to each user's home cloud server. So many recommendations are available in the cloud for retrieval. The scalability issue was mitigated somewhat in [6] with a cache management design by which a capability-limited IoT device only keeps trust information of a subset of nodes of interest and performs minimum computation to update trust. The method, however, would still break down when there are a huge number of nodes in a large-scale IoT system.

Relative to [2, 3, 4, 5, 6] our work focuses on hierarchical trust management for achieving efficiency and scalability, without compromising trust accuracy.

Chen, et al. [7] proposed a centralized IoT trust management model based on fuzzy reputation. However, their trust management model considers a very specific IoT environment populated with immobile sensors only without considering node mobility or scalability issues.

Saied et al. [15] proposed a context-aware and multi-service approach for centralized trust management in IoT systems against malicious attacks. However, it requires a centralized trusted entity to collect and disseminate all trust data. Their work does not address how trust data may be collected and how the centralized trusted entity may be implemented to scale to a large number of IoT devices.

Nitti et al. [14] proposed two models for IoT trust management: a subjective model (called SubjectiveTrust) for distributed trust management with each node maintaining its own trust and service experience data, and an objective model (called ObjectiveTrust) with a set of nodes forming a centralized repository for storing trust and service experience data for all the nodes in the system. Both subjective and objective models rely on a friendship social network graph as input to know the "centrality" of a node to another node or all nodes in the network. The distributed trust management protocol (SubjectiveTrust) is used by each individual node to assess its "subjective trust" toward a peer IoT since each node maintains its own data. A node assesses the trust score of another node through a weighted sum of the following three scores: centrality, own service experiences, and recommendations filtered by credibility. The centrality score (in the range of 0 to 1) of *j* as evaluated by *i* is computed by the degree of common friends between *i* and *j*. The credibility score of *k* (a recommender that provides a recommendation to *i* about *j*) as evaluated by *i* is computed by a weighted sum of own service experiences of *i* toward *k* and the centrality score of *k* as evaluated by *i.* For scalability, each node only stores trust information about its neighbor nodes in the social network graph. When a node wants to know the trust value of a remote node, a search procedure is invoked to first find a path leading to the remote node [23] whose trust value is then computed through the trust chain found. While it is scalable in terms of the storage cost because a node only needs to store trust data about its neighbors, it is expensive (and hence is not scalable) in terms of the communication and computational cost and the delay may be intolerable. The objective model (ObjectiveTrust) requires a set of pre-trusted nodes be in place for storing trust information of all nodes in the system. Unlike their subjective model, their objective model assesses the trust score of a node through a weighted sum of the centrality score and the average opinion score (long term and short term) after applying the recommender's credibility score to filter untrustworthy recommendations. Specifically, their objective model computes the centrality score (in the range of 0 to 1) of *j* based on if *j i*s central in the network and if it is involved in many transactions. It computes a recommender's credibility score by taking into consideration of possible collusion attacks. So the credibility score of *k* (a recommender that provides opinions about *i*) is proportional to *k*'s trust score, but inversely proportional to the capability of *k,* the strong object relationship (including ownership, co-location, co-work, social, and parental) between *i* and *k*, and the number of transactions between *i* and *k*. A problem of their objective model is that it is not clear how user-owned IoT devices can serve the role of pre-trusted nodes. Also their objective model essentially is to compute the "objective trust" (common belief or reputation), not the "subjective trust" of an IoT device, so it does not preserve the notion that trust is subjective and is inherently one-to-one. This is especially problematic for IoT systems since IoT devices are owned by humans who have social relationships among themselves and the trust of one user toward another user is inherently one-to-one and subjective.

Relative to [7, 14, 15, 23] our work also leverages centralized entities (i.e., cloud servers) for storing trust data. However, our work preserves the notion of "subjective trust" evaluation despite the fact that trust computation is performed by cloud servers. Unlike [7, 14, 15, 23], we address the issue of user profile and trust data management for large-scale IoT systems, and we propose a scalable *report-and-query* design for supporting scalable mobile cloud IoT trust management. Lastly, unlike [14] which must rely on the existence of a friendship social network graph as input for specifying social relationships, we collect social relationships between each pair of IoT devices dynamically when IoT devices encounter each other. Our social similarity calculation method is more scalable than [14] as it is difficult if not impossible to construct an accurate friendship social network graph when there is a large number of IoT devices arriving and leaving the system dynamically.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2018.2886379, IEEE Transactions on Network and Service Management

Among the contemporary IoT trust management protocols cited above, we select Adaptive IoT Trust [6] and ObjectiveTrust [14] as baseline trust protocols against which IoT-HiTrust is compared for performance analysis. The reason we select Adaptive IoT Trust is that it is a proven protocol for distributed IoT trust management and it outperforms existing P2P trust protocols including EigenTrust [10], PeerTrust [20], and ServiceTrust [17]. The reason we select ObjectiveTrust is that it is the only centralized IoT trust management protocol to-date that considers social standing and relationships for credibility rating and recommendation filtering. Relative to Adaptive IoT Trust and ObjectiveTrust, our work addresses scalability while achieving "subjective trust" evaluation accuracy. Further, we demonstrate that our hierarchical trust management protocol outperforms these two baseline IoT trust protocols with two real-world mobile cloud IoT applications.

III. System Model

A. 3-Tier Cloud-Cloudlet-Device Architecture

IoT-HiTrust is built on top of a 3-tier mobile cloud architecture [28] as illustrated in Figure 1 for hierarchical IoT trust management. IoT devices (labeled as *d*'s) are sitting at the bottom tier of the hierarchy. Cloudlets (labeled as *CL*'s) are at the middle tier. A public cloud comprising a number of "home" cloud servers (labeled as *CS*'s) is at the top tier. A user is assigned to a fixed "home" cloud server based on load balancing reasons and once the assignment is done, a user's home cloud server is not changed dynamically.

An IoT device runs IoT-HiTrust by creating virtual machines (VMs) running on the local cloudlet it is currently under as well as on its home cloud server to solve platform/operating system heterogeneity problems. A user's queries and reports are sent to its local cloudlet's VM which subsequently relays them to the home cloud server's VM. In the literature [28], cloudlets can be either "heavyweight" IoT devices (e.g., PCs, servers, notebooks, etc.) collocated with Wi-Fi hotspots with moderate communication, computational and storage capability, or mobile operator owned powerful base stations. The communication between an IoT device and a local cloudlet therefore is typically through Wi-Fi hotspots in the formal case, and through mobile operator's base stations in the latter case. In this paper for generality we consider the former case. The communication between a cloudlet and the cloud is through the Internet. Using IoT-HiTrust, the cloud can periodically evaluate service trustworthiness of all IoT devices in a cloudlet region and select a group of IoT devices to form the region's cloudlet. In return for the surrogate services provided by these IoT devices, users owning these IoT devices are granted access privileges to cloud resources. We will call these IoT devices selected to govern a region's cloudlet as "cloudlet devices," with the understanding that cloudlet devices are just heavyweight IoT devices.

At the bottom tier sit "lightweight" IoT devices owned by users (e.g., smart phones, sensors, PDAs, etc.). Lightweight IoT devices typically are carried by their owners and can move from one cloudlet to another due to mobility. When a lightweight IoT device is disconnected from the current cloudlet, it can simply connect to a new regional cloudlet for service continuity. Since neither the IoT device nor the local cloudlet stores service trustworthiness data, an IoT device can simply recreate a VM to run on the new local cloudlet without involving VM content transfer from the old regional cloudlet to the new regional cloudlet. To save energy and bandwidth, an IoT device always communicates with the cloud through its regional cloudlet.

Figure 1 shows two cloudlets, CL_1 and CL_2, each with three heavyweight IoT devices serving as cloudlet devices. The "logical" cloud has 5 cloud servers, CS_1, CS_2, CS_3, CS_4 and CS_5. In Figure 1, CS_2 is the home cloud server of user u_2 and CS_3 is the home cloud server of user u_3. Since user u_3 owns two IoT devices, d_{31} and d_{32}, these two IoT devices' home cloud sever is also CS_3. Each cloudlet only relays requests/responses from/to IoT devices under its region. In addition, each cloudlet caches trust information. In case of Internet disconnection, a cloudlet can operate in disconnection mode [16] to answer user queries issued from IoT devices in its region. The regional size covers the radio range to ensure that mobility and instability of radio environments will not be a major factor to prevent nodes under a cloudlet region from communicating directly with the cloudlet. When an IoT device in one cloudlet region moves to another cloudlet region, it performs a registration/deregistration action to the two involving cloudlets. As illustrated in Figure 1, user u_3 moves across the cloudlet boundary, causing deregistration with the old cloudlet CL_1 and registration with the new cloudlet CL_2. A pointer is recorded by CL_1 so that a message for u_3 can be redirected to CL_2.

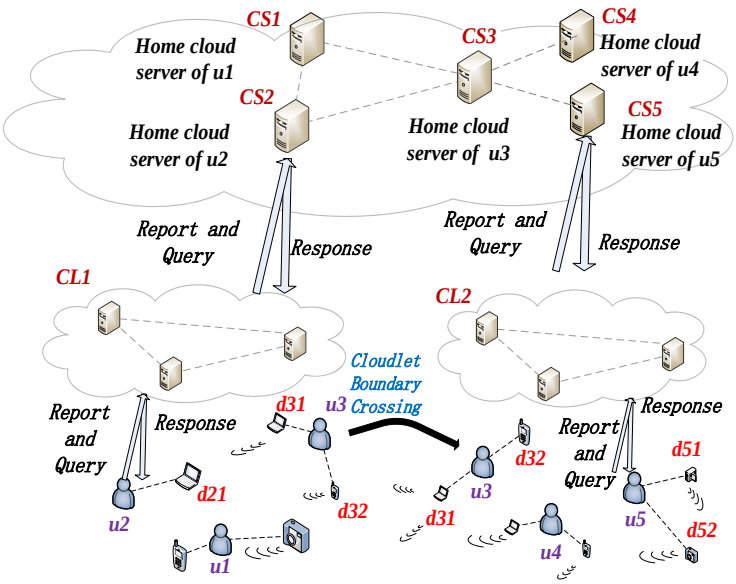

Figure 1: Information Flow in the 3-Tier Cloud-Cloudlet-Device Architecture.

B. Threat Model

By a malicious node, we refer to a node having only self-interest. A smart malicious node can choose to provide good or bad service depending on whether it would benefit itself and its allies (other malicious nodes or friends) which altogether can collude to monopoly service. By trustworthiness of a node in the context of service management, we mean (a) when acting as

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2018.2886379, IEEE Transactions on Network and Service Management

a service provider, whether it can deliver high service quality, and (b) when acting as a service recommender, whether it is truthful in providing the service ratings of service providers it claims to have service experiences on. A malicious node in general can perform communication protocol attacks to disrupt network operations. We assume such attack is handled by intrusion detection techniques [13] and is not addressed in this paper. We are concerned with trust-related attacks that can disrupt trust-based service management.

In this paper we consider a malicious IoT device (because its owner is malicious) capable of performing the following trust-related attacks:

1. *Self-promoting attacks*: a malicious node can promote its importance (by providing good recommendations for itself) for it to be selected as an SP, but then can provide bad or malfunctioned service. We address this attack by not allowing self-recommendations in trust computation (see Section IV.B for detail).
2. *Bad-mouthing attacks*: a malicious node can ruin the reputation of a well-behaved device (by providing bad recommendations against it) so as to decrease the chance of that good device being selected as an SP. Bad-mouthing attack is a form of collusion attack when multiple malicious nodes perform bad-mouthing attacks to a good node to ruin the service trustworthiness of the good node. We address this attack by "social similarity based recommendation filtering" to filter out untrustworthy service rating recommendations (see Section IV.A for detail).
3. *Ballot-stuffing attacks*: a malicious node can boost the reputation of a malicious node (by providing good recommendations) so as to increase the chance of that bad device being selected as an SP. Ballot-stuffing attack is also a form of collusion attack when multiple malicious nodes perform ballot-stuffing attacks to boost the service trustworthiness of one another. We address this attack by "social similarity based recommendation filtering" to filter out untrustworthy service rating recommendations (see Section IV.A for detail).
4. *Discriminatory attacks* (or conflicting behavior attacks): a malicious node can discriminatively attack non-friends or nodes without strong social ties (without many common friends) because of human nature or propensity towards friends in social IoT systems. While serving as a recommender, if the target node is a friend or a malicious node, it can provide a good service recommendation (i.e., ballot stuffing attacks) even if the target node does not provide good service. On the other hand, if the target node is a non-friend, it can provide a bad service recommendation (i.e., bad-mouthing attacks) even if the target node provides good service. We address this attack by considering trust being formed not only from recommendations, but also from self-observations (see Section IV.B for detail). Self-observations are one-to-one in nature, so the service behavior of a malicious node performing discriminatory attacks on a service requester will be remembered by the service requester. Consequently, our trust formation design has inherently addressed discriminatory attacks.
5. *Opportunistic service attacks*: a malicious node can provide good service to gain high reputation opportunistically especially when it senses its trust standing is dropping because of providing bad service. With a good trust standing, it can effectively collude with other bad nodes to perform bad-mouthing and ballot-stuffing attacks. We assume that a malicious node has a low trust score threshold below which it will behave (providing good service) in order to raise its trust standing, and a high trust score threshold above which it will misbehave (providing bad service) to take advantage of its high trust standing for self-gain. We address this attack by "adaptive filtering" (see Section IV.B for detail) which adjusts the weights associated with *direct trust* and *indirect trust* to adapt to new evidence (oscillating service experiences) exhibited from opportunistic service attacks, thus effectively reflecting the true service trustworthiness level of an IoT device dynamically.

IV. IoT-HiTrust Protocol Design

In this section we provide a detailed description of our IoT-HiTrust protocol design for trust-based service management of mobile cloud IoT systems. We first describe our recommendation filtering design based on social similarity. We then describe our report-and-query design for scalability. Then we describe the actions taken by the entities in our 3-tier mobile cloud hierarchy (IoT devices, cloudlets, and cloud servers) for IoT-HiTrust protocol execution. Lastly we describe how we make our 3-tier mobile cloud hierarchy resilient to network disconnection.

A. Recommendation Filtering based on Social Similarity

Our trust model is based on social relationships among human owners of IoT devices. A user upon receiving a recommendation from an IoT device, will measure the trustworthiness of the recommender (or rater) so as to apply "recommendation filtering" based on its social relationships with the recommender (or rater). We consider three core social metrics for measuring social relationships which are multifaceted: friendship (representing intimacy), social contact (representing closeness), and community of interest (representing knowledge and standard on the subject matter). The idea is that two users sharing similar social relationships are likely to have similar views towards services provided by a trustee IoT device. Social relationships between owners are translated into social relationships between IoT devices as follows:

1. *Friendship*: Each owner has a list of friends (i.e., other owners), representing its social relationships. This friendship list varies dynamically as an owner makes or denies other owners as friends. If the owners of two IoT devices are friends, then it is likely they will be cooperative with each other. The friendship list contains only direct friends of an owner, not friends of a friend. Friendship is measured by the degree of commonality of direct friends. That is, if two owners have about the same set of direct friends (including each other),

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2018.2886379, IEEE Transactions on Network and Service Management

they have a strong tie in friendship.

2. *Social Contact*: A device may be carried or operated by its owner in certain environments (e.g., at work, school or home). Two devices have high social contact opportunities when their owners go to the same locations. The social contact relationship is measured by the degree of commonality of social contacts.
3. *Community of Interest* (CoI): Each owner has a list of communities of interest such as health, sport, travel, etc. Nodes belonging to a similar set of communities likely share similar interests or capabilities [10]. The CoI relationship is measured by the degree of commonality of communities of interest.

To facility measuring social similarity with other owners, an IoT device belonging to user u_x maintains the following three lists (F, S, C) in its profile (as illustrated in Figure 2):

1. Friends of u_x, denoted by a set $F_x = \{u_1, u_2, \dots\}$;
2. Locations that u_x frequently visited for social contact, denoted by a set $S_x = \{Loc_1, Loc_2, \dots\}$;
3. Communities of interest that u_x is a member of, denoted by a set $C_x = \{CoI_1, CoI_2, \dots\}$.

A user may designate one of its IoT devices to update such information and share the information with other IoT devices it owns. Social similarity is inherently between two users, but it is propagated to IoT devices owned by the two users, so social similarity is also between two devices.

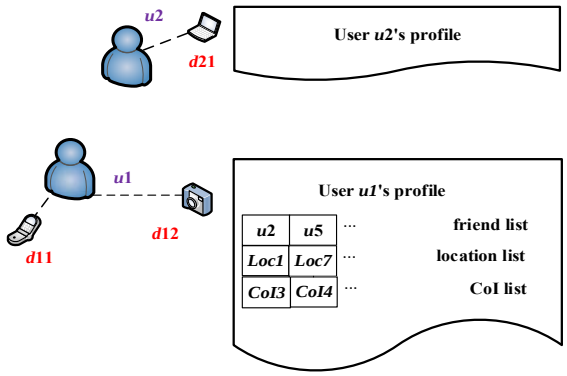

Figure 2: Each User Stores Its Friend, Location, and CoI Lists for Detecting Social Similarity with other Users.

B. Report-and-Query Design for Scalability

We propose a simple report-and-query design for an IoT device to communicate with its local cloudlet for trust-based service management.

1) Report Design with Privacy Considerations

An IoT device (on behalf of its owner) dynamically collects and reports to the cloud two pieces of sensitive information about other devices: (a) a service rating $f_{x,i}(t)$ in the range of 0 to 1 toward device d_i's service quality after a service is received and rated by its owner (user u_x) at time t, and (b) a social similarity score in the range of 0 to 1 toward another user acting as a recommender. The former is to assess the trustworthiness of the service quality of a service provider device based on self-experience. The latter is to assess the trustworthiness of a recommender who recommends the service quality of a service provider. No other information about other devices is kept in an IoT device. Moreover, these two pieces of information are reported to the cloud and saved in the cloud. As a result, an IoT device will not store service trustworthiness data of another IoT device. Consequently, when a device is compromised, only the private/sensitive information regarding the owner of the compromised IoT device itself will be leaked out through its user profile which contains the owner's friendship, social contact, and CoI information.

A user would not want to reveal its social relationship data to another user when they encounter each other and exchange their friendship, social contact, and CoI information to compute mutual social similarity. This is achieved by our privacy-preserving social similarity computation design described below. Each user maintains its (F, S, C) profile separately. When user u_x encounters user u_y, they exchange their (F_x, S_x, C_x) and (F_y, S_y, C_y) profiles to measure their mutual social similarity. To preserve energy, they can exchange the profile information the very first time they encounter or periodically. This is especially so if user profile information does not change much over time. To preserve privacy, they only want to reveal common elements in the *F, S,* and *C* lists (if any) but do not want to let the other party know their entire (F, S, C) profile. To achieve this, users u_x and u_y can first authenticate each other using standard PKI. User u_x can then use a cryptographic hash function in combination with a secret session key *K* (established via PKI during user authentication) to generate a hash-based message authentication code HMAC(*K, p*) for $p \in (F_x, S_x, C_x)$ and then transmit HMAC(*K, p*) along with HMAC(*K*, HMAC(*K, p*)) to u_y. When u_y receives the message, it can unilaterally generate HMAC(*K*, HMAC(*K, p*)) using HMAC(*K, p*) sent by u_x. If this matches with HMAC(*K*, HMAC(*K, p*)) sent by u_x, then u_y verifies the message received is indeed sent by u_x. Then u_y can compare HMAC(*K, p*) with HMAC(*K, q*) for $q \in (F_y, S_y, C_y)$. If HMAC(*K, p*)=HMAC(*K, q*) then *p*=*q* and a common friend, location, or CoI (corresponding to *F, S,* or *C*) is identified. If HMAC(*K, p*)≠HMAC(*K, q*), it prevents the identities of uncommon friends/locations/CoIs from being revealed.

We adopt "cosine similarity" to measure the distance of two social relationship lists (see Figure 2), with 1 representing complete similarity and 0 representing no similarity. The physical meaning of cosine similarity is the cosine of the angle between the two vectors deriving from the two lists, with the cosine value of 1 meaning totally identical lists and the cosine value of 0 meaning totally non-overlapping lists. Computational efficiency is the main reason why we choose cosine similarity to measure the similarity of two lists in high-dimensional positive spaces because of limited computational capacity of IoT devices. Specifically, the following three similarity metrics are measured as follows:

- **Friendship Similarity** (sim_f): The friendship similarity is a powerful social relationship (intimacy) for screening recommendations. After two users u_x and u_y exchange their friend lists, F_x and F_y, they compute the "cosine similarity" sim_f as follows:

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2018.2886379, IEEE Transactions on Network and Service Management

$$sim_f(u_x, u_y) = \frac{|F_x \cap F_y|}{\sqrt{|F_x| \cdot |F_y|}} \quad (1)$$

Where the notation $|A|$ represents the cardinality of set A.

- **Social Contact Similarity** (sim_s): The social contact similarity represents closeness and is an indication if two nodes share the same location-based social contacts (e.g., co-workers at work, classmates at school, and co-residents at home) and thus share the same sentiment towards devices which provide the same service. The operational area could be partitioned into sub-grids. User u_x records the IDs of sub-grids it has visited in its *location list* S_x for social contact. After two users u_x and u_y exchange their location lists, S_x and S_y, they could compute sim_s in the same way of computing sim_f as follows:

$$sim_s(u_x, u_y) = \frac{|S_x \cap S_y|}{\sqrt{|S_x| \cdot |S_y|}} \quad (2)$$

- **Community of Interest Similarity** (sim_c): Two users in the same CoI share similar social interests and most likely have common knowledge and standard toward a service provided by the same device. Also very likely two users who have used services provided by the same IoT device can form a CoI (or are in the same CoI). After two users u_x and u_y exchange their device lists, C_x and C_y, they could compute sim_c as follows:

$$sim_c(u_x, u_y) = \frac{|C_x \cap C_y|}{\sqrt{|C_x| \cdot |C_y|}} \quad (3)$$

The above three social similarity measures (sim_f, sim_s, sim_c) are computed upon the encountering event of user u_x with user u_y, and are reported by user u_x and user u_y through their local cloudlets to the home cloud servers of user u_x and user u_y, respectively.

When the home cloud server of u_x receives $sim_i(u_x, u_y)$, $i \in \{f, s, c\}$, from user u_x which just encounters user u_y, it computes the social similarity between users u_x and u_y (who now serves as a rater or recommender) as a weighted combination of all social similarity metrics as follows:

$$sim(u_x, u_y) = \sum_{i \in \{f,s,c\}} w_i \cdot sim_i(u_x, u_y) \quad (4)$$

where $0 \le w_f, w_s, w_c \le 1$, with $w_f + w_s + w_c = 1$, are social similarity weight parameters to be dynamically adjusted by IoT-HiTrust to maximize trust protocol performance.

2) Query Design

Whenever a user wants to know the trust value of an IoT device, it simply sends a query to its home cloud server. For example, in Figure 1, u_2 will send a query to its home cloud server CS_2 to know its "subjective trust" toward d_{31} which belongs to u_3.

Let $t_{x,i}$ denote the "subjective trust" of user u_x toward d_i. The home cloud server of u_x computes $t_{x,i}$ by combining $u_x's$ direct trust toward d_i ($t_{x,i}^d$) based on self-observation rating reports, and $u_x's$ indirect trust toward d_i ($t_{x,i}^r$) based on other users' rating reports, as follows:

$$t_{x,i} = \mu_{x,i} \cdot t_{x,i}^d + (1 - \mu_{x,i}) \cdot t_{x,i}^r \quad (5)$$

Here, $\mu_{x,i}$ is a weight parameter ($0 \le \mu \le 1$) to weigh the importance of direct trust relative to indirect trust. The selection of $\mu_{x,i}$ is critical to trust evaluation. We apply adaptive filtering developed in [6] to adjust $\mu_{x,i}$ dynamically to effectively cope with malicious attacks and to improve trust accuracy.

The direct trust $t_{x,i}^d$ in Equation 5 is computed by Beta Reputation [9] under which the trust value is modeled as a random variable in the range of [0, 1] following the Beta (α, β) distribution. The numbers of positive and negative experiences are modeled as binomial random variables. Since the beta-binomial is a conjugate pair, this leads to a posterior beta distribution with updated parameters. Specifically, we can calculate direct trust of u_x toward device d_i, $t_{x,i}^d$, as follows:

$$t_{x,i}^d = \alpha/(\alpha + \beta) \quad (6)$$

where α is the amount of positive service experience with time decay and β is the amount of negative service experience with time decay, calculated as $\alpha = \sum_{all} f_{x,i}(t)\, e^{-\lambda_d(t_{now}-t)}$ and $\beta = \sum_{all}(1 - f_{x,i}(t))\, e^{-\lambda_d(t_{now}-t)}$ where $f_{x,i}(t)$ is a service rating received from user u_x at time t about d_i's service quality, t_{now} is the current time, and λ_d is the decay parameter to discount old service experiences. Here $f_{x,i}$ contributes to positive service experience and $1 - f_{x,i}$ contributes to negative service experience. If $t = t_{now}$ then the service rating has the highest credibility of 1; otherwise, the credibility of the service rating decays over time exponentially. The summation is over all service ratings received from user u_x, including old and new service ratings maintained in user $u_x's$ cloud server.

To compute indirect trust of u_x toward device d_i, $t_{x,i}^r$, the home cloud server of u_x first locates social similarity records $sim(u_x, u_z)'s$ in its local storage. The home cloud server of u_x then selects top-n_r raters with the highest similarity scores with u_x among all and calculates the indirect trust ($t_{x,i}^r$) towards device d_i as follows:

$$t_{x,i}^r = \sum_{u_z \in U} \frac{sim(u_x, u_z)}{\sum_{u_w \in U} sim(u_x, u_w)} \cdot t_{z,i}^d \quad (7)$$

The indirect trust model as indicated in Equation 7 is essentially a weighted sum of service ratings reported by other IoT devices (acting as recommenders) with a higher weight giving to a recommender with a higher social similarity. It is based on a widely accepted concept that mobile IoT systems can be characterized as a hybrid of P2P and social networks and IoT trust management must take into account social relationships among device owners in order to maximize protocol performance [6, 14]. Here, U is a set of up to n_r raters whose $sim(u_x, u_z)$ scores are the highest, $u_z \in U$ is a rater selected, and $t_{z,i}^d$ is the service rating provided by u_z toward device d_i. We note that $t_{z,i}^d$ is stored in the home cloud server of u_z but it is obtainable after the home cloud server of u_x communicates with the home cloud server of u_z. In Equation 7, the service rating provided

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2018.2886379, IEEE Transactions on Network and Service Management

from u_z toward d_i $(\text{i.e.}, t_{z,i}^d)$ is weighted by the ratio of the similarity score of u_x toward u_z to the sum of the similarity scores toward all raters. That is, if the similarity score of u_x toward u_z is high relative to that of u_x toward other raters, then the home cloud server of u_x will put a relatively high weight on the rating $t_{z,i}^d$ provided by u_z to compute $t_{x,i}^r$.

C. Protocol Execution Description

In this section, we elaborate in detail the actions taken by the entities in our 3-tier mobile cloud hierarchy (IoT devices, cloudlets, and cloud servers) while executing IoT-HiTrust. For user u_x, we use CL_x and CS_x to refer to its local cloudlet and home cloud server, respectively, in the 3-tier mobile cloud hierarchy. We note that while CS_x is fixed, CL_x is dynamically changed as the user roams from one region to another.

Figure 3 shows a flowchart of our trust management protocol execution from the perspective of user u_x. As illustrated in Figure 3, the actions performed by u_x are as follows:

(1) When u_x just receives a service completed by d_i belonging to u_y, u_x (using its primary IoT device) reports to CS_x through CL_x a service rating report ($f_{x,i}$ along with the timestamp at which service is rendered) indicating the extent to which it is satisfied with the service provided by d_i.

(2) When u_x encounters u_y, they exchange their (F_x, S_x, C_x) and (F_y, S_y, C_y) profiles so as to measure their mutual social similarity $sim(u_x, u_y)$ (Equations 1, 2, 3 and 4). Then u_x and u_y report it to their respective home cloud servers CS_x and CS_y through their local cloudlets CL_x and CL_y, respectively. Energy constrained devices may report $sim(u_x, u_y)$ only the first time or periodically to conserve energy.

(3) When u_x wants to know service trustworthiness of d_i belonging to u_y, u_x (using its primary IoT device) sends a query to CS_x through CL_x and waits for a reply back from CL_x returning the trust score of d_i.

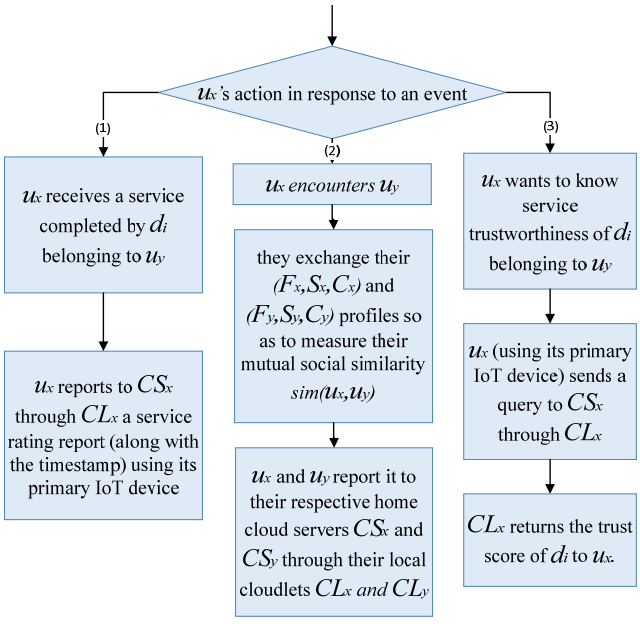

Figure 3: Action Flowchart for User u_x.

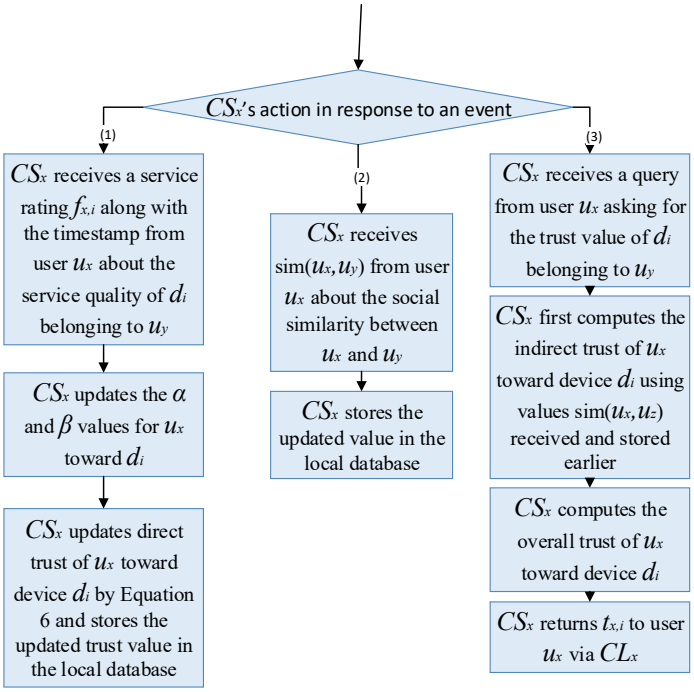

Figure 4: Action Flowchart for Cloud Server CS_x.

Figure 4 shows a flowchart of our trust management protocol execution from the perspective of CS_x, the home cloud server of user u_x. As illustrated in Figure 4, the actions performed by CS_x are as follows:

(1) When CS_x receives a new service rating $f_{x,i}(t)$ along with the timestamp t from user u_x about the service quality of d_i belonging to u_y, CS_x updates α and β values for u_x toward d_i based on $u_x's$ own old and new service ratings toward d_i with trust decay over time. Then, CS_x updates direct trust of u_x toward device d_i $(t_{x,i}^d)$ by Equation 6 and stores the updated $t_{x,i}^d$ value in the local database.

(2) When CS_x receives $sim(u_x, u_y)$ from user u_x about the social similarity between u_x and u_y, CS_x stores the updated $sim(u_x, u_y)$ value in the local database.

(3) When CS_x receives a query from user u_x asking for the trust value of d_i belonging to u_y, CS_x first computes the indirect trust of u_x toward device d_i $(t_{x,i}^r)$ by Equation 7 using $sim(u_x, u_y)$ values received earlier and stored in its local database. Then, CS_x computes the overall trust of u_x toward device d_i $(t_{x,i})$ by Equation 5 using $t_{x,i}^d$ and $\mu_{x,i}$ values stored in the local database. Lastly, CS_x returns $t_{x,i}$ to user u_x via CL_x.

Figure 5 shows a flowchart of our trust management protocol execution from the perspective of CL_x, the local cloudlet of user u_x. As illustrated in Figure 5, the actions performed by CL_x are as follows:

(1) When CL_x receives a service rating $f_{x,i}$ along with the timestamp from user u_x about the service quality of d_i belonging to u_y, CL_x forwards it to $u_x's$ home cloud server CS_x.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2018.2886379, IEEE Transactions on Network and Service Management

(2) When CL_x receives $sim(u_x, u_y)$ from user u_x about the social similarity between u_x and u_y, CL_x forwards it to $u_x's$ home cloud server CS_x.

(3) When CL_x receives a query from user u_x asking for the trust value of d_i belonging to u_y, CL_x forwards the query to $u_x's$ home cloud server CS_x. Then, after receiving $t_{x,i}$ as a reply from CS_x, CL_x forwards it to user u_x.

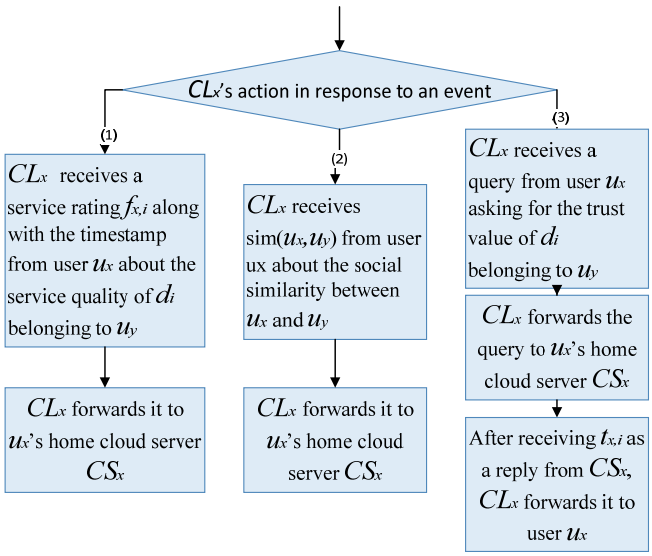

Figure 5: Action Flowchart for Cloudlet CL_x.

D. Dealing with Network Disconnection

A cloudlet may lose connectivity with the cloud if all heavyweight IoT devices selected as the cloudlet devices for the cloudlet experience network disconnection in emergency situations such as a network service disruption. A cloudlet must maintain service continuity to IoT devices during network disconnection. In case the cloudlet can still communicate with a neighbor cloudlet which still has connectivity with the cloud, then all reports, queries, and responses can route through the neighbor cloudlet. Otherwise, it will have to operate in disconnected mode [16] using cached data. For a heavy IoT device that is typically not mobile or only moves within the region, the home regional cloudlet has all the data it needs for answering a user query regarding the trustworthiness of the IoT device, since it caches all reports, responses, and social similarity reports with the IoT device which most of the time stays within the local cloudlet region. So all the local cloudlet has to do is to follow the computational procedure described earlier to assess the trustworthiness of the IoT device, as if the computation is performed by the cloud itself. For a lightweight IoT device, the home region cloudlet may lose some precision in estimating the trustworthiness of the IoT device because the local cloudlet may not have all the data needed. For example, the local cloudlet may miss some reports of a lightweight IoT device when the IoT device moves away from its region. The trust accuracy is largely affected by the mobility of the user carrying the IoT device. In Section V we will assess the extent to which the trust accuracy is impacted under various mobility scenarios.

E. System Responsiveness

Our solution greatly improves responsiveness of the system for two reasons. First, all IoT devices in the system independently and concurrently perform trustworthiness evaluation toward service providers or recommenders they encounter and report the trustworthiness data to the cloud through local cloudlets, as explained in Section IV.C and illustrated in Figures 3, 4, and 5. As a result, trustworthiness data can be quickly accumulated in the cloud concurrently by all the IoT devices in the system, allowing the system to quickly answer a query regarding the service trustworthiness of a device. Secondly, our solution does not rely on the existence of a social network graph as input for specifying social relationships. Such approaches (e.g., [14, 23]) require the cloud to collect social and interaction relationships information from individual IoT devices dynamically in order to maintain an accurate and up-to-date social network graph. The amount of traffic generated by cloud-to-IoT query messages and IoT-to-cloud reply messages due to a large number of IoT devices simultaneously communicating with the cloud will not only consume IoT energy but also cripple the cloud communication network and reduce system responsiveness. Our solution collects social relationships between each pair of IoT devices dynamically only when IoT devices encounter each other. As a result, only IoT-to-cloud report traffic is being generated sparingly at the encountering moments, as explained in Section IV.B.1 and illustrated in Figure 3. This avoids a huge amount of traffic from being generated simultaneously by a large number of IoT devices and thus greatly improves system responsiveness. In Section V.A we perform a complexity analysis of the communication and storage cost to back up our claim that our 3-tier cloud-cloudlet-device hierarchical trust-based service management protocol provides adequate system responsiveness. Further improvement in responsiveness is possible by applying intelligent cache management (e.g., as in [6]) so that cloudlets cache trustworthiness data for IoT devices (acting as a service provider or as a recommender) under their regions, thereby allowing a query regarding the service trustworthiness of a local IoT device to be answered by a local cloudlet using the cached trustworthiness data, without having to route the query to the cloud for query processing. The topic of cache coherence protocol design is outside the scope of this paper.

V. IoT-HiTrust Protocol Performance

In this section, we analyze IoT-HiTrust system performance. We first perform a complexity analysis of the communication and storage cost for evaluating scalability. Then we conduct a performance analysis using ns-3 network simulator [24]. We compare IoT-HiTrust with two baseline trust protocols, Adaptive IoT Trust [6] and ObjectiveTrust [14]. See Section II for the detail of these two baseline trust protocols chosen for our comparative analysis.

A. Scalability Analysis

Table 1 summarizes the complexity analysis results. We evaluate the scalability of IoT-HiTrust against Adaptive IoT Trust [6] and ObjectiveTrust [14] based on the complexity analysis results.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2018.2886379, IEEE Transactions on Network and Service Management

Table 1: Complexity Analysis of IoT-HiTrust against Adaptive IoT Trust [6] and ObjectiveTrust [14].

Protocol	Communication Complexity	Storage Complexity
IoT-HiTrust	$O(\sum_{j=1}^{N_T} \lambda_{ij} + q_{ij} + \pi_{ij})$	$O(1)$
Adaptive IoT Trust	$O(\sum_{j=1}^{N_T} \pi_{ij})$	$O(N_T)$
ObjectiveTrust	$*O\left(\sum_{i=1}^{N_T} \lambda_{ij} + q_{ij}\right)$	$O(1)$

$*$assuming existence of a global friendship social network graph as input.

Communication Cost Complexity: For IoT-HiTrust, the communication cost per IoT device (from node *i*'s perspective) is of complexity $O(\sum_{j=1}^{N_T} \lambda_{ij} + q_{ij} + \pi_{ij})$ where N_T is the number of IoT devices in the system, λ_{ij} is the service request rate of node *i* to node *j*, q_{ij} is the query rate of node *i* about node *j*'s trust status, and π_{ij} is the encountering rate of node *i* with node *j* which can be derived by analyzing the encounter or interaction pattern, e.g., a power-law distribution, as supported by the analysis of many real traces [27]. Upon completing a service from node *j*, node *i* sends its service quality assessment toward node *j* to its home cloud server. When node *i* wishes to find out if node *j* is trustworthy, it sends a query to its home cloud server. Lastly, upon encountering node *j*, node *i* exchanges its user profile with node *j's* user profile while preserving privacy to compute three social similarity measures (sim_f, sim_s, sim_c) between node *i* and node *j* according to Equations 1, 2, and 3, after which node *i* sends (sim_f, sim_s, sim_c) to its home cloud server for storage. The one extra message needed for sending computed social similarity values to its home cloud server does not change the complexity.

For ObjectiveTrust [14], the communication cost per IoT device (from node *i*'s perspective) is of complexity $O\left(\sum_{j=1}^{N_T} \lambda_{ij} + q_{ij}\right)$. Upon completing a service from node *j*, node *i* sends its service quality assessment toward node *j* to the cloud. When node *i* wishes to find out if node *j* is trustworthy, it sends a query to the cloud. There is no need for node *i* to exchange social relationships upon encountering each other, or report the social similarity between node *i* and node *j* to the cloud because ObjectiveTrust assumes the existence of a friendship social network graph as input for specifying social relationships (e.g., centrality and credibility) and such information is already loaded in the cloud. Due to this assumption, ObjectiveTrust's communication cost complexity is lower than that of IoT-HiTrust by $O(\sum_{j=1}^{N_T} \pi_{ij})$.

For Adaptive IoT Trust [6], the communication cost per node (from node *i*'s perspective) is $O(\sum_{j=1}^{N_T} \pi_{ij})$ because upon node *i* encountering node *j*, node *i* exchanges its user profile with node *j's* user profile for computing social similarity and also exchanges its service quality experiences with node *j's* service quality experiences toward other nodes in the system for computing trust scores toward other nodes in the system. There is no communication cost for node *i* to send its own service quality assessment results and social similarity results to the cloud because node *i* stores all service quality assessment results (including from itself and from all other nodes) as well as social similarity results in its local storage. When node *i* wishes to know node *j*'s trust status it simply looks up its trust data in the local storage. Compared with IoT-HiTrust, Adaptive IoT Trust communication cost complexity is lower by $O\left(\sum_{j=1}^{N_T} \lambda_{ij} + q_{ij}\right)$ since node *i* sends neither trust data nor trust status queries to the cloud.

Storage Cost Complexity: For IoT-HiTrust, the storage cost per node (from node *i*'s perspective) is O(1) for storing its own user profile only (i.e., friend, social contact, and community-interest lists) because all service quality experience, social similarity, and trust data are stored in the cloud. The storage cost per cloud server is $O(N_T/N_C)$ for storing service quality experiences, social similarity, and trust data of N_T/N_C devices, where N_T is the number of IoT devices and N_C is the number of cloud servers, as the load is shared by all cloud servers (through the use of a fair hash function).

For ObjectiveTrust [14], the storage cost per node is also O(1) because all data are also stored in the cloud.

For Adaptive IoT Trust [6], the storage cost per node is $O(N_T)$ because for every other IoT device, a storage space is needed for storing service quality experiences, social similarity, and trust data. Apparently Adaptive IoT Trust is not scalable when N_T is sufficiently large.

Scalability Evaluation: Adaptive IoT Trust is not scalable in the storage cost when N_T is sufficiently large. IoT-HiTrust and ObjectiveTrust are both scalable in the communication and storage cost. However, IoT-HiTrust achieves better scalability than ObjectiveTrust for two reasons: (a) with the hierarchical mobile cloud IoT architecture under IoT-HiTrust, an IoT device only communicates with its local cloudlet over a short radio-range distance when forwarding service quality experience results and social similarity information to the cloud or querying trust data from the cloud, whereas under ObjectiveTrust, an IoT device communicates with the cloud over a long haul distance; (b) IoT-HiTrust collects social relationships between each pair of IoT devices dynamically when IoT devices encounter each other, while ObjectiveTrust must rely on the existence of a friendship social network graph for specifying social relationships. Point (b) above is especially problematic for scalability of ObjectiveTrust because it will be costly if not impossible to obtain a global social network graph with a huge number of IoT devices changing their social profiles dynamically.

Table 2: Parameter List for Performance Evaluation.

Parameter	Meaning	Default
N_T	Number of IoT devices	2000
N_u	Number of users	500
N_C	Number of cloud servers	10

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2018.2886379, IEEE Transactions on Network and Service Management

P_M	% of malicious users	20-40%
P_C	% of high centrality nodes	0-40%
λ	Service request rate per node	1/day
λ_d	Time decay parameter for service rating	10^{-4}
σ_c	Standard deviation of error	1%
$m \times m$	Cloudlet regional area	16×16
$SWIM_S$	SWIM slope	1.45
$SWIM_{pt}$	SWIM maximum pause time	4 hrs
$SWIM_{npp}$	SWIM # popular places per node	[10, 25]
$n_{f,lc}$	# friends per low-centrality node	[10, 50]
$n_{f,hc}$	# friends per high-centrality node	[100, 500]
N_{CoI}	# communities of interest	50
n_{CoI}	# communities of interest per node	[10, 25]
n_r	# of recommenders accepted	10
$w_f: w_s: w_c$	Weight ratio of social relations	1/3:1/3:1/3

B. Environment Setup for Simuation Evaluation

The experiment setup follows the model parameters as listed in Table 2. We select the values for the parameters listed in Table 2 to model a relatively large IoT system to illustrate scalability. We consider a large IoT $m \times m = 16 \times 16$ cloudlet regional area with N_T = 2000 IoT devices, where each cloudlet region (location) is also a square with the width and height equal to wireless radio range so that nodes within a grid can communicate with the local cloudlet in the grid. The boundary locations are wrapped around (i.e., a torus is assumed) to avoid end effects. These IoT devices are randomly assigned to N_U = 500 users, with each user having 4 devices on average. The number of cloud servers in the data cloud center is N_C = 10 such that each cloud server can approximately handle N_T/N_C = 200 IoT devices. Note that N_C = 10 is an arbitrary choice, so in the simulation we know which user/device is map to which cloud server.

The % of malicious users (P_M) is a model parameter whose effect will be analyzed via a sensitivity analysis. Malicious users are chosen randomly from N_U = 500 users. A malicious user will perform self-promoting, bad-mouthing, ballot-stuffing, discriminatory, and opportunistic service attacks as described in Section II.C. In particular, a malicious user u_y can provide a bad recommendation $t_{y,i}^d$=0 (see Equation 7) against a good device *i* for bad-mouthing attacks, and conversely a good recommendation $t_{y,i}^d$=1 for a malicious device *i* for ballot-stuffing attacks. Our protocol handles ballot-stuffing and bad-mouthing attacks by recommendation filtering (see Section IV.A) during the computation of indirect trust $t_{x,i}^r$ based on Equation 7.

The % of high centrality users (P_C) is also a model parameter whose effect will be analyzed. Users are connected through social networks represented by a friendship matrix and a CoI matrix where an entry at (*x, y*) is 1 means that user *x* and user *y* are friends and members of a CoI, respectively. Each low centrality user has $n_{f,lc}$ = [10, 50] friends, and each high centrality user has $n_{f,hc}$ = [100, 500] friends populated randomly. There are N_{CoI} = 50 CoIs and each user has n_{CoI} = [10, 25] CoIs.

We consider all users moving according to the small world in motion (SWIM) mobility model [12], modeling human social behaviors for the purpose of assessing the social contact similarity metric between each pair of users. In SWIM [12], a node has a home location and $SWIM_{npp}$= [10, 25] popular places populated randomly out of the $m \times m$ locations in the system. A node makes a move to one of the population places based on a prescribed pattern. The probability of a location being selected is higher if it is closer to the node's home location or if it has a higher *popularity* (visited by more nodes). When reaching the destination, the node pauses at the destination location for a period of time following a bounded power law distribution [27]. We set the slope of the SWIM mobility model ($SWIM_S$) to 1.45 (as in [12]) and the upper-bound pause time ($SWIM_{pt}$) to 4 hours. The encounter time interval for any two nodes is a bounded power-law distribution between [10 minutes, 2 days], which models the social contact behavior of any two nodes.

Three IoT devices in a region are selected as cloudlet devices periodically responsible for caching and relaying service experience reports, trust score queries, and responses for IoT devices in the region. Direct trust of node *i* toward node *j* is assessed upon completion of a service request from node *i* to node *j*. Each node requests services from a selected device with a time interval following an exponential distribution with parameter λ, with 1/day being the default unless otherwise specified. The trust update interval Δt is 2 hours. The system runs continuously although trust convergence is achieved in less than 200 hours.

The user satisfaction levels of service experience (i.e., $f_{x,i}$ in the range of [0, 1] from user u_x about d_i's service quality) are from a real web service dataset [21] and are used as "ground truth" based on which the accuracy of our trust protocol is assessed. Since the direct trust of user u_x toward service provider d_i (i.e., $t_{x,i}^d$) is calculated by Equation 5 with "ground truth" user service experiences as input, $t_{x,i}^d$ essentially is equal to ground truth. However, we account for the presence of noise in the IoT environment (i.e., error of assessing user satisfaction level received) by considering a standard deviation parameter σ_c (set to 1% as default) to reflect the deviation of the actual user satisfaction level recorded in the database from the direct trust evaluation outcome $t_{x,i}^d$. The decay parameter λ_d is set to 10^{-4} as in [26] for heavy discounting old experiences. Initially, $t_{x,i}$ is set to 0.5 (ignorance) by user u_x for all *i*'s. Then, trust is updated dynamically as nodes encounter each other, as services are requested and rendered, and as trust feedback are acquired. n_r=10 for the set size of U in Equation 7 for calculating $t_{x,i}^r$. We consider $w_f = w_l = w_c = 1/3$ considering friendship, social contact, and community of interest are equally important.

C. IoT-HiTrust Performance Characteristics

In this section, we report ns-3 simulation results on IoT-HiTrust performance characteristics in response to user and environment dynamics.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2018.2886379, IEEE Transactions on Network and Service Management

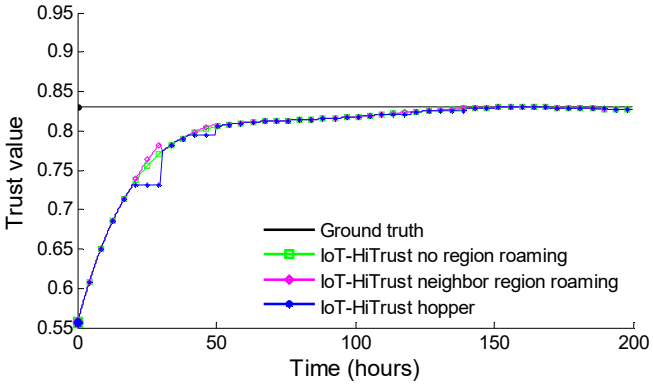

Figure 6: Trust Value of a Good Node under Intermittent Disconnection.

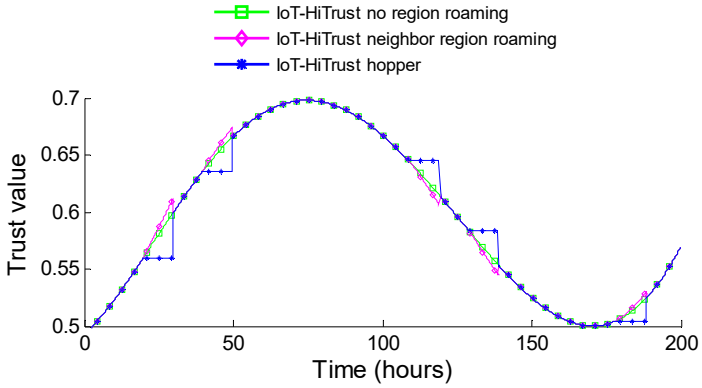

Figure 7: Trust Value of a Bad Node under Intermittent Disconnection.

Figure 6 shows IoT-HiTrust performance in terms of trust accuracy, convergence, and resiliency against attacks. It shows the progressive trust value of a "good" IoT device as assessed by the cloudlet in the home region where the IoT device initially resides vs. time for the case in which P_M = 30%. The "ground truth" trust value of this good device is depicted by the solid line at 0.83. Note that the "ground truth" trust value of this good node is not 1 because $f_{x,i}$ (service satisfaction experience of user u_x toward d_i) retrieved from the trace dataset [21] is not necessarily 1. The progressive trust value measured by IoT-HiTrust is depicted by dashed lines. We consider the scenario in which the "good" IoT device's cloudlet is disconnected due to network disconnection at times (marked by zig-zag patterns) during which it can only perform disconnected trust assessment. The label "no region roaming" means that the target IoT device stays in the same region all the time. The label "neighbor region roaming" means that the target IoT device stays in the home region but roams to neighbor regions from time to time. The label "hopper" means that the target IoT device moves across region boundary frequently. We see that trust accuracy decreases as the target "good" IoT device moves across the regional boundary more frequently. However, all cases eventually converge after sufficient data are collected to allow accurate trust assessment.

Correspondingly, Figure 7 shows IoT-HiTrust performance in terms of the trust value of a "bad" IoT device vs. time when P_M = 30%. This bad node's trust value is up and down because of opportunistic service attacks performed by the bad node. We again confirm that trust accuracy decreases as the target "bad" IoT device moves across the regional boundary more frequently. However, trust accuracy is restored as soon as the home regional cloudlet is reconnected. Figures 6 and 7 verify that our IoT-HiTrust protocol is effective to deal with intermittent disconnection for providing service continuity, especially for IoT devices that do not move much such as heavyweight IoT devices.

In summary, relative to prior work [2, 3, 4, 5, 6], Figures 6 and 7 demonstrate that our hierarchical trust management achieves efficiency and scalability, without compromising trust accuracy.

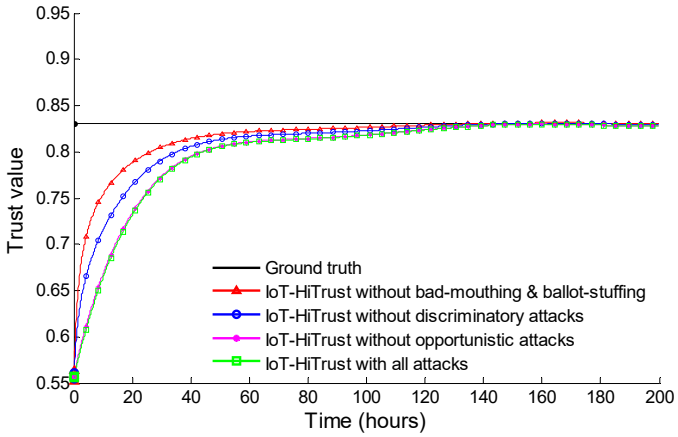

Figure 8: Trust Value of a Good Node under Various Attack Types.

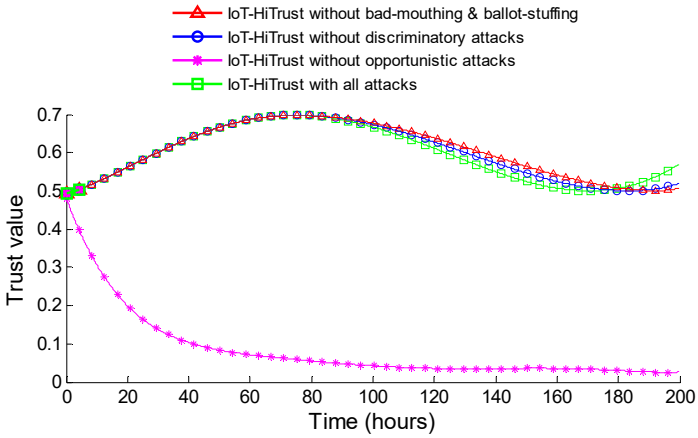

Figure 9: Trust Value of a Bad Node under Various Attack Types.

We test the sensitivity of the results w.r.t. the attack type in Figures 8 and 9.

Figure 8 demonstrates the effect of the attack type on trust accuracy, convergence, and resiliency of a "good" target node that roams between neighbor regions. The curve labeled by

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2018.2886379, IEEE Transactions on Network and Service Management

"with all attacks" (green curve) corresponds to the "neighbor region roaming" curve in Figure 6. There are three other curves, i.e., "without bad-mouthing and ballot-stuffing attacks," "without discriminatory attacks," and "without opportunistic attacks," each with a particular attack type being removed so that we can see the effect of its absence on accuracy, convergence, and resiliency. Figure 8 indicates that recommendation attacks (i.e., bad-mouthing and ballot-stuffing attacks combined) have the biggest effect on accuracy, convergence, and resiliency because without them (red curve) the system can approach ground truth (black curve) in the shortest amount of time. Discriminatory attacks also have some effect on accuracy, convergence, and resiliency because without them (blue curve) the system can converge faster than with them (green curve), although the sensitivity is not as high. Lastly, opportunistic attacks have the least effect on accuracy, convergence, and resiliency because without them (pink curve) the system does not converge any faster than with them (green curve). The reason is that a bad node performing opportunistic attacks will only affect its own trust value, not the "good" target node's trust value. Therefore, the most severe attack type is the recommendation attack as it will ruin the reputation of the "good" target node.

Figure 9 demonstrates the effect of the attack type on trust accuracy, convergence, and resiliency of a "bad" node that roams between neighbor regions. Similarly the curve labeled by "with all attacks" (green curve) corresponds to the curve in Figure 7. There are three other curves, i.e., "without bad-mouthing and ballot-stuffing attacks," "without discriminatory attacks," and "without opportunistic attacks," each with a particular attack type being removed so as to see its absence on accuracy, convergence, and resiliency. Since the target node is a "bad" node, we see from Figure 9 that opportunistic attacks have the most severe effect on accuracy, convergence, and resiliency because without them (pink curve) the system converges to ground truth (zero for a bad node) while with them the system converges to the up and down curve (green curve). This is so because the "bad" target node performing opportunistic service attacks will alternate between behave and misbehave to keep its trust value between the high threshold and low threshold. We see from Figure 9 that the other two attack types do not have a high impact on the trust value of this "bad" target node. In practice, a bad node will always perform opportunistic service attacks to disguise itself as a good node without being caught. The best the system can do is to accurately track a bad node's trust status to refrain it from providing bad service and to decrease the chance of selecting bad nodes for providing service. This is illustrated by two real-world mobile cloud IoT applications described in Sections VI and VII below.

D. Sensitivity and Comparative Analysis

In this section, we report the sensitivity of IoT-HiTrust performance with respect to the % of malicious users (P_M) and the % of high centrality users (P_C). We again compare IoT-HiTrust performance with Adaptive IoT Trust [6] and ObjectiveTrust [14], which we have described in detail in Section II.

We first analyze the effect of the % of malicious users (P_M). Figure 10 compares IoT-HiTrust (red surface) with Adaptive IoT Trust (orange surface) and ObjectiveTrust (green surface) as P_M (% of malicious nodes) varies from 20% to 40% for a non-malicious node randomly chosen. All other parameters use the default settings as in Table 2. The ground truth trust value of the non-malicious node is marked by a solid line at 0.83. We first note that for all protocols, the estimated trust score of the non-malicious nodes approaches the ground truth value as time progresses as more evidence is collected. Also, the deviation from the ground truth trust score increases as P_M increases because more malicious nodes perform attacks. We see that IoT-HiTrust (red) consistently outperforms Adaptive IoT Trust (orange) and ObjectiveTrust (green), especially when the % of malicious nodes is high (40%). We attribute IoT-HiTrust performing better than Adaptive IoT Trust to the fact that IoT-HiTrust can leverage cloud service to aggregate broad evidence from all nodes to achieve the minimum trust bias. We attribute IoT-HiTrust performing better than ObjectiveTrust to the fact that unlike ObjectiveTrust, IoT-HiTrust considers "subjective trust" which takes a customer's own service experiences into consideration and it can dynamically adjust the weight associated with a customer's own service experiences to effectively cope with malicious attacks and improve trust accuracy.

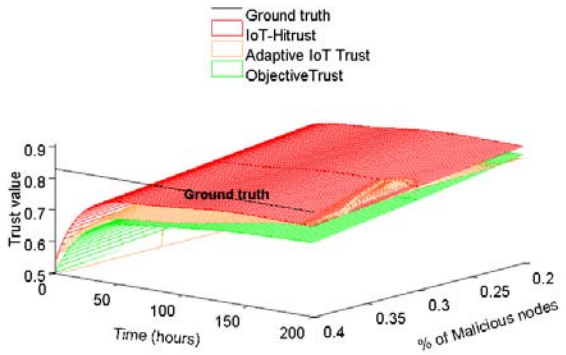

Figure 10: Performance Comparison of Trust Convergence, Accuracy and Resiliency in 3-D View with P_M (% of Malicious Nodes) ranging from 20% to 40%.

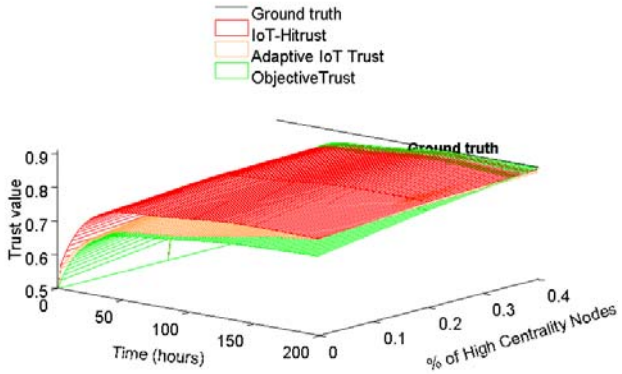

Figure 11: Performance Comparison of Trust Convergence, Accuracy and Resiliency in 3-D View with P_C (% of High Centrality Nodes) ranging from 0% to 40%.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2018.2886379, IEEE Transactions on Network and Service Management

Next we analyze the effect of % of high centrality users (P_C). Figure 11 compares IoT-HiTrust (red surface) with Adaptive IoT Trust (orange surface) and ObjectiveTrust (green surface) as P_C (% of high centrality nodes) varies from 0% to 40% for a non-malicious node randomly chosen. All other parameters use the default settings as in Table I. We see IoT-HiTrust (red) also consistently performs better than Adaptive IoT Trust (orange) and ObjectiveTrust (green), especially when P_C is low. All protocols perform comparably when P_C is high. The reason is that all protocols consider social relationships (including friendship which maps to centrality) for recommendation filtering. So when P_C is high, all have excellent recommendation filtering taking place, resulting in the estimated trust score approaching the ground truth trust value. When P_C is extremely high (40%), ObjectiveTrust has a slight edge over IoT-HiTrust and Adaptive IoT Trust, because ObjectiveTrust uses a weighted sum of centrality and filtered opinions for trust computation, and directly uses centrality (by means of the overall trust score computation) as credibility for recommendation filtering. One should note that, however, a social network normally has only a small number of nodes with high centrality [23], so for a typical social network with less than 20% high centrality nodes, IoT-HiTrust (red) still outperforms ObjectiveTrust (green).

It is interesting to note that Figure 11 also analyzes the effect of insufficient knowledge of friend devices (or missing friendship) on trust-based service management, through analyzing the effect of % of high centrality users (P_C) in the network. When there are not many high centrality users in the system, there is a weak tie in friendship for users in the network and hence the friendship social similarity is low. Figure 11 shows that the trust value of a non-malicious service provider decreases as the % of high centrality users (P_C) decreases because a device may unnecessarily filter out many trustworthy recommenders due to missing friendship. On the other hand, when P_C is high and hence the friendship social similarity is high, an IoT device will have access to many trustworthy recommenders, resulting in the estimated trust score of the non-malicious service provider in question approaching the ground truth trust value (solid line). Figure 11 demonstrates that IoT-HiTrust (red surface) consistently performs better than Adaptive IoT Trust (orange surface) and ObjectiveTrust (green surface), the effect of which is especially pronounced when a device does not have much knowledge about friend devices, i.e., when P_C is low.

In summary, relative to Adaptive IoT Trust [6] and ObjectiveTrust [14], our work addresses scalability while achieving "subjective trust" evaluation accuracy.

VI. Case Study 1: Smart City Travel Service Composition

The first case study is taken from [5]: Bob has never traveled to Washington DC, so he is excited but also nervous about the quality of service he will receive during his visit. He is aware of the fact that DC is a smart city so he registers his smartphone to the *travelers-in-Washington-DC* social network. He also downloads an *augmented map* social IoT application [34] to run on his smartphone, allowing his Near Field Communications (NFC) equipped smartphone to browse a tag-augmented DC map wherever he goes sightseeing. This tag-augmented map automatically connects Bob's smartphone to IoT devices available upon encounter, which provide information, food, entertainment (e.g., ticket purchasing), and transportation services. Bob instructs his smartphone to make selection decisions dynamically, so it can leverage new information derived from direct experiences and recommendations received from IoT devices it encounters. In response to a service request issued by Bob, his smartphone must (a) gather sensing data or information collected from the physical environment based on either self-observations or recommendations; (b) formulate a service plan based on the results gathered; and (c) invoke necessary services to meet Bob's service demand and requirements.

To this end, the augmented map travel service composition application running on Bob's smartphone composes a service workflow plan as shown in Figure 12 in response his service request "Fill me with the best grilled hamburger within 20 minutes under a $30 budget." With the service plan formulated, Bob's smartphone selects the best service providers out of a myriad of service providers to execute the service plan. The objective of this trust-based service composition application running on Bob's smartphone is to select the most trustworthy IoT nodes for providing services specified in the flow structure subject to the time and budget constraints (20 minutes and 30 dollars) such that the overall *trustworthiness* score representing the goodness of the service composition is maximized.

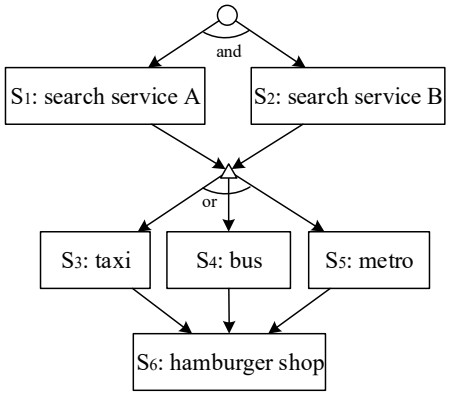

Figure 12: A Service Flow Structure for the Smart City Travel Service Composition Application.

In Figure 12, there are 6 atomic services connected by three types of workflow structures: *sequential*, *parallel* (AND), and *selection* (OR). Each service would have multiple service provider candidates. In this case, the overall trustworthiness score of this service composition application can be calculated recursively in the same way the reliability of a series-parallel connected system is calculated. Specifically, the trustworthiness score of a composite service (whose trustworthiness score is T_s) that consists of two subservices (whose trustworthiness scores are T_1 and T_2) depends on the structure connecting the two subservices as follows:

a) Sequential structure: $T_s = T_1 \times T_2$;

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2018.2886379, IEEE Transactions on Network and Service Management

b) Selection structure (OR): $T_s = \max(T_1, T_2)$;
c) Parallel structure (AND): $T_s = 1 - (1 - T_1) \times (1 - T_2)$.

For the flow structure in Figure 12, the outmost structure is a sequential structure connecting (S_1 S_2), (S_3 S_4 S_5), and S_6 out of which (S_1 S_2) is a parallel structure and (S_3 S_4 S_5) is a selection structure.

Figure 13 shows the ns-3 simulation results. The 3-tier hierarchical mobile cloud environment is setup the same way as discussed in Section V (see Table 2). We observe that trust-based service composition with IoT-HiTrust (red line) outperforms both Adaptive IoT Trust (yellow line) and ObjectiveTrust (green line) as time progresses when more trust evidence is gathered.

Figure 14 shows the percentage of bad nodes selected for the augmented map travel service composition application. Again as time progresses, IoT-HiTrust outperforms all baseline schemes in selecting a good SP to execute the service request. Initially Adaptive IoT Trust (which is a distributed trust protocol) performs better than both IoT-HiTrust and ObjectiveTrust (both are centralized trust protocols) because centralized trust protocols initially do not have access to a broad range of trust evidence collected from other IoT devices. The advantage of IoT-HiTrust over Adaptive IoT Trust increases as time progresses after the system gathers sufficient service data from all nodes having service experience with a target node. We also note that the advantage persists even the percentage of malicious nodes is 30% in the system. On the other hand, IoT-HiTrust performs better than ObjectiveTrust because it considers "subjective trust" which takes a customer's own service experiences with a target service provider into consideration as it computes the trust score of a customer toward a target service provider. In the case in which the service experience is plenty, IoT-HiTrust will dynamically put a higher weight on direct service experience and conversely a lower weight on the recommendations to compute the trust score for the purpose of minimizing trust bias. In the case in which the service experience is not plenty but recent, IoT-HiTrust will adaptively adjust the weights on direct service experiences and recommendations such that the computed score would match the recent service satisfaction outcome. In both cases, it shields the customer from malicious recommendation attacks, which is a common problem for reputation-based trust management protocols like ObjectiveTrust.

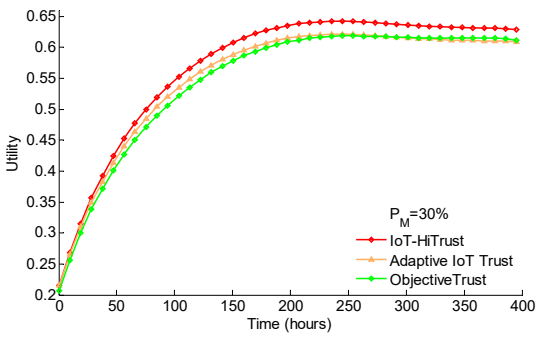

Figure 13: Utility Score of the Smart City Travel Service Composition Application.

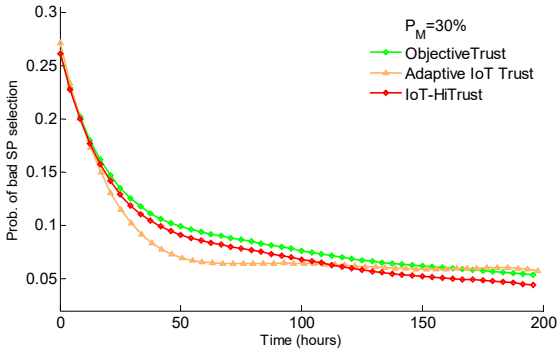

Figure 14: Probability of a Bad SP Being Selected for the Smart City Travel Service Composition Application.

VII. Case Study 2: Air Pollution Detection and Response

The second case study is for the *Fairfax County Hazard Detection and Response Team* charged to monitor the pollution levels of CO, NO_2, SO_2, and O_3 for all cities under the county so as to take appropriate actions if the air pollution level is above a tolerance threshold. Since the area to be covered is rather large, the county officials only install a few county-sensors in more strategic and populated areas to collect air pollution data. To cover the whole county area air quality detection, the county officials also encourage environment-health-conscious civilians driving or carrying air pollution detection capable vehicles or smartphones [8] to report air pollution data.

In case of emergency, the county officials can request IoT devices in a particular location to immediately report their sensing results to their home cloud servers through their local cloudlets. Because the county officials have registered this cloud service, a home cloud server upon receiving a sensing report will inform the county officials (running as an IoT device) of the sensing report. Also the county officials send queries via IoT-HiTrust to get the trustworthiness scores of these IoT devices who had reported sensing results. To know if a location has acceptable air quality, the county officials (running as node i) accept results (S_j) from 200 most trustworthy IoT devices (which have the highest t_{ij} trust values as determined by IoT-HiTrust) for the air quality detection service out of a total of 2000 nodes, and compute a trust-weighted average $\sum_{j=1}^{200}(t_{ij}/\sum_{j=1}^{200} t_{ij}) \times S_j$ for each air pollutant (e.g., CO). If the level exceeds a minimum threshold (e.g., above 70 ppm for CO), the county officials push alerting text to IoT devices in the affected area.

Using the ns-3 simulator, we simulate the above system populated with 2000 IoT devices capable of detecting and reporting CO air pollutant levels. The 3-tier hierarchical mobile cloud environment is setup the same way as discussed in Section V (see Table 2). The CO level is simulated to be in the range of [60, 70 ppm] in various locations. The percentage of bad nodes is set at P_M =30%. A malicious node always reports CO readings above 70 ppm in the range of [70, 120 ppm] regardless of location in order to confuse the county official. Also a malicious node always performs bad-mouthing attacks (saying a good node's sensing result is not trustworthy in the user

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2018.2886379, IEEE Transactions on Network and Service Management

satisfaction report) and ballot-stuffing attacks (saying a bad node's sensing result is trustworthy).

We again compare IoT-HiTrust with Adaptive IoT Trust and ObjectiveTrust. We measure two performance metrics for performance analysis: (a) the trust-weighted average CO reading vs. ground truth (i.e., the actual CO level at a specific location and a particular time); (b) the accuracy of selecting trustworthy participants.

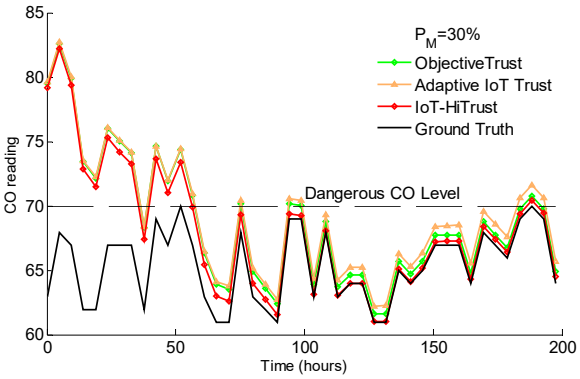

Figure 15: Performance Comparison of Trust-Weighted Average CO Readings for the Air Pollution Detection and Response Application.

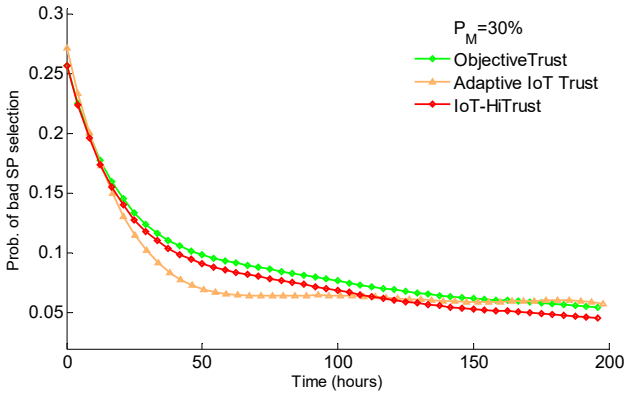

Figure 16: Percentage of Bad IoT Devices Selected to Provide CO Sensing Service for the Air Pollution Detection and Response Application.

Figure 15 shows the trust-weighted average CO readings vs. time (each time point is a CO detection service request) with the percentage of bad nodes P_M set at 30%. We observe that IoT-HiTrust (red line) leveraging the proposed mobile cloud hierarchy can provide CO readings very close to ground truth (black line) as time progresses. Further, IoT-HiTrust outperforms Adaptive IoT Trust (yellow line) and ObjectiveTrust (green line) in terms of accuracy, convergence, and resiliency. We mark a "Dangerous CO Level" line at which the CO reading is equal to or above 70 in the graph. We see that in many sensing time points such as at 65, 75, 90, 100, and 185, IoT-HiTrust would report the CO level is not dangerous as the ground truth is, but either Adaptive IoT Trust or ObjectiveTrust would falsely report the CO level is dangerous since the trust-weighted CO level average computed is above 70. This demonstrates that IoT-HiTrust is more resilient to malicious attacks (30% are malicious) than either Adaptive IoT Trust or ObjectiveTrust in this application. Figure 16 shows the percentage of bad nodes selected to provide sensing results. We see again IoT-HiTrust (red line) outperforms Adaptive IoT Trust and ObjectiveTrust as time progresses.

We attribute the superiority of IoT-HiTrust over Adaptive IoT Trust to its ability to effectively aggregate trust evidence from all nodes who have had sensing service experiences with a target IoT device, leveraging our scalable report-and-query design, not being limited by node encountering experiences as in Adaptive IoT Trust. We attribute the superiority of IoT-HiTrust over ObjectiveTrust to its ability to accurately compute the "subjective trust" which takes a customer's own service experiences into consideration as opposed to the "objective trust" which only takes the common belief or reputation into consideration as in ObjectiveTrust, and also to its ability to dynamically adjust the weights associated with direct trust and indirect trust to minimize trust bias, based on the customer's past and recent own service experiences.

VIII. Conclusion

In this paper, we designed and analyzed a scalable hierarchical trust management protocol called IoT-HiTrust for large mobile cloud IoT systems. We verified that IoT-HiTrust is effective for dealing with intermittent disconnection and cloud failure while preserving desirable trust accuracy, convergence and resiliency properties, especially for IoT devices that do not move much such as heavyweight IoT devices. We also demonstrated its applicability by applying IoT-HiTrust to a smart city travel service composition application and an air pollution detection and response application. Our results support its superiority over Adaptive IoT Trust [6] and ObjectiveTrust [14] in achieving scalability and maximizing application performance, without compromising trust accuracy, convergence and resiliency properties.

In the future, we plan to further validate our 3-tier cloud-cloudlet-device hierarchical trust-based service management with more real-world mobile cloud IoT applications, including environmental monitoring and road/traffic monitoring applications [11], IoT services applications [6, 18], and IoT health applications [1, 34]. We also plan to investigate caching mechanisms at the cloudlet level that can improve the overall system performance. Last but not least, we are currently investigating a more holistic design to manage integrated mobility, service, and trust information of a large number of IoT devices, in a scalable, secure, reliable, and efficient manner. A possible solution is to integrate the tiered cloud architecture presented in this work with existing design concepts of hierarchical mobility management [29], resilient failure recovery management [30, 31], and admission control [32, 33]. While a node in a hierarchical mobility management architecture is a router responsible for keeping track of location information only (where and how to route), a node in a hierarchical cloud management architecture is a cloud server responsible for keeping track of integrated information including location, trust, and service information.

A lower-level cloud server (e.g., a cloudlet) keeps track of IoT devices in its directly covered service area. A higher-level cloud server (e.g., a public cloud) in the architecture keeps track of status of all IoT devices covered by all lower-level cloud servers below it. Should an IoT device roam from one cloud server area to another, a "service handoff" ensues, causing this IoT device's location, trust, and service information to be transferred between the two involving cloud servers such that the new local cloud server can immediately answer user queries regarding this IoT device that just roams into its area. Such an IoT architecture can track IoT devices not only in trust status, but also in service and mobility status dynamically to achieve the potential of anytime anywhere service-oriented IoT applications.

ACKNOWLEDGEMENTS

This work is supported in part by the U.S. AFOSR under grant number FA2386-17-1-4076. This work is also partially supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. 2017-0-00664, Rule Specification-based Misbehavior Detection for IoT-Embedded Cyber Physical Systems).

REFERENCES

[1] H. Al-Hamadi and I.R. Chen, "Trust-Based Decision Making for Health IoT Systems," *IEEE Internet of Things Journal*, vol. 4, no. 5, Oct. 2017, pp. 1408-1419.

[2] F. Bao and I. R. Chen, "Dynamic Trust Management for Internet of Things Applications," *2012 International Workshop on Self-Aware Internet of Things*, San Jose, California, USA, 2012.

[3] F. Bao and I.R. Chen, "Trust Management for the Internet of Things and Its Application to Service Composition," *IEEE WoWMoM 2012 Workshop on the Internet of Things*, San Francisco, CA, USA, 2012.

[4] F. Bao, I.R. Chen, and J. Guo, "Scalable, Adaptive and Survivable Trust Management for Community of Interest Based Internet of Things Systems," *11th International Symposium on Autonomous Decentralized System, Mexico City,* Mexico, 2013.

[5] I. R. Chen, F. Bao, and J. Guo, "Trust-based Service Management for Social Internet of Things Systems," *IEEE Transactions on Dependable and Secure Computing*, vol. 13, no. 6, Nov-Dec 2016, pp. 684-696.

[6] I.R. Chen, J. Guo, and F. Bao, "Trust Management for SOA-based IoT and Its Application to Service Composition," *IEEE Transactions on Services Computing*, vol. 9, no. 3, 2016, pp. 482-495.

[7] D. Chen, G. Chang, D. Sun, J. Li, J. Jia, and X. Wang, "TRM-IoT: A Trust Management Model Based on Fuzzy Reputation for Internet of Things," *Computer Science and Information Systems,* vol. 8, no. 4, pp. 1207-1228, Oct 2011.

[8] S. Devarakonda, et al., "Real-time Air Quality Monitoring Through Mobile Sensing in Metropolitan Areas," *UrbComp*, Chicago, Illinois, USA, 2013.

[9] A. Jøsang, and R. Ismail, "The Beta Reputation System," *Bled Electronic Commerce Conference*, Bled, Slovenia, 2002, pp. 1-14.

[10] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The EigenTrust algorithm for reputation management in P2P net-works," *12th International Conference on World Wide Web*, Budapest, Hungary, May 2003.

[11] W.Z. Khan, Y. Xiang, M.Y. Aalsalem, and Q. Arshad, "Mobile Phone Sensing Systems: A Survey," *IEEE Communications Surveys and Tutorials*, vol. 15, no. 1, pp. 402–427, 2013.

[12] S. Kosta, A. Mei, and J. Stefa, "Small World in Motion (SWIM): Modeling Communities in Ad-Hoc Mobile Networking," *7th IEEE Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, Boston, MA, USA, 2010.

[13] R. Mitchell and I. R. Chen, "Modeling and Analysis of Attacks and Counter Defense Mechanisms for Cyber Physical Systems," *IEEE Transactions on Reliability*, vol. 65, no. 1, March 2016, pp. 350-358.

[14] M. Nitti, R. Girau, and L. Atzori, "Trustworthiness Management in the Social Internet of Things," *IEEE Transactions on Knowledge and Data Management*, vol. 26, no. 5, 2014, pp. 1253-1266.

[15] Y. B. Saied, A. Olivereau, D. Zeghlache, and M. Laurent, "Trust management system design for the Internet of Things: A context-aware and multi-service approach," *Computers and Security*, vol. 39, Nov. 2013, pp. 351-365.

[16] M. Satyanarayanan, et al., "The role of cloudlets in hostile environments," *IEEE Pervasive Computing*, Oct. 2013, pp. 40-49.

[17] Z. Su, L. Liu, M. Li, X. Fan, and Y. Zhou, "ServiceTrust: Trust Management in Service Provision Networks," *IEEE International Conference on Services Computing*, Santa Clara, 2013, pp. 272-279.

[18] Y. Wang, I.R. Chen, J.H. Cho, A. Swami, and K.S. Chan, "Trust-based Service Composition and Binding with Multiple Objective Optimization in Service-Oriented Ad Hoc Networks," *IEEE Trans. Services Computing*, vol. 10, no. 4, 2017, pp. 1939-1374.

[19] Y. Wang, I.R. Chen, and D.C. Wang, "A Survey of Mobile Cloud Computing Applications: Perspectives and Challenges," *Wireless Personal Communications*, vol. 80, no. 4, 2015, pp. 1607-1623.

[20] L. Xiong, and L. Liu, "PeerTrust: Supporting Reputation-Based Trust for Peer-to-Peer Electronic Communities," *IEEE Trans. on Knowledge and Data Engineering*, v.16, pp. 843-857, July 2004.

[21] Z. Zheng, Y. Zhang, and M. R. Lyu, "Investigating QoS of Real-World Web Services," *IEEE Transactions on Services Computing*, vol. 7, no. 1, pp. 32-39, 2014.

[22] J. Guo, I.R. Chen, and J.J.P. Tsai, "A Mobile Cloud Hierarchical Trust Management Protocol for IoT Systems," *5th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering*, San Francisco, April 2017.

[23] M. Nitti, L. Atzori, and I.P. Cvijikj, "Friendship Selection in the Social Internet of Things: Challenges and Possible Strategies," *IEEE Internet of Things Journal*, vol. 2, no. 3, 2015, pp. 240-247.

[24] ns-3 Network Simulator, http://www.nsnam.org, Release ns-3.27 with core, network, Internet, and mobility models, Oct. 2017.

[25] I.R. Chen, J. Guo, F. Bao, and J.H. Cho, "Trust Management in Mobile Ad Hoc Networks for Bias Minimization and Application Performance Maximization," *Ad Hoc Networks*, vol. 19, August 2014, pp. 59-74.

[26] I.R Chen, F. Bao, M.J. Chang, and J.H. Cho, "Dynamic Trust Management for Delay Tolerant Networks and Its Application to Secure Routing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 5, 2014, pp. 1200-1210.

[27] T. Karagiannis, J.-Y. Le Boudec, and M. Vojnović, "Power Law and Exponential Decay of Intercontact Times between Mobile Devices," *IEEE Trans. Mobile Computing*, vol. 8, no. 10, 2007, pp. 1377-1390.

[28] J.B. Abdo and J. Demerjian, "Evaluation of Mobile Cloud Architecture," *Pervasive and Mobile Computing*, vol. 39, 2017, pp. 284-303.

[29] B. Gu and I. R. Chen, "Performance Analysis of Location-aware Mobile Service Proxies for Reducing Network Cost in Personal Communication Systems," *Mobile Networks and Applications,* vol. 10, no. 4, 2005, pp. 453-463.

[30] I.R. Chen, B. Gu, S.E. George, and S.T. Cheng, "On Failure Recoverability of Client-Server Applications in Mobile Wireless Environments," *IEEE Trans. Reliability*, vol. 54, no. 1, 2005, pp. 115-122.

[31] I.R. Chen and F.B. Bastani, "Effect of Artificial-Intelligence Planning Procedures on System Reliability," *IEEE Trans Reliability*, vol. 40, no. 3, pp. 364-369, 1991.

[32] S.T. Cheng, C.M. Chen, and I.R. Chen, "Dynamic Quota-based Admission Control with Sub-rating in Multimedia Servers," *Multimedia Systems*, vol. 8, no. 2, 2000, pp. 83-91.

[33] I.R. Chen, O. Yilmaz, and I.L. Yen, "Admission Control Algorithms for Revenue Optimization with QoS Guarantees in Mobile Wireless Networks," *Wireless Personal Communications*, vol. 38, no. 3, 2006, pp. 357-376.

[34] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A Survey," *Computer Networks,* vol. 54, pp. 2787-2805, Oct. 2010.

[35] J. Guo, I.R. Chen, and J.J.P. Tsai, "A Survey of Trust Computation Models for Internet of Things Systems," *Computer Communications*, vol. 97, 2017, pp. 1-14.

[36] J.H. Cho and I.R. Chen, "PROVEST: Provenance-based Trust Model for Delay Tolerant Networks," *IEEE Transactions on Dependable and Secure Computing,* vol. 15, no. 1, 2018, pp. 151-165.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSM.2018.2886379, IEEE Transactions on Network and Service Management

AUTHOR BIOGRAPHIES

Ing-Ray Chen received the BS degree from the National Taiwan University, and the MS and PhD degrees in computer science from the University of Houston. He is a professor in the Department of Computer Science at Virginia Tech. His research interests include mobile computing, wireless systems, security, trust management, and reliability and performance analysis. Dr. Chen currently serves as an editor for *IEEE Transactions on Services Computing, IEEE Transactions on Network and Service Management,* and *The Computer Journal*. He is a recipient of the IEEE Communications Society William R. Bennett Prize in the field of Communications Networking.

Jia Guo received his BS degree in computer science from Jilin University, China in 2011 and his PhD degree in computer science from Virginia Tech in 2018. His research interests include trust management, mobile ad hoc and sensor networks, Internet of things, delay tolerant computing, and secure and dependable computing.

Ding-Chau Wang received the BS degree from Tung-Hai University, Taichung, Taiwan, and the MS and PhD degrees in computer science and information engineering from National Cheng Kung University, Tainan, Taiwan. He is currently an associate professor in the Department of Information Management at Southern Taiwan University of Science and Technology, Tainan, Taiwan. His research interests include game-based learning, Internet of things, mobile computing, security, database systems and performance analysis.

Jeffrey J.P. Tsai received a Ph.D. degree in Computer Science from the Northwestern University, Evanston, Illinois. He is the President of Asia University, Taiwan, and a professor in the Department of Bioinformatics and Biomedical Engineering at Asia University. Dr. Tsai was a Professor of Computer Science at the University of Illinois, Chicago. His current research interests include bioinformatics, ubiquitous computing, services computing, intrusion detection, knowledge-based software engineering, formal modeling and verification, distributed real-time systems, and intelligent agents. Dr. Tsai received an IEEE Technical Achievement Award and an IEEE Meritorious Service Award from IEEE Computer Society. He is a Fellow of the AAAS, IEEE, and SDPS.

Hamid Al-Hamadi received the B.S. degree in information technology from Griffith University, Brisbane, Australia, in 2003 and the M.S. degree in information technology from the Queensland University of Technology, Brisbane, Australia, in 2005 and the Ph.D. degree in computer science from the Virginia Polytechnic Institute and State University, VA, USA in 2014. He has experience working as a Network Engineer at Kuwait National Petroleum Company and at Tawasul Telecom, Kuwait. Currently, he is an Assistant Professor with the Department of Computer Science, Kuwait University, Kuwait. His current research interests include Internet of Things, security, mobile cloud, trust management, and reliability and performance analysis.

Ilsun You received the M.S. and Ph.D. degrees in computer science from Dankook University, Seoul, South Korea, in 1997 and 2002, respectively, and the Ph.D. degree from Kyushu University, Japan, in 2012. He is currently Chairperson of the Department of Information Security Engineering, Soonchunhyang University, Asan, South Korea. His main research interests include Internet security, authentication, access control, and formal security analysis. He is a Fellow of the IET. He is the EiC of the Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications. He is on the Editorial Board of Information Sciences, Journal of Network and Computer Applications, International Journal of Ad Hoc and Ubiquitous Computing, Computing and Informatics, Journal of High Speed Networks, Intelligent Automation and Soft Computing, and Security and Communication Networks.