Machine to Machine Trust in the IoT Era

Ling Liu[(+)], Margaret Loper[(*)], Yusuf Ozkaya[(+)], Abdurrahman Yasar[(+)], Emre Yigitoglu[(+)]

[(+)] School of Computer Science
[(*)]Georgia Tech Research Institute
Georgia Institute of Technology

Abstract

Machine to machine communications are at the center stage of the Internet of things (IoT). Connecting the physical world with the digital world not only creates new opportunities for innovation and discovery, but also opens doors for misuse and abuse. This paper argues that reputation based trust can be an effective countermeasure for securing machine-to-machine communications. We propose to establish machine-to-machine trust by taking into account both transaction/interaction service behaviors and feedback rating behaviors in the presence of bogus transactions and dishonest feedbacks. Our machine-to-machine trust model, called M2MTrust, introduces two novel trust metrics: (1) pairwise similarity based feedback credibility and (2) threshold-controlled trust propagation. We compute the direct trust from machine A to machine B by utilizing their pairwise rating similarity as the weight to the normalized aggregate of ratings that A has given to B. Our direct trust computation model can effectively constrain malicious nodes to gain direct trusts from dishonest feedback ratings by leveraging feedback credibility. Furthermore, our threshold-controlled trust propagation mechanism can successfully block the trust propagation from good nodes to malicious nodes. We conduct extensive experiments using simulation and real datasets and the experimental results show that M2MTrust significantly outperforms other trust metrics in terms of both attack resilience and performance in the presence of dishonest feedbacks and sparse feedback ratings against four representative attack models.

1. Introduction

The Internet of Things (IoT) refers to the capability of enabling physical objects to be connected, tracked, coordinated, or controlled by using sensors, actuators and Internet technology. By bringing machines into the connected world, be it hand-held devices, smart phones, self-driving vehicles, consumer appliances, and wireless terminals, the IoT holds the promise of enabling machines with embedded actuators and sensors to be programmed to take action on their own. Machine to machine communications will be at the center stage of the Internet of everything (IoE). Connecting the physical world with the digital world not only creates new opportunities for science and engineering discovery, for business and industry innovation, and for new life-enhancing experiences but also opens doors for misuse and abuse, as well as new privacy risks and security violations. For example, the connections that allow remote machines to take action without a human operator are subject to hacking by criminals or terrorists. The machines in the connected world may be infected by external and side channel attacks, such as Trojan horse programs, viruses, dataflow replay and DDoS attacks. Trust and reputation management are recognized as a popular and yet effective countermeasure for secure machine to machine (M2M) interactions.

We argue that in-depth understanding of trust and reputation is critical for agent societies where agents can be machines with embedded actuators and sensors, or humans with hand-held devices. Trust operates at many levels of interactions in agent societies, including human-to-human, human-to-machine, and machine-to-machine interactions. Trust has multi-facet and can play many roles in many contexts. For instance, the use of reputation mechanisms is one way to establish trusts based on interactions and feedback

ratings of the interactions. Many computational and theoretical models and approaches to reputation have been developed in recent years (for ecommerce, social networks, blogs, etc.). To ensure reputation based trust is established reliably, one needs to ascertain reliable interactions and transactions by identity and associated trustworthiness. A high quality trust is capable of reflecting the trade-off between individual utility and collective interest.

Trust and reputation also involve deception, privacy, security and control [Du+14, Hwa09, Var14]. Furthermore, many cloud infrastructure providers requite the IoT multi-agent systems and applications to be responsible for the application level security, privacy and trust vulnerabilities [Ber08, Li+13, Ris09]. For example, Amazon [AWS] states that security of tenant virtual machines is the responsibility of tenants since they are free to run any of operating systems or applications, although it claims to safeguard the underlying infrastructure. Also attackers can pretend to be legitimate (good) service providers but provide untrustworthiness service components. The service components provided by authentic providers may also embody security threats that can be exploited by attackers. Furthermore, multiple malicious attackers may launch colluding attacks on certain targeted service functions [BarXX,Du+14]. The reputation-based trust management can effectively measure resource provision QoS through trust-guided selection of services providers [Hwa09, Ima13, Li+10, San14].

In this paper, we present a reputation based machine-to-machine trust framework, called M2MTrust, to facilitate a network of machines to accomplish a collaboration task with high quality and high reliability. Each of these machines can be viewed as a collaborative agent in a networked multi-agent system. M2MTrust is novel in two aspects: First, M2MTrust utilizes machine to machine interaction experiences to establishes direct trust between a pair of machines. Example interactions can be transactional services and feedback ratings. To increase attack resilience of our M2MTrust model in the presence of malicious services (e.g., inauthentic file downloads) and dishonest feedback, M2MTrust promotes a clean distinction of transactional experience based reputation from feedback referral based reputation and compute the direct trust from machine A to machine B based on both the feedback ratings that B have received from A as well as the rating similarity that both A and B have given to the same set of machines that they have interacted with historically. We show that the rating similarity based feedback credibility can be instrumental for improving the attack resilience of the simply rating aggregation based direct trust computation. Second, M2MTrust employs a trust propagation kernel to handle the rating sparseness and cold start problem with feedback rating based trust computation [Kam03, Xio04]. However, we argue that though the uniform trust propagation kernel is popular and simple to implement, it is vulnerable to some strategic malicious attacks [Kam03, Fan41]. To counter such vulnerabilities, we develop a threshold-based controlled trust propagation kernel. This allows M2MTrust to propagate the trust of an agent (e.g., machine A) to only those of the agents to whom A has rated before (i.e., A's neighbor agents in the rating network) and with whom A also has similar transactional and feedback rating behaviors. Combined with the above two novel features, M2MTrust makes it much harder for malicious entities to gain trust from good entities, and enables good entities to share their experiences and feedback within the circle of agents who share similar transactional and feedback behaviors. Our controlled trust propagation scheme is highly attack resilient in the presence of dishonest feedback, sparse ratings, and a large number of malicious entities, because M2MTrust can significantly block the trust propagation paths from good entities to malicious ones. We conduct extensive evaluations of our proposed M2MTrust in terms of its effectiveness and efficiency against four representative attack models using both simulation and realistic datasets. Our experimental results show that M2MTrust significantly outperforms other reputation-based trust models.

The rest of this paper is organized as follows. Section 2 gives an overview of the basic concepts and terminology, the attack models and the core components of a reputation-based trust model. We describe in detail our M2MTrust model in Section 3. We report the experimental results in Section 4 and conclude the paper in Section 5.

2. OVERVIEW

2.1 Preliminary

Machine to Machine Interaction Network. Let G=<V, E> represent a large-scale machine to machine interaction network of n entities, $|V|=n$. Each entity is connected to a minimum number ($m>1$) of other entities. An edge $e=(i, j)$ is in E if two entities i and j are connected in the network G. Example transactions performed in such a network could be content/file sharing or scientific computation. It is well known that the real world machine to machine interaction network typically follows a zipf skewed degree distribution such that large number of entities connect to a small number of entities and only a few entities connect to a large number of entities. Thus an entity may be served by another entity that is several hops away in the network G. When an entity receives several responses for its service request, it may rely on their reputation trust scores to determine which one is selected to be its service provider.

Machine to Machine Transaction Rating Network. Upon the completion of a transaction between a pair of entities, the service entity can rate the provider entity in terms of its quality of service (QoS) provisioned, denoted by $tr(i, j)$. An entity i of the system can rate another entity j if it has an actual transaction with entity j. The rating can be either binary [Kam03,Ric03] or multi-scale [Fen12,Su+13]. For example, with binary rating model, i can give j the positive feedback rating by $tr(i, j) = 1$ or negative rating by $tr(i, j) = -1$. By default, $tr(i, j) = 0$ for $i, j = 1, \ldots, n$, and it implies that i has never had any transaction with j. Based on the rating relationship between a pair of machines, we can construct a machine to machine rating network. Note that the n entities in the rating network are the same as the machine-to-machine interaction network. However, two entities that have edges in the rating network may not be connected in the machine-to-machine interaction network and vice versa.

Simple Rating Aggregation. Let s_{ij} denote the simple aggregate rating that entity i gives to another entity j. We can define s_{ij} by the sum of individual feedback ratings: $s_{ij} = \sum tr(i, j)$. For binary rating, this is equivalent to the difference between the satisfied transaction number $sat(i, j)$ and unsatisfied transaction number $unsat(i, j)$ that entity i has received from entity j. Namely $s_{ij} = sat(i, j) - unsat(i, j)$.

Normalized Rating Aggregate and Direct Trust Score. It is well understood that using s_{ij} to define the direct trust value that i gives to j is problematic [Xio04], because this can introduce certain vulnerability due to some unwanted bias. For example, with 21 positive rating and 1 negative rating will receive the same trust score of 20 as the entity with 100 positive rating and 80 negative ratings. However, it is obvious that the entities that received large proportion of negative ratings are at the risk of being dishonest or malicious raters. Thus, normalized rating aggregation schemes are proposed [Kam03, Li+04] to prevent dishonest or malicious entities from colluding by giving arbitrarily high direct trust to other malicious entities, and arbitrarily low direct trust to good entities. In addition, for those entities that have not received any ratings because they have not been selected as service providers for any request (e.g., cold start or rating sparseness), a common mechanism is to use a small number of bootstrap entities to serve as pre-trusted entities (machines) in the network. One method to normalize the direct trust score that i has over j, denoted by c_{ij} , is given below [Kam03]:

$$c_{ij} = max(s_{ij}, 0) \Big/ \sum_k max(s_{ik}, 0) \ \ if \ \sum_k max(s_{ik}, 0) \neq 0$$
$$c_{ij} = p_j \ \ otherwise \qquad (1)$$

When $s_{ij} = 0$, we set $c_{ij} = p_j$ and $p_j = 1/|P|$ if $j \in P$, P is the set of pre-trusted bootstraping entities. Using a small number of pre-trusted members as the central authority of the system, it can help bootstrap the trust system initially [Kam03]. By utilizing c_{ij} $(i, j=1, \ldots, n)$, we transform the rating network of n entities into a direct trust network for the same n entities and if two entities have rating relationship, then they will have direct trust relationship.

Trust Propagation. When a trust network is very sparse, namely each entity i only trusts a small number of other entities, say j, such that $c_{ij}>0$, and for most $j \in [1, n]$, we have $c_{ij} = 0$. This will make it very hard for i to find and select the right service providers because most of time entity i may not have any other entity that can provide the service requested by i. This skewed problem has lead to the use of trust propagation kernel to compute the transitive trust that entity i has over entity j as long as j is reachable from i in the rating (also direct trust) network by graph traversal. Concretely, if j is reachable from i via another entity q in the direct trust network, then we can compute the trust c_{ij} by the weighted summation of c_{iq} and c_{qj} : $c_{ij} = \sum_q c_{iq} \cdot c_{qj}$.

Let n denote the total number of entities in the system, we can define C as the matrix $[\, c_{ij} \,]$ with n rows and n columns. Let $\vec{t}^{k+1}$ denote the global trust vector of size n to be computed at $(k+1)^{\text{th}}$ round of iterations, $0<k<n$. Then we can define $\vec{t}^{k+1} = (1-a)C^T\vec{t}^{k} + a\vec{p}$, where a is the probability of an entity knows none and relies on the pre-trusted entities to be introduced into the network of the system, and $\vec{p}$ denotes the initial trust vector with only pre-trusted entities have initial non-zero trust scores, each having the trust score of $1/|P$. For each element of the trust vector $\vec{t}^{k+1}$, say $t_i^{(k+1)}$, we can transform the above matrix form into the following:

$$t_i^{(k+1)} = (1-\alpha)(c_{1i}t_1^{(k)} + \mathrm{L} \; + c_{ni}t_n^{(k)}) + \alpha p_i \qquad (2)$$

This formula says that the reputation-based trust score of entity i can be computed by aggregating the direct trust values that entity i has received from all other entities in the trust network, e.g., c_{1i}, …c_{ji}, …, c_{ni} ($j \neq i \in [0, n]$).

Trust-enabled Service Selection. Two popular trust-enabled service provider selection schemes are deterministic method and probabilistic method [Kam03]. The deterministic method always chooses the service provider with the highest global trust score among those who respond to the service request as the provider, such as the download source for a music request. This can overload the entity with the highest global trust score. The probabilistic method chooses an entity i as the provider according to the probability generated by response entities' trust, computed by $t_i / \sum_{j=1}^{R} t_j$, ensuring that a participant with higher trust will have higher probability to be selected as service provider. The probabilistic method prevent the system from overloading entities with high trust scores. To further overcome the problem of cold start with new members, one can augment the above trust-enabled service selection method by some refinement: For example, with a small default probability, say 5~ 10%, the system may randomly select from those participants whose trust scores are zero as the service provider. This refined probabilistic selection gives newcomers some chance to build up their trust in the system.

2.2 Attack Models

We consider the following four attack models [Kam03, Ric03, Fan12, Fen12, Son05, Su+13].

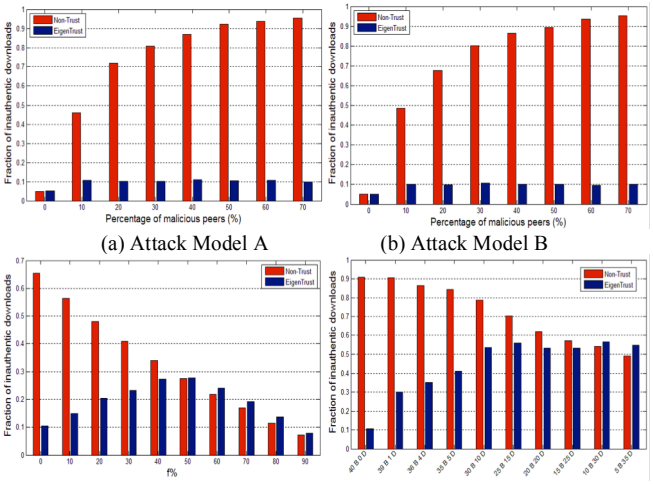
Attack Model A (*Independently Malicious*). Malicious participants are independent and provide bad services (e.g., fake data) and dishonest ratings.

Attack Model B (*Chain of Malicious Collectives*). Malicious participants collude with one another and deterministically give other colluding entities high trust score and badmouth good entities. This results in a malicious chain with entities in the chain having high direct trust values. Malicious participants always provide bad services and dishonest ratings.

Attack Model C (*Malicious Collectives with Camouflage*). Malicious participants get high direct trust score because to gain high ratings, they strategically provide good services (e.g., authentic data) in f% of all cases when selected as service providers. However, malicious participants always provide dishonest feedback ratings to good participants.

Attack Model D (*Malicious Colluding Spies*). Malicious participants are strategically organized into

two groups: one group of malicious participants (type D) who act as normal participants in providing good services to increase their positive ratings and use the trust they have gained to boost the trust score of the other group of malicious colluding participants (type B) who only provide bad services when selected as service providers. Both types of malicious entities always provide dishonest ratings to good participants.



(a) Attack Model A (b) Attack Model B

(c) Attack Model C (d) Attack Model D

Fig. 1. EigenTrust performance in Attack Models A, B, C and D.

To show how such attacks may impact on the effectiveness of a trust model, we implement EigenTrust with the normalized rating aggregate in Formula (1) for the direct trust computation and the uniform trust propagation in Formula (2) for the global trust computation. Fig.1 shows the benefit and the inherent problems of EigenTrust, which are also reported in [Kam03]. We make three interesting observations: First, the trust model works effectively compared to non-trust scenario under Attack Models A and B with up to 70% malicious participants, but it performs poorly against Attack Models C and D when the malicious participants are about 27% and 39% respectively. Second, under Attack Model C (malicious collectives with camouflage), the effectiveness of the EigenTrust model deteriorates very fast as the percentage *f* increases. When *f* is greater than 50%, EigenTrust surprisingly performs worse than non-trust scenario with higher fraction of bad services. In addition, under Attack Model D (malicious spies), as the number of malicious spies (type D) increases by 25% of the total malicious group, EigenTrust continues to deteriorate and when the malicious spies are up to 75% of the total malicious colluding group, EigenTrust performs worst than no trust scenario.

2.3 M2MTrust: Solution Approach

We identify two main reasons that the direct trust computation combined with the uniform trust propagation kernel used in EigenTrust is vulnerable under Threat models C and D. The first reason is due to the subjective assumption made by the design of the EigenTrust model and most existing trust models: namely participants performing well in their transactional services will also provide honest ratings and vice versa. Unfortunately, the correlation between these two factors is only high for good entities but can be very low lead for malicious entities. Second, both direct trust computation and the uniform trust propagation kernel completely fail to differentiate good participants from those strategically malicious collectives who act as spies or who make use of camouflage effect.

Bearing this understanding in mind, we develop M2MTrust to strengthen the robustness of both direct trust computation and the trust propagation kernel by intelligently capitalizing on the similarity-based feedback credibility and controlled trust propagation.

Concretely, instead of using normalized rating aggregate as the direct trust for each participant in the machine to machine interaction network, we define the relative feedback credibility of a participant with respect to another participant using their rating similarity. If *i* has higher similarity to *j* in their feedback behavior, then the feedback credibility of participant *i* with respect to participant *j* is high. Moreover, we also set an exponential threshold for participants *i* and *j* to judge whether *i* should propagate trust to *j*, and how much trust should be propagated if approved. This enables M2MTrust to control when and how much to propagate trust from one participant to another. We utilize this feedback credibility and threshold-based control knob to determine when and how much a participant should propagate trust to other participants

with which it has direct or indirect rating relationships, and the hop based partition size of the rating network anchored from this participant.

The problem of uniform trust propagation is the uniform treatment of good and bad participants. The uniform trust propagation model works well when there are no malicious collectives with camouflage or malicious spies as defined in Attack Models C and D respectively. However, when malicious collective with camouflage exists, as the number of iteration rounds increases, the fraction of bad services (such as inauthentic downloads) also goes up due to the increased amount of good services provided by the malicious participants as camouflage or spies, which boosts the trust scores of malicious participants while failing to raise the trust scores of good participants. Thus, we advocate the use of a controlled trust propagation kernel in our M2MTrust framework.

3. ATTACK RESILIENT TRUST PROPAGATION

3.1 Similarity Weighted Direct Trust Computation

In the M2MTrust model, we first aggregate the transaction ratings that a participant *i* gives to another participant *j* by introducing the total number of transactions as the denominator:

$$s_{ij} = \begin{cases} \dfrac{sat(i,j)}{sat(i,j)+unsat(i,j)} & sat(i,j)+unsat(i,j) \neq 0 \\ 0 & otherwise \end{cases} \quad (3)$$

Then we use this normalized s_{ij} to compute the transaction based local trust that *i* has on *j*, denoted by c_{ij}, by normalizing the aggregate transaction rating as follows:

$$c_{ij} = \begin{cases} \dfrac{max(s_{ij},0)}{\sum_j max(s_{ij},0)} & if \sum_j max(s_{ij},0) \neq 0 \\ p_j & otherwise \end{cases} \quad (4)$$

where p_j denotes the set P of pre-trusted participants of the network, and $p_j = 1/|P|$ for $j \in P$, otherwise $p_j = 0$. In addition, we further improve the computation of direct trust that *i* has for *j* by introducing feedback similarity as a weight to the normalized rating aggregate score, c_{ij}. This decision is motivated by a number of observations:

(i) Two good participants may give very similar feedback ratings to the same common set of participants with which they have had interactions or transactions in the past.

(ii) Two malicious participants, on the other hand, may give very similar feedback ratings to the same common set of (good or malicious) participants with which they have had historical transactions.

(iii) On the contrary, a good participant and a malicious participant most likely would give very different feedback ratings to the same set of participants whom they have interacted with. Thus, we can utilize such feedback similarity measure as an indicator to differentiate good participants from malicious participants.

Pairwise Feedback Similarity. In a network of *n* participants, each participant *i* has a feedback vector of size *n*, denoted by $<s_{i1}, s_{i2}, \ldots, s_{in}>$. To compute the similarity between two feedback vectors of participants *i* and *j*, we use the Weighted Euclidean Distance (WED) method, which captures the degree of "dispersion" in the historical feedback given by the participants *i* and *j*. The larger the dispersion is, the smaller the similarity will be. Thus the feedback based similarity between two participants *i* and *j* can be defined as follows:

$$sim(i,j) = \begin{cases} 1 - \sqrt{\sum_{q \in comn(i,j)} w_{(i,j,q)} \cdot (Tr(i,q) - Tr(j,q))^2} & if \ |comn(i,j)| \neq 0 \\ 0 & otherwise \end{cases}$$

$$Tr(i,q) = \frac{\sum_{k=1}^{|r(i,q)|} tr_k(i,q)}{|r(i,q)|} \qquad (5)$$

$$Tr(j,q) = \frac{\sum_{k=1}^{|r(j,q)|} tr_k(j,q)}{|r(j,q)|}$$

where $comn(i,j)$ denotes the subset of common participants that have had interaction with both i and j, $r(i,q)$ is the total number of transactions between i and q, and $tr_k(i,q)$ denotes the kth normalized local trust that participant i places on another participant j. $w_{(i,j,q)}$ denotes the normalized weight of participant q's impact on similarity measure by calculating its standard deviation:

$$w'_{(i,j,q)} = \sqrt{\frac{(Tr(i,q) - Avg(i,j,q))^2 + (Tr(j,q) - Avg(i,j,q))^2}{2}}$$

$$Avg(i,j,q) = \frac{Tr(i,q) + Tr(j,q)}{2} \qquad (6)$$

$$w_{(i,j,q)} = \begin{cases} \frac{w'_{(i,j,q)}}{\sum_m w'_{(i,j,m)}} & if \sum_m w'_{(i,j,m)} \neq 0 \\ 0 & otherwise \end{cases}$$

The use of weighted Euclidean distance allows us to leverage different weights to amplify the rating dissimilarity over the common transactional participants that are rated differently by i and j in the vectors $<s_{i1}, s_{i2}, \ldots, s_{in}>$ and $<s_{j1}, s_{j2}, \ldots, s_{jn}>$. Consider an example feedback vectors of participants i and j over the four other common participants are <0.10, 0.30, 0.02, 0.05> and <0.01, 0.05, 0.05, 0.85> respectively, we can calculate that the traditional ED based similarity, which is 0.578, and the WED based similarity, which is 0.328. Based on the intuitive analysis on the two vectors of feedback ratings, we can perceive that i and j should be dissimilar. This shows that using weighted Euclidean distance based formula is more effective.

Similarity based Feedback Credibility. We define feedback credibility and utilize it to constrain the malicious participants from receiving high feedback from other good participants even when they provide satisfactory transactions. We use the exponential function of the pairwise similarity:

$$cr_{ij} = e^{(1-1/sim(i,j))} \qquad (7)$$

This formula indicates that the feedback credibility is exponentially constrained: the feedback credibility will be high when the pairwise similarity is high, and vice versa. If $sim(i,j) = 1.0$, then $cr_{ij} = 1.0$; if $sim(i,j) = 0.0$, then $cr_{ij} \approx 0.0$. Let cf_{ij} denote the feedback credibility weighted direct trust score that participant i has placed on participant j. We compute cf_{ij} by utilizing similarity based feedback credibility as the weight to c_{ij} as follows:

$$cf_{ij} = cr_{ij} \cdot c_{ij} = e^{(1-1/sim(i,j))} \cdot c_{ij} \qquad (8)$$

This feedback credibility weighted direct trust computation formula states that a participant has high local trust value only if this participant has received high transaction based ratings (c_{ij}) and high feedback credibility at the same time. For example, the direct trust value that a good participant i places on a malicious participant j should be weighted by their similarity-based feedback credibility. Given that the good participant i and the malicious participant j will be extremely dissimilar in their feedback behaviors, thus the feedback credibility cr_{ij} is very small. By Formula (8), the weighted direct trust that i has for j will be much smaller than the normalized rating aggregate value, c_{ij}. Thus, by using feedback credibility as a weight to the transaction based rating, M2MTrust can effectively reduce the positive ratings from good participants to malicious participants no matter whether they are malicious camouflage (Attack Model C) or malicious spies (Attack Model D).

Threshold-Controlled Trust Propagation. Although we can restrain malicious participants from gaining high local trust through feedback credibility, we still cannot completely cut down the trust propagation to malicious participants once they have built up the trust propagation path (direct trust) from good participants. Thus, we propose a threshold-controlled trust propagation kernel:

$$\tau'_{ij} = 1/(1+e^{sim(i,j)}) \qquad (9)$$

To coordinate with direct trust, we map this threshold to the same interval [0, 1] by *max-min* method:

$$\tau_{ij} = \frac{\tau'_{ij} - 1/(1+e^{max(sim(u,v))})}{1/(1+e^{min(sim(u,v))}) - 1/(1+e^{max(sim(u,v))})} \qquad (10)$$

Obviously, $max(sim(u,v))= 1.0$ and $min(sim(u,v)) = 0.0$. Hence we compare the feedback credibility weighted direct trust score cf_{ij} with this exponential threshold τ_{ij}. If $cf_{ij} \geq \tau_{ij}$, we propagate trust from *i* to *j*; if $cf_{ij} < \tau_{ij}$, we block the trust propagation. We compute the global trust scores at the $(k+1)^{th}$ iteration for all *n* participants in the machine to machine network by utilizing the global trust scores computed at k^{th} iteration as follows:

$$\vec{t}^{k+1} = (1-a)L^T\vec{t}^{k} + a\vec{p} \qquad (11)$$

where *a* is a jumping factor to avoid trust propagation to be trapped in a malicious clique. Also, we set the initial trust score for a participant *i* as $t_i^0 = \sum_j cf_{ij} \cdot cf_{ji}$.

The global trust for each participant over the transactional network can be defined by using the threshold-controlled trust propagation matrix *M*:

$$M = \begin{pmatrix} \varphi_{11} \cdot cf_{11} & \cdots & \varphi_{1n} \cdot cf_{1n} \\ \vdots & \ddots & \vdots \\ \varphi_{n1} \cdot cf_{n1} & \cdots & \varphi_{nn} \cdot cf_{nn} \end{pmatrix}$$

$$\varphi_{uv} = \begin{cases} 1, & if \quad cf_{uv} \geq \tau_{uv} \\ 0, & otherwise \end{cases}$$

where φ_{uv} denotes whether the feedback credibility weighted local trust value cf_{uv} is bigger than or equal to their threshold τ_{uv}, if the local trust cf_{uv} is big enough and the threshold τ_{uv} is small enough, then *u* propagates trust to *v*, otherwise *u* discards the trust propagation to *v*.

To facilitate the comparison of different propagating weights, we need to normalize matrix *M*:

$$m_{uv} = \begin{cases} \dfrac{\varphi_{uv} \cdot cf_{uv}}{\sum_q \varphi_{uq} \cdot cf_{uq}} & if \ \sum_q \varphi_{uq} \cdot cf_{uq} \neq 0 \\ 0 & otherwise \end{cases} \qquad (12)$$

Thus we propose the following matrix formula to calculate the global trust score:

$$\vec{t}^{k+1} = (1-\alpha)M^T \cdot \vec{t}^{k} + \alpha \cdot \vec{p} \qquad (13)$$

The $(k+1)^{th}$ iteration computation relies on the k^{th} iteration:

$$t^{k+1}(i) = (1-\alpha)(m_{1i}t^k(1) + \mathrm{L} + m_{ni}t^k(n)) + \alpha \cdot p_i \qquad (14)$$

Our threshold-controlled trust propagation kernel can successfully block the trust propagation from good participants to malicious ones.

4. EXPERIMENTAL EVALUATION

We evaluate M2MTrust model in terms of efficiency, effectiveness and attack resilience. To make a fair

comparison with EigenTrust, we build a simulator on the top of TM/RM simulation platform [TM/RM] and incorporate all four attack models used into this TM/RM simulator in order to compare the performance of our M2MTrust with EigenTrust, ServiceTrust and Non-Trust scenario. Table I gives the list of the parameters. The query/answer network is setup

We evaluate M2MTrust model in terms of efficiency, effectiveness and attack resilience. To make a fair comparison with EigenTrust, we build a simulator on the top of TM/RM simulation platform [TM/RM] and incorporate all four attack models used into this TM/RM simulator in order to compare the performance of our M2MTrust with EigenTrust, ServiceTrust and Non-Trust scenario. Table I gives the list of the parameters. The query/answer network is setup in a similar fashion as [Kam03]. Both malicious and pre-trusted participants have 10 initial neighbors, and good participants have 2 initial neighbors. Initially, only pre-trusted participants have positive reputation. When a participant issues a query, the query is propagated by the scoped broadcast mechanism with the specified hop-count horizon over the entire network. Participants that receive the query will forward it to the next hop participant(s) and also check whether they have the requested file or not, if have, respond it. We set 7 hops as the default response range. Furthermore, the number of distinct files assigned to each participant follows the Zipf distribution, and popular files have more copies in the system. On the other hand, the number of queries issued for different files is also based on Zipf distribution.

TABLE I
EXPERIMENTAL CONFIGURATION

Network Structure	number of good participants	600, 600, 700, 1000
	number of pre-trusted participants	30
	number of initial neighbors of good participants	2
	number of initial neighbors of malicious participants	10
	number of initial neighbors of pre-trusted participants	10
	number of hops for query process	7
File Distribution	file distribution at good participants	Zipf distribution over 200 distinct files
	number of distinct files at good participant	uniform random distribution
	top % queries for most popular files pre-trusted participants respond to	5%
	% file categories owned by good participants in Attack Model A, B and C	15%
	% file categories owned by good participants in Attack Model D	10%
	% file categories owned by malicious participants in Attack Model A, B, D	100%
	% of file categories owned by malicious participants in Attack Model C	55%
Participant Behavior	% of download requests in which good participant $_i$ returns inauthentic file	5%
	downloads source selection algorithm	probabilistic algorithm
	probability that participant with global trust value 0 is selected	range [0%-10%]

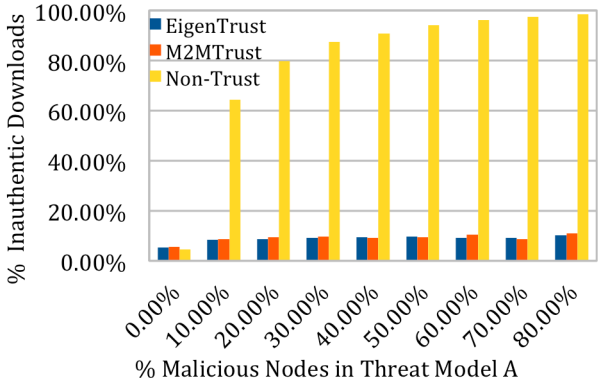


Fig. 2. Attack Model A

4.1 Performance Evaluation

We compare M2MTrust with Non-Trust and EigenTrust under the four attack models. Fig. 2 shows the results. The total numbers of transactions are 6300 in Attack Models A and B, 7300 in Attack Model C and 10300 in Attack Model D. Different from variable percentages of malicious participants in Attack Models A and B, the amounts of malicious participants are constant in Attack Models C and D, and the percentages of malicious participants are 27% and 39% respectively. We observe that M2MTrust effectively constrains the trust propagation to malicious participants from good participants due to the low rating similarity between them, in addition to the

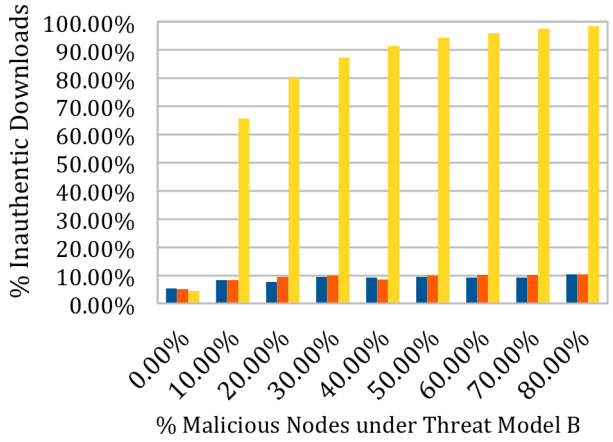


Fig. 3. Attack Model B

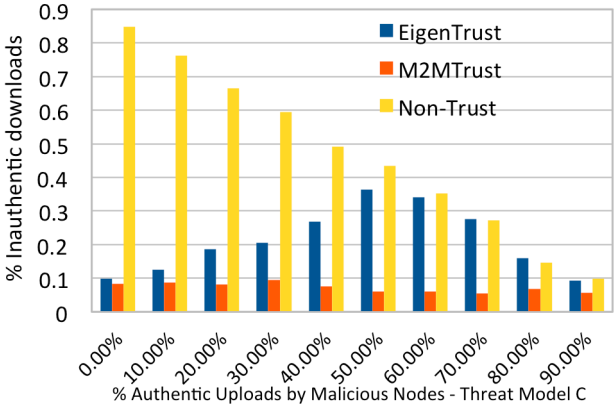


Fig. 4. Attack Model C (f=40%).

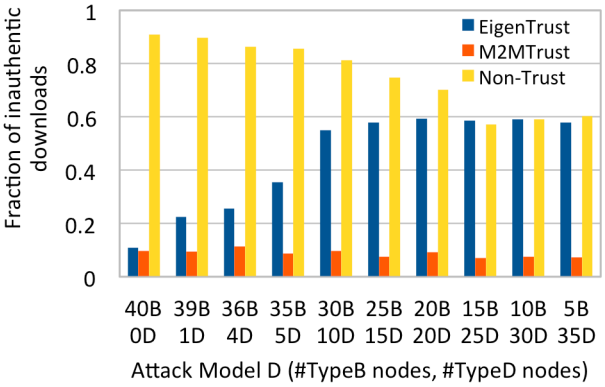


Fig. 5. Attack Model D

threshold-driven based controlled propagation kernel. In Attack Models C and D, M2MTrust also significantly outperforms Non-Trust and EigenTrust. Although the strategic malicious participants can gain trust scores by acting as regular participants to provide good services, the M2MTrust can differentiate malicious participants through from good participants by employing rating similarity based feedback credibility and threshold-controlled trust propagation kernel. Fig. 3 shows the global trust scores under Attack Model C when *f* is 40%. The global trust scores of malicious participants are very high in EigenTrust, and in contrast, M2MTrust can completely reduce the trust scores of malicious participants to zero through rating similarity weighted direct trust and the threshold controlled propagation kernel, which cut off the trust propagation paths from good participants to malicious participants effectively.

4.2 Evaluation Using Epinions Dataset

We evaluate the performance of M2MTrust using real dataset Epinions [Ric03] in terms of Attack Models. As we know, the strategic malicious participants in Attack Models C and D can gain trust scores through providing good services. Thus we create some colluding participants to learn the effectiveness of our M2MTrust metrics. Concretely, 10 malicious participants (ID: 1000-1009) are added into the Epinions dataset and connected to the 10 most highly connected participants already in the network to receive as many ratings as possible. Then, we organize them in two ways: one is to make these 10 participants form a chain to give colluding entities high direct trust ratings (say 1.0) according to configuration under Attack Model C, and the other is to divide them into two groups (Group B and Group D) with 5 participants each group according to the configuration in Attack Model D, participants in Group D provide trustworthiness services to gain high trust scores, and in return boost other participants in Group B. Since those malicious participants in Attack Model C and Group D act as regular participants, good participants will give them high feedback ratings and thus high direct trust scores. Therefore, we set an interval [0.5, 1.0] from which good (regular in Epimions) participants select their direct trust ratings to assess the malicious participants, and another interval [0, 0.05] from which malicious participants select their direct trust to assess good participants. In addition, we utilize Zipf Distribution to generate edge weight between a pair of regular participants. Fig. 6 shows the trust values of good nodes and malicious nodes under Threat model C. Fig. 7 zooms

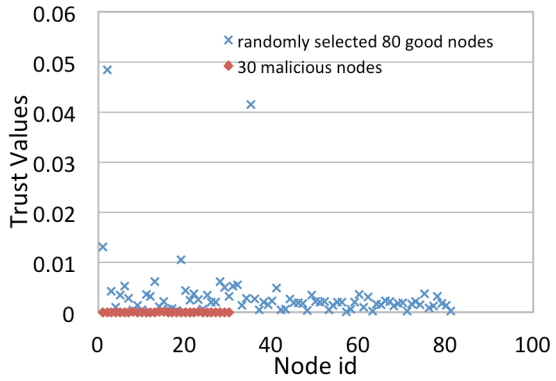


Fig. 6 Trust values of malicious and good nodes in ePinions

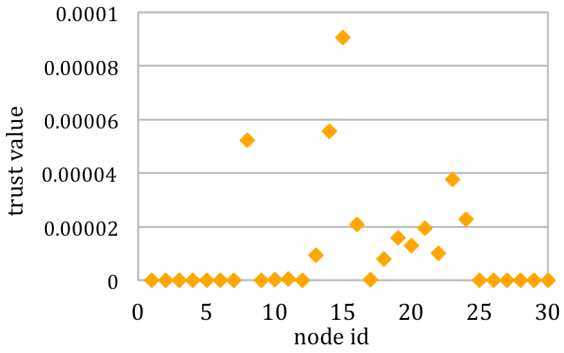


Fig. 7 Zoom-in of the trust values of malicious nodes in ePinions

into the trust values of all malicious nodes under Threat model C. Different from the configurations in EigenTrust wherein the initial trust scores of pre-trusted participants are non-zero (1/|P|, P is the set of bootstrap/pre-trust nodes) and others are zero. In M2MTrust, we compute the PageRank values as the initial trust scores. In Attack Model C, since the added participants (1000-1009) are set by the same links from/to other participants, their trust scores are equal by using the PageRank computation formula. EigenTrust increases the trust scores of these 10 malicious participants as they receive high direct trust ratings. In contrast, M2MTrust can effectively reduce the trust scores of these 10 malicious participants thanks to its rating similarity weighted direct trust and its threshold-controlled trust propagation kernel.

4.3 Computation Complexity

In EigenTrust, the time overhead mainly depends on the computation of trust scores. For each participant, its trust score is computed by aggregating the trust scores of other n-1 (n is the network size) participants, thus, for n participants, the computation complexity is $O(n^2)$. In M2MTrust, we need to compute the pairwise rating similarity. However, we do not need to compute rating similarity for every pair of participants in each iteration of the trust score refinement during the entire simulation. We just compute the pairwise similarity only when the direct trust ratings placed on the common set of participants rated by this pair of participants have been changed. Furthermore, in M2MTrust, for each participant, the threshold-controlled propagation needs $O(1)$ time to check whether a connected participant meets the threshold. The loop will continue until all the connected participants are checked. Given that the connected neighbors are no more than n. Thus, for the total participants, the computation complexity is also $O(n^2)$. Moreover, M2MTrust still needs $O(n^2)$ to compute trust scores for all the participants. In general, the computation overhead for M2MTrust is less than EigenTrust because M2MTrust can discard partial participants from being processed for trust propagation through feedback credibility when the pairwise similarity is zero, thus its time consumption is less than EigenTrust. In addition, M2MTrust can further cut off those connected participants from trust propagation when they fail to pass the threshold check, thus the computation overhead is much less than EigenTrust.

5. Conclusion

We have presented M2MTrust, an attack resilient machine-to-machine trust model and showed analytically and experimentally that M2MTrust is significantly more attack resilient than other trust metrics. Concretely, we promote three principled design goals. First, the direct trust between a pair of participants in the M2M network should be computed by taking into account both quality (satisfactory and unsatisfactory experiences) and quantity of their interactions, making it harder and

costly for attackers to manipulate the trust model. Second, our trust metrics advocate a clean separation of the transaction or interaction quality from the feedback quality in trust computation. This significantly strengthens the attack resilience. Third but not the least, M2MTrust incorporate threshold-controlled trust propagation, instead of uniform trust propagation to capture the non-uniformity of trust propagation and experience sharing among the group of connected nodes in the network. We conduct extensive experimental results using simulation and real dataset, and show that our proposed machine to machine trust model is effective terms of both performance and attack resilience in the presence of dishonest feedbacks and sparse feedback ratings against four representative attack models.

References

[AWS] "AWS security center." Available: http://aws.amazon.com/security/.

[Ber08] S. Berger et al., "TVDc: Managing Security in the Trusted Virtual Datacenter," ACM SIGOPS Operating Systems Rev., vol. 42, no. 1, pp. 40-47, 2008.

[Ris09] T. Ristenpart, E. Tromer, H. Shacham, and S. Savage, "Hey, You Get Off My Cloud! Exploring Information Leakage in Third-Party Compute Clouds," Proc. 16th ACM Conf. Computer and Communications Security (CCS), 2009.

[Du+14] J. Du, D. Dean, Y. Tan, et al. Scalable Distributed Service Integrity Attestation for Software-as-a-Service Clouds. IEEE Transactions on Parallel and Distributed System, 25(3):730-739, 2014.

[Kam03] S.D. Kamvar, M.T. Schlosser, and H. Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. Proceedings of the 12th international conference on World Wide Web, pages 640–651. ACM, 2003

[IBM] IBM Red Boo. Fundamentals of Grid Computing, Technical Report REDP-3613-00 2000.

[Li+13] X. Li and J. Du. Adaptive and attribute-based trust model for service-level agreement guarantee in cloud computing. IET Information Security, 7(1): 39-50, 2013.

[TM/RM] TM/RM simulator: http://rtg.cis.upenn.edu/qtm/p2psim.php3.

{San14] Z. Sanaei, S. Abolfazli, A. Gani, et al. Heterogeneity in Mobile Cloud Computing: Taxonomy and Open Challenges. IEEE Communication Surveys & Tutorials, 16(1): 369-392, 2014.

[Xio04] L. Xiong and L. Liu, "Peertrust: Supporting reputation-based trust for peer-to-peer electronic communities," IEEE Transactions on Knowledge and Data Engineering, 16(7):843–857, 2004.

[Ric03] Matthew Richardson, Rakesh Agrawal, Pedro Domingos. Trust Management for the Semantic Web. IN PROCEEDINGS OF THE SECOND INTERNATIONAL SEMANTIC WEB CONFERENCE, 2003.

[Gol08] J. Golbeck. Weaving a web of trust. Science, 321(5896):1640–1641, 2008.

[Wan10] Y. Wang and A. Nakao. Poisonedwater: An improved approach for accurate reputation ranking in p2p networks. Future Generation Computer Systems, 26(8):1317–1326, 2010.

[Fen10] Q. Feng, L. Liu, and Y. Dai. Vulnerabilities and countermeasures in context-aware social rating services. ACM Transactions on Internet Technology (TOIT), 11(3), 2012.

[Son05] S. Song, K. Hwang, R. Zhou, and Y.K. Kwok. Trusted p2p transactions with fuzzy reputation aggregation. Internet Computing, IEEE, 9(6):24–34, 2005.

[Bar13] A. Barsoum and A. Hasan. Enabling Dynamic Data and Indirect Mutual Trust for Cloud Computing Storage Systems. IEEE Transactions on Parallel and distributed Systems, 2013.

[Haw10] Hwang, K., Li, D.: 'Trusted cloud computing with secure resources and data coloring', IEEE Internet Comput., 2010, 14, (5), pp. 14–22.

[Awn12] X. Wang, L. Liu, J. Su, RLM: A General Model for Trust Representation and Aggregation. IEEE Transaction on Service Computing, 5(1): 131–143, 2012.

[Su+13] Zhiyuan Su, Ling Liu, Mingchu Li, et al. ServiceTrust: Trust Management in Service Provision Networks. IEEE International Conference on Services Computing (SCC), 272-279, 2013.

[Fan12] Xinxin Fan, Ling Liu, Mingchu Li and Zhiyuan Su. EigenTrust++: Attack Resilent Trust Management. Proceedings of 8th IEEE International Conference on Collaborative Computing (CollaborateCom2012).

[Var14] V. Varadharajan, and U. Tupakula, Security as a Service Model for Cloud Environment. IEEE Transactions on network and Service management, 11(1): 60-75, 2014.