SC-TRUST: A Dynamic Model for Trustworthy Service Composition in the Internet of Things

Anuoluwapo A. Adewuyi, Hui Cheng, Qi Shi, Jiannong Cao, Xingwei Wang, and Bo Zhou

Abstract— A future Internet of Things (IoT) will feature a service-oriented architecture consisting of lightweight computing platforms offering individual, loosely-coupled microservices. Often, an end-user will request a bespoke service that will require a composition of two or more microservices offered by different service providers. However, the underlying complexities of soft compositions and the increased security risks are inherent in such a massively decentralised and distributed architecture. The use of trust management to secure the IoT is well studied in the literature. However, there are limitations to its use in service compositions in the IoT. Specifically, transparent (agnostic) trust composition and decomposition remain key problems for this area. A novel model for trustworthy service compositions in the IoT, SC-TRUST, is therefore proposed to deal with these challenges. In this study, the trust properties of service compositions and the effect of service workflows on transparent trust composition and decomposition are investigated. Based on the findings, relevant trust evaluation functions are derived to guide the compositions. SC-TRUST was implemented in a suitable application and its performance, in terms of the utility derived and the trust accuracy, convergence and resiliency, was evaluated. The results show that SC-TRUST improves the quality of service compositions and adequately mitigates trust-related attacks, thus increasing both efficiency and security.

Index Terms— Service Composition, Internet of Things, Trust management, Collaborative computing, Distributed applications, Security and Privacy Protection

1 Introduction

THE increasing development and pervasiveness of the Internet of Things (IoT) have facilitated the proliferation of a new generation of smart objects [1]–[5]. The potential benefits and use cases of the IoT apply to virtually every domain of social life, including healthcare, transportation and agriculture [6]–[8]. The IoT requires the extensive interoperability of heterogeneous devices, networks, and technologies, for which the traditional internet is ill-suited. Therefore, a principal and necessary component of the future IoT will be the provision of bespoke services on the fly to satisfy dynamic user requirements. This will require the cooperation and collaboration of individual smart devices offering unique microservices. These microservices can be transparently composed, as required, to provide a service offering that is guaranteed to fulfil the user's service requests [9]–[11]. The process by which this is done is called a service composition. In this context, *transparency* means that the inner working of the composition is abstracted from the users or entities interacting with the composed service. Specifically, performance, access and location transparencies [12] are implied. The composed service appears as a single service to the user requesting it, as the composition should be performed in an agnostic manner. The details of the underlying microservices and any middleware should be abstracted from the user [13].

In a service composition, the IoT middleware consists of a service-oriented architecture (SOA) where each connected device is a service provider (SP) or a service requester (SR) [7], [14]. The underlying microservices are hosted and provided by SPs, while the middleware layer accepts service requests from end-users, performs tasks such as service discovery and aggregation, SP selection, network routing functions, and security and trust management, and delivers the results of the composed service to the requesting user [15]–[18]. The middleware itself may be hosted in the cloud or by another device which provides the composition platform. A device may provide multiple services, e.g. temperature and humidity sensing. However, each service usually belongs to a single service class, which is an abstraction for all services of the same type [10]. The output of one service may be passed as input to another. For example, the geographical coordinates from a GPS sensor may be passed to a weather forecast service and the forecast readings are returned to the user. The network between users, smart objects, and the services they offer forms the foundation of the concept referred as the Social Internet of Things (SIoT) [16], [19], [20].

This work was supported in part by the National Natural Science Foundation of China under Grant No. 61872073 and in part by the LiaoNing Revitalization Talents Program under Grant No. XLYC1902010. *(Corresponding authors: Hui Cheng and Xingwei Wang)*

- *A.A. Adewuyi, Q. Shi, and B. Zhou are with the Department of Computer Science, Liverpool John Moores University, Liverpool, L3 3AF, UK. (e-mail: a.a.adewuyi@2015.ljmu.ac.uk; q.shi@ljmu.ac.uk; b.zhou@ljmu.ac.uk).*
- *H. Cheng is with the Department of Computer Science, University of Hertfordshire, Hatfield, AL10 9AB, UK. (e-mail: h.cheng2@herts.ac.uk).*
- *J. Cao is with the Department of Computing, Hong Kong Polytechnic University, Hong Kong. (e-mail: csjcao@comp.polyu.edu.hk).*
- *X. Wang is with State Key Laboratory of Synthetical Automation for Process Industries, College of Computer Science and Engineering, Northeastern University, Shenyang, China. (email: wangxw@mail.neu.edu.cn).*

The need for secure and trustworthy service compositions is evident. First, there may be multiple SPs offering the same functionality within a service class. In composing a user-requested service, it would be necessary to select the most reliable and least malicious SP in each required service class. Therefore, some method must exist to identify malicious nodes and preclude their service offering from the composition [21]. Second, the middleware must consider the compatibility of the service components to be mixed and matched, whilst ensuring that the results satisfy the requester's utility [15], [22], [23]. The difficulty in finding universal solutions for the IoT context establishes trust as, perhaps, the most important security metric in SOA-based IoT systems. Trust management is wider in scope than traditional security management, in that it also addresses the quality of service (QoS) and reliability of SPs [16], [23], [24].

A trust management system (TMS) provides methods and mechanisms to evaluate the trustworthiness of interacting peers, based on a trust model. The trust model determines how trust is computed in a specific context. While several TMSs have been proposed for trust in generic IoT contexts, little work has been done on the management of trust in SOA-IoT contexts [14], [23]. Furthermore, these models do not consider transparent service compositions, where an SR can neither provide direct trust ratings on nor receive recommendations on SPs of the underlying services. In addition, they do not consider the trustworthiness of the composed service separately from the trust ratings of the SPs. Also, there is an implicit assumption that, due to the social nature of SOA-based IoT, a social relationship exists between the owners of the participating IoT devices. Based on this assumption, such social relationships must factor into the trust estimation. However, in a true service-based IoT where the primary incentive for interactions is to provide or request a service, no other relationship may exist among SPs and SRs outside the given service context. The implication is that these models may exclude trustworthy nodes capable of delivering reliable services and include nodes owned by SPs who have an external social relationship with the SR. Finally, these models do not respect privacy, as the middleware composing the service must be made aware of these relationships among the SPs and SR.

In summary, there is a need for a reliable trust model to guide compositions of IoT services. In a previous work, the authors proposed a trust model, CTRUST, which focused on dynamic trust management for collaborative IoT applications [25]. In CTRUST, the parametrization, weighting, maturity, and decay of trust between nodes are accurately modelled based on suitable functional attributes of the nodes in the collaboration context, and in consideration of the trustor's subjective preferences. In addition, the effects of recommendation on the trust evaluation process were studied. However, the work focused only on collaborative applications where the trustor (i.e. SR) directly requests and receives services from SPs and can, therefore, provide a direct trust rating of the nodes based on their performance, thus establishing a trustor-trustee relationship. Even though service compositions are a special category of collaborative applications, there is no direct trustor-trustee relationship, meaning that the SR is unlikely to be aware of the identities or trust scores of the SPs given that the details of the composition is abstracted from the SR. Thus, the work in this paper significantly extends our CTRUST model, with a focus on dynamic trust management for service compositions in the IoT. The extension includes the following novel contributions:

1. We study the requirements for a suitable trust model for service compositions in the IoT context.
2. We analyse existing techniques for service composition and provide methods for the derivation and aggregation of trust values based on the relevant trust parameters of SPs from different required service classes, extending our contributions in the CTRUST model.
3. We provide a method to reliably estimate the trust score of the composed service based on the indirect trust scores computed for the SPs, in accordance with the SR's requirements. Thus, it provides a novel privacy-preserving solution for transparent trust composition in the IoT.
4. Finally, our model receives trust ratings from the SR based on the satisfaction degree on the composed service, and provides a reasonable method to indirectly update the trust scores of the underlying services. This solves the problem of transparent trust decomposition.

The rest of the paper is organised as follows. Section 2 introduces related concepts, provides a comprehensive overview of service composition techniques, and highlights the necessary properties of an ideal trust model for service composition in the IoT. Section 3 specifies and analyses the SC-TRUST model design. In Section 4, the model is implemented in a service composition application and its performance is evaluated. Section 5 discusses and compares related work. Finally, Section 6 concludes this paper and proposes future work.

2 Preliminaries

This Section presents an overview on service composition in the IoT and its implications for trust modelling.

2.1 IoT Service Composition – Concepts & Techniques

The IoT provides novel ways for users to interact with things around them. The data received from various sensors in the environment can be utilized by multiple actuators to achieve a desired result. Sensors and actuators have a broader meaning in this context and are not limited to traditional electronic devices. For example, a refrigerator may "sense" it is empty of some food stuff and "actuate" an app on the owner's phone to create a shopping list of such items. This list may be passed to an online grocery delivery service at regular intervals that are determined by a bot. At such intervals, the bot "actuates" the online service to create an order and deliver the grocery. This concept of "social sensing" is one possibility of an SOA-based IoT. Usually, the service requested by a user will entail the collaboration of several microservices, as in the previous example. Therefore, a bespoke service will be composed for the user, using two or more microservices which may be offered by different SPs. This architecture offers several advantages that are key to the realisation of the IoT vision, such as modularity, increased reliability and technology heterogeneity and interoperability.

The manner and order by which services are composed plays a role in the trust evaluation strategy. Basically, service compositions may be categorised in two ways. First, the mix of different classes of services involved may be considered. A service class is a logical unit into which SPs offering a similar service may be grouped together. It is an abstraction of a service type. Every service is a member of at least one service class. Secondly, service compositions may also be classified according to the sequence or workflow in which the services are ordered. We discuss both in detail in the following subsections.

In connection with the service class, there are two kinds of service compositions: homogenous and heterogeneous. In a homogenous composition, all the underlying services are from the same class. An example would be an application that allows the SR to request the assistance of some SPs in the collaborative download of a large file by pooling their bandwidths. Each service class is a self-contained collaboration context to which a suitable IoT trust model, such as CTRUST [25], may be applied in selecting the most trustworthy SPs.

In a heterogeneous composition, the underlying services are mixed and matched from different classes. In this scenario, the trust model must identify the most suitable SPs within each service class. The SPs are chosen based on their scores on relevant parameters and in line with the SR's subjective preferences. Heterogeneous compositions have far much more applications in the IoT than homogenous ones. Consequently, they are the major focus of service composition algorithms. As noted earlier, a node may simultaneously offer multiple services from more than one class. In such cases, there may be conflicts between the services which could lead to the performance degradation of some or all the services.

The workflow refers to the order in which services are performed and composed. Generally, there are three basic types of workflow, which may be used in any combination. First, it may be a simple case of selection, where the SR receives a list of SPs along with their trust ratings. The SPs may not be from the same service class. Services may also be composed in a parallel workflow, where two or more SPs simultaneously provide the same or different services. Thirdly, services may be composed in a sequential workflow where the results from one service are provided to the next and so on, and the results of the second service are then returned to the SR.

2.2 Ideal Trust Model for IoT Service Composition

IoT applications generally consist of collaboration and service provisioning [16]. Therefore, an ideal IoT trust model should provide a balance between security, functionality and usability whilst considering the constraints imposed by the limited resources available to most IoT devices. Given that service compositions are based on the task(s) demanded by the user, the trust score of the composed service must be based on the efficient, reliable, risk-minimized completion of the task(s). A review of the existing literature suggests that no work has focused on the decomposition of trust scores assigned to composed services (that is, a top-down approach).
A suitable trust model for service compositions must include methods for reliable trust composition; it must adequately estimate the trust value of the composed service, based on the user's utility preferences and the current trust scores of candidate SPs. Also, it must adequately model trust decomposition; it should accept the trust value given by the user upon consumption of the composed service and decompose that value to update the trust scores of the underlying services in a transparent manner.

Based on the previous work done in [23]–[29], a list of suitable attributes for an ideal TMS for IoT were enumerated and discussed in our previous work [25]. The list consists of the following 9 properties: 1. *Platform Consideration,* 2. *Trust as a Decision (TaaD), 3. Contextual Trust Parametrization,* 4. *Trust Persistence,* 5. *Trust Decay,* 6. *Risk Mitigation,* 7. *Trust Accuracy*, 8. *Trust Convergence,* and 9. *Trust Resilience.* In addition to these, trust models for service composition in the IoT must also possess the following requirements:

10. *Transparent Trust Composition*: The trust model must include methods for estimating the trust score or *trustworthiness* of a composed service appropriately, considering the trust scores of the SPs, the service context, and the workflows involved in the composition. This should be done transparently to the SR; that is, the SR should not be aware of the internal details of the composition, or of the underlying services.
11. *Transparent Trust Decomposition*: Ideally, the SR would give a trust score after consuming the composed service. However, it has been established that the SRs will not interact directly with, or even know, the SPs in a service composition. Therefore, the TMS must incorporate methods to decompose the trust score from the SR and utilize it to update the trust scores of the

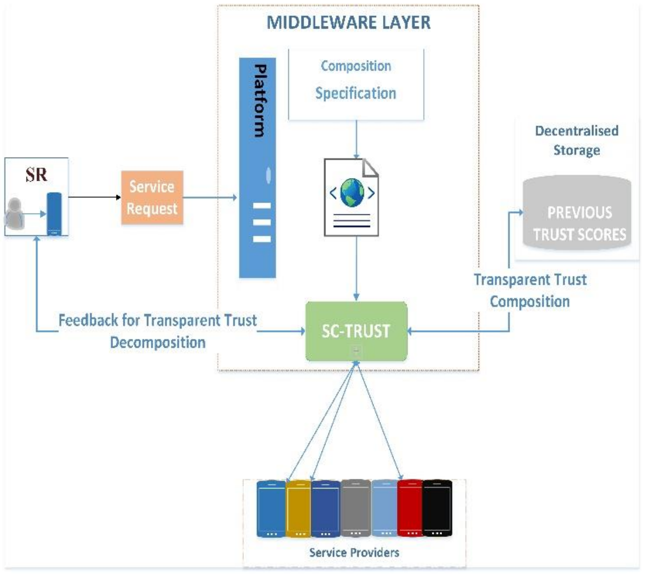


Fig. 1. Basic processes involved for trust computation in SC-TRUST.

underlying services in an impartial and transparent manner.

In Section 3, we will provide methods to extend the CTRUST model to IoT service compositions by fulfilling the last two requirements.

3 SC-TRUST Model Design and Analysis

We now propose SC-TRUST as a suitable trust model to guide compositions of IoT services. Underlying services and SPs are assumed to have trust scores assigned to them prior to the composition, based on their direct interactions with other nodes or peers outside a service composition context. Just as in CTRUST, a trust score is computed based on the objective measurement of a node's performance on functional parameters, which form the basis on which the trust assessment is made. Trust criteria are modelled based on these parameters; a trustor determines the weights of each criterion. Trust scores are stored and are used to guide future interactions, although their importance declines over time. SC-TRUST extends the trust assessment, decay recommendation and aggregation functions of CTRUST to include two novel functions for transparent trust composition and decomposition. A high-level workflow of the model is illustrated in Fig. 1. A stepwise overview of the service composition process is given below:

1. Underlying services are assessed on one or more trust criteria (parameters) relevant to their service class. The assessment results are called *partial trust scores*.
2. An SR requests a *request-granting service*, which is the middleware for composing the services. A candidate platform would be a blockchain built for this purpose, similar to those proposed in [30]–[32].
3. The middleware composes a service workflow pattern to match the SR's requests. Examples of candidate solutions exist in the literature [15], [18]. SC-TRUST can be integrated into a suitable middleware platform.
4. After the service workflow is decided, a handover is made to SC-TRUST to guide the actual composition process.
5. The prior partial trust scores on the parameters identified in (4) for each SP are aggregated according to the SR's weight assignment, to calculate a single trust score for each SP.
6. The trust score of the composed service is estimated transparently through a trust composition function based on the workflow, relevant partial trust scores of the SPs, and weights assigned by the SR.
7. Upon service consumption, the SR may provide a score on each parameter, which is used to transparently update the posterior partial trust scores of the SPs through a trust decomposition function.
8. The trust decay and recommendation functions of CTRUST are inherited and utilized as required.

We will outline the CTRUST functions mentioned in the above process and then proceed to discuss the extensions provided by the SC-TRUST model. Let ***S*** be the set containing the SR and all the SPs available for composition under the model. ***T[S]***, the trust space over S, is an octuple expressed by the following notation:

$$T[S] \equiv \left[T_{ij}, P, W_i, V_{ij}, F, t_{\frac{1}{2}}(i), C, D\right] \forall i, j \in S \tag{1}$$

Where

- T_{ij} is the trust score of SP j from the perspective of SR i;
- $P = \{p_i, p_2, p_3, \ldots, p_n\}$ is the set of all trust parameters or properties from every service class in the composition;
- $W_i = \{w_i(p_1), w_i(p_2), w_i(p_3), \ldots, w_i(p_n)\}$ is the set of weights on each parameter in P, as assigned by SR i;
- $V_{ij} = \{V_{ij}(p_1), V_{ij}(p_2), V_{ij}(p_3), \ldots, V_{ij}(p_n)\}$ is the set of values denoting SR i's perceived assessment (partial trust score) of SP j on each parameter in P;
- $F = f(W, V) \equiv T_{ij}$ is the trust aggregation function;
- $t_{\frac{1}{2}}(i)$ is the half-life of any partial trust score computed for i, that is, the time required for a partial trust score to decay to half its original value;
- C is the trust composition function; and
- D is the trust decomposition function.

3.1 Adapted CTRUST Functions

The CTRUST functions are adapted to model prior partial trust scores to determine which SPs may be selected to participate in a service composition.

3.1.1 *Trust Parameters*

For any SP $j \in S, \left(V_{ij}(p)\right)_t$ is approximately the same if measured by any $i \in S$ at any time t prior to the composition. Suppose that a service composition with n unique service classes,$Q_1 \ldots Q_n$, is set up. Set P is populated with all the parameters from each service class. Some parameters of a service class may not be useful to its role in the service composition and may be given a weight of zero, as we will show in Section 3.2.3.

3.1.2 Parameter Weights

The SR assigns weights to each parameter to reflect their relative importance based on the current subjective opinions of the SR. The weights provide a required differential factor in trust evaluation; thus an SP with a consistent behaviour and value set, V, may have a different perceived trust rating to different SRs. The weights are assigned such that:

$$W_i(p) \in [0,1] \forall i \in S, p \in P \text{ where } \sum_{p \in P} W_i(p) = 1 \quad (2)$$

SRs can assign a weight of zero to parameters they deem irrelevant to the service composition requested. They can also update their record of set W before any service composition request, according to changes, over time, in their perceptions of the relative importance of each parameter. This dynamic weighting considers changes in preference modelling; therefore, it is a more accurate adaptation of social human trust to IoT.

3.1.3 Partial Trust Scores and Aggregation

Set V_{ij} contains the normalized partial trust scores on each parameter p in P. These values are used for service composition and are updated per the SR's post-service scores. The values are computed based on objective assessments that are weighted by the appropriate entry in the set W of the SR. The values are normalised to [0, 1] so that T_{ij} is also within the same range, and the normalisation method must be defined in the trust context. There are two factors to take into account in computing $V_{ij}(p)$ at any time: its previous value before the composition and the decomposed portion of the SR's post-service score which is used to transparently update it. The prior value may be formed from direct interactions and, to a much lesser degree, indirect assessments. The post-service trust score is derived from the assessment given by the SR upon consumption of the service, by means of a trust decomposition function which will be discussed in Section 3.3. The trust aggregation function, F, specifies how the partial trust scores are aggregated to compute T_{ij}. CTRUST uses a weighted sum function, $F = W \times V$. Therefore,

$$T_{ij} = \sum_{x=1}^{n} w_i(p_x) \times V_{ij}(p_x) \, \forall i,j \in S, p_x \in P \quad (3)$$

3.1.4 Trust Decay

An exponential decay function is used; the basis for this is outlined in [25]. The function can be expressed mathematically thus:

$$\begin{aligned} \left(V_{ij}(p)\right)_{0\to t} &= \left(V_{ij}(p)\right)_0 \times \frac{1}{2}^{\frac{t}{t_{\frac{1}{2}}(i)}} \\ &= \left(V_{ij}(p)\right)_0 \times e^{-\lambda_i t} \\ &= \left(\phi_i\right)_t \times \left(V_{ij}(p)\right)_0 \forall i,j \in S, p \in P \end{aligned} \quad (4)$$

$$\lambda_i = \frac{\ln 2}{t_{\frac{1}{2}}(i)} \approx \frac{0.693}{t_{\frac{1}{2}}(i)} \quad (5)$$

$$\left(\phi_i\right)_t = e^{-\lambda_i t} \quad (6)$$

Where

- $(V_{ij}(p))_0$ is a partial trust score at the end of the last session of interactions between i and j;
- $(V_{ij}(p))_{0\to t}$ is the current value of $(V_{ij}(p))_0$ after time t of no interactions between i and j;
- λ_i is a decay constant for partial trust scores from i; and
- $(\phi_i)_t$ is the trust decay multiplier for SR i, which is a proportion of the partial trust score that has not decayed after time t of no interaction.

Equation (4) can be simplified and rewritten as:

$$\left(V_{ij}(p)\right)_{0\to t} = (\phi_i)_t \times \left(V_{ij}(p)\right)_0 \forall i,j \in S, p \in P \quad (7)$$

After an adequate number of interactions in a new session, the effective proportion of $(V_{ij}(p))_0$ that determines $V_{ij}(p)$ in a session becomes 0, based on the trust maturity factor to be defined in Section 3.1.6.

3.1.5 Trust Recommendations

A belief function is used to accurately model recommendations in CTRUST, and its mathematical expression is:

$$\beta_{ij\leftarrow k} = \left(1 - \left|\frac{V_{kj}(p) - \left(V_{ij}(p)\right)_0}{\left(V_{ij}(p)\right)_0}\right|\right) \times (1 - \phi_i) \times V_{ik}(p) \quad (8)$$

This belief function indicates how much a node i is willing to accept a recommendation on j from k, i.e. the weight that i assigns to that recommendation. Therefore, it determines i's indirect assessment of j on parameter p through k, which is defined as:

$$\Psi_{ij\leftarrow k}(p) = \beta_{ij\leftarrow k} \times V_{kj}(p), j \neq k \quad (9)$$

A node cannot provide recommendations on itself. This comprehensively defends against self-promotion attacks. Once new interactions begin between i and j, then $\beta_{ij\leftarrow k} = 0$. Accordingly, and together with the trust decay function, the ability of malicious nodes to perform ballot-stuffing, bad-mouthing or on-off attacks is severely limited. This ensures the reliability of the computed prior partial trust scores.

3.1.6 Trust Maturity

The concept of trust maturity is introduced so that trust values may be updated reliably and consistently. Let Γ be the number of interactions required to reliably measure $V_{ij}(p)$ based on direct assessments, $D_{ij}(p)$, only. Therefore, Γ direct interactions between any two nodes in any session are sufficient to attain a trust maturity or equilibrium. The value of Γ depends on the service class concerned and must be determined by direct interactions with the SPs in the class before a service composition. After Z interactions between i and j in a new session, the effective proportion of a previous trust score, $\left(V_{ij}(p)\right)_0$, is given by:

$$\mu_i = \max\left(\left(1 - \frac{Z}{\Gamma}\right) \times (\phi_i)_t, 0\right) \quad (10)$$

In other words, once $Z \geq \Gamma, \mu_i = 0$. At the start of a new session of interactions between i and j, we set the initial value of the direct trust assessment as $D_{ij}(p) = 0.5$. This is the midpoint between complete distrust (0) and perfect trust (1). It is the neutral prior probability of trust, and the default value of the partial trust score, $V_{ij}(p)$, in the absence of previous interactions or recommendations. Having discussed all the three factors (i.e. trust decay, recommendation and maturity) which affect the prior trust values of any SP *j* before the service composition, they can be aggregated for SR *i* to update SP *j*'s trust value on each parameter *p* at time *t* after the activation of the composed service, which is denoted as $\left(V_{ij}(p)\right)_t$. Let $G \subseteq S$ be the set of nodes with each having a recommendation on SP *j*. Suppose that there are *s* nodes in G: $G(1) \dots G(s)$. Then $\left(V_{ij}(p)\right)_t$ is defined as:

$$\left(V_{ij}(p)\right)_t = \begin{cases} \dfrac{D_{ij}(p)+\left(\phi_i\times\left(V_{ij}(p)\right)_0\right)+\sum_{x=1}^{s}\Psi_{ij\leftarrow G(x)}(p)}{1+\phi_i+\sum_{x=1}^{s}\beta_{ij\leftarrow G(x)}(p)} & if\ Z=0 \\ \dfrac{D_{ij}(p)+\left(\mu_i\times\left(V_{ij}(p)\right)_0\right)}{1+\mu_i} & if\ Z>0 \end{cases} \quad (11)$$

$$\forall p \in P, G(x) \in G, j \notin G$$

The first case (*Z* = *0*) applies at the start of new interactions between *i* and *j*. The purpose of the CTRUST functions is to determine which SPs would be selected *a priori* from each service class. This score becomes $(V_i(p))_o$ just before the instantiation of the composition, and it is used in estimating the trust of the composed service. The partial trust scores for each SP may be stored in a decentralised architecture, such as in a blockchain, where similar composition platforms may access these values. The details of the storage and retrieval mechanisms are abstracted for the purpose of this paper. Having adapted the functions of CTRUST, we focus on its extension for service compositions in the subsections below.

3.2 Transparent Trust Composition

Upon consumption, the SR's posterior feedback may indicate a higher or lower level of satisfaction compared to the prior estimated score. A reliable prior trust estimate should not be higher than the posterior feedback from the SR. Therefore, a novel, bottom-up approach is proposed to compose the trust value. The objective here is to *satisfice* [33], that is to initially find a suitable composition which meets the SR's threshold, even if it is not the best possible composition. This is an optimal strategy because it reduces the time, energy and computational costs involved in composing a service. Then, based on the feedback from the SR, we can update the trust scores of the SPs and compose a better service with every iteration.

The set of prior partial trust scores of each selected SP *j* is denoted now as V_j, and the set *P* contains all parameters for all service classes represented in the composition. The SR assigns weights to each of these parameters to indicate their order of relative importance. This could be implemented in several ways. One way would be to assign a default weighting of 0.5 on a scale of 0 to 1.0 for each parameter. Then the SR can adjust the weighting of any parameter as desired. Another method could be to request that the SR provide pairwise comparisons or ratios on matched parameters (e.g. taxi fare vs cleanliness, cleanliness vs vehicle emissions). Analytic Hierarchy Process (AHP) methods can then be used to elicit consistent weights for each parameter. Once the weights have been determined, we can proceed to estimate the trust score of the composed service recursively, as determined by the workflow. Each workflow is resolved to an equivalent single service characterized by an appropriate set of partial trust scores. The method by which this is done for each workflow type is discussed in the following subsections.

3.2.1 Selection Workflow

In a selection workflow, the SR must select one SP from a group of two or more SPs from the same or associated service classes and offer similar services. Therefore, we know that this group can be represented by the SP with that highest trust score, as this is the SP most likely to be chosen by the SR. Once an SP is chosen, the services of the others are not used (not at this level, at least) and therefore do not impact on the composite trust score. Therefore, to resolve or simplify this workflow, we determine the SP with the maximum prior trust score based on the partial trust values on the same set of parameters, using weights assigned by the SR. Let *Q* be the set of SPs from which a selection is to be made, and $R \subseteq P$ be the set of parameters common to every member of *Q*. Thus, we compute:

$$T_j = \sum_{p_x \in R} w(p_x) \times V_j(p_x),\ \forall j \in Q \quad (12)$$

Therefore, the set *Q* can be resolved to a single logical service that has the equivalent trust characteristics (the same set of partial trust scores, *V*) as the SP with the highest trust score as computed by the above equation. Two or more SPs may tie for the highest trust score. This does not affect the resolution of the services, because the trust score represents the estimated maximum utility that the SR may derive from the consumption of any of such services. Unless there are other non-functional constraints, we can choose the partial trust score set of any of these SPs for the equivalent logical service. This set is represented mathematically by:

$$V \overset{\text{def}}{=} V_j, \text{where } T_j = max\left(\{T_l\}_{l\in Q}\right) \quad (13)$$

3.2.2 Parallel Workflow

In a parallel workflow, multiple SPs provide services concurrently. The SPs may be of the same or different service classes. Considering the case of SPs from the same class, the collective services provided by the group of SPs could be reliably substituted by a single service with a partial trust score equal to the average group score on each trust parameter. Suppose that the SPs are from different service classes. The SPs could be logically replaced by a single service that offers both services. This single service has a set of partial trust scores that is given by the union of the sets of partial trust scores of all the SPs. Where two or more SPs have a partial trust score on the same trust parameter, the replacement service is assigned the average (arithmetic mean) value of the scores on this parameter. This is true for both cases above.

To formalise the above cases, let Q be the set of SPs in the parallel workflow and $R \subseteq P$ be the set of unique parameters for all the SPs in Q. Then the set V for a single service to replace Q is given by:

$$V \stackrel{\text{def}}{=} \left\{ \frac{\sum_{j \in Q \wedge o_j(p_x)=1} V_j(p_x)}{\sum_{j \in Q} o_j(p_x)} \right\}_{p_x \in R} \tag{14}$$

$$o_j(p_x) = \begin{cases} 1, \text{if } V_j(p_x) \in V_j \\ 0, otherwise \end{cases}, \forall j \in Q, p_x \in R$$

Here, $o_j(p_x)$ is used to decide whether partial trust score $V_j(p_x)$ is in SP j's set V_j. Every value in V is the average partial trust score computed separately on each parameter p_x in R from all the partial trust scores on p_x collated from the SPs in Q.

3.2.3 Sequential Workflow

Every service composition can eventually resolve to a sequential workflow. To understand how two microservices composed in sequence may affect the prior trust estimation of the composed service, we study one example.

Suppose that the SR is visiting a new city and does not understand its language. The SR wants a summary of Internet articles about one museum to be translated to the SR's native language. Two microservices are required for the composed service: a summarization service with both accuracy and response time parameters; then a translation service with an accuracy parameter. Suppose that the selected summarization SP has prior partial trust scores of 0.7 and 0.9 on accuracy and response time, respectively. The translation service has an accuracy of 0.8. In this case, the accuracy parameter is a shared parameter belonging to two or more service classes in the composition. If the SR-assigned weights are 0.7 and 0.3 for accuracy and response time respectively, then the prior trust score of the composed service is computed as:

$$Min.\ (0.7,\ 0.8) * 0.7 + 0.9 * 0.3 = 0.76$$

Therefore, we can now define the trust characteristics for a single logical service that is reasonably equivalent to two or more services in a sequential workflow. Let Q be the set of all SPs in a sequential workflow and $R \subseteq P$ be the set of unique parameters for all the SPs in Q. Then the set V of the single service is the union of all distinct elements which have a non-zero weighting, taken from the sets of partial trust scores of all the SPs in Q. Where two or more SPs each have a partial trust score on the same (shared) parameter, the set V contains the minimum partial trust score on this parameter. The minimum partial trust score is computed separately for each parameter in R. This is represented mathematically by:

$$V \stackrel{\text{def}}{=} \left\{ \min\left(\left\{ V_j(p_x) \right\}_{j \in Q \wedge o_j(p_x)=1} \right) \right\}_{p_x \in R} \tag{15}$$

Having discussed the methods by which each type of workflow may be resolved into and replaced by a single equivalent logical service, we can apply them recursively as required until the whole composition is replaced by an equivalent logical service that is defined by the appropriate trust characteristics in its set V. Then the estimated trust score of the composed service is equivalent to the trust score of this logical service and computed according to the following equation, where $w(p_x) = 0$ if no weight is assigned to p_x by the SR:

$$T = \sum_{p_x \in P} w(p_x) \times V(p_x) \tag{16}$$

This model is the view that the SR "sees" in interacting with the composed service. The SR need not have any knowledge of the individual underlying microservices. Also, none of the SPs gains any knowledge of either the SR or another SP from the composition process. Therefore, the method is privacy-preserving and makes it sufficiently difficult for an adversary to bad-mouth, ballot-stuff or perform on-off attacks. The SR then gives a posterior trust evaluation after consuming the service. The method by which this posterior score is transparently decomposed is discussed in the next section.

3.3 Transparent Trust Decomposition

The composed service consists of several microservices. Therefore, a method is required to reliably decompose the partial trust scores given by the SR to the underlying services in a reasonable and impartial manner. This should be done through a top-down approach to persist the logical view of the composed service to the SR and hide any details of the underlying services. While this is a novel and complex problem in the IoT, it is somewhat similar to the problem of group assessment in education [34]–[36].

Some observations can be made from the studies cited above. First, group assessments can be an effective method for both the assessor (SR) and the assessee (SPs). It can increase coordination, cooperation and collaboration among the assesses whilst providing a straightforward way for the assessor to evaluate the performance of all the group members at once. However, it may give malicious SPs an inducement to "free-ride" because the differential contributions of SPs are not acknowledged given that everyone gets the same scores on the same parameters. Also, it may deincentivize high performing SPs because they may be punished for the low performance of other SPs.

Several methods have been proposed to improve the quality and fairness of group assessments. Among them, one method assigns both a group score and an individual score to every member. The individual component is based on a separate piece of work produced by each group member. Thus, the advantage of collaboration and group assessment may be achieved whilst providing an incentive for improving individual performance. Similarly, the individual component punishes bad or low-performing members, thus creating a fair distribution of the marks among the group members. These characteristics make this method feasible in our context. Moreover, the group members are not required to provide feedback on their peers. Therefore, it maintains the privacy aspects of the composition.

Oftentimes, only one SP per service class would be required in a service composition. Although service classes may share parameters, each service class is usually differentiated by at least one parameter. This creates a method to give an individual rating to an SP of the same class. Therefore, an SP j from any service class would share at most n-1 trust ratings with another SP from a different service class, and the rating for at least one parameter would uniquely apply to j. Usually, this unique parameter is also the defining parameter regarded by most SRs. Thus, it is appropriate to expect that these defining parameters would incentivize better quality of service from good nodes and significantly lower the trust scores of malicious nodes. The set of unique parameters, $U \subseteq P$, is expressed mathematically by the following equation:

$$U \overset{\text{def}}{=} \left\{ p : p \in P \wedge \sum_{j \in S} o_j(p) = 1 \right\} \quad (17)$$

Suppose that an SP adjusts its service provision such that it receives good ratings on its unique parameters but bad ratings on the others. This is an opportunistic service attack. However, the attack fails in our model because the SP destroys its own reputation as well with no overall gains. Our model can also deter such attacks. First, the SP does not know the identities of other SPs involved in the collaboration. Second, the middleware platform is a "wholesale buyer" of services and interacts directly with the SP. The SP cannot selectively attack service compositions because it cannot tell whether its services are being utilized in a composition or otherwise. Third, given that there are costs to service provisioning and that the SP cannot gain from malicious behaviour, a rational SP would be motivated to provide good services.

On a shared parameter, the post-service trust score received from an SR is decomposed according to the individual partial trust scores as defined in Equation 11 and the composed partial trust score on that parameter as defined in Equation 15. Suppose $V(p)$ and $V^+(p)$ are the composed and post-service feedback scores, respectively, on parameter p. Then, the partial trust score of each SP sharing this parameter is updated according to the following equation:

TABLE 1
LIST OF MODEL VARIABLES

Symbol	Description	Type
T_{ij}	Trust value of j as computed by i, at the current instance and context	Derived
p	A trust metric or parameter by which trust is assessed in the current context	Design
$w_i(p)$	The importance of p as determined by i	Input
$D_{ij}(p)$	The direct assessment score of j, as measured or perceived by i, on parameter p	Derived
$\left(V_{ij}(p)\right)_t$	The trust score of j, as determined by i, on parameter p, at time t	Derived
$\left(V_{ij}(p)\right)_0$	The trust score of j, as determined by i, on parameter p, at the end of the last session	Derived
$(\phi_i)_t$	weight of $\left(V_{ij}(p)\right)_0$ in next session of interactions after time t of no interactions between i and j	Input
$\left(V_{ij}(p)\right)_0$	The real value of $\left(V_{ij}(p)\right)_0$ that determines $\left(V_{ij}(p)\right)_t$ at time t, in the current session	Derived
μ_i	Best defined as $\left(V_{ij}(p)\right)_{0\to t}$ / $\left(V_{ij}(p)\right)_0$	Derived
$\beta_{ij\leftarrow k}$	The proportion of a recommendation on j, from k, that i is willing to accept	Derived
$\Psi_{ij\leftarrow k}(p)$	The indirect assessment score of j on parameter p, as received by i from k, based on $\beta_{ij\leftarrow k}$	Derived
Γ	Number of interactions in a session required to reliably measure trust by direct assessment only	Design
N[C]	Number of all nodes in the collaboration context and community, C	Input
N[G]	Number of nodes in C that are actively in collaboration with i at the current instance	Input

$$V_j^+(p) = V_j(p) + \left(V^+(p) - V(p)\right) \times \frac{V_j(p)}{\sum_{j \in S \wedge o_j(p)=1} V_j(p)} \quad (18)$$

Where $V_j(p)$ *and* $V_j^+(p)$ are the pre-service and updated partial trust scores, respectively, for SP j on parameter p. Equation 18 utilizes a game theory approach that de-incentivizes free-loading. This is achieved by differentially penalizing each SP sharing the parameter if the post-service score is lower than the prior estimated score. Given that the SPs have zero knowledge of the partial trust scores of one another, a rational SP would act to avoid the penalty by ensuring that the posterior score received on the shared parameter is as high as possible. Also, given that an SP cannot readily tell whether its services are being offered directly to a node or to a composition platform, a rational SP would seek to avoid being penalized so that its partial trust scores can remain high enough to be selected to provide services in the future. This fosters cooperation and motivates each SP to provide their best possible service.

Table 1 provides a brief description of some relevant model variables. The above discussion shows that the SC-TRUST model is privacy-preserving and transparent to both the SP and the SR and can significantly reduce the ability to perform maliciously. SC-TRUST can also deter the middleware platform from being biased or malicious. This is because good SPs who have received good ratings from other nodes or platforms would eventually decline to interact with a misbehaving middleware platform. Therefore, the quality of its composed services would be low, and SRs would no longer subscribe to its services, leading to a loss of reputation and revenue. Given that there are other such composing platforms from which SRs can request services, the provider of the platform is motivated to remain neutral and provide trustworthy services.

4 Model Performance and Evaluation

SC-TRUST was evaluated in a collaborative download (CD) application similar to [25], but with a service composition approach, instead. The evaluation is approached along two different strategies. In the first, we evaluate the impact of the model on the level of utility gained by an SR. We compare the performance of the trust-based composition to a random composition and an SR-composed composition (that is, the SR directly shops for the individual services). In the second, we evaluate three major trust properties of the model, namely trust accuracy, convergence, and resilience. We compare the results to those obtained in CTRUST. Through this we show the efficiency of the model in service composition applications, as well as the effectiveness of the model in mitigating trust-related attacks.

Given that the model is based on CTRUST, it inherits the verified methods of trust persistence, decay, and parametrization. The trust properties of *platform consideration* and *Trust as a decision (TaaD)* are implicit in the model's design. We will show that the trust composition and decomposition methods provide reliable estimates of the trust values of an SP and approach the ground truth values. Overall, we prove that SC-TRUST includes all the attributes required in an ideal trust model for IoT service composition, as enumerated in Section 2.2.

In Section 4.1, the experimental setup is outlined, and the parameters are defined. In Section 4.2, the utility of the model is evaluated to determine the accuracy of the transparent trust composition in SC-TRUST. Lastly, the trust properties (accuracy, convergence, and resilience) of the model are evaluated in Section 4.3, and they show the effectiveness of the transparent trust decomposition method.

4.1 Context Overview: Collaborative Downloading

Collaborative downloading (CD) is a concept well studied in the literature [37]–[39]. Usually, it employs a peer-to-peer (P2P) paradigm and involves the pooling of the bandwidth of multiple devices to download a common resource. This has potential applications in SOA-IoT. We utilize the experimental setup described in our previous work [25] and adapt it from a direct collaboration to a service composition. The hardware remains the same, but the virtual environment is Mininet, an emulator for prototyping Software Defined Networks (SDN), running in VirtualBox. An emulator, rather than a simulator, is used to better

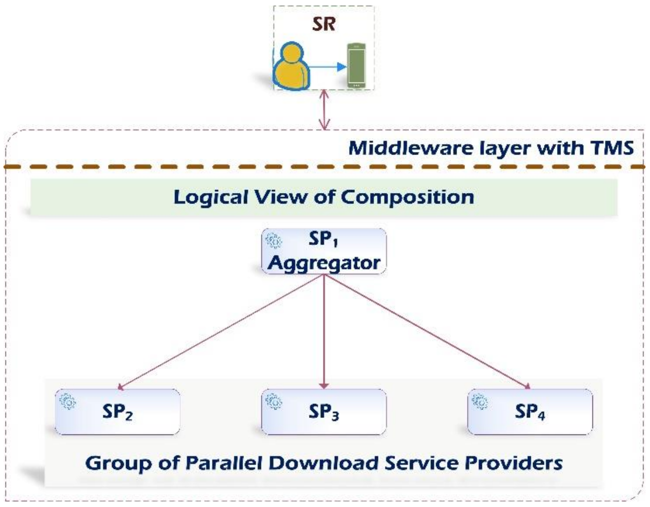

SR
Middleware layer with TMS
Logical View of Composition
SP$_1$ Aggregator
SP$_2$
SP$_3$
SP$_4$
Group of Parallel Download Service Providers

Fig. 2. Service composition for a collaborative download, consisting of one aggregation SP and three download SPs working in parallel.

model the host and network constraints in IoT networks. Mininet was chosen because it suits SDN-type networks and is easily extensible for our purpose. In addition, it is open source, written in python and well-documented. This supports reproducibility of both the experimental setup and results.

The composition process is as follows: first, an SR requests a CD session, sending the URL of the file to be downloaded as input. Then the middleware composes the services as follows: SPs which offer the download services are selected through the trust management system. The download services are composed in sequence with an aggregation service offered by a different SP. The aggregation SP divides the resource into workloads or blocks, verifies the downloaded contents, and checks for errors or malicious modifications. Then, it aggregates the verified blocks into the originally requested resource and sends the complete content to the SR over the local network connection. The communication protocol is managed by the middleware layer in a way that preserves the privacy of the SPs. This setup is illustrated in Fig. 2.

The trust parameters are based on functional requirements which are considered by researchers to be the most important for a successful collaborative download. Thus, the set P of all the trust parameters from both service classes in the composition contains three elements, namely, *Successful Completion Rate (SCR)*, *Cumulative Bandwidth Average (CBA)*, and *Inverse Risk Index (IRI)*. These have been discussed extensively in [25]; thus, we focus only on the modifications in this paper.

The SCR is designed as a parameter unique to the aggregation service class whilst the CBA and IRI are unique to the download service class. In this setup, however, there is one aggregation SP and a group of 3 to 8 download SPs for each CD session. Therefore, the download SPs share the scores assigned by the SR on the CBA parameter, according to the

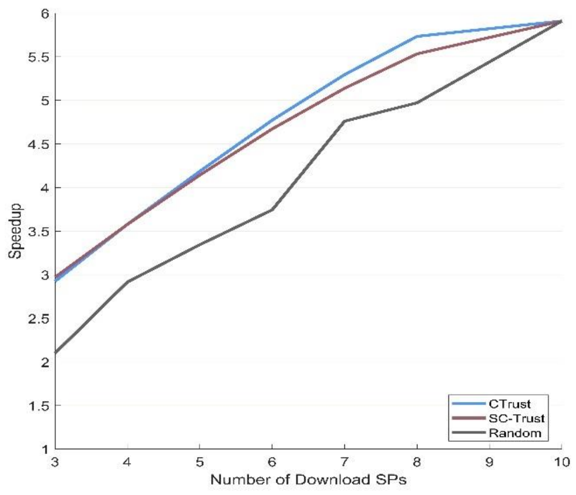


Fig. 3. Plot of Speed-up against varied sizes of the set Q of download SPs working in parallel. The figure shows the speed-up achieved using SC-TRUST in a service composition, compared to a random selection of SPs, or to CTRUST in a similar collaborative context.

TABLE 2
TWO-TAILED PAIRED SAMPLE T-TEST COMPARING THE SPEEDUP OBTAINED USING SC-TRUST, CTRUST AND RANDOM MODES FOR SELECTION OF SPS

	SC-TRUST vs. Random	SC-TRUST vs. CTRUST
Observations	100	100
P value at (α=0.05)	4.25E-14	0.053 (lowest value obtained)

transparent trust decomposition method discussed. The IRI is automatically assessed internally based on the criteria stated above. However, the SR may assign a weight to each of the parameters. It is assumed that the SR initially assigns an equal weight to each parameter, i.e. $w_1 = w_2 = w_3 = \frac{1}{3}$. The partial trust score of the composed service is computed according to the relevant transparent trust composition equation.

The purpose of these experiments is to show that the trust scores obtained by both transparent trust composition and decomposition are not significantly different from the scores that would have been elicited if the SR was in a direct collaboration with the SPs. Therefore, we show that by utilising a robust trust model as SC-TRUST, the advantages of service composition in IoT can be achieved whilst mitigating the realisation and effects of trust risks.

In our simulation, there are 5 members of the aggregation service class, and one is selected per CD session. There are 10 members of the download service class, and a range of 3 to 5 SPs are selected for each CD session. During a CD session, the aggregation SP may not be changed, but download SPs may be removed and/or added to the composition as required. The SPs are simulated using a uniform random distribution such that their behaviour should yield a partial trust score (that is, the ground truth value) in the range [0.5,1]. Thus, there is a uniform distribution of malicious or underperforming SPs as well as good, high-performance SPs. The simulation involves 100 such download sessions.

4.2 Evaluating Utility Gain in SC-TRUST

The purpose of utilising multiple download SPs is to increase the throughput, that is the speed at which the content is retrieved and delivered to the SR. Therefore, a faster download increases the utility gained by the SR. Fig. 3 illustrates the relative speedup (in comparison to the SR's average download speed) achieved for different cardinalities of the set Q of download SPs. We compare the speedups achieved using: (i) SC-TRUST in a service composition, (ii) CTRUST in a collaboration context, as in [25] and (iii) a random selection of SPs in a service composition. SC-TRUST achieves a similar speedup to CTRUST for each group size. SC-TRUST performs marginally better than CTRUST initially. This is due to the logistical overhead incurred by CTRUST, because in a direct collaboration, the service requester must select the SPs and compose the service directly.
As the session progresses, CTRUST slightly overtakes SC-TRUST in terms of speedup. This is probably due to the recurring overhead involved in communicating with the middleware layer. However, the difference between both models is statistically insignificant at a significance level, α=0.05, as shown in Table 2. Therefore, we can conclude that SC-TRUST provides as much utility gain for the SR as CTRUST, whilst providing the benefits of an automatic service composition. For example, using SC-TRUST, the SR does not need to request and assess SPs directly. Also, the SR does not incur energy and computation costs involved in running a service composition, as these are shifted to and borne by the middleware. Moreover, the SR does not need to keep a record of known SPs but can rely on the middleware to select appropriate SPs according to its service request.

Additionally, it is evident that the use of SC-TRUST increases the speedup quite significantly as compared to a random selection. We observe that SC-TRUST outperforms the random selection and that there is a consistent increase in the speedup even when the utilisation level of available download SPs is almost 80% (i.e. the number of selected SPs in set Q is 8, which is denoted as $n(Q) = 8$). Thus, SC-TRUST selects the most reliable SPs for service provision until there is no alternative. Above 6 used download SPs, the speedup of the random selection begins to sharply increase to match that of SC-TRUST and eventually a similar speedup is achieved at $n(Q) = 10$. This is because SC-TRUST accurately selects the most trustworthy SPs first. Therefore, the marginal increase in speedup reduces as $n(Q)$ approaches $n(S)$ (i.e. the total number of SPs). This is because the random mode is more likely to select trustworthy SPs, which were originally left out, as $n(Q)$ increases. When $n(Q) = n(S)$, there is no difference in speedup between both modes, as the trust model cannot perform any choice due to $\binom{n}{n} = 1$.

Therefore, given that the speedup achieved is comparable to the results that were obtained in [25] and [40], we conclude that the transparent trust composition in SC-TRUST yields an accurate trust score with a performance level equalling that of CTRUST. Also, utilising SC-TRUST in a service composition increases the utility gained by the SR with no significant overhead incurred, whilst providing the trust-based security required for the realization of all the potential benefits of an SOA-based IoT.

4.3 Evaluating Trust Model Accuracy, Resilience and Convergence

In evaluating the accuracy and convergence, it is necessary to establish the ground truth. The ground truth value is obtained by computing the trust score of an SP based on the perfect information of its behaviour and trust characteristics. This information is obtained from the record of the random trust behaviour assigned to each SP at the start of the simulation. In real-world applications, neither the SR nor the middleware would have perfect knowledge of the behaviour of any SP. Hence, there is the need for a trust model in the first instance. A trust model which performs accurately in simulations by closely matching known ground truth values in a reasonable time will perform well in real-world applications. Trust resilience is a measure of the ability of the model to adapt to changes in the behaviour of SPs and maintain a high level of performance (in terms of the utility derived by the SR) under such circumstances. This is important because the trust characteristics of SPs may be changed during a session due to malicious or non-malicious reasons. A resilient trust model must identify and adapt to these changes and converge to the new trust score quickly and accurately. By doing this, a high-level of utility is maintained (as current high-performing nodes are selected) and trust risks are minimized. The results obtained for SC-TRUST are presented in Fig. 4-7.

To generate sufficient interactions for these evaluations, the CD sessions were adapted for collaborative streaming. The only difference is that instead of waiting for the entire content to be downloaded before consumption by the SR, each downloaded block is streamed instantly to the SR. If the stream progresses successfully with no block missing, then the SCR increases. The inverse is also true. Similarly, if there is no buffering, then the CBA is increased. If the block arrives but not in time or sequence, then the SR automatically gives a negative feedback score on the CBA. SC-TRUST decomposes the SR's feedbacks. Each streaming

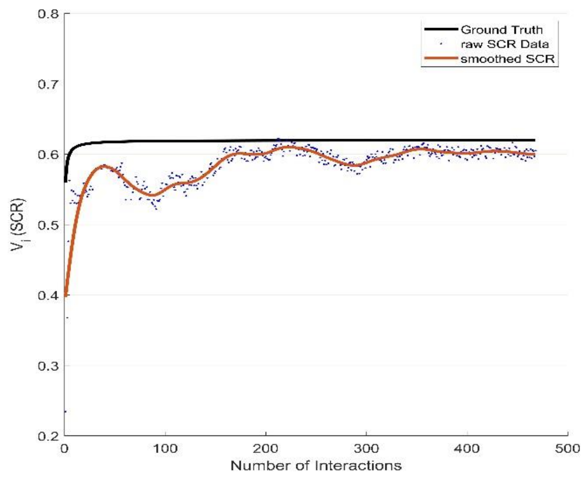


Fig. 4. Convergence of a Trust Parameter to the ground truth based on the decomposition of feedback from the SR.

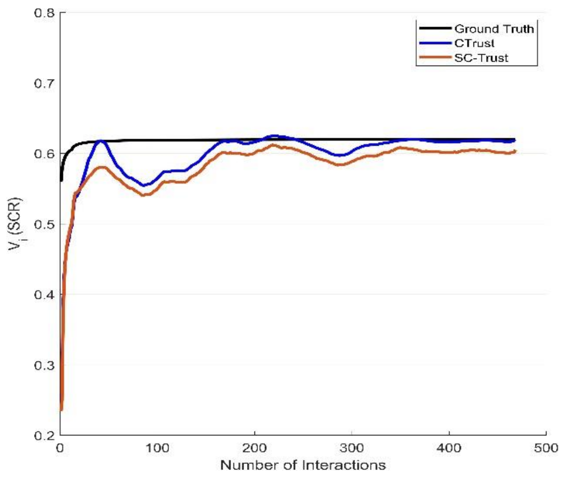


Fig. 5. Comparison of the trust accuracy and convergence properties of SC-TRUST to those of CTRUST.

session includes 400-800 blocks and an equivalent number of interactions.

Fig. 4 shows the results obtained on the SCR parameter after both trust composition and decomposition. Since the SCR parameter is unique to the aggregation service class, the decomposed trust scores only affect the selected aggregation SP. A smoothed plot of the decomposed SCR values in Fig. 4 shows that it converges from default state (no previous interaction) to the ground truth in less than 400 interactions. This is slightly higher than the ≈300 interactions required by CTRUST to converge to the ground truth. The reason is that the interactions in this simulation are shorter than those in the CTRUST simulation. Overall, the duration of the sessions in both models are similar. Therefore, SC-TRUST shows a high degree of convergence and accuracy. The fluctuations seen in the raw data are expected, as the utility of the SR and its perception of the composed service are sensitive to changes in the behaviour of the SP.

However, if the trust characteristics of the SP are consistent, then the decomposed trust value will always converge to the ground truth.

In Fig. 5, the SCR measured by SC-TRUST is compared to that of CTRUST. We see that while the value of CTRUST is closer to the ground truth, SC-TRUST follows the same pattern with a slight lag in the measurement of the trust score. However, this difference is statistically insignificant. It should be recalled that these values were decomposed transparently from the SR's feedbacks. Therefore, we can conclude that the trust decomposition method in the model not only converges to the ground truth but offers a level of accuracy on par with CTRUST, but in a service composition context. It is observed that the trend in both plots is similar. This is because SC-TRUST is built on top of CTRUST and utilizes some of the latter's methods.

In Fig. 6, the trust characteristic (ground truth) of the aggregation SP is modified to a higher value. It is observed that SC-TRUST converges quickly to the new ground truth within 400 interactions. Also, it should be observed that SC-TRUST is conservative in trust evaluation; that is, the composed or decomposed trust value is never higher than the ground truth. This is in accordance with the specifications in Sections 3.2 and 3.3; therefore, the SR always receives the estimated level of utility or higher, but never lower. This, in turn, increases the SR's trust in both the composing platform and composition services offered on the platform. As previously noted, SC-TRUST slightly lags behind CTRUST in the measured trust value, but this difference is insignificant and expected. CTRUST measures the trust scores from direct interactions; therefore, it is not suitable for transparent trust computations required in these service compositions.

Finally, in Fig. 7, we investigate the trust properties of SC-TRUST by measuring a shared parameter. The CBA is a parameter common to all download SPs. Therefore, it is difficult to accurately decompose the SR's feedback in a manner fair to each SP. However, due to the internal use of the IRI parameter to identify malicious and underperforming SPs, SC-TRUST performs reasonably well in converging to the ground truth on this parameter. It is observed that while CTRUST may produce trust scores higher than the ground truth (and therefore misleading the SR in a service composition), both the composed and decomposed trust scores in SC-TRUST are almost always lower than or equal to the ground truth. After 450 interactions, the trust characteristic of this SP is changed to a lower value. Again, SC-TRUST adapts and converges to the ground truth in a reasonable time, no more than required to establish and converge to the initial ground truth value. After 850 interactions, the value of SC-TRUST is a little higher than the ground truth. In this exceptional case, however, the increase over the ground truth is less than 1%. Therefore, SC-TRUST produces reliable and highly accurate trust scores,

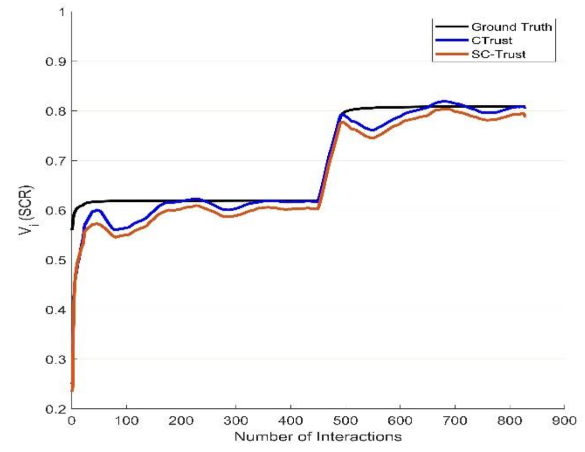


Fig. 6. Comparison of the trust resilience of SC-TRUST to that of CTRUST on a unique parameter.

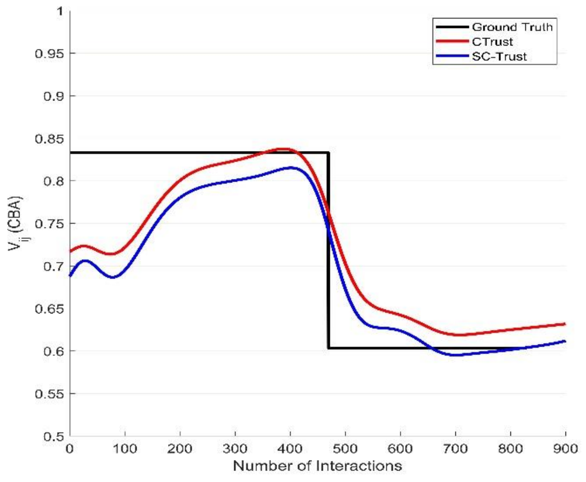


Fig. 7. Comparison of the trust resilience of SC-TRUST to that of CTRUST on a shared parameter.

within the margin of low and acceptable errors.

From the above analysis, it is evident that SC-TRUST meets the requirements 10 and 11 of an ideal trust model for IoT service compositions, as enumerated in Section 2.2. In addition, as it inherits the properties of CTRUST, SC-TRUST satisfies requirements 1-9 as well. In comparison to the few existing trust models for service composition, SC-TRUST produces a more accurate and reliable score. For example, the trust scores produced by the models in both [14] and [22] diverge significantly from the ground truth when the percentage of malicious SPs is greater than 30%. In contrast, SC-TRUST shows a high degree of resilience such that even when half of the SPs are malicious, it retains its high accuracy and convergence.

5 Related Work

Only a few trust models for service composition exist in the literature. This is due, in part, to the complexity of modelling trust in a decentralized manner [41]. Moreover, no previous works have provided methods for transparent trust composition and decomposition in IoT. Existing models measure trust mainly on social trust parameters, such as friendship, honesty, cooperativeness and kindness, based on direct observations and recommendations [16], [23], [42], [43]. The reasoning behind this modelling paradigm is the notion of the "social" nature of service-oriented IoT applications. Hence some social characteristics of human relationships are inaccurately applied and modelled in IoT. It is important to observe that the social nature of the IoT is much different from human social interactions. In IoT, the focus is much more on the ecosphere of humans, the environment, and smart things. Users interacting with services need not a relationship with the devices providing such services, or their owners. Depending on the type of service being provided, the SRs may not need to be co-located or share the same community of interest with SPs or their owners.

The primary goal of the SR in trusting a service or SP is the provision of a reliable, quality service that meets the specification of the SR without posing any risks. Thus, while a social interaction exists in the IoT, the relationships formed, and the parameters for measuring those relationships, are based on the required functional properties of the interaction context; that is the request and provision of a service. Moreover, it is not reasonable to directly apply the aspects of trust, as it applies in human relationships, to evaluate computational trust without some conceptual adaptation represented by mathematical models [44]. Even in human circles, the meaning of trust must be implied from the context and depends on the subjective assessment of the trustor [45], [46]. Even in human relationships, trust is predominantly functional, except for some cases of absolute trust or distrust. Therefore, in adapting trust to the SIoT, the notions of context and function must be preserved because they form the basis of trust [28].

In [47], a self-enforcing, privacy-preserving and decentralised TMS for SIoT is proposed. The protocol makes use of non-interactive zero-knowledge proofs in securing the network and for the reliable update of trust scores. However, the cryptographic protocol used is computationally intensive. While the computational overhead is manageable on systems with adequate computing resources, it is impractical for use on IoT devices typically with limited computing resources. Also, the model does not satisfy the trust resilience property because it does not adapt to changes in the node behaviour during a trust session. Moreover, the model neither assigns weights to different trust parameters to indicate the trustor's subjective preferences, nor includes trust parametrization. Therefore, it cannot be applied to different contexts.

In [48], a trustworthiness inference framework for SIoT is proposed based on familiarity and similarity trust, with contextual information based on time and location. The model does not consider the notion of functional trust or service contexts, which are important to guide service compositions. A trust-based service architecture for IoT is proposed in [49], with emphasis on improved efficiency of IoT services. However, the model operates in a centralized cloud architecture, thus limiting its applicability to service compositions in the IoT. The model contains no notion of trust parametrization or service contexts. In [50], a trust architecture for software-defined networks in IoT is proposed. The model uses reputation evaluation for trust establishment, with no notion of objective, functional parameters on which trust could be measured in a service-based application.

In [51], a reputation-based trust system is proposed for IoT applications. However, a rigorous analysis conducted in [25] shows that reputation-based models used in IoT are vulnerable to trust-related attacks such as bad-mouthing, ballot stuffing, and opportunistic service attacks, especially in a decentralized architecture. A similar argument may be applied to the trust-enhanced recommender system proposed in [52]. TMSs for dynamic trust management for SOA-IoT applications and service management in SIoT are proposed in [14], [22], [23]. The models produced are similar and measure the trust between nodes based on similarities in friendship, social contact, and community of interests. Functional constraints of the service contexts are not considered. Likewise, the models do not account for an SR's preferences and requirements, which should guide the service composition in an ideal trust model. Indeed, the models focus on achieving "subjective trust", which is a recommendation score primarily based on similarities and relationships between the owners of the IoT devices. This is contrary to the actual social nature of service applications in IoT, which should be based on the service context. However, the models do consider transparent trust composition for IoT based on the workflow, using a method derived from reliability/fault analysis. Thus, the trust score of a service composed of two microservices in sequence (series) is given to be the product of trust scores of the microservices. This assumption does not hold in many service applications, as discussed below.

It is possible to compose a service of high trust from microservices having low or average trust scores. Take the case of a pizzeria which provides a pizza ordering service, makes excellent pizza but has slow delivery times. As a result, it is given an average trust score by an SR that gives significant weighting to delivery times. Suppose this SR also has a trust relationship with a ride-hire service which has exceptionally low wait times but rude drivers. The SR is uneasy for the duration of the ride because of the driver's continuous boorish remarks; therefore, the SR gives a low rating to this service. Suppose that the two services are composed sequentially with the pizzeria producing the pizza and the fast ride-hire service delivering it to the SR. If the composition is done transparently to the SR, then the composed service would have a high trust rating, because the pizza is delivered fast and the SR does not hear the driver's remarks. This is contrary to the results that the reliability formula mentioned earlier would have produced: low trust X average trust = lower trust. Also, reliability analysis states that the reliability of a group of components is increased when the components operate in parallel, for redundancy. However, this does not necessarily apply to trust composition in IoT service composition. For example, composing two highly trusted services, one with an average delay and the other with an average energy efficiency, in a parallel workflow may increase both the response time and energy usage; consequently, the SR may assign a lower trust score to the resulting composed service.

From these examples, it is evident that to accurately compose trust in a service composition, there must be a mechanism to assign weights to each parameter, such that parameters irrelevant to the service composition (such as the driver's behaviour in the previous example) would have little or no effect on the overall trust score; this is lacking in most of the models reviewed. Moreover, most of these models do not consider the temporal nature of trust and its effect on the decay of previously accumulated trust. It is necessary to track the behaviour of SPs in relation to the functional trust parameters and to detect and respond to changes over time. In contrast, SC-TRUST has been designed and evaluated to address these research gaps and satisfy the requirements for an ideal trust model for service composition in the IoT, thus justifying its novel contribution to this research area.

6 Conclusion

In this paper, we discussed the suitable trust properties required by a trust model and TMS to guide service compositions in the IoT. We studied the classifications of compositions with respect to service classes and workflows. We also analysed the effects of the workflows on transparent trust composition, aggregation and decomposition. Based on these analyses, SC-TRUST was designed as a suitable model for service compositions in the SOA-based IoT context. The simulation evaluations show that the use of SC-TRUST in a service composition increased the utility gained. Also, SC-TRUST showed a robust performance in trust accuracy, convergence, and resilience. Therefore, the model minimizes the impact of trust-related attacks, including ballot stuffing, bad-mouthing, and opportunistic service attacks. SC-TRUST was modelled with the platform characteristics of IoT in consideration, so its trust evaluations and algorithms require minimal computational resources. In addition, the flexibility of the design ensures that the model can be easily applied to any service composition context. Thus, SC-TRUST addresses critical gaps by providing a dynamic, systematic, and holistic approach to trust modelling in SOA-based IoT, especially for trust-based service compositions. In addition, the elegant solutions for transparent trust composition and decomposition are novel contributions.

In this work, we have not considered the effect of constraints, such as price and energy, on the service composition. It is observed that, by increasing performance and reducing trust attacks, SC-TRUST reduces overall energy usage. This requires further evaluations beyond the scope of this paper. Also, the prices which SPs charge for their services may affect their selection depending on the limit of the SR's budget. However, the price is not a trust characteristic, as it is not a functional parameter of the composition context, but rather an external constraint on the process. It may be argued that more trustworthy compositions would generally cost more because the price is an incentive for an SP to produce better services. It would be beneficial to study the effects of external constraints, such as price and an SR's budget, on the service composition process and the trust scores of SPs. We will consider these issues as part of our future work.

References

[1] S. Li, L. Da Xu, and S. Zhao, "5G Internet of Things: A survey," *Journal of Industrial Information Integration*, vol. 10, pp. 1–9, Jun. 2018.

[2] C. L. Hsu and J. C. C. Lin, "An empirical examination of consumer adoption of Internet of Things services: Network externalities and concern for information privacy perspectives," *Computers in Human Behavior*, vol. 62, pp. 516–527, Sep. 2016.

[3] P. Rawat, K. D. Singh, and J. M. Bonnin, "Cognitive radio for M2M and Internet of Things: A survey," *Computer Communications*, vol. 94, pp. 1–29, Nov. 2016.

[4] E. Borgia, "The Internet of Things vision: Key features, applications and open issues," *Computer Communications*, vol. 54, no. 7, pp. 1–31, Dec. 2014.

[5] D. Miorandi, S. Sicari, F. De Pellegrini, and I. Chlamtac, "Internet of things: Vision, applications and research challenges," *Ad Hoc Networks*, vol. 10, no. 7, pp. 1497–1516, Sep. 2012.

[6] J. Ding, M. Nemati, C. Ranaweera, and J. Choi, "IoT Connectivity Technologies and Applications: A Survey," *IEEE Access*, vol. 8, pp. 67646–67673, 2020.

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

[8] F. Li, Y. Lu, X. Wang, Y. Bi, T. Pan, Y. Zhang, W. Li, and Y. Wang, "Software-Defined Networking Assisted Content Delivery at Edge of Mobile Social Networks," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 8122 - 8132, Sep. 2020.

[9] M. Hamzei and N. Jafari Navimipour, "Toward Efficient Service Composition Techniques in the Internet of Things," *IEEE Internet of Things Journal*, vol. 5, no. 5, pp. 3774–3787, Oct. 2018.

[10] M. Sun, Z. Shi, S. Chen, Z. Zhou, and Y. Duan, "Energy-Efficient Composition of Configurable Internet of Things Services," *IEEE Access*, vol. 5, pp. 25609–25622, 2017.

[11] Z.-Z. Liu, D.-H. Chu, Z.-P. Jia, J.-Q. Shen, and L. Wang, "Two-stage approach for reliable dynamic Web service composition," *Knowledge-Based Systems*, vol. 97, pp. 123–143,

Apr. 2016.

[12] J. Crowcroft, *Open Distributed Systems*. USA: Artech House, Inc., 1996.

[13] E. Yahyapour *et al.*, *Towards a Service-Based Internet*, vol. 6481. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010.

[14] 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, pp. 684–696, Nov. 2016.

[15] A. H. H. Ngu, M. Gutierrez, V. Metsis, S. Nepal, and M. Z. Sheng, "IoT Middleware: A Survey on Issues and Enabling technologies," *IEEE Internet of Things Journal*, pp. 1–1, 2016.

[16] M. S. Roopa, S. Pattar, R. Buyya, K. R. Venugopal, S. S. Iyengar, and L. M. Patnaik, "Social Internet of Things (SIoT): Foundations, thrust areas, systematic review and future directions," *Computer Communications*, vol. 139, pp. 32–57, May 2019.

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

[18] M. A. Razzaque, M. Milojevic-Jevric, A. Palade, and S. Cla, "Middleware for internet of things: A survey," *IEEE Internet of Things Journal*, vol. 3, no. 1, pp. 70–95, Feb. 2016.

[19] 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, pp. 3594–3608, Nov. 2012.

[20] L. Atzori, A. Iera, and G. Morabito, "SIoT: Giving a Social Structure to the Internet of Things," *IEEE Communications Letters*, vol. 15, no. 11, pp. 1193–1195, Nov. 2011.

[21] Y. Ben 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 & Security*, vol. 39, no. PART B, pp. 351–365, Nov. 2013.

[22] I.-R. Chen, J. Guo, D.-C. Wang, J. J. P. Tsai, H. Al-Hamadi, and I. You, "Trust-Based Service Management for Mobile Cloud IoT Systems," *IEEE Transactions on Network and Service Management*, vol. 16, no. 1, pp. 246–263, Mar. 2019.

[23] 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, pp. 482–495, May 2016.

[24] Z. Yan, P. Zhang, and A. V. Vasilakos, "A survey on trust management for Internet of Things," *Journal of Network and Computer Applications*, vol. 42, pp. 120–134, Jun. 2014.

[25] A. A. Adewuyi, H. Cheng, Q. Shi, J. Cao, A. MacDermott, and X. Wang, "CTRUST: A dynamic trust model for collaborative applications in the internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 5432–5445, Feb. 2019.

[26] J. Guo, I.-R. Chen, and J. J. P. Tsai, "A survey of trust computation models for service management in internet of things systems," *Computer Communications*, vol. 97, pp. 1–14, Jan. 2017.

[27] D. Gessner, A. Olivereau, A. S. Segura, and A. Serbanati, "Trustworthy Infrastructure Services for a Secure and Privacy-Respecting Internet of Things," in *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*, Jun. 2012, pp. 998–1003.

[28] Z. Lin and L. Dong, "Clarifying Trust in Social Internet of Things," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 2, pp. 234–248, Apr. 2017.

[29] J. Guo and I.-R. Chen, "A Classification of Trust Computation Models for Service-Oriented Internet of Things Systems," in *2015 IEEE International Conference on Services Computing*, Jun. 2015, pp. 324–331.

[30] W. Viriyasitavat, L. Da Xu, Z. Bi, and A. Sapsomboon, "Blockchain-based business process management (BPM) framework for service composition in industry 4.0," *Journal of Intelligent Manufacturing*, May 2018.

[31] P. Wang *et al.*, "Smart Contract-Based Negotiation for Adaptive QoS-Aware Service Composition," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 6, pp. 1403–1420, Jun. 2019.

[32] C. Yu, L. Zhang, W. Zhao, and S. Zhang, "A blockchain-based service composition architecture in cloud manufacturing," *International Journal of Computer Integrated Manufacturing*, pp. 1–15, Feb. 2019.

[33] H. A. Simon, "Rational choice and the structure of the environment.," *Psychological Review*, vol. 63, no. 2, pp. 129–138, Mar. 1956.

[34] M. M. Rahman, "Assessing small group assignments," in *7th Brunei International Conference on Engineering and Technology 2018 (BICET 2018)*, 2018, pp. 1-4.

[35] L. Johnston and L. Miles, "Assessing contributions to group assignments," *Assessment and Evaluation in Higher Education*, vol. 29, no. 6, pp. 751–768, Dec. 2004.

[36] J. Goldfinch and R. Raeside, "DEVELOPMENT OF A PEER ASSESSMENT TECHNIQUE FOR OBTAINING INDIVIDUAL MARKS ON A GROUP PROJECT," *Assessment & Evaluation in Higher Education*, vol. 15, no. 3, pp. 210–231, Sep. 1990.

[37] G. Ananthanarayanan, V. N. Padmanabhan, L. Ravindranath, and C. a. Thekkath, "COMBINE," in *Proceedings of the 5th international conference on Mobile systems, applications and services - MobiSys '07*, 2007, pp. 286-298.

[38] A. Sharma, V. Navda, R. Ramjee, V. N. Padmanabhan, and E. M. Belding, "Cool-Tether : Energy Efficient On-the-fly WiFi Hot-spots," *ACM International Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, pp. 109–120, 2009.

[39] P. Jassal, K. Yadav, A. Kumar, V. Naik, V. Narwal, and A. Singh, "Unity: Collaborative downloading content using co-located socially connected peers," in *2013 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, Mar. 2013, pp. 66–71.

[40] G. Iosifidis, L. Gao, J. Huang, and L. Tassiulas, "Efficient and Fair Collaborative Mobile Internet Access," *IEEE/ACM Transactions on Networking*, vol. 25, no. 3, pp. 1386–1400, Jun. 2017.

[41] U. E. Tahta, S. Sen, and A. B. Can, "GenTrust: A genetic trust management model for peer-to-peer systems," *Applied Soft Computing*, vol. 34, pp. 693–704, Sep. 2015.

[42] J.-H. Cho, K. Chan, and S. Adali, "A Survey on Trust Modeling," *ACM Computing Surveys*, vol. 48, no. 2, pp. 1–40, Nov. 2015.

[43] F. Bao and I.-R. Chen, "Dynamic trust management for internet of things applications," in *Proceedings of the 2012 international workshop on Self-aware internet of things - Self-IoT '12*, 2012, pp. 1-6.

[44] W. Leister and T. Schulz, "Ideas for a Trust Indicator in the Internet of Things," in *SMART 2012—The First International Conference on Smart Systems, Devices and Technologies*, 2012, pp. 31–34.

[45] D. H. Mcknight and N. L. Chervany, "The Meanings of Trust," 1996.

[46] Y. D. Wang and H. H. Emurian, "An overview of online trust: Concepts, elements, and implications," *Computers in Human Behavior*, vol. 21, no. 1, pp. 105–125, Jan. 2005.

[47] M. A. Azad, S. Bag, F. Hao, and A. Shalaginov, "Decentralized Self-Enforcing Trust Management System for Social Internet of Things," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 2690–2703, Apr. 2020.

[48] H. Xia, F. Xiao, S. Zhang, C. Hu, and X. Cheng, "Trustworthiness Inference Framework in the Social Internet of Things: A Context-Aware Approach," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, Apr. 2019, pp. 838–846.

[49] T. Wang, G. Zhang, A. Liu, M. Z. A. Bhuiyan, and Q. Jin, "A Secure IoT Service Architecture With an Efficient Balance Dynamics Based on Cloud and Edge Computing," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4831–4843, Jun. 2019.

[50] J. Chen, Z. Tian, X. Cui, L. Yin, and X. Wang, "Trust architecture and reputation evaluation for internet of things," *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, no. 8, pp. 3099–3107, Aug. 2019.

[51] S. Asiri and A. Miri, "An IoT trust and reputation model based on recommender systems," in *2016 14th Annual Conference on Privacy, Security and Trust (PST)*, Dec. 2016, pp. 561–568.

[52] N. S. Nizamkari, "A graph-based trust-enhanced recommender system for service selection in IOT," in *2017 International Conference on Inventive Systems and Control (ICISC)*, Jan. 2017, pp. 1–5.

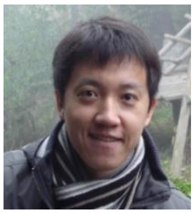
Anuoluwapo A. Adewuyi (S'18) received the M.Sc. degree in Advanced Computer Science with IT Management from the University of Manchester in 2015. He is currently a Ph.D. student and researcher within the PROTECT Research Centre of Liverpool John Moores University (LJMU). His main research interests are cryptography, trust management, IoT security and secure service composition. He is a student member of both the IEEE and IEEE Computer Society.

Hui Cheng (M'18) received the B.Sc. and M.Sc. degrees in Computer Science from Northeastern University, China in 2001 and 2004, and the Ph.D. degree in Computer Science from Hong Kong Polytechnic University in 2007. He was a Senior Lecturer at Liverpool John Moores University from October 2013 until June 2018. He is currently a Senior Lecturer at the University of Hertfordshire. His research interests include trust management, artificial intelligence, dynamic optimization, optical networks, and mobile networks. He is a member of the IEEE.

Qi Shi received the Ph.D. degree in Computing from the Dalian University of Technology, China. He was a Research Associate on an EU Research Project with the University of York, U.K. Afterwards, he joined Liverpool John Moores University (LJMU) as a Lecturer and then a Reader. He is currently a Professor in Computer Security and the Director of the PROTECT Research Centre, LJMU. He has many years of research experience in many security related areas. He has authored or co-authored over 250 papers in international conference proceedings and journals and served as a member for a number of journal editorial boards and conference committees.

Jiannong Cao (M'93–SM'05–F'15) received the B.Sc. degree in computer science from Nanjing University, Nanjing, China, in 1982, and the M.Sc. and Ph.D. degrees in Computer Science from Washington State University, Pullman, WA, USA, in 1986 and 1990. He is currently a Chair Professor of distributed and mobile computing with the Department of Computing and the Director of the University Research Facility in Big Data Analytics, Hong Kong Polytechnic University, Hong Kong. His current research interests include parallel and distributed computing, wireless networks and mobile computing, big data and cloud computing, pervasive computing, and fault tolerant computing. Prof Cao has served as the Chair and a member of Organizing and Technical Committees of many international conferences, including PERCOM, INFOCOM, SMARTCOMP, ICMU, ICPP, MASS, ICPADS, IWQoS, ICDCS, DSN, SRDS, ICNP, and RTSS. He has also served as an Associate Editor and an Editorial Board member of many international journals, including IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Cloud Computing, IEEE Transactions on Computers, IEEE Network, ACM Transactions on Sensor Networks, Elsevier's Pervasive and Mobile Computing Journal, Springer's Peer-to-Peer Networking and Applications, and Wiley's Wireless Communications and Mobile Computing.

Xingwei Wang received the B.S., M.S., and Ph.D. degrees in computer science from Northeastern University, Shenyang, China, in 1989, 1992, and 1998, respectively. He is currently a Professor with the School of Computer Science and Engineering, Northeastern University, China. His current research interests include future Internet, cloud computing, mobile computing, and mobile social networks.



Bo Zhou is a Reader in Network Security in the School of Computer Science and Mathematics at Liverpool John Moores University in the UK. He studied Telecommunications for BSc at the Northeastern University of China and MSc at Queen Mary, University of London in the UK, before completing PhD in Network Security at Liverpool John Moores University. His research interests include Network Security, Security Visualisation, Hardware Security, Intelligent Transportation Systems, and Machine Learning. Dr Zhou has published over 50 research papers in these areas.