CATrust: Context-Aware Trust Management for Service-Oriented Ad Hoc Networks

Yating Wang, Ing-Ray Chen, Jin-Hee Cho, Ananthram Swami, Yen-Cheng Lu, Chang-Tien Lu, Jeffrey J.P. Tsai

***Abstract* — We propose a context-aware trust management model called CATrust for service-oriented ad hoc networks such as peer-to-peer and Internet of Things networks wherein a node can be a service requester or a service provider. The novelty of our design lies in the use of logistic regression to dynamically estimate trustworthiness of a service provider based on its service behavior patterns in response to context environment changes. We develop a recommendation filtering mechanism to effectively screen out dishonest recommendations even in extremely hostile environments in which the majority recommenders are dishonest. We demonstrate desirable convergence, accuracy, and resiliency properties of CATrust. We also demonstrate that CATrust outperforms contemporary peer-to-peer and Internet of Things trust models in terms of service trust prediction accuracy against collusion recommendation attacks.**

***Index Terms* — Trust, service-oriented ad hoc networks, logistic regression, recommendation attacks, statistical analysis.**

I. INTRODUCTION

With the proliferation of fairly powerful mobile devices and ubiquitous wireless technology, traditional ad hoc networks now migrate into a new era wherein a node can provide and receive service from other nodes it encounters and interacts with. This paper proposes a new trust model for service-oriented ad hoc networks (SOANETs) consisting of service providers (SPs) and service requesters (SRs). Peer-to-peer (P2P) networks [14] [22] [33], Internet of things (IoT) systems [3] [20], and mobile ad hoc networks (MANETs) [4] [5] [6], especially mobile cloudlets [27] are realizations of SOANETs populated with SPs and SRs. One can view a SOANET as an instance of Internet of Things (IoT) systems with a wide range of mobile applications including smart-city, smart tourism, smart car, smart environmental monitoring, and healthcare [2].

We exemplify a prototypical SOANET application by a dynamic service composition and binding environmental monitoring application in a smart city setting as illustrated in Figure 1 with multi-objective optimization (MOO) for three service quality criteria: quality of information, service delay, and service cost [26]. In this smart city application there would be many environmental sensors embedded in the city, including smart phones carried by human beings conscious of environmental health, public transportation vehicles (buses, taxis, trains and park vehicles), city light lamps, utility poles, buildings, and infrastructures providing environmental monitoring service of air pollutants such as CO, NO_2, SO_2, and O_3 in the same proximal location. While service delay and service cost are easily measureable physical quantities, quality of information is specific to the application domain. In environmental monitoring service, quality of information is measured by the extent to which the output contributes to the ground truth data [23]. For SPs providing sensing service, quality of information is measured by the extent to which the sensing data contribute to the ground truth picture [17]. In this SOANET application, a requested service issued by a SR is first decomposed into abstract services each of which is then bound to a SP selected by the SR, with the goal of maximizing the quality of information while minimizing service cost and service delay. Essentially, service composition is formulated as a workflow problem based on the SR's location and the availability of SPs around the SR's location, while service binding is formulated as a node-to-service assignment problem. This application can base on IEEE 802.11 technology, allowing node-to-node communication for up to 200m covering the monitoring area to be surveyed when a SR moves to the environment. As an environmental monitoring service or sensing service is inherently location based, it is expected that a service request issued will be considered only by SPs providing the required services near the SR's location. However not all of these SPs will be trustworthy, so trust management to cope with misbehaving SPs and recommendation filtering is required to screen out untrustworthy recommendations.

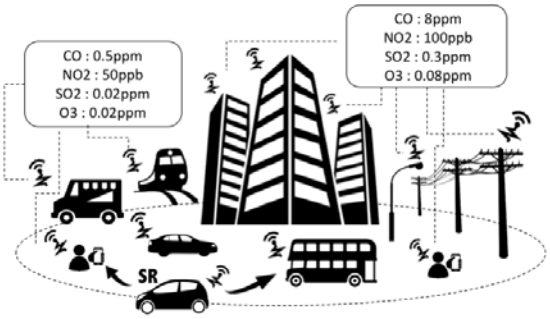


Figure 1: A Prototypical Environmental Monitoring SOANET Application.

In this work, we take a context-aware approach for trust management, treating channel conditions, node status, service payoff, and social disposition as "context" information. We use

†Yating Wang, Ing-Ray Chen, Yen-Cheng Lu, and Chang-Tien Lu are with the Department of Computer Science, Virginia Tech, Falls Church, VA 22043. E-mail: {yatingw, irchen, kevinlu, ctlu}@vt.edu.
**Jin-Hee Cho and Ananthram Swami are with Computational and Information Sciences Directorate, U.S. Army Research Laboratory, Powder Mill Rd. Adelphi, MD 20783. E-mail: {jinhee.cho, ananthram.swami} @us.army.mil.*
‡Jeffrey J.P. Tsai is with the Department of Bioinformatics and Biomedical Engineering, Asia University, Taichung, Taiwan 41354. Email: jjptsai@gmail.com.

the word "context" interchangeably with "context environment" to refer to these environmental and operational conditions a SP is in. We view trust as the probability that a SP will provide a satisfactory service in a particular context environment, as expected by a SR. We note that the probability of getting a satisfactory service from a SP can be extended to mean that the SR is willing to depend on the SP in a given situation with a feeling of relative security, even though negative consequences are possible [13].

The novelty of our work lies in the use of a robust inference model based on logistic regression to robustly yet accurately predict how a SP's service quality will be in response to context changes. This allows us to reason about a node's service behavior pattern, given the operational and environmental context information as input. We name our context-aware trust management protocol "CATrust." This paper substantially extends our conference paper [30] with a thorough analysis of the convergence, accuracy and resiliency properties of our protocol design, a comparative performance analysis with two contemporary P2P/IoT trust protocols, and a novel recommendation filtering technique to address the case where malicious recommenders form a majority.

This paper makes the following contributions:

- To the best of our knowledge, we are the first to propose a context-aware trust management for SOANETs. Similar to existing P2P or IoT trust protocols, we also predict a SP's trustworthiness based on the SP's behavior. The difference lies in our ability to associate a SP's service behavior with context information. The end result is that there is one service trust value for each context environment, instead of one service trust value for all context environments (as in existing protocols). This context-aware design greatly improves service trust prediction accuracy.
- We propose a novel recommendation filtering mechanism to effectively screen out dishonest recommendations even when most of the recommendations are dishonest.
- We demonstrate that CATrust is highly resilient toward collusion recommendation attacks. CATrust significantly outperforms contemporary P2P and IoT trust protocols in terms of accuracy and resiliency against recommendation attacks.

This paper is organized as follows: In Section II, we survey trust models and trust-based defenses against malicious attacks, especially collusion recommendation attacks. We contrast and compare existing approaches with our approach. In Section III, we discuss the system model including service-oriented ad hoc networks, node service behavior model, threat model, and performance metrics. In Section IV, we discuss our trust management protocol design for CATrust in detail. In Section V, we analyze the convergence, accuracy and resiliency properties of CATrust. In Section VI, we perform a comparative analysis of CATrust against two contemporary P2P and IoT trust models, namely, Beta Reputation [12] and Adaptive Trust Management [3]. In Section VII, we discuss the computational feasibility and applicability issues. Lastly in Section VIII we conclude the paper and outline future research directions.

II. Related Work

We focus our attention on existing trust management models for P2P networks, IoT networks, and MANETs since these systems are realizations of SOANETs.

Trust protocols based on Bayesian probability are popular because of sound statistical basis. Yu et al. [34] applied Bayesian inference to measure the reputation of a MANET node assuming that a node's behavior in each observation period is identically and independently distributed, and follows the binomial distribution. Therefore, a node's reputation is determined by the numbers of positive and negative samples observed. Chen et al. [3] considered a Bayesian framework as the underlying model for direct trust assessment from user satisfaction experiences in IoT systems. A shortcoming of the above models based on Bayesian probability is that the trust value is not associated with context since it is just based on user satisfaction experiences.

Belief theory or subjective logic trust models [11] [13] introduce uncertainty into trust calculation. Balakrishnan et al. [1] developed a subjective logic based model for a MANET node to evaluate its opinion towards another node using evidence-to-opinion mapping operators. Twigg [24] developed a subjective logic based trust model for selecting trustworthy nodes for secure routing in P2P networks.

Fuzzy logic based trust models are also well studied in the literature [29] [32]. Instead of using a binary set, a membership function is defined indicating the degree to which a node is considered trustworthy. Wang and Huang [29] computed the fuzzy trust values of candidate paths based on node reputation, bandwidth, and hop count for route selection. Xia et al. [32] applied fuzzy inference rules for trust prediction, considering past and current service experiences for predicting the service capability of a transmitter node. One drawback of fuzzy logic-based trust prediction is that it requires domain experts to do parameter tuning and set the fuzzy rules incorporating the knowledge of the causal relationship between the input and output parameters.

Relative to the works cited above based on Bayesian probability, belief theory, and fuzzy logic, we take an entirely different approach. We develop a regression-based trust model utilizing *logistic regression* to estimate the trustworthiness of a SP in response to context environment changes.

There is little work in the literature on applying regression for trust computation. To date, it has been used only by [15] [25] for finding the best weights to assign to observations or trust components. Specifically, Li et al. [15] proposed an auto-regression-based technique to learn the weights from historical observations to predict future outcomes. Venkataraman et al. [25] developed a regression-based trust model to learn the optimal weights of multiple trust metrics, where each trust metric is assessed separately using Bayesian inference. Unlike [15] [25], we do not use regression to learn weights to be applied to observations or trust properties. Instead, we apply logistic regression analysis to learn the service behavior patterns of a SP in response to context environment changes and consequently predict the SP's dynamic trust in terms of its service quality trust. For this

reason, we do not consider [15] [25] as baseline cases for performance comparison.

We consider Beta Reputation [12] and Adaptive Trust Management [3] (both measuring service quality trust) as baseline cases for performance comparison in this paper. Beta Reputation [12] is the most popular trust protocol for P2P systems to-date. It applies the concept of *belief discounting* for recommendation filtering. Adaptive Trust Management [3] is a very recent IoT/P2P trust protocol proven to outperform other contemporary IoT/P2P protocols. It applies the concept of *collaborative filtering* for recommendation filtering and the concept of *adaptive filtering* for dynamically adjusting the weights of direct trust obtained through self-observations and indirect trust obtained through recommendations to maximize protocol performance.

A considerable amount of work has been done in the area of trust-based defenses against attacks in P2P networks, IoT networks, and MANETs [3] [4] [6] [5] [14] [22] [28] [31] [33]. Chen et al. [4] proposed the concept of trust bias minimization by dynamically adjusting the weights associated with direct trust (derived from direct evidence such as local observations) and indirect trust (derived from indirect evidence such as recommendations) so as to minimize trust bias. Cho et al. [5] [6] proposed the use of trust thresholds to filter out untrusted recommendations. EigenTrust [14], PeerTrust [33], and ServiceTrust [22] considered various methods to aggregate recommender feedbacks weighted by the recommender's trustworthiness based on factors that affect a recommender's trustworthiness, including transaction context, community context, and credibility in terms of the trust and personalized similarity between the trustor and the recommender, etc. to filter out distrusted feedbacks. A common challenge with the above approaches is that dynamically tuning trust parameters may perform poorly when malicious recommenders form a majority, especially if a node does not have enough service experiences with other nodes and must rely on recommendations for decision making. CATrust leverages a robust statistical kernel to tolerate false recommendations to effectively achieve resiliency against recommendation attacks. Furthermore, CATrust filters out false recommendations based on a novel threshold-based filtering mechanism such that if the difference between the predicted service quality trust and the recommended service quality trust under the same context environment is above a threshold, then the recommendation is filtered. We demonstrate that CATrust significantly outperforms contemporary P2P/IoT trust models including Beta Reputation [12] and Adaptive Trust Management [3], especially when observations for recommenders are limited.

III. System Model

A. Service-Oriented Ad Hoc Networks

We consider location-based service requests. That is, a SR requests a service, and SPs in the same location (within radio range) respond to the request. As illustrated in Figure 1 for the prototypical service composition and binding environmental monitoring SOANET application, there are multiple SPs competing for a requested service in the same proximal location. The SR selects the SP with the highest trust value, given the current context environment as input. The reason we develop CATrust specifically for SOANETs but not for other types of service ecosystems is that a SP's service quality criteria (such as quality of information, service delay, and service cost considered in [26]) are inherently associated with rapid context environment changes in SOANETs.

B. Node Service Behavior Model

We consider the notion of context-sensitive service behavior, i.e., a SP's service behavior may change dynamically, as the SOANET operational and environmental conditions change dynamically due to node mobility, channel contention, node status, and social disposition toward other nodes in the system. Trust therefore is dynamic because SPs are heterogeneous in terms of capability and attitude, and adapt to context changes. We call an operational or environmental condition that may affect a SP's service behavior a *context* variable. While CATrust can handle any context variable, we consider profit-awareness, capability-limitation, and energy-sensitivity as three distinct context variables for the following reasons: (a) a profit-aware SP is more likely to provide quality service when the SR offers a higher price [9] [35]; (b) a SP is likely to provide inferior service when it is limited in resources and capability [18]; and (c) a SP is more likely to provide inferior service when the cost of servicing the task is high [17]. For example, in a congested environment the probability of wireless channel contention and signal interference will be high, so it will cost more for a SP to execute a service because the SP needs to consume more energy in listening to the channel and repeating packet transmission.

The service quality provided by a SP is determined by its service behavior in response to changes in the context environment. We call the resulting service quality "ground truth" service quality as it is intrinsically related to a SP's service behavior. A malicious node (defined below), however, may provide inferior service for self-interest even if it is capable of providing satisfactory service.

C. Threat Model

As every node in a SOANET can be a SP or a SR itself, it wants to be selected to provide service for profit when it is a SP and wants to find the best SPs for best service available when it is a SR. Therefore, by a malicious node we do not refer to a node that is compromised by enemies and attempts to destroy or disrupt the operation of the system. Rather, we refer to a node that acts for its own benefits and can collude with other malicious nodes to monopolize service even if the service it provides is inferior. For convenience, we will use the word "bad" interchangeably with "malicious," and the word "good" interchangeably with "non-malicious."

We assume that a malicious node as a recommender will perform the following collusion recommendation attacks:

- *Bad-mouthing attacks*: a malicious node can ruin the service trust of a non-malicious node by providing bad recommendations. It can collude with other malicious nodes to ruin the service trust of a good node.

TABLE I: Behavior of a Malicious Recommender.

Trustor	Trustee	Bad-Mouthing Attack	Ballot-Stuffing Attack
malicious	*malicious*		
malicious	*non-malicious*		
non-malicious	*malicious*		√
non-malicious	*non-malicious*	√	

TABLE II: Behavior of a Malicious Service Provider.

Service Requester	Conflicting Behavior Attack	Random Attack
malicious	√	
non-malicious	√	√

- *Ballot-stuffing attacks*: a malicious node can boost the service trust of another malicious node by providing good recommendations so as to increase the chance of that malicious node being selected as a SP. Malicious nodes can collude with each other to boost their service trust values.

Table I summarizes the attack behavior of a malicious node as a recommender, depending on the nature of the trustor and trustee nodes. If the trustor SR is non-malicious and the trustee SP is malicious, a malicious recommender will perform ballot-stuffing attacks. If the trustor SR is non-malicious and the trustee SP is also non-malicious, a malicious recommender will perform bad-mouthing attacks.

We assume that a malicious node as a SP will perform the following service attacks:

- *Conflicting behavior attacks*: a malicious node can selectively provide satisfactory service within its service capability for some SRs while providing unsatisfactory service for others. We assume that malicious nodes know each other, so with conflicting behavior attacks a malicious node will provide satisfactory service to other malicious nodes, but unsatisfactory service to non-malicious nodes.
- *Random attacks*: While performing conflicting behavior attacks, a malicious node can perform *random attacks*, i.e., providing unsatisfactory service to non-malicious nodes only randomly, so as to avoid being labeled as a low service trust node and risk itself not being selected as a SP by non-malicious SRs in the future.

Table II summarizes the attack behavior of a malicious SP, depending on the nature of the SR.

We will first analyze the convergence, accuracy and resiliency properties of CATrust against collusion recommendation attacks in Section V. Then we will address how CATrust can deal with conflicting behavior and random attacks in Section VII.

D. Performance Metrics

The performance metrics for comparative performance analysis are false negative probability (P_{fn}) and false positive probability (P_{fp}) defined as follows:

- *False negative probability* (P_{fn}) is the missing bad service probability. That is, it is the conditional probability that SR *i* will misidentify SP *j* as being able to provide satisfactory service, given that SP *j* actually provides unsatisfactory service. The term false negative is consistent with that used in intrusion detection [22], although the target subject in intrusion detection is the node itself (missing a malicious node) rather than the service provided (missing a bad service provided by a node). P_{fn} can be calculated by:

$$P_{fn} = \frac{\widehat{N}^c}{N^c} \quad (1)$$

$\widehat{N}^c$ is the number of cases SR *i* believes SP *j* will provide satisfactory service while SP *j* actually provides unsatisfactory service, and N^c is the number of cases SP *j* will provide unsatisfactory services if selected. The lower the false negative rate the better the performance.

- *False positive probability* (P_{fp}) is the missing good service probability. That is, it is the conditional probability that SR *i* will misidentify SP *j* as being unable to provide satisfactory service, given that SP *j* actually provides satisfactory service. Again the term false positive is consistent with that used in intrusion detection [19], although the target subject in intrusion detection is the node itself (misidentifying a non-malicious node) rather than the service provided (missing a good service provided by a node). P_{fp} can be calculated by:

$$P_{fp} = \frac{\widehat{N}^m}{N^m} \quad (2)$$

$\widehat{N}^m$ is the number of cases SR *i* believes SP *j* will not provide satisfactory service while SP *j* actually provides satisfactory service and N^m is the number of cases SP *j* will provide satisfactory service if selected. The lower the false positive rate the better the performance.

IV. CATRUST DESIGN

A. Problem Definition and Design Objective

The central idea of CATrust is that instead of directly predicting service quality, we predict the probability of delivering satisfactory service, i.e., service trust. Our rationale is that, given several SP candidates, a SR is able to select the most reliable SP with least risk. Such risk information might not be straightforward if we predict service quality only. However, with service trust information, we can easily infer the risk associated with a decision.

For ease of discussion, we list the symbols used and their meanings in Table III. A symbol may be associated with a special character to denote a special meaning, with "$\_$" (underscore) denoting a vector, "$\hat{}$" (hat) denoting an inferred/predicted value, and "$\tilde{}$" (tilde) denoting a set of self-observations and recommendations. We use *i* to refer to a SR, *j* to refer to a SP, and *k* to refer to a recommender. We assume transportation/link failures can be identified by protocols in lower networking layers. We assume that the observation and assessment of received service is error-free. Hence, SR *i*'s judgment of SP *j*'s service quality is the actual service quality, i.e., $s_{ij}^t = s_j^t$.

TABLE III: Notation.

Notation	Meaning
i	Node i normally referring to a service requestor (SR)
k	Node k normally referring to a recommender
j	Node j normally referring to a service provider (SP)
T_j^t	j's service trustworthiness at time t
T_{ij}^t	j's service trustworthiness at time t as predicted by node i
s_j^t	Actual service quality delivered by j at time t
$\hat{s}_{ij}^t$	s_j^t as predicted by i
s_{ij}^t	i's self-observation of service quality of j at time t
$\underline{\tilde{S}}_{ij}^t$	A vector of self-observations and recommendations received by SR i for service quality of SP j at time t
$\underline{\tilde{S}}_{ij}$	$\left[\underline{\tilde{S}}_{ij}^{t_0}, \dots, \underline{\tilde{S}}_{ij}^{t_n}\right]$, a vector of $\underline{\tilde{S}}_{ij}^t$ over $[t_0, \dots, t_n]$
x_m^t	m^{th} context variable value observed at time t
$\underline{x}^t$	$[x_1^t, \dots, x_M^t]$, a vector of M context variable values observed at time t
$\underline{X}$	$\left[\underline{x}^{t_0}, \dots, \underline{x}^{t_n}\right]$, a vector of $\underline{x}^t$ over $[t_0, \dots, t_n]$
$\underline{\beta}_j$	$\left[\beta_j^1, \dots, \beta_j^M\right]$, a vector of regression coefficients for M context variables
$\underline{\hat{\beta}}_{ij}$	i's estimate of $\underline{\beta}_j$

The problem at hand is for SR i to predict whether SP j will perform satisfactorily or not for a requested service in a particular context environment, given a history of evidence. The objective is to achieve high prediction accuracy in terms of correctly predicting bad service while not missing good service from a SP. Here we note that a node, malicious or not, can provide good service or bad service depending on the context environment.

Within a specific type of service, SR i's observation s_{ij}^t at time t of the service quality received from SP j is either "satisfactory" or "unsatisfactory." If the service quality is satisfactory, then s_{ij}^t=1 and SP j is considered *trustworthy;* otherwise, s_{ij}^t=0 and SP j is considered *untrustworthy*. Let the operational and environmental conditions at time t be characterized by a set of distinct M context variables $\underline{x}^t = [x_1^t, \dots, x_M^t]$. Then, SP j's *service trust* is the probability that SP j is capable of providing satisfactory service given context variable $\underline{x}^t$, i.e., $T_j^t \triangleq \Pr(s_j^t = 1)$.

Let k ($k \neq i$) be a recommender who had prior service experience with SP j and is asked by SR i to provide its feedback regarding SP j. The recommendation from node k is in the form of $[\underline{x}^t, s_{kj}^t]$ specifying the context $\underline{x}^t$ under which the observation s_{kj}^t was made. Since k might launch recommendation attacks, it might report a dishonest observation to i, in which case s_{kj}^t reported is $1 - s_{kj}^t$. Let $\underline{\tilde{S}}_{ij} = \left[\underline{\tilde{S}}_{ij}^{t_0}, \dots, \underline{\tilde{S}}_{ij}^{t_n}\right]$, $i \neq j$, denote the cumulative evidence gathered by SR i over $[t_0, \dots, t_n]$, including self-observations and recommendations. Also let $\underline{X} = \left[\underline{x}^{t_0}, \dots, \underline{x}^{t_n}\right]$ denote the corresponding context variable value vector over $[t_0, \dots, t_n]$.

CATrust learns the service behavior pattern of SP j based on $\underline{\tilde{S}}_{ij}$ and $\underline{X}$, and predicts the probability that SP j is trustworthy at time t_{n+1}, given $\underline{x}^{t_{n+1}}$ as input. Suppose that node j follows service behavior pattern $\underline{\beta}_j$. Then, our prediction problem is to estimate $T_{ij}^{t_{n+1}} = \Pr(\hat{s}_{ij}^{t_{n+1}} = 1 | \underline{x}^{t_{n+1}}, \underline{\hat{\beta}}_{ij})$, where $\underline{\hat{\beta}}_{ij}$ is node i's estimate of $\underline{\beta}_j$. Essentially, $T_{ij}^{t_{n+1}}$ obtained above is the service trust of SP j at time t_{n+1} from SR i's perspective.

B. Trust Computation

We utilize a sigmoid function to link the binary observation of service quality with context variables in a continuous range. More specifically, we utilize robust logistic regression [16] to analyze the relation between $\underline{\tilde{S}}_{ij}$ and $\underline{X}$. While many forms exist for relating $\underline{\tilde{S}}_{ij}$ and $\underline{X}$, we adopt a linear model for its simplicity and effectiveness, treating SP j's behavior pattern $\underline{\beta}_j = \left[\beta_j^1, \dots, \beta_j^M\right]$ essentially as a vector of M regression coefficients matching the context variable vector $\underline{x}^t = [x_1^t, \dots, x_M^t]$. Later in Section VII, we discuss the feasibility of using other models.

We assume observations are mutually independent and that the order of observations can be changed. Following the linear model for a classical logistic regression problem, j's service trustworthiness at time t is modeled by:

$$\begin{aligned} T_j^t &= \Pr(s_j^t = 1 | \underline{x}^t, \underline{\beta}_j) \\ &= \left(1 + \exp(-(\underline{x}^t)^\top \underline{\beta}_j)\right)^{-1} \end{aligned} \tag{3}$$

Or, equivalently, following the logit function definition, $\text{logit}(y) = \ln\left(\frac{y}{1-y}\right)$, we have:

$$\text{logit}(T_j^t) = (\underline{x}^t)^\top \underline{\beta}_j \tag{4}$$

The logit function defined in (4) is the link function in logistic regression for transforming the prediction from a binary service outcome (0 or 1) to a continuous outcome upon which linear regression may be conducted. With (3), i estimates j's trust T_j^t based on its estimated $\underline{\beta}_j$. To do so, i needs to estimate $\underline{\beta}_j$, but it only has noisy observations of the service history. We model this by:

$$z_{ij}^t = \text{logit}(T_{ij}^t) = (\underline{x}^t)^\top \underline{\beta}_j + \varepsilon_{ij}^t \tag{5}$$

where T_{ij}^t is j's service trustworthiness at time t as predicted by node i, and ε_{ij}^t is an independent error term following the logistic distribution with the cumulative distribution function $\frac{1}{1+e^{-y}}$, $y \in (-\infty, \infty)$.

A malicious recommender k can modify all unsatisfactory observations on j to "1" in a ballot-stuffing attack, while reversing all satisfactory services to "0" in a bad-mouthing attack. Malicious behaviors result in "outliers" which will lead to inaccurate estimation. We adopt robust logistic regression [16] to replace the error term ε_{ij}^t in (5) with a noise term that follows the standard t-distribution with ν_0 degrees of freedom to tolerate recommendation attacks without overly sacrificing solution accuracy. The logistic distribution is known to resemble the t-distribution in shape but has heavier tails. We

set $\nu_0 = 7$ as in [16] to make the t-distribution also possess heavy tails, which increases the ability to absorb outlier errors and provides robust estimates of $\underline{\beta_j}$.

After replacing ε_{ij} by a standard t-distribution random variable, denoting $(\underline{x}^t)^\top \underline{\beta_j}$ as u^t, and providing a value for the hyper parameter ν, z_{ij}^t in (5) has the following density function:

$$f_\nu(z) = (\pi\nu)^{-\frac{1}{2}}\Gamma\left(\frac{\nu+1}{2}\right)\Gamma^{-1}\left(\frac{\nu}{2}\right)\left(1+\frac{(z-u)^2}{\nu}\right)^{-\frac{\nu+1}{2}} \quad (6)$$

where Γ is the gamma function. We apply Bayesian inference based on the data augmentation algorithm with Markov Chain Monte Carlo (MCMC) [8] [21] to infer $\underline{\beta_j}$ given historic observations $\underline{\tilde{s}_{ij}}$, as follows:

$$p\left(\underline{\beta_j}\middle|\underline{\tilde{s}}_{ij}\right) = \int p\left(\underline{\beta_j}\middle|\underline{z}_{ij},\ \underline{\tilde{s}}_{ij}\right) p\left(\underline{z}_{ij}\middle|\underline{\beta_j},\underline{\tilde{s}}_{ij}\right) d\underline{z}_{ij} \quad (7)$$

where $\underline{z}_{ij} = [z_{ij}^{t_0}, \dots, z_{ij}^{t_n}]$ is a latent variable set introduced by the data augmentation algorithm in order to construct known $p\left(\underline{\beta_j}\middle|\underline{z}_{ij},\ \underline{\tilde{s}}_{ij}\right)$. However, due to the t-distribution for ε_{ij}^t, there is no closed-form solution for $p\left(\underline{\beta_j}\middle|\underline{z}_{ij},\ \underline{\tilde{s}}_{ij}\right)$. To circumvent this, z_{ij}^t is approximated by a scale mixture of Gaussian distribution, i.e., $z_{ij}^t|\omega_{ij}^t \sim \mathcal{N}(u^t, (\omega_{ij}^t)^{-1})$ with $\omega_{ij}^t \sim \Gamma(\nu/2, \nu/2)$ where $\mathcal{N}(u^t, (\omega_{ij}^t)^{-1})$ is Gaussian distribution with mean $u^t$and variance $(\omega_{ij}^t)^{-1}$ and $\Gamma(\nu/2, \nu/2)$ is Gamma distribution with shape $\nu/2$ and scale $\nu/2$. Let $\underline{\omega}_{ij} = [\omega_{ij}^{t_0}, \dots, \omega_{ij}^{t_n}]$. Then (7) can be rewritten as:

$$p\left(\underline{\beta_j}\middle|\underline{\tilde{s}}_{ij}\right) = \int\int p\left(\underline{\beta_j}\middle|\underline{z}_{ij}, \underline{\omega}_{ij}, \underline{\tilde{s}}_{ij}\right) p\left(\underline{z}_{ij}, \underline{\omega}_{ij}\middle|\underline{\beta_j}, \underline{\tilde{s}}_{ij}\right) d\underline{\omega}_{ij} d\underline{z}_{ij} \quad (8)$$

Assuming a Gaussian priori to $\underline{\beta_j}$ with known mean and variance defined above, the posterior $\underline{\beta_j}|\underline{z}_{ij}, \underline{\omega}_{ij}, \underline{\tilde{s}}_{ij}$ will follow a Gaussian distribution and the posterior $\underline{\omega}_{ij}|\underline{z}_{ij}, \underline{\beta_j}$ will follow a Gamma distribution [8] [21]. Meanwhile, $\underline{z}_{ij}|\underline{\beta_j}, \underline{\tilde{s}}_{ij}$ is a truncated t-distribution, depending on the value of $\underline{\tilde{s}}_{ij}$. We then apply an iterative sampling procedure to first draw a new $(\underline{z}_{ij}, \underline{\omega}_{ij})$ and then produce a new $\underline{\beta_j}$ from $p\left(\underline{\beta_j}\middle|\underline{z}_{ij}, \underline{\omega}_{ij},\ \underline{\tilde{s}}_{ij}\right)$. This process is repeated iteratively until the Markov Chain is stabilized. The final $\underline{\beta_j}$ obtained is node i's estimated $\underline{\beta_j}$, i.e., $\underline{\hat{\beta}}_{ij}$. Node i can then compute T_{ij}^{t+1} by (3), given $\underline{x}^{t+1}$ and $\underline{\hat{\beta}}_{ij}$ as input.

C. Recommendation Filtering

The logistic regression model with t-distribution error enables fairly robust learning of a SP's service behavior pattern based on a trustor's self-observations and the recommenders' history records. However, for any statistical model, there is a breakpoint. In our case, if the percentage of malicious nodes is too high, it can hardly differentiate honest from dishonest recommendations, thus resulting in a low accuracy rate. One possible solution is to seek socially connected peers (friends) who are most likely to deliver honest recommendations. However, the limitation is that it depends on the availability of friends in the neighborhood and also it requires friends to have direct service experiences with the targeted SP, a condition that may be difficult to meet in SOANET environments.

We propose a novel threshold-based recommendation filtering mechanism. The main idea is to compare the SR's prediction of the service trust toward the trustee with the recommender's report toward the same trustee under the same context environment. Specifically each time node k, serving as a recommender, propagates a recommendation about node j in the form of $[\underline{x}^t,\ s_{kj}^t]$, node i will apply the current predictor $\underline{\hat{\beta}}_{ij}$ with $\underline{x}^t$ given as input to compute T_{ij}^t. Ideally, if k delivers honest s_{kj}^t, T_{ij}^t should be closer to s_{kj}^t than to $1 - s_{kj}^t$. To filter out malicious recommendations, i uses a threshold parameter T^{th}. If $|T_{ij}^t - s_{kj}^t|$ is larger than T^{th}, s_{kj}^t is considered modified and consequently $[\underline{x}^t,\ s_{kj}^t]$ is rejected by i; otherwise, it will be accepted by i and put in the training set for updating $\underline{\hat{\beta}}_{ij}$. In case a recommender provides several recommendations for several distinct context environments, the average difference can be first computed before applying threshold-based recommendation filtering. The threshold parameter T^{th} is an important parameter whose effect on protocol performance will be analyzed in Section V.

D. Characteristics of Context Variables

Context variables must obey the property that in similar context, the service from a SP performs similarly. Therefore, context variables are inherently tied to a SP's service behavior and the service quality criteria defined by an application. In our example SOANET application [26] illustrated in Figure 1, service quality is defined by three criteria, namely, QoI, service delay, and service cost. Consequently, energy (which influences QoI [17]), local traffic (which influences service delay), and incentive (which influences service cost [9]) are natural choices for this application. As these context variables have clear physical meanings, the range of a context variable can be defined accordingly. For example, the energy context variable can be categorized as [high, medium, low] denoting the energy status of a SP. The incentive context variable can be the price paid to a SP upon satisfactory completing of service, in the range of [minimum price, maximum price]. The local traffic context variable can be the number of neighbors simultaneously transmitting packets with the range of [0, maximum node density × radio range area]. A range with a finer granularity allows CATrust to more accurately learn a SP's service behavior pattern and service quality at the expense of computational complexity (see Section VII.A for a discussion). Another property that must be satisfied is that a context variable must be measurable at runtime. For the example SOANET application, the energy status of SP j can be measured by the SR by counting the ratio of the number of acknowledgement packets received from SP j over the total

number of transmitted packets from the SR to SP *j* during the encounter interval. The incentive to SP *j* is determined by the SR itself so it is easily measurable by the SR. The local traffic can be estimated by the SR based on the collision probability or the packet retransmission probability after transmitting a sequence of packets for initiating a service request.

V. Analysis of Convergence, Accuracy, and Resiliency Properties of CATrust

In this section, we analyze the convergence, accuracy, and resiliency properties of CATrust against collusion recommendation attacks. We first describe the environment setup and then we present the results.

Table IV: Parameters and their Values.

Notation	Meaning	Default Value
A	Operational area	800x800 m^2
S	Speed	[1.0, 2.5] m/s
P	Pause time	[0, 60] s
W	Movement time	[5*60, 15*60] s
R	Radio range	220 m
α	Encountering rate	5/hr
λ	Service request rate	5/encounter
L	Length of measurement interval	24 hr
t_{unit}	Trust update interval	7.2min
P_{ssr}	Satisfactory service ratio	60%
p_{unit}	Unit price	1
v_0	Degree of freedom	7
P_b	Percentage of bad nodes	[10%-70%]
T^{th}	Recommendation filtering threshold	[0-1]
n_n	Number of nodes	50
n_c	Number of context variables	3
n_r	Number of service records per node	$\alpha L\lambda$

A. Environment Setup

Table IV lists a set of parameters and their values/ranges used in our analysis. We simulate a SOANET with n_n nodes and the operational area being a rectangular area *A*, The radio transmission range is *R*, and the mobility model is the Random Waypoint mobility (RWM) model [10] without considering mobility dependency among nodes or geography obstacles. Under RWM, every node moves randomly, with speed S, movement time W, and pause time P defining the movement pattern such that the average encounter rate between any two nodes is approximately α. We simulate encountering events (i.e., when nodes are in the same subarea within radio range) at which service requests are issued and service quality received are recorded, and recommendations are exchanged. Only one service type is considered for simplicity. A node acting as a SR has a service request rate of λ upon encountering a potential SP. A node can provide recommendations only to nodes with which it has had service experiences. The measurement time interval for data collection is *L*. The trust update interval is t_{unit}.

We consider three context variables in the experiment, namely, energy-sensitivity ($\underline{x_e}$), capability-limitation ($\underline{x_c}$), and profit-awareness ($\underline{x_p}$). The values of context variables are generated as follows: x_e^t is measured by the number of neighbors sharing the channel as more energy is consumed for channel contention and packet retransmission when there are more nodes sharing the channel. x_c^t is measured by the number of service requests to be processed in a SP's queue as high traffic to the SP hinders its processing capability. x_p^t is SP's potential gain upon satisfactory service completion. The potential gain consists of two parts: the asked price P_{ask} from a SP, calculated by multiplying the queue length with the unit price p_{unit}, and the overpaid price p_{over} by the SR that represents the overpaying incentive, modeled by a normal distribution with mean and variance being 50% and 12.5% of the asked price P_{ask}, respectively. Once $[x_e^t, x_c^t, x_p^t]$ is generated, we generate s_g^t (ground truth service satisfaction) such that the average satisfactory service ratio is P_{ssr}. A SP, whether malicious or not, has its own P_{ssr} and has specific context environment instances under which it can provide satisfactory service within its capability.

Figure 2 shows a snapshot of s_g^t (ground truth service satisfaction) vs. $[x_e^t, x_c^t, x_p^t]$ for a SP with P_{ssr}= 60%.

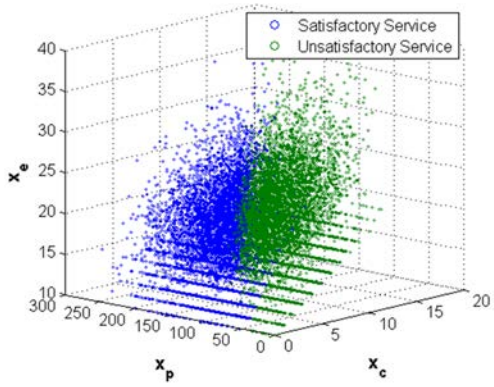
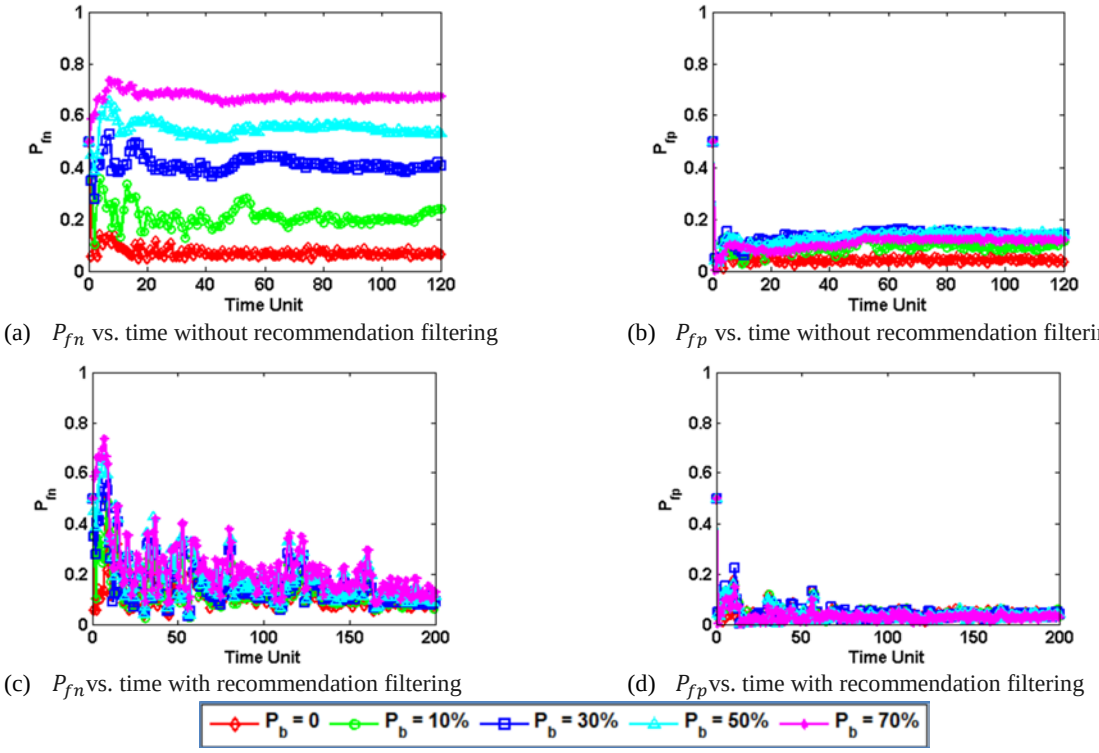


Figure 2: Visualization of Synthetic Data.

The hostility level is described by the percentage of malicious nodes (P_b) whose effect will be analyzed. Malicious nodes are randomly picked and will perform attacks as described in the threat model. For CATrust, the degree of freedom v_0 of the t-distribution error is set to 7 (as in [16]) for approximating the original logistic distribution error. We analyze the effect of the recommendation filtering threshold parameter (T^{th}) on protocol performance.



(a) P_{fn} vs. time without recommendation filtering

(b) P_{fp} vs. time without recommendation filtering

(c) P_{fn}vs. time with recommendation filtering

(d) P_{fp}vs. time with recommendation filtering

Figure 3: Convergence, Accuracy, and Resiliency Behavior of CATrust for a Malicious Trustee SP. The Top Graphs are without Recommendation Filtering. The Bottom Graphs are with Recommendation Filtering.

B. Convergence, Accuracy and Resiliency Numerical Results

We use MATLAB to implement the algorithm and collect numerical data for analyzing the convergence, accuracy, and resiliency properties of CATrust against collusion recommendation attacks. The performance metrics are false negative probability (P_{fn}) and false positive probability (P_{fp}) described earlier. Each data point reported is the average of 100 randomly generated test cases in a test data set $\left[\underline{x}_e, \underline{x}_c, \underline{x}_p, \underline{s}_g\right]$. Specifically, for the case in which s_g (ground truth) is 1 (satisfactory service) and the service trust predicted by CATrust is T_{ij}^t, then $P_{fp} = 1 - T_{ij}^t$ because $1 - T_{ij}^t$ is the belief that the service provided will be unsatisfactory so it is the missing good service probability. For the case in which s_g (ground truth) is 0 (unsatisfactory service) and the service trust predicted by CATrust is T_{ij}^t, then $P_{fn} = T_{ij}^t$ because T_{ij}^t is the belief that the service provided will be satisfactory so it is the missing bad service probability.

Figure 3 shows P_{fn}/P_{fp} vs. time as P_b (the percentage of malicious nodes) varies in the range of 0-70%, for a malicious trustee SP randomly picked (with P_{ssr}=0.6). The top (bottom) 2 graphs are without (with) recommendation filtering. With recommendation filtering, if the difference between the predicted service trust and the recommended service trust under the same context environment is above a threshold, then the recommendation is filtered. We intentionally used the same (and full) scale for all graphs, so we can visually see the sensitivity of P_{fn}/P_{fp} values with respect to P_b.

First of all, we see fast convergence behavior in both cases without much sensitivity to P_b. However, without recommendation filtering, the prediction accuracy of P_{fn} (see Figure 3(a)) is inversely related to P_b. The reason is that without recommendation filtering, as P_b increases, a SR will receive more and more high but false service trust recommendations from more malicious nodes performing ballot-stuffing attacks. These malicious trust recommendations cause the SR to misidentify bad service provided by the malicious node. Second, from Figure 3(c), we observe that with recommendation filtering, CATrust is able to effectively filter out false recommendations and, as a result, converges to the same low P_{fn}value for high accuracy eventually. Note that the ideal P_{fn}value is 0. Hence, a low P_{fn}value close to 0 after convergence means high accuracy. This demonstrates that CATrust with recommendation filtering is resilient to ballot-stuffing attacks, even in extremely hostile environments. Last, from comparing Figures 3(b) and 3(d), we observe that recommendation filtering has a relatively small effect on

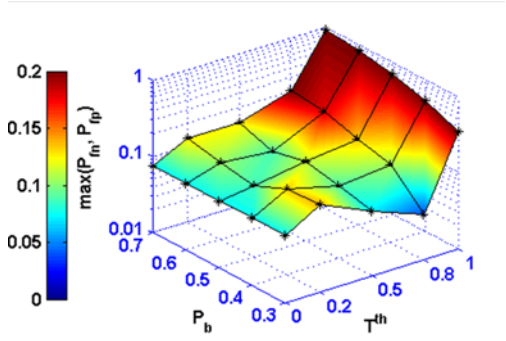
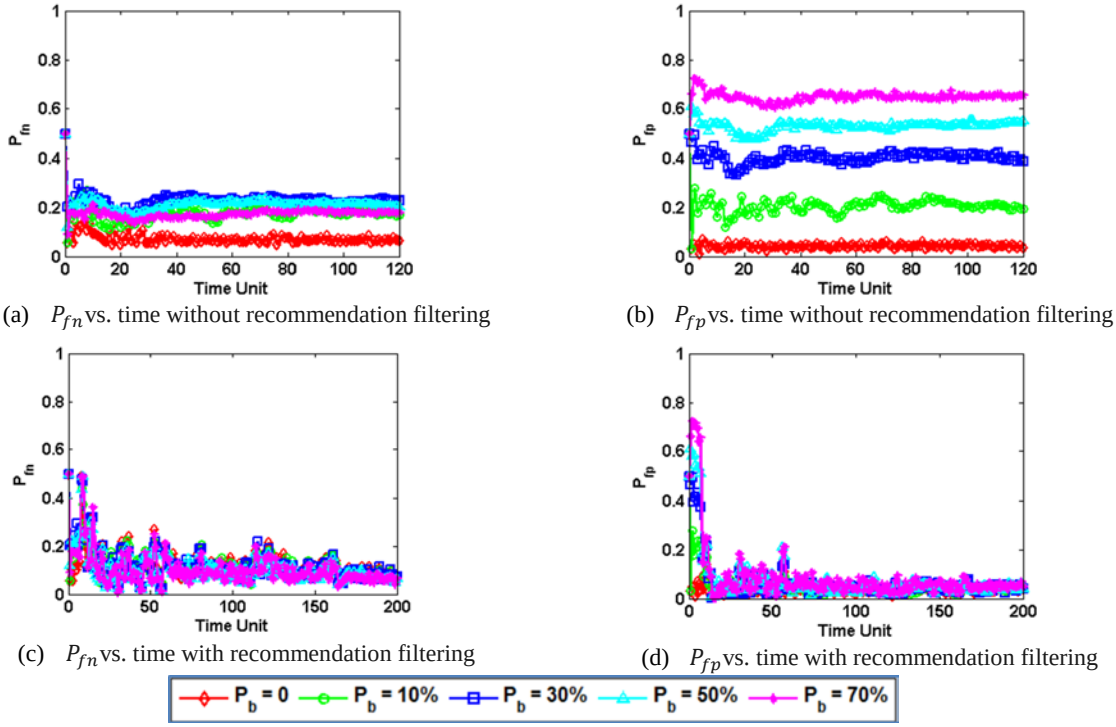


Figure 4: Effect of Recommendation Filtering Threshold T^{th} on $\max(P_{fn}, P_{fp})$ for a Malicious Trustee SP.



(a) P_{fn} vs. time without recommendation filtering

(b) P_{fp} vs. time without recommendation filtering

(c) P_{fn} vs. time with recommendation filtering

(d) P_{fp} vs. time with recommendation filtering

Figure 5: Convergence, Accuracy and Resiliency Behavior of CATrust for a Non-malicious Trustee SP. The Top Graphs are without Recommendation Filtering. The Bottom Graphs are with Recommendation Filtering.

CATrust's prediction accuracy of P_{fp}. The reason is that ballot-stuffing attacks can boost bad services but cannot further boost already good services provided by a malicious node.

Figure 4 analyzes the sensitivity of CATrust performance with respect to the recommendation filtering threshold T^{th}, a design parameter in our trust protocol design. For a service-oriented application, what matters to the end user is not to miss a good service (i.e., low P_{fp}) and not to misidentify a bad service as a good service (i.e., low P_{fn}). In many applications, minimizing both P_{fn} and P_{fp} is desirable. Hence, we use min max(P_{fn}, P_{fp}) as the performance metric to identify the best T^{th} for performance maximization. There is a tradeoff between P_{fn} and P_{fp}. That is, as the minimum trust threshold T^{th} increases, the false negative probability P_{fn}decreases while the false positive probability P_{fp} increases. We observe from Figure 4 that there exists an optimal T^{th} at which max(P_{fn}, P_{fp}) is minimized, given P_b as input. For example when $P_b = 0.3$, the optimal value T^{th} is 0.8, but when $P_b = 0.4$, the optimal value T^{th} is 0.5. This result suggests that one should dynamically adjust the filtering threshold T^{th} to adapt to changes in hostility conditions in order to maximize application performance, i.e., minimizing both P_{fn} and P_{fp}.

Correspondingly, Figure 5 shows P_{fn}/P_{fp} vs. time as P_b varies in the range of 0-70%, for a non-malicious trustee SP with P_{ssr} =0.6. Here the trustee SP is non-malicious, so malicious nodes will perform bad-mouthing attacks to ruin its service trust, which, as opposite to ballot-stuffing attacks, will affect P_{fp} more than P_{fn}. We observe from Figure 5(b) that without recommendation filtering, the prediction accuracy of P_{fp} is indeed inversely related to P_b. The reason is that without recommendation filtering, as P_b increases, a SR will receive more and more low but false service trust recommendations from more malicious nodes performing bad-mouthing attacks. These malicious recommendations cause the SR to misidentify good service provided by the non-malicious SP, which is downgraded due to bad-mouthing attacks. From Figure 5(d), we again observe that with recommendation filtering, CATrust is able to effectively filter out false recommendations and, as a result, converges to the same low P_{fp}value eventually as time progresses. This result demonstrates that CATrust with recommendation filtering is resilient to bad-mouthing attacks. From comparing Figure 5(a) with Figure 5(c), we observe that recommendation filtering has a relatively small effect on CATrust's high prediction accuracy of P_{fn}. The reason is that bad-mouthing attacks (on a non-malicious node) can effectively downgrade good services but cannot downgrade already bad services provided by a non-malicious node.

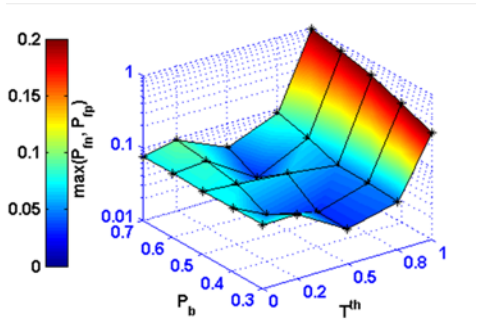
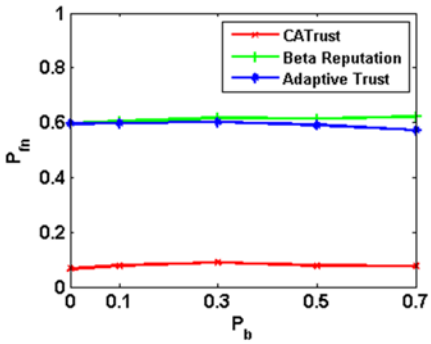
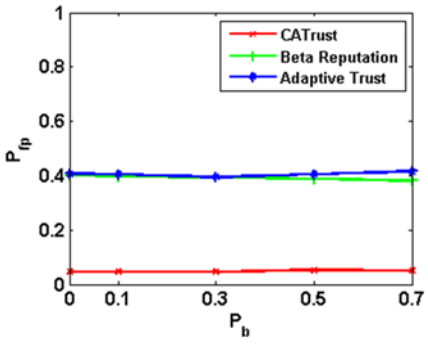


Figure 6: Effect of Recommendation Filtering Threshold T^{th} on max(P_{fn}, P_{fp}) for a Non-malicious Trustee SP.

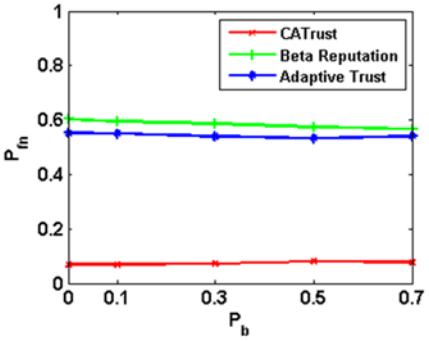


(a) P_{fn} vs. P_b for a malicious trustee SP.

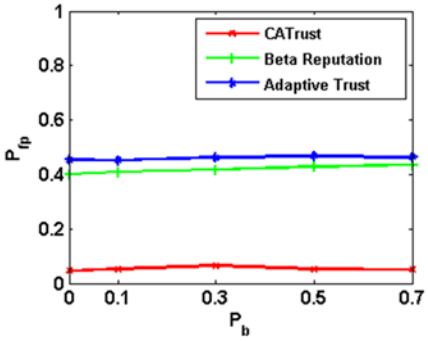


(b) P_{fp} vs. P_b for a malicious trustee SP.

Figure 7: Performance Comparison of CATrust vs. Beta Reputation and Adaptive Trust for a Malicious Trustee SP.



(a) P_{fn} vs. P_b for a non-malicious trustee SP.



(b) P_{fp} vs. P_b for a non-malicious trustee SP.

Figure 8: Performance Comparison of CATrust vs. Beta Reputation and Adaptive Trust for a Non-malicious Trustee SP.

Figure 6 analyzes the sensitivity of CATrust performance with respect to the recommendation filtering threshold T^{th} when the trustee SP is non-malicious. We again observe from Figure 6 that there exists an optimal T^{th} at which $\max(P_{fn}, P_{fp})$ is minimized, given P_b as input. We conclude that adjusting T^{th} dynamically to maximize application performance is a viable design. Our analysis paves the way for realizing adaptive control for protocol performance optimization.

VI. Performance Comparison

In this Section, we compare protocol performance of CATrust against Beta Reputation [12] and Adaptive Trust Management [3]. For fair comparison, we compare all three protocols at their optimizing conditions. See Section II for a description of these two baseline schemes and the reasons we select them for performance comparison.

Figure 7 shows performance comparison in terms of converged P_{fn}/P_{fp}values for a malicious SP with P_{ssr}=0.6, as $P_b$varies in the range of [0, 70%]. We observe that while all three protocols are resilient against collusion recommendation attacks, CATrust performs best by a wide margin. We notice that because the trustee SP is malicious, P_{fn}will be affected more than P_{fp}via ballot-stuffing attacks in this case.

Correspondingly Figure 8 compares performance in terms of converged P_{fn} / P_{fp} values for a non-malicious SP with P_{ssr} =0.6, as P_b varies in the range of [0, 70%]. We again observe that CATrust outperforms the other two by a wide margin. We notice that because the trustee SP is non-malicious, P_{fp}will be affected more than P_{fn}via bad-mouthing attacks in this case.

The superiority of CATrust over Beta Reputation and Adaptive Trust Management as demonstrated in Figures 7 and 8 is attributed to the fundamental difference in trust protocol design logic. CATrust infers a service trust value for *each* context environment based on the trustee node's predicted service behavior in that context environment, while Beta Reputation or Adaptive Trust Management just maintains one service trust variable across all context environments. Consequently, for a malicious trustee SP (as in Figure 7), P_{fn} (missing a malicious node's bad service) tends to converge to the malicious node's average service trust value which is equivalent to the malicious node's satisfactory service ratio $P_{ssr} = 0.6$. For a non-malicious trustee SP (as in Figure 8), P_{fp}(missing a non-malicious node's good service) tends to converge to $1 - P_{ssr} = 1 - 0.6 = 0.4$. In contrast, our CATrust protocol is not bound by the satisfactory service ratio. Rather, by learning the trustee node's service behavior, CATrust infers a service trust value as close to the ground truth service satisfaction as possible in a particular context environment. The association of service trust with context results in high prediction accuracy, simply because the trust value inferred is tied to a specific context environment. Beta Reputation and Adaptive Trust Management, on the other hand, can only infer the average trust value across all context

environments as context information is not taken into consideration in their trust protocol design.

VII. Discussion

A. Computation Feasibility

In this subsection, we discuss the computational feasibility for a SOANET node to execute CATrust to learn the behavior patterns of other nodes ($\underline{\beta_j}'s$ for individual SPs) at runtime. Based on our trust propagation and aggregation protocol design, a SR stores a new service record of length $n_c + 1$ (for n_c context variables and user satisfaction) toward a SP after having a direct service experience with the SP. Two nodes encountering each other exchange their past service records toward all other nodes in the system. The memory complexity per node is linear in O($n_n n_c n_r$), where n_n is the number of nodes, n_c is the number of context variables, and n_r is the number of service records (as defined in Table IV), because every node needs to store n_r service records (each of size n_c) for each of the other n_n-1 nodes. The communication cost complexity per node is O($n_n n_c \alpha L$) where α is the encountering rate and L is the length of the measurement interval, because every node potentially can provide n_n-2 service recommendation records (each of size n_c) toward the other n_n-2 nodes whenever it encounters another node. Last, the computational complexity is O($n_n k \max(n_c, n_r)$), where k is the number of iterations needed for reaching convergence, because every node needs to update n_r latent variables corresponding to the $n_r$service records and $\underline{\beta_j}$ of size n_c for node j in each iteration and this computational procedure is applied to each of the other n_n-1 nodes in the system. In general, $n_c \ll n_r$ so the computational complexity is O($k n_n n_r$). Further, the magnitude of k largely depends on the granularity of context variable values, e.g., a range of (high, medium, low) for energy is of low granularity, while a range of [0-10] joule is of high granularity. By controlling data granularity, k is a small constant relative to n_n or n_r, so in practice the computational complexity of CATrust is just O($n_n n_r$).

As a comparison, the memory complexity, message complexity, and computational complexity for both Beta Reputation [12] and Adaptive Trust Management [3] are O($C n_n$), O($\alpha L C n_n$), and O($n_n n_r$), respectively, where *C=2* for Beta Reputation (2 positive/negative service counts) and *C=5* for Adaptive Trust Management (2 positive/negative service counts and 3 social similarity lists). We first observe that CATrust has the same order of computational complexity O($n_n n_r$) as Beta Reputation and Adaptive Trust Management. With $n_c \approx C$ (that is, the number of context variables is between 2 to 5), CATrust, Beta Reputation, and Adaptive Trust Management have comparable communication overhead. Last, CATrust has a higher memory overhead by a factor of n_r. In practice, the memory overhead is lower because one may be interested in only the most recent n_r service records (e.g., in the past hour or day). This memory requirement can still be excessive for SOANET nodes with limited memory space. We refer the readers to a caching design [3] as a possible solution to mitigate this problem. For the experimental setting specified in Table IV, a node with a 2.4 GHz i7 CPU with 8GB RAM took 2.63s real time to learn a SP's behavior pattern and predict its trust. For a less powerful node, it may take minutes rather than seconds to compute the result. Fortunately, the computational procedure needs to be executed only periodically in the background by a SR after new observations are collected. Before the next trust update time arrives, a SR can simply use learned behavior patterns ($\underline{\beta_j}'s$ for individual SPs) for decision making.

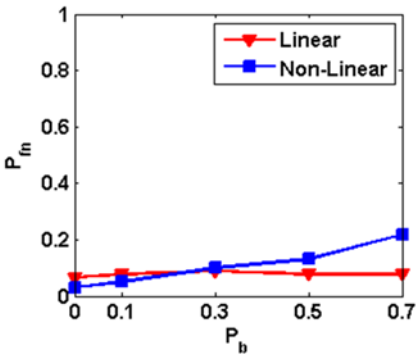
B. Dealing with Conflicting Behavior and Random Attacks

In this subsection, we discuss the applicability of CATrust in environments with conflicting behavior and random attacks.

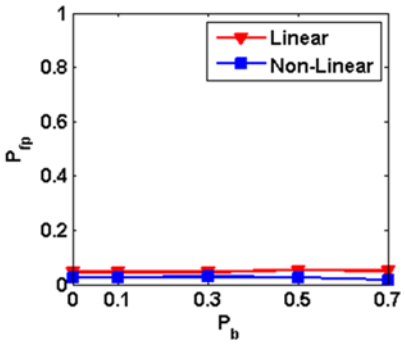
With conflicting behavior attacks, a malicious SP can selectively provide satisfactory service for some SRs while providing unsatisfactory service for others. In general, the relationship between a SR and a SP determines the SP's service behavior toward the SR. This is naturally solved by CATrust since it is based on SR-SP pairing. More specifically, if SP *j* who is capable of providing good service in a context environment provides bad service to SR *i*, then SR *i* will consider SP *j*'s bad service as SP *j*'s service behavior in this context environment. In effect, from the perspective of SR *i*, SP *j*'s s_g (ground truth service satisfaction) is changed from 1 (satisfactory service) to 0 (unsatisfactory) which is learned by logistic regression. As a result, SR *i* will predict unsatisfactory service being provided by SP *j* even though SP *j* is capable of providing satisfactory service in the same context environment.

With random attacks, a malicious node will provide bad service only randomly so as not to risk itself being labeled as a node providing bad service and not being selected for service. Again random attack can be naturally covered by CATrust since it is based on SR-SP pairing. From the perspective of SR *i* who is under random attacks by SP *j*, SP *j*'s s_g (ground truth service satisfaction) is sometimes 1 (satisfactory service) and sometimes 0 (unsatisfactory), which is learned by logistic regression. As a result, SR *i* will predict sometimes satisfactory service and sometimes unsatisfactory service being provided by SP *j* even though SP *j* is capable of providing satisfactory service in the same context environment. Consequently, the degree to which the malicious node can disguise itself as a SP providing good service is simply proportional to 1 − random attack probability. As long as SP *j's* random attack probability is not zero, the random attack behavior will be learned by SR *i*.

C. Linear Model vs. Non-linear Model Comparison

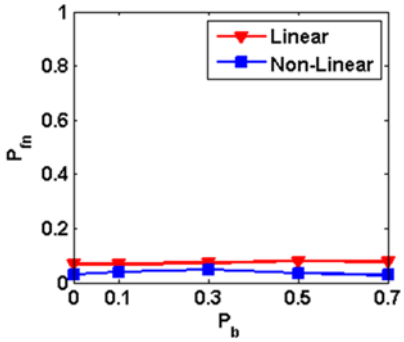


(a) P_{fn} vs. P_b for a malicious trustee SP.

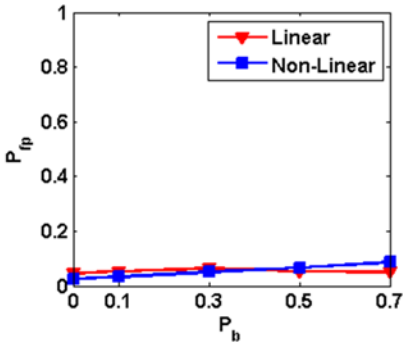


(b) P_{fp} vs. P_b for a malicious trustee SP.

Figure 9: Performance Comparison of Linear CATrust vs. Non-Linear CATrust for a Malicious Trustee SP.



(a) P_{fn} vs. P_b for a non-malicious trustee SP.



(b) P_{fp} vs. P_b for a non-malicious trustee SP.

Figure 10: Performance Comparison of Linear CATrust vs. Non-Linear CATrust for a Non-malicious Trustee SP.

In many applications, context variables may not be independent, including covariate relationship between service observations and correlation between context variables. The results which we have reported above are based on a simple linear model with computational complexity of $O(n_n n_r)$ (see VII.A) to model the relation between context variables and observations. In this subsection, we conduct a comparative analysis to test if the prediction accuracy may improve further with a non-linear model at the expense of added computational complexity. The non-linear model implemented is the multi-layer feedforward neural network (FNN) algorithm [7] with n_c nodes in the input layer and n_c^2 nodes in the hidden layer, resulting in computational complexity of $O(n_n n_r n_c^3)$ to process n_r service records for each of the other n_n-1 nodes.

Figures 9 and 10 compare linear CATrust vs. non-linear CATrust performance in terms of converged P_{fn}/P_{fp}values for a malicious trustee SP and a non-malicious trustee SP, respectively, with P_{ssr}=0.6, as $P_b$varies in the range [0, 70%]. Note that the setup is the same as in Figures 7 and 8. Also note that regardless of node type (malicious or non-malicious), the probabilities of misidentifying a node's bad service and good service are measured by $P_{fn}$and P_{fp}, respectively. As shown in Figure 9 (for a malicious trustee SP) as P_b increases, linear CATrust performs better than non-linear CATrust in P_{fn}, while non-linear CATrust performs better than linear CATrust in P_{fp}. Because the trustee SP is malicious in this case, P_{fn} (the probability of the SP's bad service being missed) increases as P_b increases via ballot-stuffing attacks. We observe that non-linear CATrust is less resilient to ballot-stuffing attacks than linear CATrust. The reason is that FNN uses mean square error as the objective function known to be sensitive to contaminated data [7]. On the other hand in Figure 10 (for a non-malicious trustee SP) as P_b increases, non-linear CATrust performs better than linear CATrust in P_{fn}, while linear CATrust performs better than non-linear CATrust in P_{fp}. Because the trustee SP is non-malicious in this case, P_{fp} (the probability of the SP's good service being missed) increases as P_b increases via bad-mouthing attacks. We again observe that non-linear CATrust is less resilient to bad-mouthing attacks than linear CATrust. On the whole there is a virtual tie between the linear and non-linear models. However, the much higher $P_{fn}$for a malicious trustee SP as P_b increases (see Figure 9(a)) and the much higher computation complexity make the non-linear model an undesirable choice for runtime execution.

VIII. Conclusion

We proposed a novel regression-based trust model, CATrust, for evaluating service trust in service-oriented ad hoc networks. CATrust assesses each SP in terms of its service behavior patterns in response to context environment changes. The net effect is that we are able to learn and then predict its service behavior in a particular context environment, instead of judging its trustworthiness from satisfactory/unsatisfactory

service history across all context environments. We also built a novel threshold-based recommendation filtering mechanism to effectively filter out dishonest recommendations. A salient feature of our model is that it can accommodate all context environment variables deemed critical to a SP's service behavior.

We analyzed convergence, accuracy and resiliency properties of CATrust and validated the theory via simulation. We conducted sensitivity analysis of CATrust performance with respect to key design parameters. We also conducted a comparative analysis of CATrust with the Beta reputation scheme with belief discounting [12] and Adaptive Trust Management with collaborative filtering [3]. Our results validated by simulation demonstrate that CATrust outperforms these existing approaches in both the missing good service and missing bad service probabilities. Finally, we discussed applicability of CATrust in terms of computational feasibility, dealing with conflicting behavior attacks and random attacks, and performance characteristics of CATrust implemented with the linear model vs. the non-linear model.

For future work, we plan to further validate CATrust with real-world data such as geo-distributed services data collected using PlaneLab. We also plan to apply CATrust to user-centric social P2P/IoT applications characterized with various application-specific QoS and social context environment variables to further demonstrate its utility. Lastly, we plan to further test the resiliency of CATrust against more complicated environmental and operational scenarios such as noisy environments and application-specific mobility patterns, as well as more sophisticated attack behaviors such as opportunistic, collusion, and insidious attacks [19].

ACKNOWLEDGMENT

This work is supported in part by the U. S. Army Research Laboratory and the U. S. Army Research Office under contract number W911NF-12-1-0445. This research was also partially supported by the Department of Defense (DoD) through the office of the Assistant Secretary of Defense for Research and Engineering (ASD (R&E)). The views and opinions of the author(s) do not reflect those of the DoD or ASD (R&E).

REFERENCES

[1] V. Balakrishnan, V. Varadharajan, and U. Tupakula, "Subjective Logic Based Trust Model for Mobile Ad hoc Networks," in *The 4th International Conference on Security and Privacy in Communication Networks*, 2008, pp. 1-11.

[2] E. Borgia, "The Internet of Things Vision: Key Features, Applications and Open Issues," *Computer Communications*, vol. 54, pp. 1-31, December 2014.

[3] 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*, 2016, in press.

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

[5] 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," in *Int'l. Conf. Computational Science and Engineering*, 2009, pp. 641-650.

[6] 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, pp. 1001-1012, May 2012.

[7] M.T. El-Melegy, M.H. Essai, and A.A. Ali, "Robust Training of Artificial Feedforward Neural Networks," in *Fondations of Computational Intelligence Volume 1*, A.-E. Hassanien et al., Eds.: Springer Berlin Heidelberg, 2009, ch. 9, pp. 217-242.

[8] S. Fruhwirth-Schnatter and R. Fruhwirth, "Bayesian Inference in the Multinomial Logit Model," *Austrian Journal of Statistics*, vol. 41, no. 1, pp. 27-43, 2012.

[9] H. Gao et al., "A Survey of Incentive Mechanisms for Participatory Sensing," *IEEE Communications Survey & Tutorials*, vol. 17, no. 2, pp. 918-943, 2015.

[10] D.B. Johnson, D.A. Maltz, and J. Broch, "DSR: The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad Hoc Networks," in *Ad Hoc Networking*, C.E. Perkins, Ed.: Addison-Wesley, 2001, ch. 5, pp. 139-172.

[11] A. Jøsang, "A Logic for Uncertain Probabilities," *International Journal of Uncertainty, Fuzziness and Knowledge-based Systems*, vol. 9, no. 3, pp. 279-311, 2001.

[12] A. Jøsang and R. Ismail, "The Beta Reputation System," in *15th Bled Electronic Commerce Conf.*, 2002, pp. 1-14.

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

[14] S.D. Kamvar, M.T. Schlosser, and H. Garcia-Molina, "The Eigentrust Algorithm for Reputation Management in P2P Networks," in *2003 World Wide Web Conf.*, 2003.

[15] Z. Li, X. Li, V. Narasimhan, A Nayak, and I Stojmenovic, "Autoregression Models for Trust Management in Wireless Ad Hoc Networks," in *IEEE Global Telecommunications Conf.*, 2011, pp. 1-5.

[16] C. Liu, "Robit Regression: A Simple Robust Alternative to Logistic and Probit Regression," in *Applied Bayesian Modeling and Causal Inference*, A. Gelman and X. L. Meng, Eds. London: Wiley, 2004, ch. 21.

[17] C.H. Liu, J. Fan, P. Hui, J. Wu, and K.K. Leung, "Towards QoI and Energy-Efficiency in Participatory Crowdsourcing," *IEEE Trans. Vehicular Technology*, vol. 64, no. 10, pp. 2684-4700, 2015.

[18] M. Mathew and N. Weng, "Quality of Information and Energy Efficiency Optimization for Sensor Networks via Adaptive Sensing and Transmitting," *IEEE Sensors Journal*, vol. 14, no. 2, pp. 341-348, 2014.

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

[20] 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, pp. 1-11, 2014.

[21] V. Roy, "Convergence Rates for MCMC Algorithms for a Robust Bayesian Binary Regression Model," *Electronic Journal of Statistics*, vol. 6, pp. 2463-2485, 2012.

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

[23] D.J. Thornley, R. Young, and J. Richardson, "From Mission Specification to Quality of Information Measures Closing the Loop in Military Sensor Networks," in *Annual Conf. Int'l Technology Alliance*, 2008, pp. 1-2.

[24] A. Twigg, "A Subjective Approach to Routing in P2P and Ad Hoc Networks," in *Trust Management*, Paddy Nixon and Sotirios Terzis, Eds.: Springer Berlin Heidelberg, 2003, pp. 225-238.

[25] R. Venkataraman, M. Pushpalatha, and T. Rama Rao, "Regression-based trust model for mobile ad hoc networks," *Information Security, IET*, vol. 6, no. 3, pp. 131-140, September 2012.

[26] Y. Wang, I.R. Chen, J.H. Cho, A. Swami, and K. Chan, "Trust-based Service Composition and Binding with Multiple Objective Optimization in Service-Oriented Ad Hoc Networks," *IEEE Transactions on Services Computing*, 2016, in press.

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

[28] X. Wang et al., "Trust and Independence Aware Decision Fusion in Distributed Networks," in *2013 IEEE Int'l. Conf. Pervasive Computing and Communication Workshops*, 2013, pp. 481-486.

[29] J.L. Wang and S.P. Huang, "Fuzzy Logic Based Reputation System for Mobile Ad Hoc Networks," in *Knowledge-Based Intelligent Information and Engineering Systems*, B. Apolloni, R.J. Howlett, and L. Jain, Eds.: Springer Berlin Heidelberg, 2007, vol. 4693, pp. 1315-1322.

[30] Y. Wang et al., "LogitTrust: A Logit Regression-based Trust Model for Mobile Ad Hoc Networks," in *PASSAT*, 2014.

[31] "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," *IEEE Standard 802.11*, Jun. 1999.

[32] H. Xia, Z. Jia, L. Ju, and Y. Zhu, "Trust Management Model for Mobile Ad Hoc Network Based on Analytic Hierarchy Process and Fuzzy Theory," *IET Wireless Sensor Systems*, vol. 1, no. 4, pp. 248-266, 2011.

[33] L. Xiong and L. Liu, "PeerTrust: Supporting Reputation-Based Trust for Peer-to-Peer Electronic Communities," *IEEE Trans. Knowledge and Data Engineering*, vol. 16, no. 7, pp. 843-857, 2004.

[34] Y. Yu, L. Guo, X. Wang, and C. Liu, "Routing security scheme based on reputation evaluation in hierarchical ad hoc networks," *Computer Network*, vol. 54, no. 9, pp. 1460-1469, 2010.

[35] V.A. Zeithaml, "Consumer Perceptions of Price, Quality, and Value: A Means-End Model and Synthesis of Evidence," *Journal of Marketing*, vol. 52, pp. 2-22, 1988.

AUTHOR BIOGRAPHIES

Yating Wang received her Bachelor degree from Hubei University of Technology, Wuhan, China in 2007. She received her PhD degree in Computer Science from Virginia Tech in 2016. Her research interests include security, computer networks, wireless networks, mobile computing, trust management, and reliability and performance analysis.

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

Jin-Hee Cho received the BA from the Ewha Womans University, Seoul, Korea and the MS and PhD degrees in computer science from the Virginia Tech. She is currently a computer scientist at the U.S. Army Research Laboratory, Adelphi, Maryland. Her research interests include network security, trust and risk management, cognitive modeling, and network science. She received the best paper awards in IEEE TrustCom09 and BRIMS13. She is a recipient of the IEEE Communications Society William R. Bennett Prize in the field of Communications Networking, and a receipient of the Presidential Early Career Awards for Scientists and Engineers.

Ananthram Swami is with the U.S. Army Research Laboratory (ARL) as the Army's ST (Senior Research Scientist) for Network Science. He is an ARL Fellow and Fellow of the IEEE. He has held positions with Unocal Corporation, the University of Southern California (USC), CS-3 and Malgudi Systems. He was a Statistical Consultant to the California Lottery, developed a MATLAB-based toolbox for non-Gaussian signal processing, and has held visiting faculty positions at INP, Toulouse, and Imperial College, London. He received the B.Tech. degree from IIT-Bombay, the M.S. degree from Rice University, and the Ph.D. degree from USC, all in Electrical Engineering. His research interests are in the broad area of network science with applications in composite tactical networks.

Yen-Cheng Lu is a Ph.D. candidate in the Computer Science Department at Virginia Tech. He received his B.S. in Applied Mathematics from National Sun Yat-Sen University, Taiwan in 2008. His research interests are in the areas of statistical machine learning and data mining, especially in outlier detection, spatio-temporal analysis, text mining, and transportation applications.

Chang-Tien Lu received the MS degree in Computer Science from Georgia Tech in 1996 and the PhD degree in Computer Science from the University of Minnesota in 2001. He is a professor in the Department of Computer Science, Virginia Tech. His research interests include spatial databases, data mining, geographic information systems, and intelligent transportation systems. Dr. Lu served as the Vice Chair of the ACM Special Interest Group on Spatial Information (ACM SIGSPATIAL) from 2011 to 2014. Dr. Lu serves as an editor for *ACM Transactions on Spatial Algorithms and Systems*. He is a member of IEEE.

Jeffrey J.P. Tsai received a Ph.D. degree in Computer Science from the Northwestern University, Evanston, Illinois. He is the President of Asia University, Taiwan, and a professor in the Department of Bioinformatics and Biomedical Engineering at Asia University. Dr. Tsai was a Professor of Computer Science at the University of Illinois, Chicago. His current research interests include bioinformatics, ubiquitous computing, services computing, intrusion detection, knowledge-based software engineering, formal modeling and verification, distributed real-time systems, and intelligent agents. He was an Associate Editor of the *IEEE Transactions on Knowledge and Data Engineering* and is currently an Associate Editor of the *IEEE Transactions on Services Computing*. He is currently the Co-Editor-in-Chief of the *International Journal on Artificial Intelligence Tools* and *Book Series on Health Informatics*. Dr. Tsai received an IEEE Technical Achievement Award and an IEEE Meritorious Service Award from IEEE Computer Society. He is a Fellow of the AAAS, IEEE, and SDPS.