IoT

MDPI

Article

Can We Trust Trust Management Systems?

Claudio Marche [1,*] and Michele Nitti [1,2,*]

[1] Department of Electrical and Electronic Engineering (DIEE), University of Cagliari, Via Marengo 3, 09123 Cagliari, Italy
[2] Research Unit of Cagliari, National Telecommunication Inter University Consortium, Viale G. P. Usberti 181A, 43124 Parma, Italy
* Correspondence: claudio.marche@unica.it (C.M.); michele.nitti@unica.it (M.N.)

Abstract: The Internet of Things is enriching our life with an ecosystem of interconnected devices. Object cooperation allows us to develop complex applications in which each node contributes one or more services. Therefore, the information moves from a provider to a requester node in a peer-to-peer network. In that scenario, trust management systems (TMSs) have been developed to prevent the manipulation of data by unauthorized entities and guarantee the detection of malicious behaviour. The community concentrates effort on designing complex trust techniques to increase their effectiveness; however, two strong assumptions have been overlooked. First, nodes could provide the wrong services due to malicious behaviours or malfunctions and insufficient accuracy. Second, the requester nodes usually cannot evaluate the received service perfectly. For this reason, a trust system should distinguish attackers from objects with poor performance and consider service evaluation errors. Simulation results prove that advanced trust algorithms are unnecessary for such scenarios with these deficiencies.

Keywords: trustworthiness management; Internet of Things; attacks on trust

Citation: Marche, C.; Nitti, M. Can We Trust Trust Management Systems? *IoT* **2022**, *3*, 262–272. https://doi.org/10.3390/iot3020015

Academic Editor: Hyun-Ho Choi

Received: 16 February 2022
Accepted: 21 March 2022
Published: 23 March 2022

1. Introduction

Many forms of social and economic transactions are built on an entity's expectation that one's transaction partner will not behave opportunistically and deviate from previously made agreements. This expectation is commonly conceptualized as trust: although we experience and rely on trust during our interactions in everyday life, trust can have many definitions, so that it is challenging to define it accurately.

Trust, defined as "the subjective probability by which an individual, the trustor, expects that another individual, the trustee, performs a given action on which its welfare depends" [1], can be regarded as the fabric of many scenarios involving interactions between entities. Whenever entity outcomes depend on others, its expectation regarding their benevolent versus malevolent intentions towards it critically shape its behaviour.

Trust is also typically regarded as essential to cooperation and it has thus been recognized as a critical factor for the Internet of Things (IoT) [2]. In the IoT scenario, the requester has the role of the trustor and has to trust that the provider, who is then the trustee, will provide the required service. However, misbehaving devices may perform several types of attacks for their own gain towards other IoT nodes: they can provide false services or false recommendations, and they can act alone or create a group of colluding devices to monopolize a class of services. If not handled adequately, attacks and malfunctions would outweigh any of the benefits of the IoT [3]. In this way, Wang et al. in [4] and Samuel et al. in [5] represent two important instances that take advantage of blockchain. In the first work, the authors provide trust management in vehicular crowdsourcing networks (VCNs); thanks to the immutability and transparency offered by blockchain, they guarantee high reliability for service producers and consumers. In addition, the second study illustrates an efficient mechanism to provide trust in smart grids towards interaction analysis between nodes and

storing information in the blockchain. Moreover, the importance of trustworthiness in edge computing is depicted in [6]. The authors investigate the problem of learning-aided computation for collaborative mobile edge computing (MEC) and propose an online mechanism based on the trust concept.

Trust management allows for multiple objects to share opinions about the trustworthiness of other devices. The goal of a trust management system (TMS) is then to guarantee that actions taken by entities in a system reflect their reputation values, and to prevent these values from being manipulated by unauthorized entities. Reputation is a measure derived from direct or indirect knowledge or experience on earlier interactions of entities and is used to assess the level of trust in an entity. TMSs play an important role in IoT as they help entities overcome perceptions of the uncertainty and risk of a transaction and promote users' acceptance and consumption of IoT services and applications.

The literature on TMSs concentrates on analysing the different phases involved in the process of managing the trust, its properties, the available techniques used to compose trust, the existing computation models and their effectiveness as defensive mechanisms against malicious attacks. Regardless of the proposed model, all the analysed papers are tested under two strong assumptions:

- A node provides the wrong service intentionally; however, well-behaving devices can show poor performance, due to errors, poor accuracy or technical problems in general. This problem is usually overlooked by trust algorithm models while it is indeed fundamental for them to be able to distinguish a malicious node from a poorly behaving one.
- The requesting node is able to accurately estimate the service received; notwithstanding this, requesters do not usually have ground truth information regarding the service, so that its evaluation is hardly accurate and even good services, and thus benevolent providers, can be poorly evaluated.

This means that even during a benevolent transaction, i.e., in a transaction involving only benevolent nodes, there could be two possible sources of errors, namely the entity providing the service and the requester evaluating it, without necessarily any malicious node involved. The presence of these errors can confuse TMSs and thus make it difficult to isolate only malicious nodes.

In this paper, we claim that these assumptions are too strong and should be lifted to a certain degree in order to better evaluate the TMSs; otherwise, it becomes difficult to estimate and appreciate their performance since simpler approaches can achieve better results. Our paper works in this direction and thus provides the following contributions:

- We briefly survey the main techniques used to form the overall trust out of multiple trust properties.
- We discuss the deficiencies of the scenarios used to test TMSs and show how, in such scenarios, advanced techniques are not necessary to deal with the trust management problem.

The rest of the paper is organized as follows: Section 2 presents the main techniques adopted to compute trust and discusses the deficiencies of the scenarios used to test them. In Section 3, we survey the common scenarios' features and show the possible types of attacks on trust. Section 4 illustrates how a basic model can outperform complex ones with an overly simplified scenario and presents how different models perform under different conditions. Finally, Section 5 provides some final remarks and a discussion about the importance of a complex scenario.

2. How to Construct Your Trust Model

The goal of any TMS is to identify malicious behaviours as soon as possible in order to isolate the nodes implementing such behaviours and discourage them from acting maliciously. With this goal in mind, many researchers have proposed different approaches

which make use of direct observation and indirect recommendations [7]. A trustworthiness management model can be classified following five dimensions [8]:

- The Composition refers to the trust attributes considered in the computation that are QoS or Social components;
- The scheme used, distributed, semi-distributed or centralized, is defined in the Propagation dimension;
- The Aggregation depicts how the trust attributes are aggregated to result in a trust degree;
- The Update concerns how trust is updated, periodically or when an event occurs;
- Finally, the Formation illustrates the overall trust as a single trust dimension or a combination of multiple ones.

Regardless of the different model proposed, all the papers follow a common creation process for trust management. At first, the authors present a scenario where objects could exchange information, describing the role of the requester and of the provider. Therefore, a trust model can be illustrated on the basis of the five dimensions previously described. Finally the model is tested under specific assumptions and compared with other trust models already accepted by the community. Regarding the simulations, the authors usually consider many types of attacks, intelligent or not, and show how the model behaves under different attack combinations.

Information about providers is gathered as direct experience when a node observes another node and itself calculates the trust value, or indirect recommendations; the providers therefore take trust information through other objects in the network. Both historical data and recommendations can be used in making decisions and, in particular, to select the best provider for the required services. Major trust management techniques investigated in the literature include approaches such as weighted sum, fuzzy logic, Bayes distribution, game theory, regression analysis or other machine learning methods.

However, all the trust models need some inputs and attributes that are associated with the main characteristics of the transaction (requester and provider) and the network [9]. The trust attributes concern QoS or social area, focusing on the objects or the connected network. Among QoS attributes, the leading role is represented by computation capabilities and transaction service quality, which consider features of the service providers and the evaluated service. Moreover, social attributes include credibility, centrality and a relationship factor, which take care of the nodes' role in the network and their reputation.

Thanks to the concept of reputation, the trust models measure the degree of trust and reduce risk in service transactions [10]. Different approaches can be used to propagate trust among objects. A first centralized method consists of a central entity that engages all the tasks of computation, trust management, storage and dissemination of information. On the other hand, in a decentralized approach each object itself manages the reputation of other nodes and its update, while in a semi-distributed technique, a set of nodes take care of the reputation from other nodes in the network and communicate with each other to maintain global trust.

The papers in the literature evaluate the models in terms of effectiveness and novelty but overlook essential deficiencies of the scenarios used to test TMSs. In the following, we present a research study evaluation and an important valuation for classifying most of the approaches in the state of the art. Figure 1 illustrates the distribution of the 43 analysed research papers over time according to their publishers (IEEE, Springer, Elsevier, MDPI, ACM, Hindawi, Intelligent Networks and Systems Society, The Institution of Engineering and Technology, and Bentham Science Publishers). We considered studies published online in recent years, from 2017 to 2021, which were published to provide a management method to guarantee trust in the IoT. The following string was defined to process the investigation:

- ("Trustworthiness" OR "Trust model" OR "Trust management" OR "Trust Technique") AND ("IoT" OR "Internet of Things").

The systematic review analyses all the resulting papers from our investigation in order to provide a response to the following analytical questions (AQ) in agreement with our paper's goals:

– AQ1: Can providers make errors when providing a service?
All the analysed trust management models rarely dealt with errors in service providing; among all the analysed papers, only [11–13] take into account this problem. Some errors involve benevolent nodes in a transaction: well-behaving devices could provide the wrong services due to errors, poor performance, poor accuracy or technical problems in general. The majority of the state-of-the-art models do not consider any type of error in the trust composition; even so, a few evaluate the simulations in a scenario with errors, not distinguishing provider errors from malicious behaviours. However, considering the probability of error in the trust composition could improve the performance of algorithms against errors.
– AQ2: How are the services evaluated? Is there any evaluation system?
When a node receives the requested service, it needs to check if it is consistent and then rate it. Therefore, the feedback has the role of being the source of a trust model, which takes care of the services and their processes. Unfortunately, the community barely examines the mechanism of evaluation, and usually, in their algorithms, the nodes provide perfect feedback in a discrete dimension, where 0 represents a poor service and 1 a good one. Among the analysed works, refs. [14–16] propose an evaluation system for receiving services, which is used to rate the interactions with the other nodes in the network. On the other side, most proposed models focus only on trust computation techniques and superficially treat feedback generation.
– AQ3: Do trust algorithms contemplate the possibility of the requester making errors in the evaluation?
As we previously remarked, the state-of-the-art works rarely evaluate errors: the models do not tackle any discrimination between service evaluation errors (requester side) and service providing errors (provider side). However, few authors contemplate a feedback algorithm in their algorithms, even though they presuppose perfect evaluation competence in the requesters. Examining the probability of errors in the evaluation service phase could improve the scenario, which would be more comparable to reality. Therefore, evaluating the feedback could increase the number of applications making use of a trust algorithm. In all the analysed papers, only [17–19] consider errors in the feedback evaluation process.

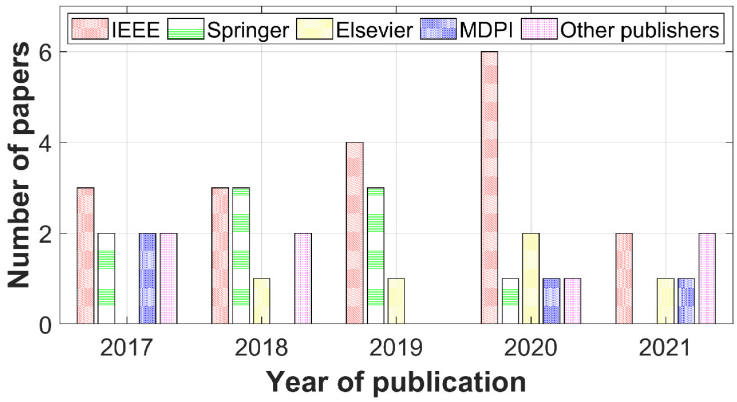

Figure 1. Distribution of research papers.

The majority of the state-of-the-art models exhibit the same assumptions: each requester can evaluate perfectly the received service, and any probability of error is contemplated in the algorithm creation. However, a trust model should consider a probability of error, whether from the requester side (service evaluation) or the provider side (sending a service).

In the following, we show how these assumptions are essential for a trust model. In addition, we illustrate how a simple trust algorithm could replace a complex one if it does not consider errors, with the best results in terms of malicious behaviour detection.

3. Scenario Description

Among all the proposed approaches, only a handful tested their trust management techniques on real-world IoT applications. Usually, due to the lack of real data concerning several aspects of object behaviour, the authors resort to simulations to test TMSs and to analyse their performance. The scenarios set up for the experimental evaluation share some common features:

- The number of devices in the network is established.
- The interaction model among nodes is defined, so that a device can send service requests to other nodes in the network. A pool of possible service providers are returned to the requester so that it can choose the most reliable one based on the TMS proposed.
- Every set of simulations defines the number of transactions carried out among the devices, so that the TMS is able to converge and reach its steady-state.
- Additional features are considered in the scenario based on particular components used to compute trust in the proposed system: e.g., if social aspects are taken into account, metrics such as centrality and friendship relations are added to the network.
- The percentage of malicious devices in the network is determined and sometimes used as a variable to test and understand the upper limit of malicious nodes that the trust system is able to identify.

However, malicious nodes can implement several types of behaviours and trust-related attacks, which represent the different solutions they adopt to avoid being detected. Among the possible types of attack, the most tested ones are [7]:

Malicious with Everyone (ME): this is the simplest attack and it is used as a baseline to test TMSs. A malicious node acts maliciously with everyone, so that it always provides bad services and recommendations, regardless of the requester [20].

Discrimination Attack (DA): a malicious node modifies its behaviour based on the service requester. This means that a node can discriminate nodes based on some characteristics, such as their typology or some social metrics. As a result, some devices can consider the node as benevolent while others can label it as malevolent [21].

On–Off Attack (OOA): a node periodically changes its behaviour, by alternately being benevolent (ON) and malevolent (OFF). During the ON state, the node builds up its trust, which is then used to attack the network [22].

Opportunistic Service Attack (OSA): a malicious node provides good services only when it senses that its trust reputation is dropping. In this way, the node tries to maintain an acceptable level of trust in order to still be selected as a service provider [23].

The general scenario is then represented by a node requesting a particular service; every node in the network can provide one or more services, so that a service discovery component in the network is needed to return to the requester a list of potential providers. At this point, the TMS has to help the requester to select one of the providers based on their level of trust. The trust level is usually computed based on the previous interactions among the nodes or by requesting recommendations to neighbouring nodes. Indeed, after every transaction, the requester assigns feedback to the selected provider to evaluate the service.

All the analysed models are then usually evaluated mainly in terms of success rate, i.e., considering the percentage of successful transactions compared to the number of total transactions. Indeed, every attack has its own impact on the success rate which also depends on the TMS implemented: some algorithms are specifically designed to identify certain attacks, while others aim to provide an acceptable level of defence against most attacks. For this reason, the impact of an attack depends on the proposed model and, most importantly, on the scenario used to test it. This is specifically the goal of the paper, i.e., to

highlight which elements must be described in order to better appreciate the performance of a model.

However, all the features described before are usually considered as ideals, such as that a provider has infinite resources and always has the requested data available or that it can, in any case, retrieve the data with excellent accuracy and therefore without the possibility of making errors. Similarly, there are no communication errors or intermediate nodes that could alter the content of the service. Finally, once a piece of data has been requested, the requester is perfectly capable of evaluating the quality of the received service and provide an accurate evaluation.

Figure 2 provides a simple example of a generic network and the main steps of a transaction between two nodes: node *i* requesting a service and node *j* providing it. The red lines highlight two of the possible sources of errors: indeed, even if both node *i* and node *j* are benevolent, the transaction could become unsuccessful if node *j* has difficulties acquiring the service with the requested accuracy or if node *i* evaluates it poorly.

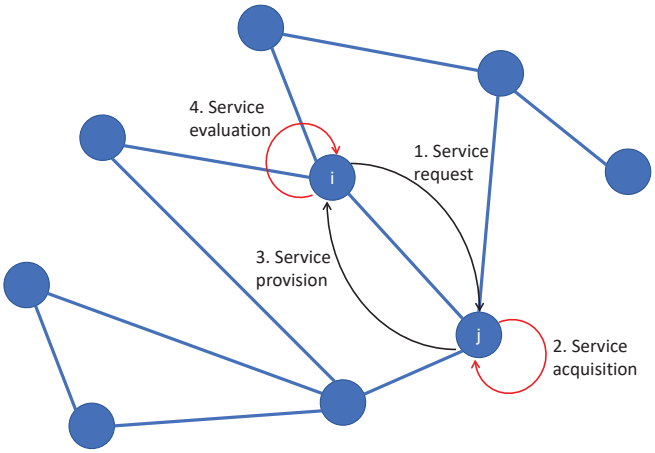

Figure 2. Trust management system—general scenario.

The goal of any TMS is to compute and list the trust level of all the available providers and to able to distinguish between malicious nodes and possible errors. This step is fundamental to help the requester to identify the most reliable node that requires the service and to avoid any malicious providers. However, we argue that the complexity of the TMSs proposed in the literature and the scenarios used to test them do not match, and therefore that they are not able to show their true performance. In the following section, we show how a simplified approach is able to outperform well-known TMSs available in the literature.

4. Let Us Test It

In this section, we test and compare different models in order to show the importance of accurately describing the scenario used to test the models. Our tests show that advanced techniques are not necessary in the scenarios commonly used in the literature. On the other hand, complex solutions become essential when models do not overlook error probability in service providing or evaluation.

4.1. Simulation Setup

In order to test the different models, we needed an unbiased network in which objects can communicate with each other. For this we resorted to the well-known IoT/SIoT dataset available here (http://www.social-iot.org/index.php?p=downloads, accessed on 10 December 2021) and to the query generation model described in [24]. The network consists of 16,216 devices owned by 4000 users from the municipality of Santander (Spain). The authors provide a set of applications that could be requested by the user, in which every app makes use of one service or more. Each object can offer or request services useful

for the requested application. Moreover, a query model of interactions between devices was offered and used in order to best fit a real IoT scenario. We decided to consider a sub-network of around 800 devices to increase the probability of interaction. Each device makes a request based on the query generation model and waits for a service from a set of possible providers. The provider was selected according to the TMS model.

Moreover, the provider can implement two main behaviours: one is always benevolent, providing good services, and the other one is malicious. The malicious nodes were designed according to Section 3. The ME behaviour represents the simplest attack, and it provides only scarce services. Similarly, nodes performing DA attacks act maliciously but modify their behaviour based on the service requester and provide bad services to a subset of devices; e.g., in the social trust models, the nodes provide good services only to friends, while in the IoT scenario only to a random subset of nodes considering the same number of the previous SIoT approach. The other two types of attacks are, respectively, OOA and OSA. An OOA node changes its behaviour every two interactions, whereas the last attack tries to maintain a trust level higher than 90% of its maximum value.

4.2. Experimental Evaluation

We used as a baseline to compare other models the simplest approach possible (labelled as Basic Approach in the figures): this approach makes use only of the requester experience, i.e., the requester does not make use of recommendations from its neighbours, and only the last interaction with the provider is considered in order to evaluate its trust. This means that, since the general scenario used to test TMSs only considers the ability for the requester to perfectly evaluate the service received, the trust can only assume two values: 1 if the service was good and 0 otherwise. Moreover, if the trust of a provider reaches 0, i.e., the last service was not evaluated positively by the requester, the provider will no longer selected.

We first evaluated the performance of the basic approach by analysing the success rate, i.e., the ratio between the number of successful transactions and the total number of transactions, or by directly calculating the level of trust computed by a node. We compared its performance with well-known models in the research community. In the first work [20], Nitti et al. proposed a trust model designed for the Social IoT (SIoT) scenario. The authors propose a decentralized architecture in which each node computes the trust values of providers on the basis of its own experience and on the opinion of the neighbours. The trust is evaluated considering quality of service (QoS) parameters, such as transaction service quality and computation capability, and social metrics, such as centrality, relationship factors and credibility. Each node computes the trust value of providers applying a static weighted sum, considering all the mentioned parameters and feedback of past interactions. Another work designed for the social scenario is presented in [25]. Chen et al. illustrate a scheme for service access based on recommendations. The authors considered both QoS metrics, such as energy status, quality reputation and social relationships. In the scheme, each node has its own vision of the network and relies on the recommendations from its friends to speed up the evaluation of trust. The final trust values are computed based on the parameters and past performances toward a weighted sum. The other two models, i.e., those from Adewuyi et al. [26] and Mendoza et al. [27], are designed for a generic IoT scenario. In the first work, the authors propose a subjective approach to evaluate and manage trust between nodes in collaborative applications. A trust aggregation function based on a weighted sum is used to calculate trust values. A concept based on trust decay is introduced to address the issue of trust update, and many resources, such as recommendations and past experience, are used for the computation. In addition, in the second study, the authors present a distributed trust management based on only direct information acquired by communication between nodes. The model considers only service quality attributes; it assigns positive trust value to the node that collaborates in service providing and a negative trust value to the node that refuses to cooperate. No social attributes are considered, and the model proposes mitigating attacks towards a reward and punishment mechanism and analysing QoS attributes.

Figure 3 shows the transaction success rate when malicious nodes implement the ME attack, i.e., nodes that act maliciously with everyone providing inadequate services, under the scenario conditions described in Section 3 at varying percentages of the malicious nodes. We considered that 10%, 20% and 30% of the nodes are malicious and that every time a requester is looking for a service there is an average of 60 possible providers. All the models had a good reaction to this attack and were able to achieve a high success rate, always higher than 95%, 93% and 80%. However, it is clear that the basic approach is able to outperform the other approaches: this happens since as soon as a requester detects a provider implementing the ME attack, that requester is labelled as malicious, its trust reaches 0 and it is never selected again. Moreover, we can see how models have different behaviours for various percentages of errors, and each algorithm performs differently with respect to others for several percentages of errors.

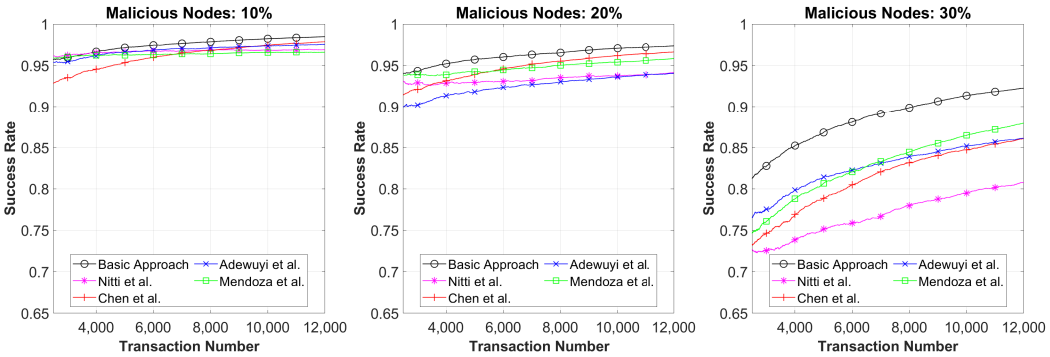

Figure 3. Transaction success rate for different trust management models for the ME attack at varying percentages of the malicious nodes.

As the first result, the simplest algorithm would seem to overcome the complex ones; this is possible because most trust models overlook the important issue of errors in the scenario used to test them. Benevolent providers could supply inadequate services due to malfunctions or scarce accuracy; in the same way, requesters are not able to accurately evaluate received services, and they could consider a good service as a bad one. To test this critical condition, we inserted an error percentage in which a node could make an error in service providing or service evaluation. Moreover, an important parameter that has to be considered is represented by the providers' availability. Figure 4 shows the performance of the trust models based on the error percentage, with different averages of available providers. Each graph exhibits the success rate after 12,000 transactions for all the evaluated models at the variation of error percentage. The results illustrate how the basic approach performs well for a high number of providers, even with the increasing percentage of error. However, the lower the average of the providers, the higher the probability of the blocking of the basic approach. The blocking problem is represented by the number of malicious nodes discarded by the simplest algorithm, which does not allow for the selection of malicious providers, i.e., the requesters do not select any provider. The probability of block increases with reductions in the number of providers, regardless of the error percentage. With the minimum number of available providers, that is, only one provider, the smallest error level provokes the interruption of the algorithm and its uselessness.

In order to overcome the blocking problem, we integrated the simplest algorithm with a tolerance mechanism: each requester considers a window of past interactions, which can be used to evaluate the trust of the providers. Then the trust is calculated as the average of the past feedback, i.e., evaluations of the historical interactions. Therefore, to improve the functioning of the basic approach, we needed to increase the simple algorithm complexity. The larger the window, the higher the probability of attack by malicious nodes; otherwise, the narrower the window, the higher the probability of errors. We next wanted to show the behaviour of the new basic approach with different dimensions of transaction windows

considering an average of 20 providers. Figure 5 illustrates how the models respond to a network with a mix of all the attacks analysed. The results are shown in terms of transaction success rate considering 10% of malicious nodes for each type of attack, for a total of 40% malicious nodes. The graph depicts how the complex approaches are able to converge well and better than the basic approach with different dimensions of the window. Until 15% of error percentage, the narrowest window operates well in terms of success rate, whereas by increasing the error the best results were revealed with the larger window. In any case, the basic approach suffers from complex attacks which only the well-designed trust models can overcome. By analysing which attacks had a higher impact, we can see how the basic approach better manages simple attacks; however, it suffers from smart attacks, such as OSA and OOA, which are not sufficiently detected.

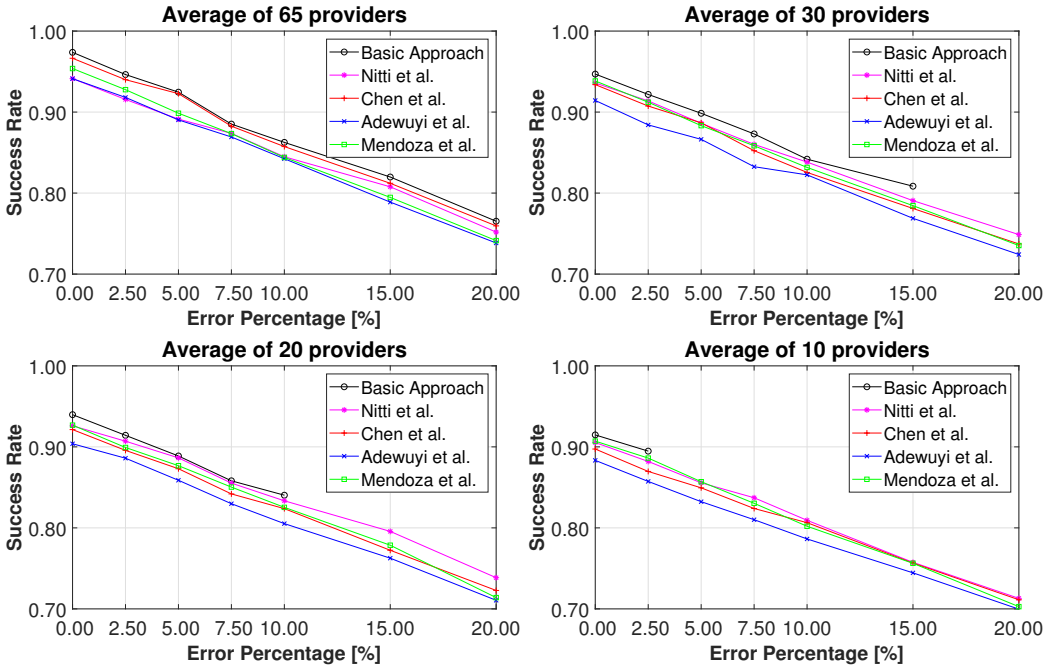

Figure 4. Transaction success rate for different averages of providers and error percentages.

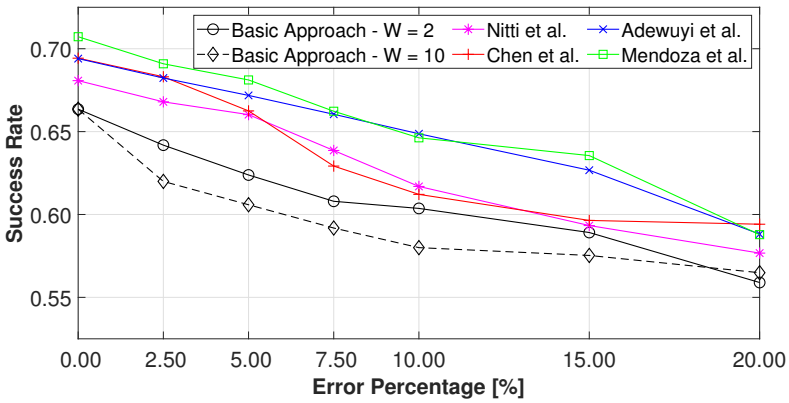

Figure 5. Transaction success rate with all types of malicious attacks.

5. Conclusions and Future Directions

This paper discussed and analysed two important deficiencies of the scenarios used to test TMSs. The first concerns errors in providing services: a node provides the wrong service due to malicious behaviours or malfunctions and poor accuracy (errors in service providing). Moreover, the requester node usually is not able to perfectly evaluate the received service; thus, benevolent providers could be poorly evaluated (errors in service

requesting). We have shown how these essential assumptions must be considered in the scenario used to test a trust model and how any complex algorithm is necessary otherwise.

We compared and evaluated the performance of the simplest trust mechanism and four well-known TMSs, where two of them were designed for the SIoT and the other two for a generic IoT scenario. The first results showed how the simplest is the best algorithm in terms of success rate when the scenario overlooks the two ignored errors in service providing and requesting and when the malicious nodes implement the simplest attack ME. By contrast, when the scenario becomes nearest to a real one, the simplest algorithm must be replaced with a complex model. Then we updated the scenario according to a percentage of error in service requesting and providing and considered different averages of available providers. The experiment illustrates the importance of declaring the scenario and how complex algorithms are needed to detect smart attacks.

Therefore, the statement of a real scenario is important to understand which metrics or parameters are more suitable: e.g., TMS could perform well in a scenario with a higher number of providers and a low error rate; otherwise, other models could work with a high percentage of specific smart attacks. In this way, a model could be selected instead of another that shows the worst performance in an evaluated scenario. The best results for TMSs are obtained with the specific scenario set to design them.

In conclusion, the designing process of a TMS model should consider several parameters. First, the average number of selected providers is essential to understand how robust the model is against errors; e.g., in models with a low average of providers, a high percentage of errors impacts more than a model with a high average of providers. Moreover, taking into consideration errors in providing and evaluating services helps the network owner to best fit the model with the utilised scenario. In this way, a trust model could be used instead of another in a scenario with less or more accurate devices. Finally, a complete description of the attacks evaluated by the model is useful for selecting the right attack detection algorithm for the evaluated scenario.

Author Contributions: Conceptualization, C.M. and M.N.; methodology, C.M. and M.N.; software, C.M.; validation, C.M. and M.N.; formal analysis, C.M. and M.N.; investigation, C.M. and M.N.; resources, C.M. and M.N.; data curation, C.M. and M.N.; writing—original draft preparation, C.M. and M.N.; writing—review and editing, C.M. and M.N.; visualization, C.M. and M.N.; supervision, M.N.; project administration, C.M. and M.N.; funding acquisition, M.N. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 957228.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used in this study are openly available in http://social-iot.org/index.php?p=downloads (accessed on 10 December 2021) at [10.1016/j.comnet.2020.107248].

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Gambetta, D. Can we trust trust? In *Trust: Making and Breaking Cooperative Relations*; Citeseer: Princeton, NJ, USA, 2000; Volume 13, pp. 213–237.
2. Pourghebleh, B.; Wakil, K.; Navimipour, N.J. A comprehensive study on the trust management techniques in the Internet of Things. *IEEE Internet Things J.* **2019**, *6*, 9326–9337. [CrossRef]
3. Li, K.; Tian, L.; Li, W.; Luo, G.; Cai, Z. Incorporating social interaction into three-party game towards privacy protection in IoT. *Comput. Netw.* **2019**, *150*, 90–101. [CrossRef]
4. Wang, D.; Chen, X.; Wu, H.; Yu, R.; Zhao, Y. A Blockchain-Based Vehicle-Trust Management Framework Under a Crowdsourcing Environment. In Proceedings of the 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), Guangzhou, China, 10–13 November 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1950–1955.

5. Samuel, O.; Javaid, N.; Khalid, A.; Imrarn, M.; Nasser, N. A trust management system for multi-agent system in smart grids using blockchain technology. In Proceedings of the Globecom 2020—2020 IEEE Global Communications Conference, Taipei, Taiwan, 7–11 December 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–6.
6. Li, Y.; Wang, X.; Gan, X.; Jin, H.; Fu, L.; Wang, X. Learning-aided computation offloading for trusted collaborative mobile edge computing. *IEEE Trans. Mob. Comput.* **2019**, *19*, 2833–2849. [CrossRef]
7. Altaf, A.; Abbas, H.; Iqbal, F.; Derhab, A. Trust models of internet of smart things: A survey, open issues, and future directions. *J. Netw. Comput. Appl.* **2019**, *137*, 93–111. [CrossRef]
8. Guo, J.; Chen, R.; Tsai, J.J. A survey of trust computation models for service management in internet of things systems. *Comput. Commun.* **2017**, *97*, 1–14. [CrossRef]
9. Azzedin, F.; Ghaleb, M. Internet-of-Things and information fusion: Trust perspective survey. *Sensors* **2019**, *19*, 1929. [CrossRef] [PubMed]
10. Sharma, A.; Pilli, E.S.; Mazumdar, A.P.; Gera, P. Towards trustworthy Internet of Things: A survey on Trust Management applications and schemes. *Comput. Commun.* **2020**, *160*, 475–493. [CrossRef]
11. Xia, H.; Xiao, F.; Zhang, S.S.; Hu, C.Q.; Cheng, X.Z. Trustworthiness inference framework in the social Internet of Things: A context-aware approach. In Proceedings of the IEEE Infocom 2019—IEEE Conference on Computer Communications, Paris, France, 29 April–2 May 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 838–846.
12. Boudagdigue, C.; Benslimane, A.; Kobbane, A.; Liu, J. Trust management in industrial Internet of Things. *IEEE Trans. Inf. Forensics Secur.* **2020**, *15*, 3667–3682. [CrossRef]
13. Abdalzaher, M.S.; Muta, O. A game-theoretic approach for enhancing security and data trustworthiness in IoT applications. *IEEE Internet Things J.* **2020**, *7*, 11250–11261. [CrossRef]
14. Suhail, S.; Hong, C.S.; Lodhi, M.A.; Zafar, F.; Khan, A.; Bashir, F. Data trustworthiness in IoT. In Proceedings of the 2018 International Conference on Information Networking (ICOIN), Chiang Mai, Thailand, 12–14 January 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 414–419.
15. Chen, G.; Zeng, F.; Zhang, J.; Lu, T.; Shen, J.; Shu, W. An adaptive trust model based on recommendation filtering algorithm for the Internet of Things systems. *Comput. Netw.* **2021**, *190*, 107952. [CrossRef]
16. Su, R.; Sfar, A.R.; Natalizio, E.; Moyal, P.; Song, Y.Q. PDTM: Phase-based dynamic trust management for Internet of things. In Proceedings of the 2021 International Conference on Computer Communications and Networks (ICCCN), Athens, Greece, 19–22 July 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 1–7.
17. Lv, Z.; Han, Y.; Singh, A.K.; Manogaran, G.; Lv, H. Trustworthiness in industrial IoT systems based on artificial intelligence. *IEEE Trans. Ind. Inform.* **2020**, *17*, 1496–1504. [CrossRef]
18. Ma, Z.; Liu, L.; Meng, W. Towards multiple-mix-attack detection via consensus-based trust management in IoT networks. *Comput. Secur.* **2020**, *96*, 101898. [CrossRef]
19. Salimitari, M.; Bhattacharjee, S.; Chatterjee, M.; Fallah, Y.P. A prospect theoretic approach for trust management in IoT networks under manipulation attacks. *ACM Trans. Sens. Netw. (TOSN)* **2020**, *16*, 1–26. [CrossRef]
20. Nitti, M.; Girau, R.; Atzori, L. Trustworthiness management in the social internet of things. *IEEE Trans. Knowl. Data Eng.* **2014**, *26*, 1253–1266. [CrossRef]
21. Wang, D.; Muller, T.; Liu, Y.; Zhang, J. Towards robust and effective trust management for security: A survey. In Proceedings of the 2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications, Beijing, China, 24–26 September 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 511–518.
22. Caminha, J.; Perkusich, A.; Perkusich, M. A smart trust management method to detect on-off attacks in the internet of things. *Secur. Commun. Netw.* **2018**, *2018*, 6063456. [CrossRef]
23. Chen, R.; Guo, J.; Bao, F. Trust management for SOA-based IoT and its application to service composition. *IEEE Trans. Serv. Comput.* **2014**, *9*, 482–495. [CrossRef]
24. Marche, C.; Atzori, L.; Pilloni, V.; Nitti, M. How to exploit the Social Internet of Things: Query Generation Model and Device Profiles' Dataset. *Comput. Netw.* **2020**, *174*, 107248. [CrossRef]
25. Chen, Z.; Ling, R.; Huang, C.M.; Zhu, X. A scheme of access service recommendation for the Social Internet of Things. *Int. J. Commun. Syst.* **2016**, *29*, 694–706. [CrossRef]
26. Adewuyi, A.A.; Cheng, H.; Shi, Q.; Cao, J.; MacDermott, Á.; Wang, X. CTRUST: A dynamic trust model for collaborative applications in the Internet of Things. *IEEE Internet Things J.* **2019**, *6*, 5432–5445. [CrossRef]
27. Mendoza, C.V.L.; Kleinschmidt, J.H. A distributed trust management mechanism for the Internet of things using a multi-service approach. *Wirel. Pers. Commun.* **2018**, *103*, 2501–2513. [CrossRef]