LogitTrust: A Logit Regression-based Trust Model for Mobile Ad Hoc Networks

Yating Wang[1], Yen-Cheng Lu[1], Ing-Ray Chen[1], Jin-Hee Cho[2], Ananthram Swami[2], Chang-Tien Lu[1]
[1]Computer Science Department, Virginia Tech
[2]US Army Laboratory
{yatingw,kevinlu,irchen,ctlu}@vt.edu, {jinhee.cho, ananthram.swami}@us.army.mil

Abstract

Mobile ad hoc networks (MANETs) have been utilized to execute many applications in diverse environments. Trust is an effective mechanism to cope with misbehaving nodes. However, implementing trust in MANETs is confronted by several obstacles, i.e., no centralized authority, dynamic environments, and limited observations which hinder trust accuracy. In this work, we propose a novel logit regression-based trust model called LogitTrust to model dynamic trust for service-oriented MANETs wherein a node can be a service requester (SR) or a service provider (SP). The novelty of our design lies in the use of logit regression to dynamically estimate trust of SPs based on their distinct behavior patterns in response to environment changes. We demonstrate that LogitTrust outperforms traditional approaches based on Bayesian Inference with belief discounting in terms of trust accuracy and resiliency against attacks, when given the same amount of limited observations, while maintaining a low false positive rate.

1 Introduction

With the proliferation of fairly powerful mobile devices and ubiquitous wireless technology, traditional mobile ad hoc networks (MANETs) now migrate into a new era wherein a node can provide and receive service from other nodes it encounters and interacts with. This paper studies trust model for MANETs consisting of service providers (SPs) that provide services, and service requesters (SRs) that request services.

Trust is an effective mechanism to cope with misbehaving SPs. A node can assess the trust levels of the SPs it interacts with, and propagate its observations of service performance as recommendations to other nodes, so that a well-behaving SP is more likely to be selected to provide services. Govindan and Mohapatra [8] summarized trust in MANET scenarios as a subjective assessment towards another node with respect to its reliability and accuracy of information under specific context. Consequently, trust indicates a SR's (or a trustor's) belief/confidence/expectations on a SP's (or a trustee's) future endeavor in terms of honesty, integrity, ability, availability and quality of service (QoS) representing a trustee's trustworthiness.

One challenge for implementing trust management in MANETs is to reliably estimate the trust levels of participants in a fully distributed manner, in contrast with an e-commerce system with a centralized authority for trust management. In most existing works, e.g., [3, 24], each node observes direct evidence for direct trust assessment and propagates its observations to other nodes as recommendations for indirect trust assessment. However, a malicious node may violate this protocol. Sun *et al.* [20] described several attacks to trust management in distributed networks, including bad-mouthing, on-off, conflicting behavior, Sybil and newcomer attacks. The conflicting behavior attack above is a dynamic attack since whether a node behaves reliably or not depends on its social relationship with the node it interacts with. Also how to cope with bad-mouthing or ballot-stuffing attacks with limited evidence about a recommender is challenging for MANETs since nodes may not have direct experiences with other nodes.

Another major challenge is that the QoS received by a SR from a SP may significantly deviate from the actual service provided by the SP due to node mobility and resource constraints (e.g., bandwidth, processing power, battery) in MANET environments. This blurs the SR's view on the ground truth of the SP. For supporting QoS, MANETs may adopt IEEE 802.11 [9] in distributed coordination function (DCF) mode, so it is likely that the waiting time to access channel may cause a significant delay. In the literature [1, 12, 27], this issue is tackled by breaking QoS into multiple trust components, each being assessed separately, and then these trust components are integrated together via trust formation into an overall trust value. However, the way to select and form multiple trust components into a single trust value is often devised in an ad hoc manner and remains an open issue.

In this work, we take a different approach. We define trust as the probability that a SP will provide a satisfactory service as expected by a SR. Unlike prior work, we consider a probabilistic statistical classification trust model. The novelty lies in the use of logit regression to accurately predict how a SP will behave in response to operational and environmental changes. This allows us to reason a node's behavior pattern, given the operational and environmental conditions as input. We name our proposed scheme as LogitTrust. To be specific, this work has the following contributions:

- To the best of our knowledge, we are the first to propose a logit regression-based trust model to estimate dynamic trust. Our model has its root in regression analysis. Thus, it provides a sound theoretical basis to predict the behavior of a SP in response to operational and environmental changes.

- We demonstrate the solution accuracy of our computational trust model over traditional trust models, especially in highly dynamic MANET environments. We also demonstrate the computational feasibility of running LogitTrust on MANET nodes without sacrificing solution accuracy.
- LogitTrust is highly resilient toward recommendation attacks including ballot-stuffing and bad-mouthing attacks. We demonstrate that LogitTrust significantly outperforms Bayesian inference with belief discounting [10, 19] in terms of prediction accuracy rate and false positive rate.

This paper is organized as follows: In Section 2, we survey trust models in MANETs and trust-based defenses against attacks. We contrast and compare existing approaches with our approach. In Section 3, we discuss the system model including dynamic trust and behavior pattern factors, attacker behaviors, and the problem definition. In Section 4, we give an overview of LogitTrust. In Section 5, we illustrate the utility of LogitTrust by a trust-based SP selection case study. In Section 6, we conduct an extensive simulation experimentation to validate the accuracy and robustness of our model. We discuss the computational feasibility in Section 7. Lastly in Section 8 we conclude the paper and outline future research directions.

2 State of the Art

Two widespread trust models for inferring trust in MANETs are Bayesian inference and fuzzy logic.

Bayesian inference applies Bayes theorem and treats trust as a random variable following a probability distribution (e.g., Beta distribution) with its value being updated upon new observations. Yu et al. [28] 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. Sun et al. [21] utilized Bayesian inference for trust bootstrapping, where a node's performance evidence in each service trial is accumulated. However it is ineffective to capture dynamic trust. Buchegger and Le Boudec [4] adopted a modified version of the Bayesian approach by assigning more weights to current evidence and reducing weights on past evidence, using the mean of posterior distribution to compute trust. Similar approaches were made in several works for modeling dynamic trust in MANETs [5–7] by considering trust decay over time. However, there is no theoretical basis for this approximation.

Fuzzy logic-based approaches have been implemented in reactive secure routing [23, 26] for MANETs. Xia et al. [26] computed trust by fuzzy logic rules, utilizing a directed graph for a trustee node with a set of vertices (nodes) and trust-weighted edges (direct trust links) where the input is the historical trust values toward the trustee node. Wang and Huang [23] applied fuzzy logic to compute a fuzzy value for each candidate path for making a routing decision. The fuzzy value is determined by node reputation, bandwidth, and hop count. One drawback of fuzzy logic-based trust inference 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 inference and fuzzy logic, we take an entirely different approach. We develop a regression-based trust model utilizing logit regression to estimate dynamic trust. To the best of our knowledge, only [13, 22] leveraged regression for trust computation in MANETs. Li et al. [13] proposed an autoregression-based trust management technique to learn the weights for history observations so as to predict the future outcome. Venkataraman et al. [22] developed a regression-based trust model for MANETs with the objective to learn the weights optimally combining several trust metrics, where each trust metric is assessed separately using Bayesian inference. Unlike [13, 22], we do not use regression to learn weights to observations or trust properties. Instead, we apply logit regression analysis to learn the behavior patterns of a node in response to operational and environmental changes. We aim to provide a generic trust model to predict a node's dynamic behavior patterns and accordingly a node's dynamic trust.

A significant amount of work has been done in the area of trust-based defenses against attacks in MANETs [5–7, 25]. In particular, Chen et al. [5] 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. However, these methods involve tuning trust parameters and may perform poorly when a node does not have enough experiences with the recommenders. Different from the work above [5–7, 25], LogitTrust does not apply trust thresholds to filter trust recommendations. Instead, it leverages a robust statistical kernel to tolerate outlier recommendations to effectively achieve resiliency against recommendation attacks. We demonstrate that our regression-based trust model significantly outperforms Bayesian inference with belief discounting especially when the direct observation towards the recommenders is limited.

3 System Model

3.1 Dynamic Trust and Behavior Pattern Factors

We consider the notion of dynamic trust, i.e., a SP's trust level changes dynamically as the MANET operational and environmental conditions change dynamically due to node mobility, changes in traffic patterns, and limited resources. Trust is dynamic because SPs are heterogeneous in terms of attitude and adaptability to operational and environmental changes. A SP may be profit-seeking, so when a SR pays a higher than basic pricing for the service, it may be more willing to apply available resources to execute the task. A SP may be limited in resources and task execution capability such that its service waiting list may be relatively long and its service delivery speed is largely deterred. Also a SP may be sensitive to the number of neighbors because more neighbors might increase the probability of wireless channel contention and signal interference, so it needs to consume more energy in listening to the channel and repeating packet transmission. Without loss of generality, we consider *energy-sensitivity*, *capability-limitation*, and *profit-awareness* as three distinct "behavior pattern factors" reflecting the extent to which a SP would behave in response to changes in MANET operational and environmental conditions.

3.2 Attacker Model

As every node in a MANET 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. We consider social selfishness to model malicious behavior in this work. Every node may exhibit social selfishness based on the social relationship with other nodes it interacts with. Consequently, every node may exhibit the following recommendation attack behaviors:

1. *Trustor-based recommendation attacks (TORA):* A node serving as a recommender provides false recommendations about a trustee. The objective is to prevent the trustor from learning the right behavior pattern and hence decrease the trustor's decision quality. Based on the notion of social selfishness, if the trustor is a friend, then it tends to speak the truth about a trustee. On the other hand, if the trustor is not a friend, then it tends to lie. A node may treat another node as a friend, acquaintance or stranger. We model a recommender's TORA behavior by a probability function $p^{TORA}(F)$ where F specifies the friendship relation between the recommender and the trustor.

2. *Trustee-based recommendation attacks (TERA):* A node serving as a recommender may perform reputation attacks on the trustee. Based on the notion of social selfishness, if the trustee node is not a friend, it tends to perform bad-mouthing attack to diminish the trustee node's reputation. On the other hand, if the trustee node is a friend then it tends to promote the trustee node by performing ballot-stuffing attacks. We model a recommender's bad-mouthing attack behavior by a probability function $p^{TERA}_{bma}(F)$ where F specifies the friendship relation between the recommender and the trustee. On the other hand, We model a recommender's ballot-stuffing attack behavior by a probability function $p^{TERA}_{bsa}(F)$ where F specifies the friendship relation between the recommender and the trustee.

3. *TERA-if-TORA:* A node serving as a recommender first decides whether to perform false recommendation attack based on its relationship with the trustor node. If yes, it performs bad-mouthing or ballot-stuffing attack based on its relationship with the trustee node.

3.3 Problem Definition

The problem at hand is for SR i to predict whether SP j will perform satisfactorily or not for a specific requested service, given a history of evidence. The objective is to achieve prediction accuracy and resiliency against recommendation attacks described in Section 3.2.

Within a specific type of service, assuming SR i's observation s^t_{ij} at time t of the service quality received from SP j is either "satisfactory" or "unsatisfactory", we define that if the QoS is satisfactory, then SP j is considered *trustworthy* denoted as 1; otherwise, it is *untrustworthy* denoted as 0. In other words, a SP is considered *trustworthy* at time t if it can be observed that the service provided at time t is satisfactory. Let the operational and environmental conditions at time t be characterized by three distinct "behavior pattern factors" (*energy-sensitivity*, *capability-limitation*, and *profit-awareness* as discussed in Section 3.1) denoted by a column vector $\mathbf{x}^t = [x^t_e, x^t_c, x^t_p]^\top$. Then, *trust* is the probability θ^t_j that SP j is capable of providing satisfactory service to SR i under the operational and environmental conditions at time t described by $\mathbf{x}^t$.

Let $s^t_j = s^t_{ij} \cup \{s^t_{kj}, k \neq i\}$ where k is a recommender who had a 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 $\{\mathbf{x}^t,\ s^t_{kj}\}$ specifying the operational and environmental conditions at time t ($\mathbf{x}^t$) under which the observation (s^t_{kj}) is made. Further, let $s_j = \{s^t_j, t = 1, \ldots, T\}$ denote the set of evidence gathered by SR i, including self observations and recommendations over $[0,\ T]$. Also let $\mathbf{x} = \{\mathbf{x}^t, t = 1, \ldots, T\}$, denote the corresponding operational and environmental conditions over $[0,\ T]$. LogitTrust solves this problem by learning the behavior pattern of SP j represented by a latent variable $\boldsymbol{\beta}_j$ between s_j and $\mathbf{x}$, and predicting s^{T+1}_j given $\mathbf{x}^{T+1}$, i.e., $E[s^{T+1}_j|\mathbf{x}^{t+1}]$. This conditional expectation will be real-valued, between 0 and 1, representing the trust level of SP j at time $T+1$ from SR i's perspective.

4 LogitTrust

4.1 Design Principle

The idea behind LogitTrust is to utilize logistic regression [14] to analyze the relation between regressor variables $\mathbf{x}$ and binary response observations s_j described earlier in Section 3.3. LogitTrust is flexible in accommodating any environment-specific behavior pattern factor (x_i) deemed appropriate for an application. LogitTrust uses a linear predictor as in [16], namely,

$$E[s^t_j|\mathbf{x}^t, \boldsymbol{\beta}_j] = \theta^t_j = \frac{1}{1 + e^{-(\mathbf{x}^t)^\top \boldsymbol{\beta}_j}} \tag{1}$$

where $\boldsymbol{\beta}_j = [\beta_{ej}, \beta_{cj}, \beta_{pj}]^\top$ is a column vector of coefficients and $\theta(\mathbf{x}^t)$ is in the range of $[0, 1]$. If $(\mathbf{x}^t)^\top \boldsymbol{\beta}_j \ll 0$, θ^t_j is less than 0.5. Therefore the delivered QoS is more likely to be *unsatisfactory*; otherwise, if $(\mathbf{x}^t)^\top \boldsymbol{\beta}_j \gg 0$, the QoS is more likely to be *satisfactory*. Following Eq. 1, we have

$$\ln\left(\frac{\theta_j^t}{1-\theta_j^t}\right) = (\mathbf{x}^t)^\top \boldsymbol{\beta}_j, \text{ or } \mathrm{logit}(\theta_j^t) = (\mathbf{x}^t)^\top \boldsymbol{\beta}_j \qquad (2)$$

Proposition 1 *The probability that the service provided by SP j to SR i is satisfactory is equivalent to the probability that s_j^{t*} is larger than zero, where*

$$s_j^{t*} = logit(\theta_j^t) + \varepsilon_j, \qquad (3)$$

and $\varepsilon_j \sim logistic(0,1)$ with cumulative density function $\frac{1}{1+e^{-x}}$, $x \in (-\infty, \infty)$.

Proof.

$$\begin{aligned} & \Pr(s_j^{t*} > 0|\mathbf{x}^t, \boldsymbol{\beta}_j) \\ = \; & \Pr(\mathrm{logit}(\theta_j^t) + \varepsilon_j > 0|\mathbf{x}^t, \boldsymbol{\beta}_j) \\ = \; & \Pr(\varepsilon_j > -\mathrm{logit}(\theta_j^t)|\mathbf{x}^t, \boldsymbol{\beta}_j) \\ = \; & \Pr(\varepsilon_j > -(\mathbf{x}^t)^\top \boldsymbol{\beta}_j) \\ = \; & \Pr(\varepsilon_j < (\mathbf{x}^t)^\top \boldsymbol{\beta}_j) \\ = \; & \frac{1}{1+e^{-(\mathbf{x}^t)^\top \boldsymbol{\beta}_j}} \\ = \; & \theta_j^t \end{aligned}$$

■

Here we note that the observation history $\{\mathbf{x},\ s_j\}$ is a collection of self observations of SR i and recommendations provided by recommenders $k \neq i$ upon encountering in MANET environments.

4.2 Dealing with Recommendation Attacks as Outliers

LogitTrust can subsume malicious recommenders who intentionally flip the actual observations. That is, a malicious recommender k will flip its recommendation in the form of $\{\mathbf{x}^t,\ 1 - s_{kj}^t\}$ when it performs a recommendation attack. We call such modified recommendations as *outliers* which can mislead the inferred $\boldsymbol{\beta}_j$ to deviate from the true behavior. If we apply a logistic distribution with thin tails for the error term, the solution accuracy is likely to be sensitive to outliers, so recommender attacks can impact the prediction accuracy. To tolerate outliers without overly sacrificing solution accuracy, we replace the latent error in logistic distribution in Eq. 3 with a white noise in t-distribution [14] whose mean is zero and degree of freedom is ν with a scale parameter σ. The t-distribution has heavier tails when ν is finite and it is more robust to outliers, since its heavy-tail characteristic increases the probability of samples occurring at a point far away from the mean, and, subsequently, increases the ability to absorb errors and protect the estimation process for $\boldsymbol{\beta}_j$.

Replacing ε_j by a standard t-distribution random variable and denoting $(\mathbf{x}^t)^\top \boldsymbol{\beta}_j$ as u_j^t, given the optimal degree of freedom (ν_0) for tolerating outliers, we have $s_j^{t*} \sim t(u_j^t, 1, \nu_0)$. We then apply the Expectation Maximization (EM) algorithm combined with iterative re-weighted least-squares to estimate $\boldsymbol{\beta}_j$ as in [14]. The relation between the t-distribution and the normal distribution is that a t-distribution can be approximated by an infinite sum of normal distributions each having a different variance [2]. Thus, we have the following priors in terms of the weight w^t for each $s_j^{t*}, i = 1, \ldots, I$:

$$\begin{aligned} s_j^{t*}|(w^t, \beta_j) &\sim \mathcal{N}(u_j^t, \frac{1}{w^t}) \\ w^t|\beta_j &\sim \Gamma(\nu_0/2, \nu_0/2) \end{aligned}$$

In the *E-step* of EM, we compute the expectation of $w^t, t = 1, \ldots, T$ with current $\boldsymbol{\beta}_j$. In the *M-step* of EM, we compute a new $\boldsymbol{\beta}_j$ that achieves the maximum likelihood. Finally, we compute $E[s_j^{T+1}|\mathbf{x}^{T+1}, \boldsymbol{\beta}_j]$.

Algorithm 1 LogitTrust

```
Input: $\mathbf{x}$, $s_j$, $\nu_0$, $\mathbf{x}^{T+1}$
Output: $E[s_j^{T+1}|\mathbf{x}^{T+1}, \boldsymbol{\beta}_j]$
1:  $k \leftarrow 0$
2:  $\hat{\boldsymbol{\beta}}_j^{(k)} \leftarrow \mathbf{1}$
3:  while not converged do
4:      for $t \leftarrow 1$ to $T$ do
5:          $u^t \leftarrow (\mathbf{x}^t)^\top \hat{\boldsymbol{\beta}}_j^{(k)}$
6:          $\hat{w}^t \leftarrow \frac{s_j^t-(2s_j^t-1)F_{\nu_0+2}(-(1+2/\nu_0)^{1/2}u^t)}{s_j^t-(2s_j^t-1)F_{\nu_0}(-u^t)}$
7:          $\hat{s}_j^{t*} \leftarrow u^t + \frac{(2s_j^t-1)f_{\nu_0}(u^t)}{s_j^t-(2s_j^t-1)F_{\nu_0+2}(-(1+2/\nu_0)^{1/2}u^t)}$
8:      end for
9:      $S_0 \leftarrow \sum_{t=1}^T \hat{w}^t \mathbf{x}^t (\mathbf{x}^t)^\top$
10:     $S_1 \leftarrow \sum_{t=1}^T \hat{w}^t \mathbf{x}^t \hat{s}_j^{t*}$
11:     $k \leftarrow k+1$
12:     $\hat{\boldsymbol{\beta}}_j^{(k)} \leftarrow S_0^{-1} S_1$
13: end while
14: return $E(s_j^{T+1}|\mathbf{x}^{T+1}, \hat{\boldsymbol{\beta}}_j) \leftarrow \frac{1}{1+exp(-(\mathbf{x}^{T+1})^\top \hat{\boldsymbol{\beta}}_j^{(k)})}$
```

4.3 Computational Procedure

Algorithm 1 LogitTrust above specifies the input requirement, including $\mathbf{x}$, s_j (explained above), ν_0 (the degree of freedom) and $\mathbf{x}^{T+1}$ (the behavior pattern factors exhibited by SP j at time $T+1$). The output is $E(s_j^{T+1}|\mathbf{x}^{T+1}, \hat{\boldsymbol{\beta}}_j)$, i.e., the trust level of SP j at time $T+1$. Lines (1-2) specify the initialization step where k is the iteration index and $\mathbf{1}$ means an all-one column vector. Lines (3-13) execute the EM algorithm to infer $\hat{\boldsymbol{\beta}}_j$. Lines (4-10) are for the *E-step* and lines (11-12) are for the *M-step*. In the *E-step*, given the current $\hat{\boldsymbol{\beta}}_j$, we compute the conditional expectation of each record's weight (line 6) and infer the latent variable s_j^{t*} that determines the observed value (line 7), where $F_\nu(x)$ is the cumulative density of a standard t-distributed random variable x given the number of degrees of freedom ν and $f_\nu(x)$ is its probability density. Parameter w^t scales down the impact of those records with a high variance in the estimation process. This is particularly useful in scenarios in which we have no evidence on the reliability of the recommender who might perform recommendation attacks, or we have an outlier due to unobserved effect. S_0 and S_1 are two terms for partial log-likelihood, i.e., $l(\boldsymbol{\beta}_j|\mathbf{x}, s_j^*, w, \nu_0)$.

In the *E-step*, the log-likelihood function is updated and consequently the estimate of $\hat{\boldsymbol{\beta}}_j$ is renewed by maximum likelihood estimation (line 12). Finally line 14 is the trust prediction step that applies Eq. 1 to compute the trust level of SP j at time $T+1$.

4.4 Service History

In LogitTrust the service history toward SP j provided by a recommender k is in the form of $\{\mathbf{x}^t, s^t_{kj}\} = \{[x^t_e, x^t_c, x^t_p]^\top, s^t_{kj}\}$. This is obtained when node k itself serves as a SR and observes the service quality of SP j as the self observation part of the service history maintained by node k. If SR k is satisfied with the service quality provided by SP j, $s^t_{kj} = 1$; otherwise, $s^t_{kj} = 0$. In addition, SR k also records down the values of x^t_e, x^t_c and x^t_p as follows: x^t_e is estimated by the 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^t_c is estimated by the packet traffic to SP j as more traffic to SP j hinders its processing capability; x^t_p is simply the negotiated price for the service provided by SP j. SR k then records $\{[x^t_e, x^t_c, x^t_p]^\top, s^t_{kj}\}$ as part of its service history toward SP j. Upon request from SR i, node k will provide it to SR i as a recommendation for SP j.

5 Case Study

In this section we illustrate the applicability of LogitTrust with a trust-based SP selection case study. SR i with a service request in hand encounters SP j who claims it can provide service, so SR i wants to know whether it should select SP j for service. SR i uses the predicted trust level of SP j obtained from executing LogitTrust based on the service history of SP j collected prior to the encounter time, as well as the operational and environmental conditions at the encounter time to make its decision as follows: if the trust level of SP j is less than 0.5, SR i rejects SP j. Otherwise, SR i selects SP j for service following a Bernoulli trial with the trust level as the probability of success. For ease of comparison, below we first discuss the solution based on Beta reputation with belief discounting [10, 19]. Then we discuss the solution based on LogitTrust.

5.1 Solution based on Beta Reputation with Belief Discounting

Let $s^+_{ij} = \mathbf{card}(\{s^t_{ij} = 1, t = 1, \ldots T\})$ and $s^-_{ij} = \mathbf{card}(\{s^t_{ij} = 0, t = 1, \ldots T\})$ be the numbers of satisfactory and unsatisfactory services, respectively, received from SP j by SR i (self experiences) over [0, T]. The service history of SP j provided by a recommender k to SR i upon request is in the form of (s^+_{kj}, s^-_{kj}). When SR i receives a recommendation toward SP j from recommender k, it applies belief discounting to merge the recommendation (s^+_{kj}, s^-_{kj}) with its own evaluation (s^+_{ij}, s^-_{ij}) such that the lesser SR i trusts the recommender, the more the recommendation is discounted. For more details on belief discounting, we refer the readers to [10, 19]. After merging self observations with recommendations, let $s^+_j = \mathbf{card}(\{s^t_j = 1, t = 1, \ldots T\})$ and $s^-_j = \mathbf{card}(\{s^t_j = 0, t = 1, \ldots T\})$ be the numbers of satisfactory and unsatisfactory services, respectively, by SP j over [0, T], covering both self observations and recommendations. Then, the average success rate of SP j can be calculated as:

$$r_{succ} = \frac{s^+_j}{s^+_j + s^-_j} \tag{4}$$

which is the mean of a beta distribution with parameters s^+_j and s^-_j, representing the trust level of SP j.

5.2 Solution based on LogitTrust

SR i utilizes the service history of SP j in the form of $(\mathbf{x}, s_j)$ consisting of SR i's own observations and recommendations received prior to the current time $T+1$, and the current operational and environmental conditions in the form of $\mathbf{x}^{T+1}$ and executes LogitTrust ("Algorithm 1: LogitTrust") to infer the trust level of SP j at time $T+1$.

6 Performance Evaluation

In this section we conduct a performance evaluation of LogitTrust in trust accuracy and resilience against attacks. In Section 6.1 we discuss experimental settings, including the environment setup, synthetic data preparation, attack implementation, and performance metrics. Then in Section 6.2 we do a comparative analysis of LogitTrust against Bayesian inference using Jøsang's Beta reputation system [10] as a baseline system for the case in which there is no malicious attack. Finally in Section 6.3 we conduct a comparative analysis of LogitTrust against Beta reputation incorporating belief discounting [19] for the case in which there are malicious attacks. Our comparative performance analysis is fair. All schemes are given the same amount of observations and recommendations toward a trustee via service history information sharing as input.

6.1 Experimental Settings

6.1.1 Environment Setup

To model social selfishness behavior, we use human-activity data based on tracing 76 users in 5 days from a MANET application called *MobiClique* [17], where each mobile user carries a Bluetooth device with 200MHz T1 processor, 65MB RAM and 128MB ROM MicroSD, and the radio range is between 10 and 20 meters. A user chooses a SP for service from among its one-hop neighbors. Upon encountering, the service history sharing is performed as described in Section 4.4. We retrieve three data sets: *TransmissionDB* (originally called *transmission* in [17]), *FacebookFriendDB* (*friend1*), and *MobiCliqueFriendDB* (*friend2*). The first data set contains data transmission logs of pairwise communications among the 76 users upon encountering in the MANET environment. The last two data sets record the pair-wise friendship relationships among the 76 users in Facebook before the application is played (representing friends), and in MobiClique after the application is played (representing acquaintances). On average, each user has about 5 to 10 friends and acquaintances.

6.1.2 Synthetic Data Preparation

Our subject node is a SP (called SP j) that has n_{hist}=204 interactions with 24 distinct SRs asking for its service upon encountering based on *transmissionDB* in *MobiClique*. We first generate j's three behavior pattern factor values $\{x_e^t, x_c^t, x_p^t\}$ at time t for each of the n_{hist}=204 interactions. The "energy sensitivity" behavior pattern factor, x_e, is modeled by considering the channel access delay. We assume that the channel access delay follows a Gaussian distribution with mean $\mu_w = 100ms$ and variance $\sigma_w = 50ms$, depending on the SP's attempt probability whose magnitude is determined by the number of neighbors according to the simulation result from IEEE 802.11 DCF in [18]. The "capability limitation" behavior pattern factor, x_c, is modeled by the queueing delay. We assume that service tasks are to be processed FIFO, follow a Poisson process with rate $\lambda_{arr} = 5$ tasks/sec, and each consumes the same amount of time. The "profit-awareness" behavior pattern factor, x_p, is modeled by SP j's potential gain upon a satisfactory service completion. SP j's potential gain consists of two parts: the asked price P_{std} from SP j computed by a linear function associated with the length of its processing queue, and the overpaid price by the SR with a probability of p_{op} that represents the overpaying incentive. We model the overpaid amount by a normal distribution with the mean and variance being a percentage of P_{std} provided by the SP. The three behavior pattern factor values $\{x_e^t, x_c^t, x_p^t\}$ reflect the environmental and operational state SP j is in while it responds to SR i's request at time t. The ground truth or actual behavior pattern $\boldsymbol{\beta}_j$ of SP j is generated based on the success rate r_{succ} of SP j as defined by Eq. 4, which controls the service quality of SP j, i.e., s_j^+ services are satisfactory and s_j^- services are unsatisfactory. After we generate $\{x_e^t, x_c^t, x_p^t\}$ and $\boldsymbol{\beta}_j$, we generate the ground truth s_{ij}^t at time t by applying Eq. 1. Whether s_{ij}^t is 1 or 0 is decided by the predictor $\boldsymbol{\beta}_j$ confined by the success rate r_{succ}.

SR i ranks the service history records of SP j collected in the order of self, friend, acquaintance and stranger so it places more confidence on self experiences over recommendations from friends, acquaintances, and strangers (in this order). In the experiment, we perform a sensitivity analysis of LogitTrust performance with respect to the amount of recommendations received from friends, acquaintances, and strangers. We set the degree of freedom ν_0=4 for the t-distribution error as in [11].

In the testing phase, a randomly generated $\{x_e^t, x_c^t, x_p^t\}$ is fed to each SR as input, and the SR predicts the trust level of SP j by the inferred $\hat{\boldsymbol{\beta}}_j$. The prediction result from each SR is compared with ground truth to evaluate the prediction accuracy of the SR. The results reported are based on the averages of all SRs over 100 $\{x_e^t, x_c^t, x_p^t\}$ sets randomly generated.

6.1.3 Attack Implementation

We consider three hostility situations: low, medium, and high. We use shorthand notations f, a, and s for friend, acquaintance, and stranger, respectively. The low hostility situation is modeled by ($p^{TORA}(f)$, $p^{TORA}(a)$, $p^{TORA}(s)$) = (0, 0.1, 0.2), ($p_{bma}^{TERA}(f)$, $p_{bma}^{TERA}(a)$, $p_{bma}^{TERA}(s)$) = (0, 0.1, 0.2), and ($p_{bsa}^{TERA}(f)$, $p_{bsa}^{TERA}(a)$, $p_{bsa}^{TERA}(s)$) = (0.2, 0.1, 0.0). The medium hostility situation is modeled by ($p^{TORA}(f)$, $p^{TORA}(a)$, $p^{TORA}(s)$) = (0.1, 0.3, 0.5), ($p_{bma}^{TERA}(f)$, $p_{bma}^{TERA}(a)$, $p_{bma}^{TERA}(s)$) = (0.1, 0.3, 0.5), and ($p_{bsa}^{TERA}(f)$, $p_{bsa}^{TERA}(a)$, $p_{bsa}^{TERA}(s)$) = (0.5, 0.3, 0.1). The high hostility situation is modeled by ($p^{TORA}(f)$, $p^{TORA}(a)$, $p^{TORA}(s)$) = (0.2, 0.6, 1.0), ($p_{bma}^{TERA}(f)$, $p_{bma}^{TERA}(a)$, $p_{bma}^{TERA}(s)$) = (0.2, 0.6, 1.0), and ($p_{bsa}^{TERA}(f)$, $p_{bsa}^{TERA}(a)$, $p_{bsa}^{TERA}(s)$) = (1.0, 0.6, 0.2). Here we note that for TERA, the probability of bad-mouthing attacks is lower for a friend (the trustee) because a socially selfish node normally would not want to ruin the reputation of a friend. On the other hand, the probability of ballot-stuffing attacks is higher for a friend because a socially selfish node normally would want to boost the reputation of a friend.

6.1.4 Performance Metrics

1. *Similarity* is estimated based on the distance between the inferred behavior pattern $\hat{\boldsymbol{\beta}}_j$ and the actual behavior pattern $\boldsymbol{\beta}_j$. It represents the accuracy of behavior learning. We measure similarity by (a) cosine similarity and (b) Euclidean distance (also known as mean square error). The cosine similarity is defined as:

$$\textit{cosine similarity}(\boldsymbol{\beta}_j, \hat{\boldsymbol{\beta}}_j) = \frac{\boldsymbol{\beta}_j \cdot \hat{\boldsymbol{\beta}}_j}{\|\boldsymbol{\beta}_j\| \|\hat{\boldsymbol{\beta}}_j\|} \qquad (5)$$

The more similar the two behavior pattern vectors are, the closer the cosine similarity value is to 1. The Euclidean distance is defined as:

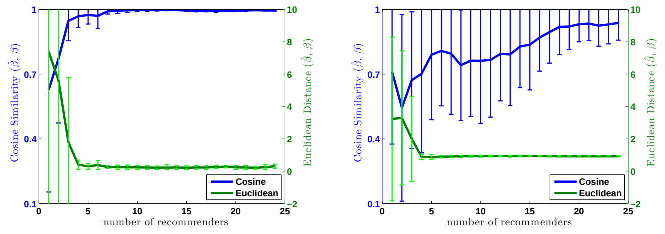
$$\textit{Euclidean distance}(\boldsymbol{\beta}_j, \hat{\boldsymbol{\beta}}_j) = \sqrt{(\boldsymbol{\beta}_j - \hat{\boldsymbol{\beta}}_j)^2} \qquad (6)$$

The more similar the two behavior pattern vectors are, the closer the Euclidean distance is to 0.

2. *Success rate ($\mathcal{S}_{i,j}$)* of service received is defined as the ratio of the number of satisfactory services received from SP j (i.e., s_{ij}^+) over the total number of service purchases by SR i (i.e., $s_{ij}^+ + s_{ij}^-$). It provides the decision accuracy. Therefore, the success rate $\mathcal{S}_{i,j}$ is calculated as:

$$\mathcal{S}_{i,j} = \frac{s_{ij}^+}{s_{ij}^+ + s_{ij}^-} \qquad (7)$$

Here we note that a service request at time t is associated with $\{x_e^t, x_c^t, x_p^t\}$ as input and SR i predicts SP j's trust level as $E[s_j^t | \mathbf{x}^t, \boldsymbol{\beta}_j]$, i.e., the probability that SP j can deliver a satisfactory service quality at time t. Thus, the prediction success rate of SR i may be higher than r_{succ} if SR i can predict SP j's behavior accurately given $\{x_e^t, x_c^t, x_p^t\}$ as input specifying the operational and environmental conditions at time t.



(a) Similarity (r_{succ}=30%) (b) Similarity (r_{succ}=60%)

Figure 1: Similarity between Inferred Behavior Pattern and Ground Truth.

Alternatively, we consider the rate of unsatisfactory service received (or the failure rate $\mathcal{F}_{i,j}$) which is just the complement of the success rate. That is, the failure rate $\mathcal{F}_{i,j}$ for selecting SP j to provide $N_{i,j}$ services is calculated as:

$$\mathcal{F}_{i,j} = \frac{s_{ij}^-}{s_{ij}^+ + s_{ij}^-} \tag{8}$$

The failure rate can be considered as the false negative rate because a bad SP is mistrusted as a good SP.

3. *Bypassing rate* ($\mathcal{B}_{i,j}$) indicates the rate of missing a satisfactory service. Let $m_{i,j}^*$ be the number of services for which SP j is not selected by SR i for service. Let $m_{i,j}^+$ be the number of services among $m_{i,j}^*$ which SP j would have satisfactorily provided if selected. Then the bypassing rate $\mathcal{B}_{i,j}$ for SR i to miss out SP j is calculated as:

$$\mathcal{B}_{i,j} = \frac{m_{i,j}^+}{m_{i,j}^*} \tag{9}$$

The bypassing rate can be considered as the false positive rate because a good SP is missed as a bad SP.

6.2 Comparative Analysis on Trust Accuracy

Fig. 1 shows the trust convergence of the inferred behavior vector by LogitTrust to the actual behavior vector. Figs. 1(a) and 1(b) display both cosine similarity and Euclidean distance as a function of the number of recommenders (no attack behavior) for the two cases of r_{succ}=30% and 60%, respectively, with the mean and error bar marked from all SR's learning results towards SP j's behavior pattern. Both figures display a similar trend that when there are more recommenders (with no attack behavior), the behavior learning result is closer to the ground truth. When the number of recommenders is small, the learning process has larger variances due to the sample space.

Fig. 2 compares the service prediction accuracy between LogitTrust and Beta reputation under the no attack case where the first row is for $r_{succ} = 30\%$ and the second row is for $r_{succ} = 60\%$. Overall, LogitTrust produces a higher and more consistent service prediction accuracy than Beta reputation. LogitTrust consistently predicts a high service success rate over and above r_{succ}, while Beta reputation produces a service rate that stays around r_{succ}. This is because Beta reputation assumes a static hidden mean trust value considering only the numbers of positive and negative observations. Therefore, when a trust instance deviates significantly from the mean, it performs poorly due to overestimating/underestimating the trust by using the mean value. As a result, its success rate just reaches the average service success rate (r_{succ}). We also note that for Beta reputation, the values for the success rate and failure rate are not available when $r_{succ} = 30\%$. This is because when $r_{succ} = 30\%$, the trust value of SP j predicted by Beta reputation is also 0.3. Since SP j's trust value is less than 0.5, a SR following the SP selection rule will never select SP j for service. While this avoids failed service from SP j, it also misses the chances to obtain good service. This trend is shown in Figs. 2(c) and 2(f) where LogitTrust shows a much lower bypassing rate (i.e., a lower false positive rate) than Beta reputation. When $r_{succ} = 30\%$ the bypassing rate under Beta reputation (in Fig. 2(c)) is insensitive to the number of recommenders. This is because the trust value obtained by Beta reputation is close to the SP's average service rate (30%) and is always below 0.5 regardless of the number of recommenders, so the SP is not selected for service every time. As a result, the bypassing rate is the same as the percentage of the time the SP can actually provide satisfactory service, i.e., it is equal to the SP's success rate.

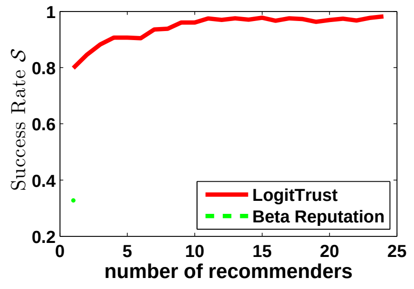
Insight: LogitTrust uses regression to learn a SP's behavior pattern and to predict its dynamic trust, and thus it provides a more accurate trust evaluation than Beta reputation with the same amount of observations. Consequently, LogitTrust improves the decision performance even in a MANET environment in which the average service success rate is low.

6.3 Comparative Analysis on Resilience against Attacks

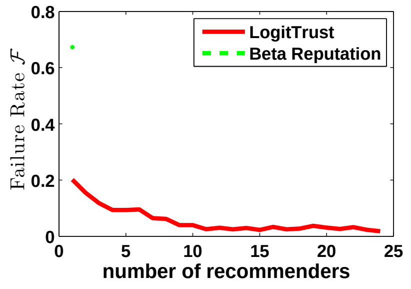
The second set of experiment focuses on the resilience against malicious recommendation attacks. Every node in the system is socially selfish and can perform TORA, TERA or TERA-if-TORA. The baseline scheme for performance comparison is Beta reputation incorporating belief discounting [19] to cope with attacks, given the same amount of observations and recommendations toward a trustee node as input.

Fig. 3 compares LogitTrust against Beta reputation with belief discounting (BD) in terms of the three performance metrics under TORA only (straight lines), TERA only ($*$ lines) and TERA-if-TORA ($+$ lines) for the case in which the average service success rate r_{succ} is 30%. The first, second and third rows of graphs are for low, medium and high hostility environments, respectively. The performance is measured as a function of the number of received recommendations ranked in the order of friend, acquaintance and stranger, i.e., the first few recommendations are from friends, followed by recommendations from acquaintances, and lastly from strangers.

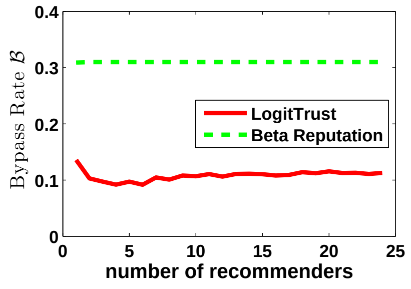
We first observe that under low to medium hostility situations, LogitTrust significantly outperforms Beta reputation in all three performance metrics. In particular, we observe that the protocol performance for both LogitTrust and Beta reputation under the low hostility environment (Figs. 3(a)-



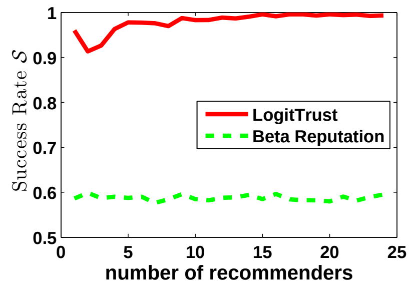
(a) Success Rate (r_{succ}=30%)



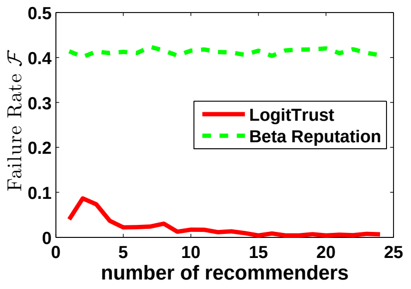
(b) Failure Rate (r_{succ}=30%)



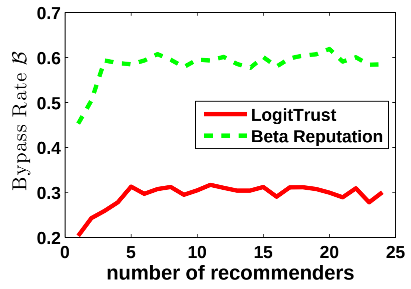
(c) Bypassing Rate (r_{succ}=30%)



(d) Success Rate (r_{succ}=60%)



(e) Failure Rate (r_{succ}=60%)



(f) Bypassing Rate (r_{succ}=60%)

Figure 2: Comparing LogitTrust against Beta Reputation in Service Success Rate, Failure Rate and Bypassing Rate.

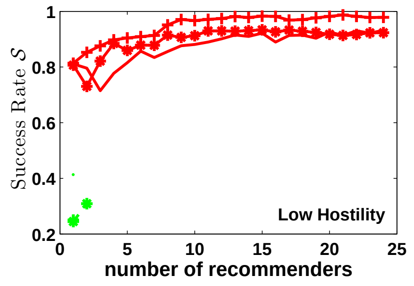
3(c)) is close to that under the no attack environment (Figs. 2(a)-2(c)) as expected for sanity check. In general LogitTrust can cope with TERA-if-TORA compared to TERA and TORA. This is because for TERA-if-TORA the overall attack probability is determined by the attack probability from the recommender to the SR and the attack probability from the recommender to the SP, so the actual probability is significantly discounted.

Under the high hostility environment, however, LogitTrust performs better than Beta reputation only if it filters out recommendations from strangers since in high hostility situations, strangers will perform attacks with probability 1. As we can see from Figs. 3(g)-3(i) when LogitTrust takes in too many false recommendations from strangers past honest recommendations provided from friends and acquaintances such that the majority of observations in the service history of SP j comprises outliers, LogitTrust starts to perform worse since the inferred behavior pattern learned by LogitTrust deviates more and more from the actual behavior. Nevertheless, with the social relationship filtering mechanism in place allowing LogitTrust to take in just 5-10 recommendations from friends and acquaintances, LogitTrust can still perform better than Beta reputation.

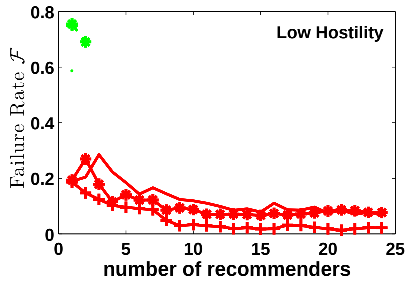
Insight: The common aspect of LogitTrust and Beta reputation with belief discounting is that both can mitigate malicious attacks. Beta reputation with belief discounting achieves resiliency by putting more weights on those recommendations from recommenders whom it trusts more. The effectiveness is sacrificed because it relies on the SR's accurate trust estimates toward the recommenders, which is difficult to achieve by Beta reputation in MANET environments wherein node-to-node interaction experiences or observations are limited. LogitTrust achieves resiliency by leveraging its robust statistical kernel to tolerate outlier recommendations and thus effectively achieve resiliency against recommendation attacks. LogitTrust can significantly perform better than Beta reputation in low to medium hostility environments, and can effectively perform better than Beta reputation in high hostility environments with as little as 5-10 recommendations from trustworthy friends and acquaintances. In this paper we only use social relationships as the basic mechanism to filter out potential false recommendations. The performance of LogitTrust may be further enhanced if we put in another mechanism such as thresholding (in addition to social relationships) to further filter out false reports.

7 Computational Feasibility

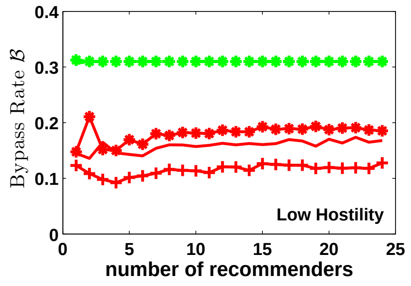
In this section, we discuss the computational feasibility for a MANET node to execute LogitTrust to learn the behavior pattern of SP j, that is, $\boldsymbol{\beta}_j$ at runtime. In our case for processing $n = 204$ records, it takes 2.63s realtime in a 2.4 GHz i7 CPU with 8GB RAM. For a less powerful MANET node, it may take minutes rather than seconds to compute the result. Fortunately, the computational procedure involving the iterative execution of the EM algorithm (see Algorithm 1: LogitTrust) 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 already learned behavior patterns ($\boldsymbol{\beta}_j$'s for individual SPs) in the system for decision making. To save storage space, a SR can keep only "useful" data leading to the learned behavior pattern toward a SP. Lastly we note that with the advent of mobile cloud computing, *true* runtime decision making may soon become a reality. This remains to be investigated in future research.



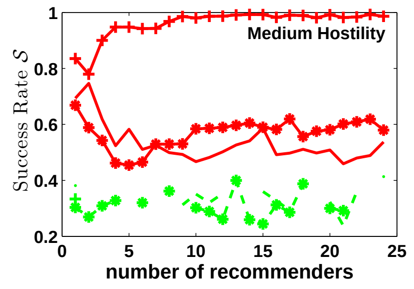
(a) Success Rate (0,0.1,0.2)



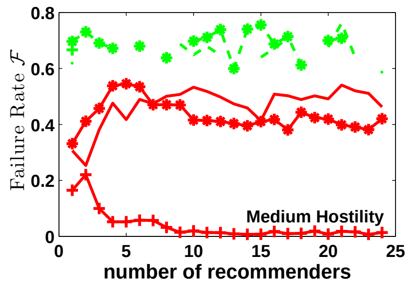
(b) Failure Rate (0,0.1,0.2)



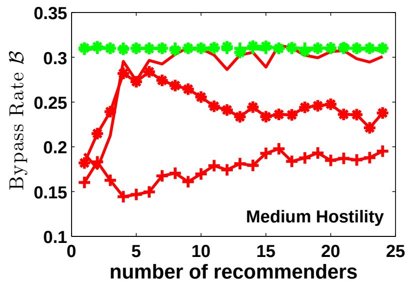
(c) Bypassing Rate (0,0.1,0.2)



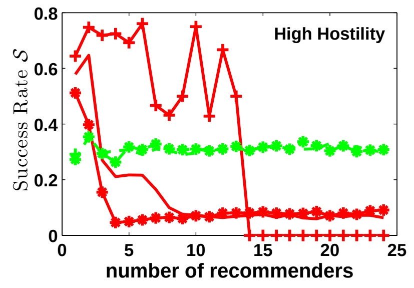
(d) Success Rate (0.1,0.3,0.5)



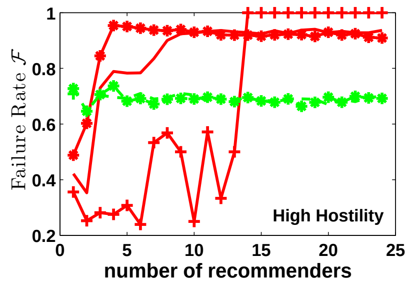
(e) Failure Rate (0.1,0.3,0.5)



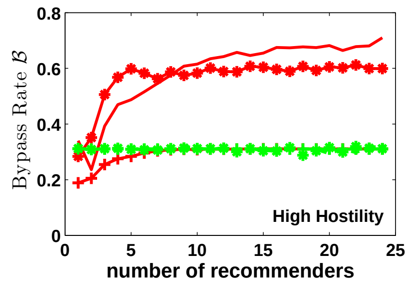
(f) Bypassing Rate (0.1,0.3,0.5)



(g) Success Rate (0.2,0.6,1.0)



(h) Failure Rate (0.2,0.6,1.0)



(i) Bypassing Rate (0.2,0.6,1.0)

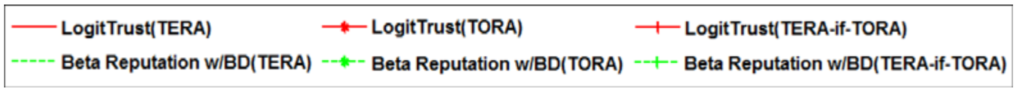

LogitTrust(TERA)
LogitTrust(TORA)
LogitTrust(TERA-if-TORA)
Beta Reputation w/BD(TERA)
Beta Reputation w/BD(TORA)
Beta Reputation w/BD(TERA-if-TORA)

Figure 3: Comparing LogitTrust against Beta Reputation with Belief Discounting in the presence of TORA, TERA, and TERA-if-TORA in Service Success Rate, Failure Rate and Bypassing Rate.

8 Conclusion

In this paper, we proposed a novel regression-based trust model, LogitTrust, for evaluating SP trustworthiness for executing and delivering a service in service-oriented MANET environments. Our model based on logit regression analysis assesses each SP in terms of its service behavior patterns in response to operational and environmental changes. The net effect is that we are able to learn and then predict its behavior in an operational and environmental setting, instead of judging its trustworthiness just from local observations or recommendations received by a SR. Our simulation results demonstrated that LogitTrust outperforms traditional approaches based on Bayesian Inference with belief discounting in terms of the rate of receiving a satisfactory service, the rate of unsatisfactory service received and the rate of satisfactory service missed in the presence of socially selfish nodes performing false recommendation attacks, when given the same amount of observations and recommendations as input.

For future work, we plan to extend the paper to address the issue of runtime learning and decision making, possibly leveraging cloud computing. In particular, we plan to develop a mechanism to learn and understand the hidden incentives behind a node's behavior patterns in response to dynamically changing environmental conditions. We also plan to further test the resiliency of LogitTrust against more complicated environmental and operational scenarios such as noisy MANET environments and application-specific behavior pattern factors as well as more sophisticated attack behaviors such as opportunistic, collusion and insidious attacks [15].

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] E. Aivaloglou and S. Gritzalis, "Trust-based data disclosure in sensor networks," in *IEEE International Conference on Communications*, June 2009, pp. 1–6.

[2] C. Archambeau, "Probabilistic models in noisy environments and their application to a visual prosthesis for the blind," *Unpublished doctoral dissertation, Université Catholique de Louvain, Belgium*, 2005.

[3] M. Blaze, J. Feigenbaum, and J. Lacy, "Decentralized trust management," in *IEEE Symposium on Security and Privacy*, May 1996, pp. 164–173.

[4] S. Buchegger and J.-Y. Le Boudec, "A robust reputation system for p2p and mobile ad-hoc networks," in *P2PEcon*, 2004.

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

[6] 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 *International Conference on Computational Science and Engineering*, August 2009, pp. 641–650.

[7] 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.

[8] K. Govindan and P. Mohapatra, "Trust computations and trust dynamics in mobile adhoc networks: A survey," *Communications Surveys Tutorials, IEEE*, vol. 14, no. 2, pp. 279–298, Second 2012.

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

[10] A. Josang and R. Ismail, "The beta reputation system," in *15th Bled Electronic Commerce Conference*, 2002.

[11] K. L. Lange, R. J. A. Little, and J. M. G. Taylor, "Robust statistical modeling using the t distribution," *Journal of the American Statistical Association*, vol. 84, no. 408, pp. 881–896, 1989.

[12] F. Li and J. Wu, "Mobility reduces uncertainty in manets," in *26th IEEE International Conference on Computer Communications*, May 2007, pp. 1946–1954.

[13] Z. Li, X. Li, V. Narasimhan, A. Nayak, and I. Stojmenovic, "Autoregression models for trust management in wireless ad hoc networks," in *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, Dec 2011, pp. 1–5.

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

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

[16] R. H. Myers, D. C. Montgomery, G. G. Vining, and T. J. Robinson, *Generalized linear models: with applications in engineering and the sciences.* John Wiley & Sons, 2012, vol. 791.

[17] A.-K. Pietilainen, "CRAWDAD data set thlab/sigcomm2009 (v. 2012-07-15)," Downloaded from http://crawdad.org/thlab/sigcomm2009/, Jul. 2012.

[18] T. Sakurai and H. Vu, "Mac access delay of ieee 802.11 dcf," *IEEE Transactions on Wireless Communications*, vol. 6, no. 5, pp. 1702–1710, May 2007.

[19] G. Shafer, *A Mathematical Theory of Evidence.* Princeton University Press, 1976.

[20] Y. Sun, Z. Han, W. Yu, and K. Liu, "A trust evaluation framework in distributed networks: Vulnerability analysis and defense against attacks," in *25th IEEE International Conference on Computer Communications*, April 2006, pp. 1–13.

[21] Y. Sun, W. Yu, Z. Han, and K. Liu, "Information theoretic framework of trust modeling and evaluation for ad hoc networks," *Selected Areas in Communications, IEEE Journal on*, vol. 24, no. 2, pp. 305–317, Feb 2006.

[22] 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, Sept 2012.

[23] J.-L. Wang and S.-P. Huang, "Fuzzy logic based reputation system for mobile ad hoc networks." in *Lecture Notes in Computer Science*, B. Apolloni, R. J. Howlett, and L. C. Jain, Eds., vol. 4693. Springer, 2007, pp. 1315–1322.

[24] X. Wang, W. Cheng, P. Mohapatra, and T. Abdelzaher, "Artsense: Anonymous reputation and trust in participatory sensing," in *INFOCOM 2013*, April 2013, pp. 2517–2525.

[25] X. Wang, J.-H. Cho, K. Chan, M. Chang, A. Swami, and P. Mohapatra, "Trust and independence aware decision fusion in distributed networks," in *2013 IEEE Pervasive Computing and Communications Workshop*, March 2013, pp. 481–486.

[26] 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," *Wireless Sensor Systems, IET*, vol. 1, no. 4, pp. 248–266, December 2011.

[27] Z. Yao, D. Kim, and Y. Doh, "Plus: Parameterized and localized trust management scheme for sensor networks security," in *Mobile Adhoc and Sensor Systems (MASS), 2006 IEEE International Conference on*, Oct 2006, pp. 437–446.

[28] Y. Yu, L. Guo, X. Wang, and C. Liu, "Routing security scheme based on reputation evaluation in hierarchical ad hoc networks," *Comput. Netw.*, vol. 54, no. 9, pp. 1460–1469, Jun. 2010.