Trust-based Service Management for Social Internet of Things Systems

Ing-Ray Chen, Fenye Bao, and Jia Guo

Abstract— A social Internet of Things (IoT) system can be viewed as a mix of traditional peer-to-peer networks and social networks, where "things" autonomously establish social relationships according to the owners' social networks, and seek trusted "things" that can provide services needed when they come into contact with each other opportunistically. We propose and analyze the design notion of *adaptive trust management* for social IoT systems in which social relationships evolve dynamically among the owners of IoT devices. We reveal the design tradeoff between trust convergence vs. trust fluctuation in our adaptive trust management protocol design. With our adaptive trust management protocol, a social IoT application can adaptively choose the best trust parameter settings in response to changing IoT social conditions such that not only trust assessment is accurate but also the application performance is maximized. We propose a table-lookup method to apply the analysis results dynamically and demonstrate the feasibility of our proposed adaptive trust management scheme with two real-world social IoT service composition applications.

Index Terms— Trust management, Internet of things, social networking, performance analysis, adaptive control, security.

1 Introduction

A social Internet of Things (IoT) system can be viewed as a mix of traditional peer-to-peer (P2P) networks and social networks, where "things" autonomously establish social relationships according to the owners' social networks, and seek trusted things that can provide services needed when they come into contact with each other opportunistically in both the physical world and cyberspace. It is envisioned that the future social IoT will connect a great amount of smart objects in the physical world, including radio frequency identification (RFID) tags, sensors [40], actuators, PDAs, and smartphones, as well as virtual objects in cyberspace such as data and virtual desktops on the cloud [2]. The emerging paradigm of the social Internet of Things (IoT) has attracted a wide variety of applications running on top of it, including e-health [9, 23], smart-home, smart-city, and smart-community [27]. We will use the terms things, objects, and devices interchangeably in the paper.

Such future social IoT applications are likely oriented toward a service oriented architecture where each thing plays the role of either a service provider or a service requester, or both, according to the rules set by the owners. Unlike a traditional service-oriented P2P network, social networking and social relationship play an important role in a social IoT, since things (real or virtual) are essentially operated by and work for humans. Therefore, social relationships among the users/owners must be taken into account during the design phase of social IoT applications. A social IoT system thus can be viewed as a P2P owner-centric community with devices (owned by humans) requesting and providing services on behalf of the owners. IoT devices establish social relationships autonomously with other devices based on social rules set by their owners, and interact with each other opportunistically as they come into contact. To best satisfy the service requester and maximize application performance, it is crucial to evaluate the trustworthiness of service providers in social IoT environments. This paper concerns trust management in social IoT environments.

The motivation of providing a trust management system for a social IoT system is clear: There are misbehaving owners and consequently misbehaving devices that may perform discriminatory attacks based on their social relationships with others for their own gain at the expense of other IoT devices which provide similar services. Further, misbehaving nodes with close social ties may collude and monopoly a class of services. Since trust provisioning in this environment inherently is fully integrated with service provisioning (i.e., one must decide whether or not to use a service provided by a device based on the trust toward the device), the notion of trust-based service management is of paramount importance.

There is a large body of trust management protocols for P2P service computing systems (e.g., [7, 13, 17, 18, 22, 26, 37, 39, 41, 43]). These P2P service systems share a common characteristic with social IoT systems in that services are provided by nodes in the system so that trust evaluation of nodes is critical to the functioning of the

• *Ing-Ray Chen, Fenye Bao and Jia Guo are with the Department of Computer Science, Virginia Tech, Falls Church, VA 22043. E-mail: (irchen,baofeyne, jiaguo)@vt.edu.*

system. However, trust protocols for P2P service computing systems lack consideration of the social aspects of IoT device owners, and are not applicable to a social IoT system comprising real or virtual heterogeneous "things" with ownership, friendship and community of interest relationships connected with each other by various ways (via the Internet), and operated by their owners with a variety of social behaviors to collect information, provide services, provide recommendations, make decisions, and take actions. On the other hand, trust protocols for social networks [10, 20, 24, 30, 38] are more concerned with trust assessment of social entities based on frequency, duration and nature of contacts (such as conversation and propagation [1]) between two social entities, without considering P2P service computing environments in which IoT devices seek and provide service when they come into contact with each other opportunistically.

To date there is little work on trust management for social IoT systems [5, 14, 32, 36], especially for dealing with misbehaving owners of IoT devices that provide services to other devices in the system. We compare and contrast our trust protocol design principles with prior work in Section 7 Related Work.

In this paper we propose an *adaptive trust management* protocol for social IoT systems. Our method is suitable to be applied to social IoT experimental platforms as discussed in [21, 44, 45]. Our goal is to enhance the security and increase the performance of social IoT applications. We aim to design and validate an adaptive trust management protocol that can dynamically adjust trust design parameter settings in response to changing environment conditions to provide accurate trust assessment (with respect to actual status) and to maximize application performance. The need for *adaptive* trust management stems from the fact that social relationships between owners and thus social behaviors of owners are evolving. An example is that owners carrying IoT devices can often move from a friendly environment (e.g., a social club) to a hostile environment (e.g., a neighborhood one does not go often).

We are particularly interested in trust protocol design that can deal with misbehaving nodes. Such a protocol must possess desirable trust *convergence*, *accuracy*, and *resiliency* properties. Our contribution relative to existing trust management protocols for IoT systems [5, 14, 32, 36] is that we develop an *adaptive* trust management protocol in social IoT systems. Unlike trust systems designed for P2P networks [26, 37, 41, 43], sensor networks [7, 12], delay tolerant networks [11, 13], and mobile ad hoc networks [17, 18], our trust management protocol takes dynamically changing social relationships among the "owners" of devices in IoT systems into account and demonstrate that the desirable *convergence*, *accuracy*, and *resiliency* properties are satisfied with extensive simulation. Further, using two real-world social IoT applications, we demonstrate that our adaptive trust management protocol is capable of adaptively adjusting the best trust parameter setting in response to dynamically changing environments to improve trust assessment accuracy and to maximize application performance, despite the presence of misbehaving nodes disrupting the functionality of a social IoT system.

The rest of the paper is organized as follows: In Section 2 we discuss the system model and assumptions. In Section 3, we describe our adaptive trust management protocol design for social IoT systems. In Section 4, we validate the convergence, accuracy, and resiliency properties of our protocol, and reveal the design tradeoff between trust convergence and trust fluctuation in adaptive trust management. In Section 5, we demonstrate the utility of our adaptive trust protocol design with two social IoT applications. Section 6 discusses the applicability, i.e., how one may apply the best protocol settings identified to achieve the desirable accuracy and maximize application performance at runtime. In Section 7, we survey related work to compare and contrast our protocol design with existing work. Finally in Section 8, we summarize the paper and outline future work.

2 System Model

2.1 User-Centric Social IoT Environments

We consider a user-centric social IoT environment with no centralized trusted authority. Each IoT device has its unique identity which can be achieved through standard techniques such as PKI. A device communicates with other devices through the overlay social network protocols, or the underlying standard communication network protocols (wired or wireless). Every device has an owner who could have many devices. Social relationships between owners is translated into social relationships between IoT devices as follows: 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 nodes are friends, then it is likely they will be cooperative with each other. A device may be carried or operated by its owner in certain community-interest environments (e.g., work vs. home or a social club). Nodes belonging to a similar set of communities likely share similar interests or capabilities.

Our social IoT model is based on social relationships among humans who are owners of IoT devices. We note that the device-to-device autonomous social relationship is also a potential for the social IoT paradigm.

2.2 Attack Model

A malicious node is dishonest and socially uncooperative in nature and can break the basic functionality of the social IoT system. A malicious node can perform the following trust-related attacks:

1. *Self-promoting attacks*: a malicious node can promote its importance (by providing good recommendations for

itself) so as to be selected as the service provider, but then it provides malfunctioned service. Our trust protocol deals with self-promoting attacks by considering honesty as a trust property to detect self-promoting attacks.

2. *Whitewashing attacks*: a malicious node can disappear and rejoin the application to wash away its bad reputation. Our trust protocol deals with whitewashing attacks by remembering trust information of each identity, and by performing trust decay over time to account for node inactivity during the period in which a node disappears from the IoT system.
3. *Discriminatory 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. Our trust protocol deals with discriminatory attacks by considering cooperativeness and community-interest as trust properties.
4. *Bad-mouthing attacks*: a malicious node can ruin the reputation of a well-behaved node by providing bad recommendations against the good node so as to decrease the chance of this good node being selected as a service provider. This is a form of *collusion attacks*, i.e., it can collaborate with other bad nodes to ruin the reputation of a good node. Our trust protocol deals with bad-mouthing attacks by considering honesty as a trust property.
5. *Ballot-stuffing attacks*: a malicious node can boost the reputation of another bad node by providing good recommendations for it so as to increase the chance of this bad node being selected as a service provider. This is also a form of *collusion attacks*, i.e., it can collaborate with other bad nodes to boost the reputation of each other. Our trust protocol deals with ballot-stuffing attacks by considering honesty as a trust property.

A collusion attack means that the malicious nodes in the system boost their allies and focus on particular victims in the system to victimize. Bad-mouthing and ballot-stuffing attacks both are a form of collusion attacks to ruin the reputation of (and thus to victimize) good nodes, and to boost the reputation of malicious nodes. A malicious node can perform Sybil and identity attacks, and in general can perform communication protocol attacks to disrupt IoT network operations. We assume such attacks will be handled by intrusion detection techniques [16, 31] and the attackers will be evicted from the system upon detection.

3 Adaptive Trust Management

Table I lists the parameters used in the paper. A design parameter is one that adaptive trust management can control to optimize performance. A derived parameter is one that is generated during runtime as a result of running the trust protocol. An input parameter is one that the operating environment dictates.

The components of adaptive trust management for a social IoT system are shown in Figure 1. Our protocol addresses all aspects of trust management: the trust *composition* component addresses the issue of how to select multiple trust properties according to social IoT application requirements. The trust *propagation* and *aggregation* component addresses the issue of how to disseminate and combine trust information such that the trust assessment converges and is accurate. The trust *formation* component addresses the issue of how to form the overall trust out of individual trust properties and how to make use of trust in order to maximize application performance. Essentially adaptive trust management is achieved by (1) selecting the best trust propagation and aggregation parameter setting to achieve trust accuracy and convergence and (2) selecting the best trust formation parameter setting to maximize application performance, in response to an evolving IoT environment.

Adaptive trust management must be distributed as a social IoT system frequently consists of free-will entities without a centralized mediator. Each node maintains its own trust assessment towards other nodes. A node is more likely to share common interests with those nodes it recently interacts with or it believes to be trustworthy. For an IoT node with a limited storage, it will only keep trust and recommendation information for a limited set of nodes which it is most interested in. In this paper we do not consider the use of caching to mitigate the limited storage issue. We refer the readers to [8] for a scalable caching storage management design to effectively utilize limited storage space without compromising trust accuracy and convergence properties. Adaptive trust management is interaction-based as well as activity-based, meaning that the trust value is updated dynamically upon an interaction event or activity. Two nodes involved in a direct interaction activity can directly observe each other and update their trust assessment. They also exchange their trust evaluation results toward other nodes as recommendations.

Table 1: List of Parameters.

Symbol	Meaning	Type
$T_{ij}^{X}(t)$	trust of *i* towards *j in X* at time *t*	derived
$D_{ij}^{X}(t)$	direct trust of *i* towards *j in X* at time *t*	derived
$R_{jk}^{X}(t)$	recommendation from *k* toward *j in X* at *t*	derived
$T_{ij}(t)$	overall trust or overall trustworthiness score of *i* towards *j* at time *t*	derived
α	weight on direct trust w.r.t. experience	design
β	weight on recommendation w.r.t. experience	design
N_T	number of IoT devices	input
N_H	number of owners	input
N_G	number of user communities	input
T	average interaction inter-interval time	input
λ	percentage of malicious nodes	input
T_c	node compromise time period	input

3.1 Trust Composition

While there is a wealth of social trust metrics available [38], we choose *honesty*, *cooperativeness*, and *community-interest* as the most striking metrics for characterizing social IoT systems, as illustrated in Figure 1 (2nd level). These trust properties are considered orthogonal but complementary to each other to characterize a node. Each trust property is evaluated separately as follows:

- The *honesty* trust property represents whether or not a node is honest. In IoT, a malicious node can be dishonest when providing services or trust recommendations. We select *honesty* as a trust property because a dishonest node can severely disrupt trust management and service continuity of an IoT application. In an IoT application, a node relies on direct evidence (upon interacting) and indirect evidence (upon hearing recommendations vs. own assessment toward a third-party node) to evaluate the honesty trust property of another node.
- The *cooperativeness* trust property represents whether or not the trustee node is socially cooperative [28] with the trustor node. A node may follow a prescribed protocol only when interacting with its friends or nodes with strong social ties (with many common friends), but become uncooperative when interacting with other nodes. In an IoT application, a node can evaluate the cooperativeness property of other nodes based on social ties and select socially cooperative nodes in order to achieve high application performance.
- The *community-interest* trust represents whether or not the trustor and trustee nodes are in the same social communities/groups (e.g. co-location or co-work relationships [3]) or have similar capabilities (e.g., parental object relationships [3]). Two nodes with a degree of high community-interest trust have more chances and experiences in interacting with each other, and thus can result in better application performance.

We note that while transaction importance or degree of friendship can complement or refine the above three trust properties, we will not consider them in this paper for simplicity. In Section 3.2 below we discuss in detail how a node evaluates other nodes in *honesty*, *cooperativeness*, and *community-interest* trust properties by combining first-hand direct observations (discussed in Section 3.2.1) and second-hand recommendations (discussed in Section 3.2.2).

3.2 Trust Propagation and Aggregation

Adaptive trust management is a continuing process which iteratively aggregates past information and new information. The new information includes both direct observations (first-hand information) and indirect recommendations (second-hand information). The trust assessment of node i towards node j at time t is denoted by $T_{ij}^{X}(t)$ where X = *honesty*, *cooperativeness*, or *community-interest*. The trust value $T_{ij}^{X}(t)$ is a real number in the range of [0, 1] where 1 indicates complete trust, 0.5 ignorance,

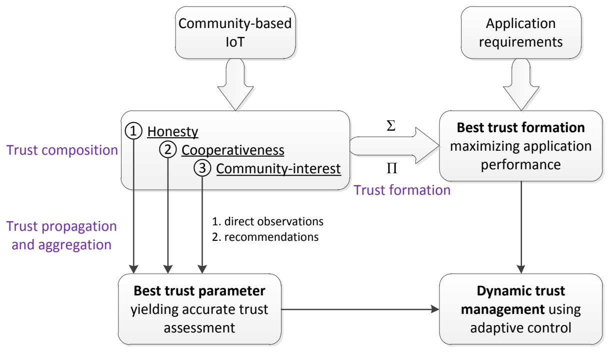

Community-based IoT
Application requirements
① Honesty
② Cooperativeness
③ Community-interest
Trust composition
Σ
Π
Trust formation
Best trust formation maximizing application performance
Trust propagation and aggregation
1. direct observations
2. recommendations
Best trust parameter yielding accurate trust assessment
Dynamic trust management using adaptive control

Figure 1: Components of adaptive trust management for a social IoT system: (1) trust composition – *honesty*, *cooperativeness*, and *community-interest*, (2) trust propagation and aggregation – combining first-hand (direct observations) and second-hand information (recommendations), (3) trust formation – forming the overall trust out of three individual trust properties, (4) adaptive trust management – adaptively adjusting parameter settings to improve trust evaluation accuracy and trust formation to maximize application performance.

and 0 distrust. In IoT environments, nodes interact with each other when they detect the presence of each other, via IoT discovery protocols such as [33]. When evaluating $T_{ij}^{X}(t)$, we adopt the following notations: node i is the trustor, node j is the trustee, node k is a recommender to provide its feedback about node j to node i.

3.2.1 $T_{ij}^{X}(t)$ Update When Node *i* Interacts with Node *j*

When node i directly interacts with (or encounters) node j at time t, node i will update its trust assessment toward node j, $T_{ij}^{X}(t)$, as follows:

$$T_{ij}^{X}(t) = (1-\alpha)T_{ij}^{X}(t-\Delta t) + \alpha D_{ij}^{X}(t) \quad (1)$$

Here, Δt is the elapsed time since the last trust update. A trust update is trigged by interaction events. Thus, the value of Δt is the time interval between two consecutive interactions. Node i will use its new trust assessment toward node j based on direct observation (i.e., $D_{ij}^{X}(t)$) and its past trust toward node j (i.e.,$T_{ij}^{X}(t-\Delta t)$) to update $T_{ij}^{X}(t)$. A parameter α ($0 \leq \alpha \leq 1$) is used here to weigh these two trust values and to consider trust decay over time, i.e., the decay of the old trust value and the contribution of the new trust value. A larger α means that trust evaluation will rely more on new direct observations.

Below we detail how each IoT device calculates $D_{ij}^{X}(t)$ for X=*honesty*, *cooperativeness*, or *community-interest*.

Honesty - $D_{ij}^{honesty}(t)$: Direct honesty trust refers to the belief of node i that node j is honest based on node i's direct interaction experiences toward node j at time t. First, node i detects bad-mouthing/ballot-stuffing attacks by node j by comparing node j's recommendation (provided to node i as a recommendation) toward another node, say, node q, against the trust value of node i toward node q itself. Node j's recommendation toward q is just $D_{jq}^{honesty}(t-\Delta t)$ if node j is an honest node not performing attacks; otherwise, it can be 0 (or 1) if node j is a dishonest node performing bad-mouthing attacks against (or ballot-staffing attacks for) node q. Node i's trust toward node q on the other hand is just $D_{iq}^{honesty}(t-\Delta t)$ kept by node i. If the percentage difference relative to $D_{iq}^{honesty}(t-\Delta t)$ is higher than a threshold, it is considered suspicious and thus a negative honesty experience. A high threshold reduces false negatives (misidentifying a bad node as a good node) but increases false positives (misidentifying a good node as a bad node), and vice versa. We set the threshold to 50% to balance false negatives with false positives. Second, node i detects self-promotion attacks by node j by detecting if node j promotes its importance by boosting its cooperativeness and/or community-interest trust (see below) so as to improve its chance of being selected as the service provider, but then provides a bad service. These direct positive/negative experiences collected are used to assess $D_{ij}^{honesty}(t)$, computed by the number of positive experiences over the total experiences collected.

Cooperativeness - $D_{ij}^{cooperativeness}(t)$: This trust property provides the degree of cooperativeness of node j as evaluated by node i based on direct observations at time t upon encountering. We use the *social friendship* [28] among device owners to characterize the cooperativeness. The rationale is that friends are likely to be cooperative toward each other. Each device keeps a list of its owner's friends which may be updated dynamically by its owner. $D_{ij}^{cooperativeness}(t)$ is computed as the ratio of common friends between i and j, i.e., $\frac{|friends(i)\cap friends(j)|}{|friends(i)\cup friends(j)|}$, where $friends(i)$ denotes the set of friends to the owner of node i. A node is included in its own friend list (i.e., $i \in friends(i)$) to deal with the case where two nodes are the only friends to each other. When node i and node j directly interact with each other, they exchange their friend lists. Node i can validate a friend in node j's list if it is their common friend. Therefore, the direct observation of cooperativeness will be close to actual status.

To preserve privacy, node i and node j with permission from their owners agree on a one-way hash function (with a session key) upon interacting while exchanging the friend lists. Thus, the friend lists exchanged are encrypted with a one-way hash function, so each party can only compare its (encrypted) list with the other party's (encrypted) list to find matches of common friends, but cannot know the other party's list. This way, only common friends will be identified while the identities of uncommon friends will not be revealed. Also, hashing is a very low cost operation and would not drain the battery of low-capacity IoT devices.

Here we note that an honest node will submit its friend list faithfully (although with hashing to hide identity). A dishonest node, however, can perform self-promotion attacks to submit a fake friend list in order to boost its "cooperativeness" trust. This in turn can lead to its "honesty" trust greatly decreased due to the honesty detection mechanisms used in our protocol design.

Community-Interest - $D_{ij}^{community-interest}(t)$: This trust property provides the degree of the common interest or similar capability of node j as evaluated by node i based on direct observations at time t upon encountering. Each device keeps a list of its owner's communities/groups of interest which may be changed dynamically. $D_{ij}^{community-interest}(t)$ is computed as the ratio of common community/group interests between nodes i and j, i.e., $\frac{|community(i)\cap community(j)|}{|community(i)\cup community(j)|}$, where $community(i)$ denotes the set of communities/groups to the owner of node i. When node i and node j directly interact with each other, with permission granted from their owners they also exchange their service and device profiles, so node i can validate whether node j and itself are in a particular community/group. Therefore, the direct observation of community-interest will be close to actual status. To preserve privacy, node i and node j agree on a one-way hash function (with a session key) upon interesting while exchanging the community lists so as not to reveal the identities of their uncommon communities. Again we note that an honest node will submit its community-interest list faithfully (although with hashing to hide identity). A dishonest node, however, can perform self-promotion attacks to submit a fake community-interest list in order to boost its "community-interest" trust. This in turn can lead to its "honesty" trust greatly decreased due to the honesty detection mechanisms used in our protocol design.

Here we note that the friend-list/community-interest list exchange is not during the service request time from node i to node j, but during encountering time of from node i with node j for the purpose assessing each other's cooperativeness and community-interest trust values.

3.2.2 $T_{ij}^{X}(t)$ Update When Node *i* Interacts with Node *k*, $k \neq j$

Whenever node i encounters node k, $k \neq j$, who had prior interaction experience with node j, node k will serve as a recommender to provide its trust recommendation about node j, $R_{kj}^{X}(t)$, to node i which will update $T_{ij}^{X}(t)$as follows:

$$T_{ij}^{X}(t) = (1-\gamma)T_{ij}^{X}(t-\Delta t) + \gamma R_{kj}^{X}(t) \qquad (2)$$

In this case, node i will not have direct interaction experience with node j and instead will use its past trust value $T_{ij}^{X}(t-\Delta t)$ and the new recommendation received from node k (second-hand information $R_{kj}^{X}(t)$ where k is the recommender) whom it interacted to update $T_{ij}^{X}(t)$. The parameter γ is used here to weigh the new recommendation vs. past experience and to consider trust decay over time.

Here we note that the recommendation $R_{kj}^{X}(t)$ provided by node k to node i about node j depends on the status of a node. If node k is a good node, node k (being a good node) will faithfully use its trust evaluation towards node j in component X as the recommendation, i.e., $R_{kj}^{X}(t)$ is simply equal to $D_{kj}^{X}(t)$. If node k is a bad node, node k (being a bad node) can perform bad-mouthing attacks by recommending

$R_{kj}^X(t)$ = 0 if node j is a good node, and can perform ballot-stuffing attacks by recommending $R_{kj}^X(t)$ = 1 if node j is a bad node.

To defend against bad-mouthing and ballot-stuffing attacks from a recommender (node k), node i uses its direct trust toward node k, $D_{ik}^X(t)$, to assess if node k is a trustworthy recommender in trust property X. Hence, we introduce another parameter $\beta \geq 0$ as follows:

$$\gamma = \frac{\beta D_{ik}^X(t)}{1 + \beta D_{ik}^X(t)} \quad (3)$$

Essentially β is a user controllable parameter to specify the impact of a recommendation on $T_{ij}^X(t)$ such that the weight γ assigned to $R_{kj}^X(t)$ in Equation 2 is normalized to $\beta D_{ik}^X(t)$ relative to 1 assigned to past information. Thus, the contribution of $R_{kj}^X(t)$ received from node k increases proportionally as either $D_{ik}^X(t)$ or β increases.

3.2.3 Trust Parameters α and β

Our trust aggregation and propagation protocol described above has two parameters: α and β. Parameter α is to tune the tradeoff between new direct trust vs. past information. Increasing α will put more weight on new direct observations $D_{ij}^X(t)$. Parameter β is to tune the tradeoff between new indirect recommendations vs. past information, taking into consideration of the trust toward the recommender. Increasing β will put more weight on new recommendations $R_{kj}^X(t)$. A key concept of our adaptive trust management protocol is that instead of having fixed weight ratios α and β, we allow the weight ratios to be adjusted dynamically in response to changing network conditions to improve trust assessment accuracy and thus provide resiliency against slandering attacks such as bad-mouthing and ballot-stuffing attacks.

3.3 Trust Formation

$T_{ij}^X(t)'s$ where X = honesty, cooperativeness, and community-interest are separately assessed by node i. How to form an overall trust out of the three $T_{ij}^X(t)'s$ is a trust formation issue and depends on the trust requirement of the IoT application running on top of our trust management protocol. The goal of our adaptive trust management design in trust formation is to dynamically discover the best way to form trust out of identified trust components to maximize the application performance, in response to dynamically changing conditions. We will discuss and illustrate our adaptive trust formation design with two real-world social IoT service composition applications in Section 5.

3.4 Cost Analysis

Based on our trust propagation and aggregation protocol design, a node updates its trust toward other nodes upon encountering or interacting with another node. Two nodes involved in a direct interaction activity (as described in Section 3.2.1) can directly observe each other and update their trust assessments. Two nodes encountering each other (as described in Section 3.2.2) exchange their trust recommendations toward other nodes. The storage cost per node is therefore O($N_T N_X$) where N_T is the number of IoT devices, and N_X is the number of trust properties (3 in our protocol design). This storage requirement can still be excessive for IoT devices with limited memory space. We refer the readers to a caching design in [8] as a possible solution to mitigate this problem without overly sacrificing trust accuracy and convergence properties. The communication overhead per node (from node i's perspective) is O($N_X \sum_{j=1}^{N_T} \pi_{ij}$) where 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 [25]. Essentially upon encountering node j, node i exchanges messages with node j for both direct trust assessment of node j of N_X trust properties (i.e., through hashed friend and community-interest lists) and indirect trust assessment of other nodes in the system (i.e., through trust recommendations). In practice the message overhead is lower because one can combine several pieces of information into one message during transmission.

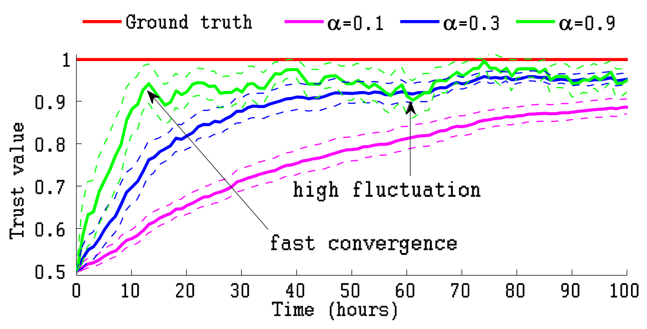
4 PROTOCOL PERFORMANCE EVALUATION

In this section, we evaluate our proposed adaptive trust management protocol based on ns3 simulation to validate the convergence, accuracy, and resiliency properties of our protocol design. The readers are referred to [6] for a formal proof. Later in Section 5, we will demonstrate the utility of our adaptive trust management protocol design with two real-world social IoT applications.

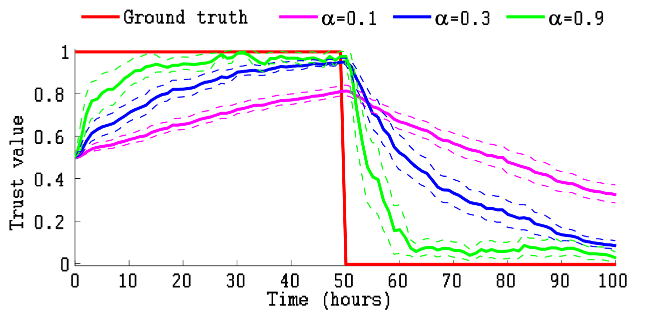
4.1 Social IoT Environment Setup

In our experimental setup the status of each node is changing dynamically. The input is specified by a set of input parameters in Table 1. We consider a hostile environment where the percentage of malicious nodes $\lambda \in [10, 90\%]$ is randomly selected out of all IoT devices. The default value of λ is 30% and we will test the sensitivity of performance results with respect to λ. A node selected to be in this "malicious" population is benign initially, but turns malicious after a period of time $T_c \in [0, 100 \text{ hrs}]$ randomly generated is elapsed, after which it will perform attacks as described in Section 2. On the other hand, a node not selected in this "malicious" population remains benign throughout the simulation. With this setup, while the objective trust or ground truth of a good node remains constant, the objective trust or ground truth of a malicious node changes dynamically.

We consider a social IoT environment with N_T=50 heterogeneous smart objects/devices with all of them providing various services. These devices are randomly distributed to N_H=20 owners. The social cooperativeness relationship among the devices is characterized by a friendship relationship (matrix) [28] among device owners. That is, if the owners of devices i and j are friends, then there is a 1 in the ij position. While our protocol allows



(a) Trust of a good node randomly picked.



(b) Trust of a malicious node randomly picked.

Figure 2: Effect of α on honesty trust evaluation (a) toward a good node and (b) toward a malicious node which turns bad at T_c=50 hours. Trust converges in both cases. There is an inherent trade-off between trust convergence time vs. trust fluctuation. Specifically, as the value of α increases, the trust value converges to ground truth faster, but the trust fluctuation also becomes higher.

dynamic friend lists, the friend list kept by each device is simulated initially and remains the same throughout the simulation. Devices are used by their owners in one or more social communities or groups. A device can belong to up to N_G=10 communities or groups. This is also simulated and remains fixed throughout the simulation. We assume that the encounter or interaction pattern follows a bounded power law distribution ([10mins, 2 days]) with the slope equal to 1.4, resulting in the average interaction-contact time T being 4 hours. The settings are close to those obtained from real traces [25]. The total simulation time is 100 hours.

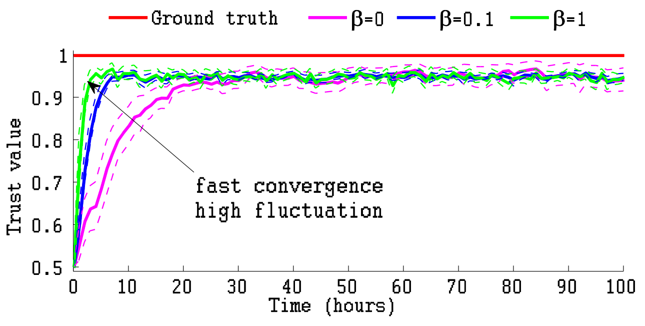
The initial trust value of all devices is set to ignorance with a trust level of 0.5. Our intent is to show that an IoT device upon hostility changes can adaptively select trust protocol settings in terms of α and β to best tradeoff the trust convergence rate and trust fluctuation rate to obtain an acceptable mean absolute error (MAE) between the trust value obtained vs. ground truth.

4.2 Effect of α on Trust Evaluation

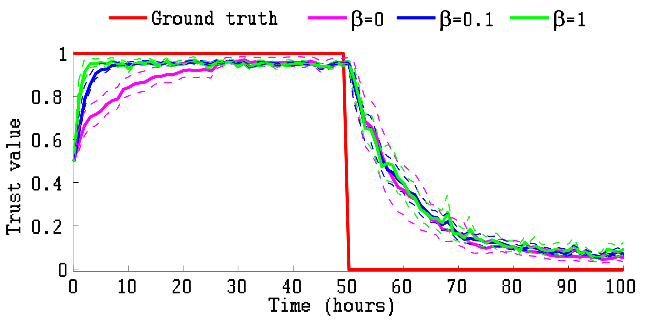
We first investigate the effect of design parameter α on trust evaluation. Recall that α is the weight associated with direct trust with respect to past experience in Equation 1. For sensitivity analysis of α, we vary α by selecting different values (0.1, 0.3, and 0.9) and set β to 0 to isolate its effect. The percentage of malicious nodes λ is 30%. Here we only give the results for the *honesty* trust property evaluation. The other two trust properties follow the same trend.

Figure 2(a) shows the effect of α on *honesty* trust evaluation toward a "good" node whose ground truth status does not change as time increases. The ground truth honesty status for this good node is constant at 1. The dash lines show the empirical confidence intervals with 90% confidence. We can see that the trust evaluation approaches ground truth as time increases. Further, we observe that as the value of α increases, the trust value converges to ground truth faster, but the trust fluctuation also becomes higher. Here we observe that the trust convergence time is 5 to 10 hours because the average inter-arrival interaction time following a bounded power law distribution is set to 4 hours (as listed in Table 1).

Figure 2(b) shows the results of trust evaluation for *honesty* toward a "malicious" node randomly selected whose status goes from benign to malicious after $T_c = 50$ is elapsed. We can see that after the status change, the trust evaluation converges towards the new ground truth status. In addition, as the value of α increases, the trust evaluation converges to the new ground truth status faster, albeit with a higher fluctuation. This result validates the convergence, accuracy, and resiliency properties of our protocol design.



(a) Trust of a good node randomly picked.



(b) Trust of a malicious node randomly picked.

Figure 3: Effect of β on honesty trust evaluation (a) toward a good node and (b) toward a malicious node which turns bad at T_c=50 hours. Trust converges in both cases. There exists a trade-off between trust convergence time vs. trust fluctuation. As β increases, the trust evaluation converges to the ground truth faster, but the trust fluctuation becomes higher. The effect of β is not as significant compared to the effect of α.

4.3 Effect of β on Trust Evaluation

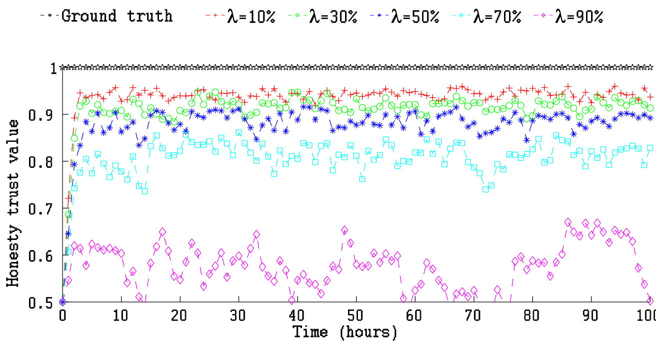
Next, we investigate the effect of design parameter β on trust evaluation. Recall that β is related to γ by Equation 3, representing the weight associated with the indirect recommendation with respect to past experience in Equation 2. For sensitivity analysis of β, we vary β by selecting different values (0, 0.1, and 1), and set α to 0.5 to isolate its effect.

Figure 3(a) shows the effect of β on *honesty* trust evaluation of a good node. Again, we can see that our trust evaluation approaches ground truth status as time increases. We also observe that as β increases, the trust evaluation converges to the ground truth faster, but the trust fluctuation becomes higher. The reason is that using more recommendations (higher β) helps trust convergence through effective trust propagation. However, one can see that the effect of β is insignificant compared to the effect of α. The reason is that very often in a social IoT environment, the chance of a trustor interacting with a recommender is higher than the chance of the trustor directly interacting with a trustee. As long as $\beta > 0$, adaptive trust management is able to effectively aggregate trust using recommendations from a large number of recommenders, thus making the effect of further increasing the value of β insignificant.

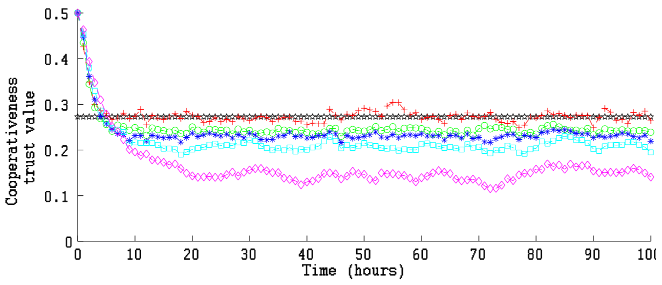
Figure 3(b) shows the effect of β on *honesty* trust evaluation of a malicious node randomly selected whose status goes from benign to malicious after $T_c = 50$ hours is elapsed. Again, we see that after the ground truth status changes, our trust protocol quickly converges towards the new ground truth status. Initially using more recommendations ($\beta > 0$) in trust evaluation helps trust convergence. However, after the status change, the convergence behavior is about the same regardless of β. This is partly because the effect of β (Figure 3(b)) is insignificant compared to the effect of α (Figure 2(b)) and partly because an honest recommender can adversely provide obsolete and inaccurate trust recommendations toward a malicious node, as it has not interacted with the malicious node since the malicious node's status change. As the trustor will not exclude these inaccurate recommendations from good nodes, using more recommendations does not accelerate the pace of trust convergence.

4.4 Adaptive Trust Management in Response to Dynamically Changing Hostility Conditions

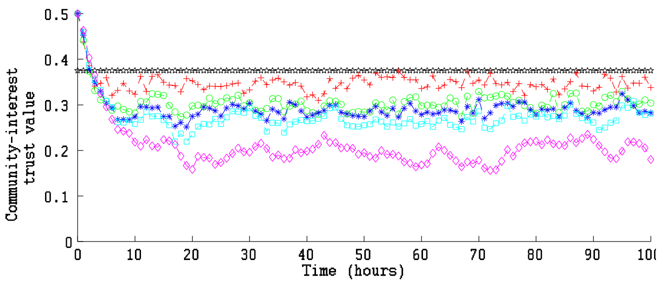
From Figures 2 and 3, one can see that the trust evaluation quickly converges and it is remarkably close to the ground truth status, demonstrating its resiliency against trust attacks. We further validate resiliency of our adaptive trust management protocol toward trust attacks in IoT environments with a varying degree of hostility. We consider five different hostile environments with the percentage of malicious nodes λ being 10%, 30%, 50%, 70%, and 90%.



(a) Honesty trust



(b) Cooperativeness trust



(c) Community-Interest trust

Figure 4: Effect of hostility on dynamic trust evaluation of a good node toward another good node. Our adaptive trust management protocol can react to changing hostility by dynamically choosing the best (α, β) values to tradeoff the trust convergence rate and trust fluctuation rate to obtain an acceptable MAE between the trust value obtained vs. ground truth.

Figures 4(a), 4(b), and 4(c) show trust evaluation results of a good node randomly picked toward another good node also randomly picked for honesty (ground truth trust = 1), cooperativeness (ground truth trust = 0.28), and community-interest (ground truth trust = 0.38), respectively. One can see that the trust evaluation quickly converges and it is remarkably close to the ground truth status (marked with solid lines) with MAE less than 10% when $\lambda \leq 50\%$, demonstrating our protocol's high resiliency to trust attacks. As λ increases, the MAE of trust evaluation inevitably increases because of more false recommendations from malicious nodes. Even when most nodes are malicious with λ going from 70 to 90%, the MAE only goes from 12 to 40%. This demonstrates our protocol's high resiliency toward attacks even in extremely hostile environments.

Here we note that given knowledge of environment hostility (expressed in terms of λ), our adaptive trust management protocol can react to changing hostility by dynamically choosing the best (α, β) values to tradeoff the trust convergence rate and trust fluctuation rate to obtain an acceptable MAE between the trust value obtained vs. ground truth. We will discuss how one may apply the analysis results at runtime in Section 6.

5 IoT Application Performance

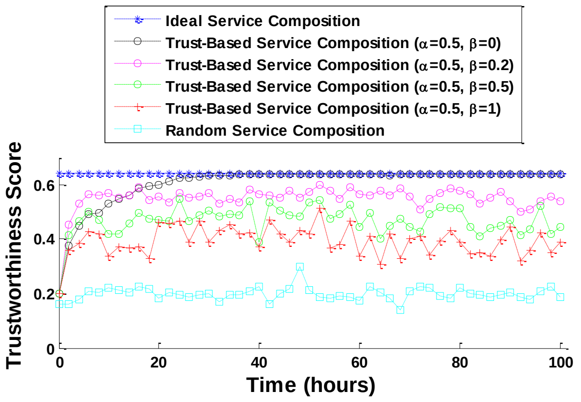
To demonstrate the effectiveness of our proposed trust protocol for IoT systems, we consider two real-world social IoT applications [2, 3, 4] which require dynamic service composition and binding [19, 29]. Such social IoT applications running on top of our trust protocol aim to first compose a service plan (this is the service composition part) and then select the most trustworthy IoT nodes (this is the service binding part) for providing services requested such that the *trustworthiness* score representing the goodness of the service composition is maximized. We compare the performance of trust-based service composition with two baseline approaches:

- *Ideal Service Composition:* it returns the maximum achievable trustworthiness score by always knowingly selecting service providers with the highest "ground truth" trustworthiness scores (based on the actual status). This scheme in practice is not achievable because we do not know ground truth status.
- *Random Service Composition:* it selects service providers randomly without regard to trust.

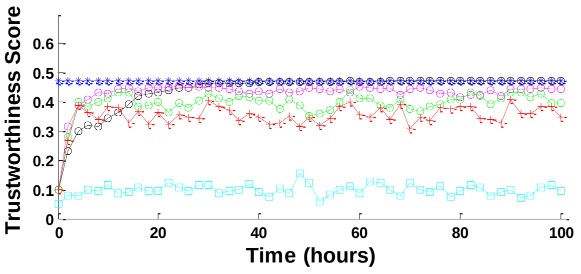
5.1 Smart City Air Pollution Detection

We consider a smart city IoT application running on Alice's smartphone for air pollution detection [4]. Alice tries to avoid stepping into high air pollution areas (in terms of the levels of carbon dioxide, PM10, etc.) for health reasons. Alice's smartphone is a member of the air pollution awareness social network. She decides to invoke her smartphone to connect to sensor devices in an area she is about to step (or drive) into. Alice knows that many IoT devices will respond, so she needs to make a decision on which sensing results to take. She instructs her smartphone to accept results only from *n*=5 most "trustworthy" sensors and she will follow a trust-weighted majority voting result. That is, each yes or no recommendation is counted as 1 weighted by Alice's trust toward the recommender. If the total trust-weighted "yes" score is higher than the total trust-weighted "no" score, Alice will step into the area; otherwise, she will make a detour to avoid the area.

This smart city air pollution detection application is essentially a simple trust-based service composition IoT application in which Alice will simply select *n*=5 IoT devices for which she trusts the most. Therefore, the *trustworthiness* score of this service composition application which it aims to maximize is simply the sum of the individual trustworthiness scores. Since this application involves a simple binary decision (yes or no), we consider a simple trust formation design as follows. If a selected service provider does not pass the minimum honesty trust



(a) Low hostility at λ =10%



(b) High hostility at λ = 50%

Figure 5: Performance comparison for the smart city air pollution detection application.

threshold, the trustworthiness score is zero; otherwise, the trustworthiness score is determined by the social ties between the service provider and the service requester, i.e., a higher trustworthiness score is given if they have more common friends, or if they share more community interests. The rationale of using honesty trust to screen service providers is to avoid malicious service. The rationale of using cooperativeness and community interest trust to subsequently rank service providers is that trustee nodes with which a trustor node has good social ties can most likely provide good service in social IoT environments. Alice decides to set the minimum honesty trust threshold as 0.5. With the reasons given above, her smartphone (as node *i*) estimates the trustworthiness score $T_{ij}(t)$ toward each service provider (node *j*) as follows:

$$T_{ij}(t) = \begin{cases} 0, & if\ T_{ij}^{honesty}(t) \leq 0.5 \\ \min\begin{pmatrix} T_{ij}^{cooperativeness}(t), \\ T_{ij}^{community-interest}(t) \end{pmatrix}, & if\ T_{ij}^{honesty}(t) > 0.5 \end{cases} \quad (4)$$

Figure 5 compares trust-based service composition against two baseline service comparison methods (random and ideal). The experimental setup is the same as that in Section 4. The performance metric is the combined trustworthiness score for *n*=5 service providers selected. We consider 4 versions of trust-based service composition by selecting 4 different sets of design parameters: $(\alpha, \beta) = (0.5, 0)$, $(\alpha, \beta) = (0.5, 0.2)$, $(\alpha, \beta) = (0.5, 0.5)$, and $(\alpha, \beta) = (0.5, 1)$ for the purpose of testing the effect of (α, β)on application performance.

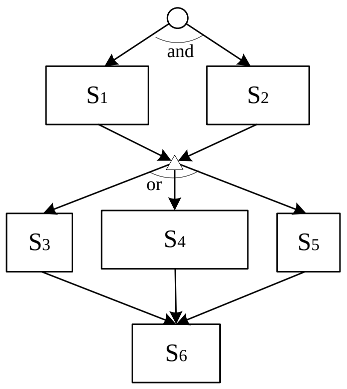


Figure 6: A service flow structure for the augmented map travel assistance application specifying the order of service execution for Bob's service request "fill me with the best grilled hamburger within 20 minutes and under a $30 budget." Here S_i represents abstract service *i* which can be a piece of information, a taxi service, or a hamburger service, and will be provided by an IoT service provider that is selected by Bob's smartphone. S_1 and S_2 are connected by a parallel structure (AND) meaning that they are services to be run concurrently (e.g., different information service providers about which hamburger shop within 20 minutes of taxi ride is the best). S_3, S_4 and S_5 are connected by a selection structure (OR) meaning that they are competitive services (e.g., different taxi companies) and only one will be chosen to execute. S_6 (e.g., the hamburger shop selected) is to be run sequentially after the upper level service bindings are completed.

We see that as λ (the percentage of malicious nodes) increases, the combined trustworthiness score obtained by each protocol decreases because of fewer good service providers exist in the social IoT environment. For example, a service provider with the highest trustworthiness score when λ =10% might become malicious when λ =50%, making the combined trustworthiness score lower under *ideal service composition*. More importantly in all cases, trust-based service composition significantly outperforms random service composition and approaches the maximum achievable performance by ideal service composition.

In Figure 5, all curves reach trust convergence although trust fluctuation is higher when β is higher because placing a higher weight on recommendations will make trust evaluation more sensitive to bad-mouthing/ballot-stuffing attacks. We see that there is a crossover point on the trustworthiness curves of two trust-based service composition methods. For example, before the crossover point, trust-based service composition under the setting of $(\alpha, \beta) = (0.5, 0.2)$ performs better, while after the crossover point, trust-based service composition under the setting of $(\alpha, \beta) = (0.5, 0)$ performs better. The reason is that while using recommendations helps trust quickly converge, it also introduces trust bias because of bad-mouthing and ballot-stuffing attacks. We observe that the crossover time point increases as the percentage of malicious nodes increases. Specifically, the crossover point is at t = 12 hours for λ = 10% and t = 26 hours for λ = 50%. Thus, in a dynamic IoT environment in which the hostility (in terms of the percentage of malicious nodes) changes over time, adaptive trust-based service management is achieved by choosing the best design parameter settings (α, β) to maximize the service composition application performance.

5.2 Augmented Map Travel Assistance

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 [2] 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 encountering, which provide information, food, entertainment (e.g., ticket purchasing), and transportation services [2]. Bob instructs his smartphone to make selection decisions dynamically, so it can leverage new information derived from direct experiences as well as recommendations received from IoT devices it encounters. In response to a service request issued by Bob, his smartphone performs the following actions:

- Formulate a service plan based on the results gathered.
- Invoke necessary services to meet Bob's service demand and requirements.

The augmented map travel assistance application running on his smartphone composes a service workflow plan as shown in Figure 6 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 the 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.

Since in this application Bob needs an overall trustworthiness score to tell him how much he can trust the service plan formulated, we consider a *scaling trust formation model* by which the trustworthiness score of node *i* toward node *j* is computed as:

$$T_{i,j}(t) = min\left(1, T_{ij}^{honesty}(t) \times \frac{T_{ij}^{cooperativeness}(t)}{T_{AVG}^{cooperativeness}(t)} \times \frac{T_{ij}^{community-interest}(t)}{T_{AVG}^{community-interest}(t)}\right) \qquad (5)$$

where $T_{AVG}^{cooperativeness}(t)$ and $T_{AVG}^{community-interest}(t)$ are the average cooperative trust and community-interest trust values, respectively, toward all IoT nodes for which node *i* had interaction experiences or received recommendations. With Equation 5, node *i* scales the honesty trust of node *j* up or down, depending on node j's cooperativeness trust and community-interest trust relative to the respective average trust value. The scaled honesty trust $T_{ij}^{honesty}(t)$ is the trustworthiness score of node *i* towards node *j*.

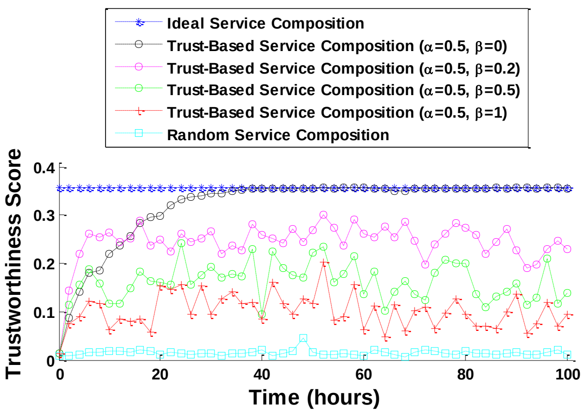
In Figure 6, there are 6 atomic services connected by three types of workflow structures: *sequential*, *parallel* (AND), and *selection* (OR). Each service would have multiple service provider candidates. In this case, the overall trustworthiness score of this service composition application can be 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:

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

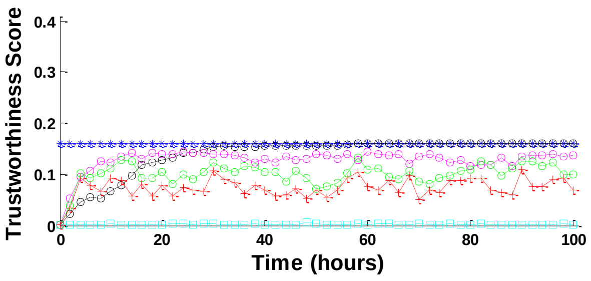
For the flow structure in Figure 6, the outermost structure is a sequential structure connecting (S_1 S_2), (S_3, S_4 S_5), and S_6 out of which (S_2 S_2) is a parallel structure and (S_3, S_4 S_5) is a selection structure.

Figure 7 compares the trustworthiness score of the trust-based service composition application against those under random service composition and ideal service composition. The experimental setup is the same as that in Section 4. Overall, the trend exhibited in Figure 7 is remarkably similar to that in Figure 5, demonstrating the tradeoff between the increase of convergence rate and the decrease of trust accuracy as β increases.

The overall trustworthiness score of the augmented map travel application (in Figure 7) is lower than that of the augmented map travel assistance application (in Figure 5). This is due to the fact that trust formation is application-dependent, so these two social IoT applications have their own ways of computing the overall trustworthiness score. The overall trustworthiness score of the former application is computed as how the overall reliability of a system comprising series-parallel connected components would be computed based on reliability theory. The overall trustworthiness score of the latter application is simply computed as the sum of component trustworthiness scores since it only involves a binary decision (yes or no) based on trust-weighted majority voting. It is noteworthy that the absolute trustworthiness score obtained is not important. What is important is the performance of trust-based service composition relative to ideal service composition which yields the best application performance. From Figure 7, we once again observe that by selecting the best (α, β) setting, trust-based service composition for the augmented map travel assistance application (with a service flow structure controlling the service execution) outperforms random service composition while approaching the best performance obtainable by ideal service composition. When there is insufficient time for a user to gather enough



(a) Low hostility at λ =10%



(b) High hostility at λ = 50%

Figure 7: Performance comparison for the augmented map travel application.

evidence, it can use a higher β to intake more recommendations. For example, Figure 7 shows that β=0.2 or 0.5 is a better choice for maximizing application performance than β=0 before *t*=10 hours when λ=10% (in Figure 7(a)) and before *t*=20 hours when λ=50% (in Figure 7(b)). Conversely, once the user has gathered sufficient evidence and the trustworthiness score is converged, it is better to use a lower β value to reduce the chance of taking in false recommendations launched by malicious nodes, especially in high hostility environments.

6 APPLICABILITY

The effectiveness of adaptive trust management relies on deploying the best protocol settings dynamically in response to changing environments. The analysis methodology proposed in the paper identifies the best protocol settings (in terms of the two design parameters α and β listed in Table 1) to best tradeoff the trust convergence rate and trust fluctuation rate for achieving the desirable accuracy and maximizing application performance, when given a set of input parameter values (defined in Table 1) as input. The analysis is performed at design time.

One way to apply the results for adaptive trust management is to build a lookup table at static time listing the best α and β settings over a perceivable range of input parameter values. The lookup table as shown in Figure 8 below would store key-value pairs where the "keys" are combinations of input parameter values, and the "values" are the best α and β design parameter values for achieving the desirable accuracy and maximizing application performance under the input parameter values. Then, at runtime, upon sensing the environment changes in terms of input parameter values, a node can perform a simple table lookup operation augmented with extrapolation or interpolation techniques to determine and apply the best α and β settings in response to dynamically changing conditions. The lookup time is O(1) and can be efficiently applied at runtime.

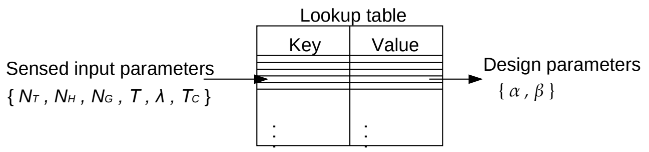


Figure 8: Lookup Table Mechanism. The "sensed input parameters" on the left are input to be sensed at runtime. The "design parameters" on the right are output as a result of a table lookup operation. Depending on data granularity, a set of input parameter values may not directly map to a set of output parameter values. Extrapolation or interpolation techniques may be used to produce the matching output.

7 Related Work

In this section, we survey recently proposed trust management protocols for IoT systems. We contrast and compare our work with existing work so as to differentiate our work from existing work and identify unique features and contributions of our trust protocol design and trust-based service management design for IoT systems.

There is little work on trust management in IoT environments for security enhancement, especially for dealing with misbehaving owners of IoT devices that provide services to other IoT devices in the system. Chen *et al.* [14] proposed a trust management model based on fuzzy reputation for IoT systems. However, their trust management model considers a very specific IoT environment populated with wireless sensors only, so they only considered QoS trust metrics like packet forwarding/delivery ratio and energy consumption for measuring trust of sensors. On the contrary, our work considers both QoS trust deriving from communication networks and social trust deriving from social networks which give rise to social relationships of owners of IoT devices in the social IoT environment. Saied et al. [36] proposed a context-aware and multiservice approach for trust management in IoT systems against malicious attacks. However it requires the presence of centralized trusted servers to collect and disseminate trust data, which is not viable in IoT environments. Relative to [36], our trust protocol is totally distributed without requiring any centralized trusted entity.

Bao and Chen [5] proposed a trust management protocol considering both social trust and QoS trust metrics and using both direct observations and indirect recommendations to update trust in IoT systems. However, the issue of adaptively adjusting trust evaluation in response to dynamically changing conditions so as to cope with misbehaving nodes and maximize the performance of IoT applications running on top of the trust management was not addressed. Relative to [5] cited above, we not only consider multiple trust properties for social IoT environments, but also analyze the tradeoff between trust convergence speed and trust fluctuation to identify the best protocol parameter settings for trust propagation and aggregation to best exploit this tradeoff for minimizing trust bias. Furthermore, it addresses the issue of trust formation for application performance maximization using service composition as an application example.

Very recently, Nitti *et al.* [32] considered social relationships of owners of IoT devices for trust management in social IoT systems. They proposed two models for trustworthiness management. Namely, a *subjective model* deriving from social networks, with each node computing the trustworthiness of its friends on the basis of its own experience and on the opinion of friendly recommenders, and an *objective model* deriving from P2P communication networks with each node storing and retrieving trust information towards its peers in a distributed hash table structure, so that any node can make use of the same information. Their objective model requires pre-trusted nodes be in place for maintaining the hash table, which is questionable in IoT environments. Their subjective model is similar in spirit to our trust model taking into consideration of the social relationships between owners of IoT devices. The fundamental difference is that our model of *objective trust* is based on ground truth or actual status, and our trust protocol dynamically adapts to changing environments by adjusting the best protocol settings to minimize trust bias (the difference between subjective trust and objective trust) and to maximize application performance.

Security has drawn the attention in IoT research [14, 15, 34, 35, 42]. Roman *et al.* [35] discussed threats to IoT, such as compromising botnets trying to hinder services and the domino effect between intertwined services and user profiling. Traditional approaches to network security, data and privacy management, identity management, and fault tolerance will not accommodate the requirements of IoT due to lack of scalability and not being able to cope with a high variety of identity and relationship types [35]. Possible solutions were proposed to each security problem, but no specific protocol or analysis was given. Ren [34] proposed a compromise-resilient key management scheme for heterogeneous wireless IoT. The proposed key management protocol includes key agreement schemes and key evolution policies (forward and backward secure key evolution). The author also designed a quality of service (QoS) aware enhancement to the proposed scheme. However, the proposed scheme does not take social relationships among IoT identities into consideration. Chen and Helal 15 proposed a device-centric approach to enhance the safety of

IoT. They designed a device description language (DDL) in which each device can specify its safety concerns, constraints, and knowledge. Nevertheless, their approach is specifically designed for sensor and actuator devices, and does not consider social relationships among device owners. Zhou and Chao [42] proposed a media-aware traffic security architecture for IoT. The authors first designed a multimedia traffic classification, and then developed this media-aware traffic security architecture to achieve a good trade-off between system flexibility and efficiency. A limitation of their work is that they only considered direct observations to traffic without considering indirect recommendations.

Relative to the security designs/mechanisms cited above, our approach is to use trust to implement security against malicious attacks. We note that our trust system can work orthogonally with these security designs/mechanisms to further enhance security of social IoT systems.

8 Conclusion

In this paper, we developed and analyzed an adaptive trust management protocol for social IoT systems and its application to service management. Our protocol is distributed and each node only updates trust towards others of its interest upon encounter or interaction events. The trust assessment is updated by both direct observations and indirect recommendations, with parameters α and β being the respective design parameters to control trust propagation and aggregation for these two sources of information to improve trust assessment accuracy in response to dynamically changing conditions. We analyzed the effect of α and β on the convergence, accuracy, and resiliency properties of our adaptive trust management protocol using simulation. The results demonstrate that (1) the trust evaluation of adaptive trust management will converge and approach ground truth status, (2) one can tradeoff trust convergence speed for low trust fluctuation, and (3) adaptive trust management is resilient to misbehaving attacks. We demonstrated the effectiveness of adaptive trust management by two real-world social IoT applications. The results showed our adaptive trust-based service composition scheme outperforms random service composition and approaches the maximum achievable performance based on ground truth. We attributed this to the ability of dynamic trust management being able to dynamically choose the best design parameter settings in response to changing environment conditions.

There are several future research areas. We plan to further test our adaptive trust management protocol's accuracy, convergence and resiliency properties toward a multitude of dynamically changing environment conditions under which a social IoT application can automatically and autonomously adjust the best trust parameter settings dynamically to maximize application performance. Another direction is to explore statistical methods to exclude recommendation outliers to further reduce trust fluctuation and enhance trust convergence in our adaptive trust management protocol design.

Acknowledgements

This work was supported in part by the US Army Research Office under Grant W911NF-12-1-0445.

References

[1] S. Adali et al., "Measuring Behavioral Trust in Social Networks," *IEEE International Conference on Intelligence and Security Informatics*, Vancouver, BC, Canada, May 2010.

[2] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A Survey," *Computer Networks,* vol. 54, no. 15, Oct. 2010, pp. 2787-2805.

[3] L. Atzori, A. Iera, G. Morabito, and M. Nitti, "The Social Internet of Things (SIoT) - When social networks meet the Internet of Things: Concept, architecture and network characterization," *Computer Networks,* vol. 56, no. 16, Nov. 2012, pp. 3594-3608.

[4] E. Borgia, "The Internet of Things vision: Key features, applications and open issues," Computer Communications, vol. 54, 2014, pp. 1-31.

[5] 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, September 2012.

[6] F. Bao, Dynamic Trust Management for Mobile Networks and Its Applications, ETD, Virginia Polytechnic Institute and State University, May 2013.

[7] F. Bao, I. R. Chen, M. Chang, and J. H. Cho, "Hierarchical Trust Management for Wireless Sensor Networks and Its Applications to Trust-Based Routing and Intrusion Detection," *IEEE Trans. on Network and Service Management,* vol. 9, no. 2, 2012, pp. 161-183.

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

[9] N. Bui, and M. Zorzi, "Health Care Applications: A Solution Based on The Internet of Things," *the 4th International Symposium on Applied Sciences in Biomedical and Communication Technologies*, Barcelona, Spain, Oct. 2011, pp. 1-5.

[10] B. Carminati, E. Ferrari, and M. Viviani, *Security and Trust in Online Social Networks*, Morgan & Claypool, 2013.

[11] I. R. Chen, F. Bao, M. 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.

[12] I. R. Chen, F. Bao, M. Chang, and J.H. Cho, "Trust-based intrusion detection in wireless sensor networks," *IEEE International Conference on Communications*, Kyoto, Japan, June 2011, pp. 1-6.

[13] I.R. Chen, F. Bao, M. Chang, and J. H. Cho, "Trust management for encounter-based routing in delay tolerant networks," *IEEE Global Telecommunications Conference (GLOBECOM 2010),* 2010, pp. 1-6.

[14] 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, Oct. 2011, pp. 1207-1228.

[15] C. Chen, and S. Helal, "A Device-Centric Approach to a Safer Internet of Things," *the 2011 International Workshop on Networking and Object Memories for the Internet of Things*, Beijing, China, Sep. 2011, pp. 1-6.

[16] J.H. Cho, I.R. Chen, and P. Feng "Effect of Intrusion Detection on Reliability of Mission-Oriented Mobile Group Systems in Mobile Ad Hoc Networks," *IEEE Trans. on Reliability,* vol. 59, 2010, pp. 231-241.

[17] J. H. Cho, A. Swami, and I. R. Chen, "Modeling and Analysis of Trust Management for Cognitive Mission-Driven Group Communication Systems in Mobile Ad Hoc Networks," *International Conference on Computational Science and Engineering*, vol. 2, 2009, pp. 641-650.

[18] J. H. Cho, A. Swami, and I. R. Chen, "Modeling and analysis of trust management with trust chain optimization in mobile ad hoc

networks," *Journal of Network and Computer Applications,* vol. 35, no. 3, 2012, pp. 1001-1012.

[19] K. Dar, A. Taherkordi, R. Rouvoy, and F. Eliassen, "Adaptable Service Composition for Very-Large-Scale Internet of Things Systems," *ACM Middleware,* Lisbon, Portugal, Dec. 2011.

[20] T. Dubois, J. Golbeck, and A. Srinivasan, "Predicting Trust and Distrust in Social Networks," *IEEE 3rd International Conference on Social Computing*, Boston, MA, USA, Oct. 2011, pp. 418-424.

[21] A. Gluhak, S. Krco, M. Nati, D. Pfisterer, N. Mitton, and T. Razafindralambo, "A Survey on Facilities for Experimental Internet of Things Research," *IEEE Communications Magazine,* vol. 49, no. 11, Nov. 2011, pp. 58-67.

[22] A. Gutscher, "A Trust Model for an Open, Decentralized Reputation System," *IFIP International Federation for Information Processing*, vol. 238, 2007, pp. 285-300.

[23] A. J. Jara, M. A. Zamora, and A. F. G. Skarmeta, "An Internet of Things-Based Personal Device for Diabetes Therapy Management in Ambient Assisted Living (AAL)," *Personal and Ubiquitous Computing,* vol. 15, no. 4, 2011, pp. 431-440.

[24] A. Josang, R. Ismail, and C. Boyd, "A Survey of Trust and Reputation Systems for Online Service Provision," *Decision Support Systems*, vol. 43, no. 2, March 2007, pp. 618-644.

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

[26] S. Lee, R. Sherwood, and B. Bhattacharjee, "Cooperative Peer Groups in NICE," *INFOCOM 2003*, vol. 2, pp. 1272-1282, San Francisco, March 2003.

[27] X. Li, R. Lu, X. Liang, X. Shen, J. Chen, and X. Lin, "Smart Community: An Internet of Things Application," *IEEE Communications Magazine,* vol. 49, no. 11, Nov. 2011, pp. 68-75.

[28] Q. Li, S. Zhu, and G. Cao, "Routing in Socially Selfish Delay Tolerant Networks," *IEEE Conference on Computer Communications*, San Diego, CA, March 2010, pp. 1-9.

[29] L. Liu, X. Liu, and X. Li, "Cloud-Based Service Composition Architecture for Internet of Things," *International Workshop on Internet of Things*, Changsha, China, August 2012, pp. 559-564.

[30] G. Liu, Y. Wang, M.A. Orgun, and H. Liu,"Discovering Trust Networks for the Selection of Trustworthy Service Providers in Complex Contextual Social Networks," *19th IEEE International Conference on Web Services*, 2012, pp. 384-391.

[31] R. Mitchell and I.R. Chen, "Effect of Intrusion Detection and Response on Reliability of Cyber Physical Systems," *IEEE Transactions on Reliability*, vol. 62, no. 1, March 2013, pp. 199-210.

[32] 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. 1-11.

[33] F. Paganelli and D. Parlanti, "A DHT-Based Discovery Service for the Internet of Things," *Computer Networks and Communications,* vol. 2012, Article ID 107041, 11 pages, 2012.

[34] W. Ren, "QoS-aware and compromise-resilient key management scheme for heterogeneous wireless Internet of Things," *International Journal of Network Management,* vol. 21, no. 4, July 2011, pp. 284-299.

[35] R. Roman, P. Najera, and J. Lopez, "Securing the Internet of Things," *Computer,* vol. 44, no. 9, Sep. 2011, pp. 51-58.

[36] 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, part B, Nov. 2013, pp. 351–365.

[37] A. A. Selçuk , E. Uzun , and M. R. Pariente, "A Reputation-based Trust Management System for P2P Networks," *Network Security*, vol.6, no.3, May 2008, pp. 235-245.

[38] W. Sherchan, S. Nepal, and C. Paris, "A Survey of Trust in Social Networks," ACM Computing Survey, Vol. 45, No. 4, Article 47, August 2013.

[39] Z. Su et al., "ServiceTrust: Trust Management in Service Provision Networks," *IEEE International Conference on Services Computing*, Santa Clara, CA, 2013.

[40] S. Tozlu, M. Senel, W. Mao, and A. Keshavarzian, "Wi-Fi Enabled Sensors for Internet of Things: A Practical Approach," *IEEE Communications Magazine,* vol. 50, no. 6, June 2012, pp. 134-143.

[41] L. Xiong, and L. Liu, "PeerTrust: Supporting reputation-based trust for peer-to-peer electronic communities," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 7, 2004, pp. 843–857.

[42] L. Zhou, and H.-C. Chao, "Multimedia Traffic Security Architecture for the Internet of Things," *IEEE Network,* vol. 25, no. 3, May-June 2011, pp. 35-40.

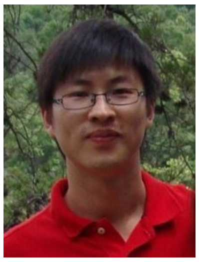
[43] R. Zhou, K. Hwang, and M. Cai, "Gossiptrust for fast reputation aggregation in peer-to-peer networks," *IEEE Transactions on Knowledge and Data Management*, vol. 20, 2008, pp. 1282–1295.

[44] A. Pintus, D. Carboni, and A. Piras, "Paraimpu: a platform for a social web of things," *21st International Conference Companion on World Wide Web*, New York, NY, USA, 2012, pp. 401-404.

[45] R. Girau, M. Nitti, and L. Atzori, "Implementation of an Experimental Platform for the Social Internet of Things," *7th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, July 2013, Taichung, Taiwan. pp.500-505.

AUTHOR BIOGRAPHIES

Ing-Ray Chen received the BS degree from the National Taiwan University, Taipei, Taiwan, 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 Communications Letters, IEEE Transactions on Network and Service Management, Wireless Personal Communications, The Computer Journal*, and *Security and Network Communications*. He is a member of the IEEE and ACM.



Fenye Bao received the B.S. degree in computer science from Nanjing University of Aeronautics and Astronautics, Nanjing, China in 2006, the M.E. degree in software engineering from Tsinghua University, Beijing, China in 2009 and his PhD degree in computer science from Virginia Tech in 2013. His research interests include trust management, security, wireless networks, wireless sensor networks, mobile computing, and dependable computing. Currently he is a technical staff member at LinkedIn, California, USA.

Jia Guo received his BS degree in computer science from Jilin University, China in 2011. His research interests include trust management, mobile ad hoc and sensor networks, Internet of things, delay tolerant computing, and secure and dependable computing. Currently he is pursuing his Ph.D. degree in the Computer Science Department at Virginia Tech.