A multi-dimensional trust-model for dynamic, scalable and resources-efficient trust-management in social internet of things

Wafa Abdelghani

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *l'Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier)*
Cotutelle internationale *Université de Sfax (FSEGS)*

Présentée et soutenue le *Date de soutenance* **par :**

Wafa ABDELGHANI

A Multi-Dimensional Trust-Model for Dynamic, Scalable and Resources-efficient Trust-Management in Social Internet of Things

JURY

Ikram AMOUS	Professeur, Université de Sfax	Directrice de thèse
Florence SEDES	Professeur, Université de Toulouse 3-Paul Sabatier	Directrice de thèse
Corinne Amel ZAYANI	Maître de conférence, Université de Sfax	Examinateur
Jean-Marc PIERSON	Professeur, Université de Toulouse 3-Paul Sabatier	Examinateur
Raoudha BEN JEMAA	Maître de conférence, Université de Sousse	Examinateur
Vasile-Marian SCUTURICI	Professeur, Université de Lyon	Examinateur
Thierry DELOT	Maître de conférence, Université de Valencienne	Rapporteur
Mahdi KHAMEKHEM	Maître de conférence, Université de Sfax	Rapporteur

École doctorale et spécialité :
MITT : Domaine STIC : Intelligence Artificielle
Unité de Recherche :
Institut de Recherche en Informatique de Toulouse (UMR 5505)
Directeur(s) de Thèse :
Florence SEDES et *Ikram AMOUS*
Rapporteurs :
Thierry DELOT et *Mahdi KHAMEKHEM*

Acknowledgement

Firstly, I would like to thank my family for supporting me spiritually throughout all the years of my studies, for accepting the fact that I am abroad past years and for all their love and encouragement. For my parents who raised me with a love of science and supported me in all my orientations.

I would like to express my sincere gratitude to my advisors Prof. Florence SEDES and Prof. Ikram AMOUS for their continuous support of my PhD study and related research, for their patience, motivation, and immense knowledge. Their guidance helped me in all the time of research and writing of this thesis. Besides my advisors, I would like to thank my thesis co-advisor: Dr Amel Corinne ZAYANI, for her insightful comments and encouragement, but also for the hard question which motivated me to widen my research from various perspectives.

My sincere thanks also go the reviewers, Pr. Thierry DELOT, and Pr. Mahdi KHEMAKHEM, to whom I associate the JURY members Pr. Jean-Marc PIERSON, Dr. Vasile-Marian SCUTURICI, Dr. Raoudha BEN JEMAA and Pr. Jean-Luc BEYLOT for taking time to evaluate my work.

I gratefully acknowledge the funding received towards my PhD from the PHC Utique program of the French Ministry of Foreign Affairs and Ministry of higher education and research and the Tunisian Ministry of higher education and scientific research in the CMCU project number 18G1431.

I thank my colleagues at IRIT for the stimulating discussions, for the sleepless nights we were working together before deadlines, and for all the fun we have had in those last years.

Last but not least, I would like to thank the one that he believed in me. I would like to thank you for being part of my life and my achievements.

I would like to say thank you to all who is dear to me.

Summary

The Internet of Things (IoT) is a paradigm that has made everyday objects intelligent by giving them the ability to connect to the Internet, communicate and interact. The integration of the social component in the IoT has given rise to the Social Internet of Things (SIoT), which has overcome various issues such as interoperability, navigability and resource/service discovery. In this type of environment, participants compete to offer a variety of attractive services. Some of them resort to malicious behaviour to propagate poor quality services. They launch so-called Trust-Attacks (TA) and break the basic functionality of the system.

Several works in the literature have addressed this problem and have proposed different trust-models. Most of them have attempted to adapt and reapply trust models designed for traditional social networks or peer-to-peer networks. Despite the similarities between these types of networks, SIoT ones have specific particularities. In SIoTs, there are different types of entities that collaborate: humans, devices, and services. Devices can have very limited computing and storage capacities, and their number can be as high as a few million. The resulting network is complex and highly dynamic, and the impact of Trust-Attacks can be more compromising.

In this work, we propose a Multidimensional, Dynamic, Resources-efficient and Scalable trust-model that is specifically designed for SIoT environments. We, first, propose features to describe the behaviour of the three types of nodes involved in SIoT networks and to quantify the degree of trust according to the three resulting Trust-Dimensions. We propose, secondly, an aggregation method based on Supervised Machine-Learning and Deep Learning that allows, on the one hand, to aggregate the proposed features to obtain a trust score allowing to rank the nodes, but also to detect the different types of Trust-Attacks and to counter them. We then propose a hybrid propagation method that allows spreading trust values in the network, while overcoming the drawbacks of centralized and distributed methods. The proposed method ensures scalability and dynamism on the one hand, and minimizes resource

consumption (computing and storage), on the other. Experiments applied to synthetic data have enabled us to validate the resilience and performance of the proposed model.

Keywords

Trust Management –Internet of Things - Social Networks – Social Internet of Things – Trust-Attacks.

Resumé

L'internet des Objets (IoT) est un paradigme qui a rendu les objets du quotidien, intelligents en leur offrant la possibilité de se connecter à Internet, de communiquer et d'interagir. L'intégration de la composante sociale dans l'IoT a donné naissance à l'Internet des Objets Social (SIoT), qui a permis de surmonter diverse problématiques telles que l'interopérabilité et la découverte de ressources. Dans ce type d'environnement, les participants rivalisent afin d'offrir une variété de services attrayants. Certains d'entre eux ont recours à des comportements malveillants afin de propager des services de mauvaise qualité. Ils lancent des attaques, dites de confiance, et brisent les fonctionnalités de base du système. Plusieurs travaux de la littérature ont abordé ce problème et ont proposé différents modèles de confiance. La majorité d'entre eux ont tenté de réappliquer des modèles de confiance conçus pour les réseaux sociaux ou les réseaux pair-à-pair. Malgré les similitudes entre ces types de réseaux, les réseaux SIoT présentent des particularités spécifiques. Dans les SIoT, nous avons différents types d'entités qui collaborent, à savoir des humains, des dispositifs et des services. Les dispositifs peuvent présenter des capacités de calcul et de stockage très limitées et leur nombre peut atteindre des millions. Le réseau qui en résulte est complexe et très dynamique et les répercussions des attaques de confiance peuvent être plus importantes.

Nous proposons un nouveau modèle de confiance multidimensionnel, dynamique et scalable, spécifiquement conçu pour les environnements SIoT. Nous proposons, en premier lieu, des facteurs permettant de décrire le comportement des trois types de noeuds impliqués dans les réseaux SIoT et de quantifier le degré de confiance selon les trois dimensions de confiance résultantes. Nous proposons, ensuite, une méthode d'agrégation basée sur l'apprentissage automatique et l'apprentissage profond qui permet d'une part d'agréger les facteurs proposés pour obtenir un score de confiance permettant de classer les noeuds, mais aussi de détecter les types d'attaques de confiance et de les contrer. Nous proposons, ensuite, une méthode de propagation hybride qui permet de diffuser les valeurs de confiance dans le réseau,

tout en remédiant aux inconvénients des méthodes centralisée et distribuée. Cette méthode permet d'une part d'assurer la scalabilité et le dynamisme et d'autre part, de minimiser la consommation des ressources. Les expérimentations appliquées sur des de données synthétiques nous ont permis de valider le modèle proposé.

Mots clés

Gestion de la confiance – Internet des Objets – Réseaux sociaux – Internet des Objets Social – Attaque de confiance.

Contents

List of Figures

List of Tables

Introduction

General context

The Internet of Things (IoT) is an emerging paradigm in which everyday objects become connected to each other and to the Internet. These objects, equipped with sensors and actuators, become able to anticipate users' needs and promise better comfort and quality of life to its users. Nevertheless, still in its infancy, the IoT raises various issues such as the heterogeneity of devices, networks and communication protocols (Singh et al. (2014)). This heterogeneity prevents IoT objects and devices from communicating, interacting and exchanging information. The Social Internet of Things (SIoT) is a variant of the IoT that addresses this issue and establishes interoperability by allowing people and objects to interact in a social framework based on a new type of navigation (Atzori et al. (2011)).

The SIoT is the result of an evolutionary process that has affected modern communication through the integration of the social component into the Internet of Things. Intelligent objects in the IoT have evolved into pseudo-social objects able to exhibit pseudo-social behavior, and then into social objects able to autonomously establish relationships with other objects, join communities and build their own social networks that may differ from those of their owners (Nitti et al. (2014b)). To do so, social IoT objects use location, category or history of interactions with other objects.

The adoption of such a vision is a promising trend that offers multiple advantages. First of all, navigability and discovery of services/resources are optimized by reducing their scope to a manageable social network (Ali (2015)). Second, scalability is guaranteed like in traditional social networks (Atzori et al. (2011)). Finally, a wealthier and more varied data source becomes available as it comes from a set, a network or a community of users. The volume and variety of the resulting contextual and social data have improved the intelligence and adaptability of IoT services (Ali (2015)).

Main issues and problematic

Despite this significant progress, the consumption of IoT services remains below expectations. Users remain suspicious and confused about adopting this new paradigm. Indeed, in this type of environment, the various actors compete to offer a variety of attractive services. Some of them resort to malicious behaviour to propagate poor-quality services. To do so, they launch different types of attacks called "Trust Attacks (TAs)" (Abdelghani et al. (2016), Bao and Chen (2012a), Chen et al. (2016b)).

Unlike security attacks, in which the attacker aims to disrupt the communication network or access data without authorization, a Trust Attack aims to distort the Reputation System. By launching a TA, the malicious user attempts to boost his or her own reputation, destroy the reputation of other legitimate users, or enhance the reputation of other malicious users. This type of attack can be dangerous in a collaborative environment such as SIoT, where exchanges and interactions are mainly based on feedback and reputation. In addition to their vulnerability to Trust Attacks, SIoT networks are complex and multi-party, as they involve three types of actor nodes. (i) User-type nodes mainly play two roles. They are both service/resource providers and service requesters/recommenders. Feedback from the service requester, represented in the form of ratings assigned after each interaction, helps to determine the "Reputation" of each node and assists other users in deciding whether or not to invoke a given service. To interact, provide and invoke

services, users will use (ii) device-type nodes. (ii) Devices, also called objects or things, are technical artefacts intended to furnish an interface between the digital world and the physical world. They provide information, knowledge or data about the physical entity they control and are able to change the physical state of a physical entity. To accomplish the tasks required, device type nodes will use (iii) service type nodes. Services are an abstraction allowing the construction of complex software systems. They are the basis of most current systems and are responsible for providing the functionalities of IoT devices (for example the detection function for a sensor).

These three types of nodes collaborate and participate in every interaction. The quality of the interaction depends on the actor-user but also on the quality of the device he uses and the quality of the service he provides through this device, thus implying three Trust-Dimensions to manage.

Finally, SIoT environments are characterized by various constraints linked either to the nature of the network such as dynamism and large number of devices or to the specificity of IoT devices such as limited computing and storage capacities. The establishment of a robust and powerful trust management mechanism in such a dynamic and constrained environment becomes challenging.

Background and related works

Trust Management Mechanisms (TMM) are designed to prevent malicious behaviour to ensure the appropriate functioning of a system. TMMs have been integrated into various types of open and distributed networking systems such as e-commerce systems, peer-to-peer networks, social networks and wireless sensor networks. They do not affect the basic functionality of a system but form a layer that is superimposed on the original system. A Trust Management Mechanism must provide methods for assessing and calculating the degree of trust, as well as for propagating, storing and updating these values. It consists of three main phases : (i) the Trust Building phase, (ii)

the Propagation phase and (iii) the Update phases. (i) The Trust Building phase consists in proposing a "Trust Assessment Model (TAM)". A TAM makes it possible to assess the system-nodes/actors based on different criteria and to estimate the degree of trust that can be granted to them. It must therefore be able to attribute a low trust-degree to malicious nodes and a high one to legitimate nodes. To do so, a TAM relies on the interaction history of each node and generally proceeds in two steps. The first step is "(i.a) the Trust-Composition step which consists of selecting Trust-Features (TF). These-Features also referred to in the literature as factors, indicators, parameters or attributes, represent descriptors that make it possible to establish the criteria considered to compare and evaluate the network nodes. These criteria depend on the nature of the network, the risks involved, the nature of the network actors (nodes) and the purpose of the trust. The TAM features must allow quantifying these criteria (abstract concepts) into concrete and measurable features. The authors of the literature have drawn inspiration from the Trust-Features proposed in traditional social networks such as "profile similarity" or "centrality in the network". These features allow the assessment of user-type nodes, but not device and service nodes. Moreover, only a minority of related work has focused on the detection of Trust-Attacks and proposed Trust-Features allowing to depict the behaviour of malicious nodes. The great majority neglected this aspect and are only interested and proposed features allowing to rank the best nodes in the network, leaving the malicious ones free to carry out their malicious behaviour.

Once the Trust-Features have been determined, the second step of a TAM is "(i.b) the Trust-Aggregation step" which consists in selecting the appropriate method to combine the values of the different factors and obtain the final trust value for each node allowing other nodes to deciding whether or not to interact with a given node. The authors of the literature have used Probability, Fuzzy Logic or Combinatorial Logic. But the most commonly used method is the Weighted Mean, which consists of assigning weights to the different selected features. The disadvantage of this method mainly concerns the choice of weights and thresholds. The latter is usually set empirically

and do not allow the detection of different types of trust attacks. If we take the example of the Self-Promoting Attack (SPA), in which a user tries to self-promote his reputation by attributing high rating to himself under a false identity. The "similarity" factor that is used to measure the similarity between two users can help reveal that it is the same user. It is therefore of paramount importance in the case of SPA attacks, whereas it is of no interest for other types of TAs, such as BSA attacks (a collusion attack where attackers help each other promote each other's reputation).

(ii) The Trust-Propagation phase consists in selecting a method to diffuse, in the network, the trust values or scores computer in the Trust-Building phase. The authors in literature are divided between those who opt for the decentralized method in which each IoT device makes its own trust-calculations and those who apply the centralized method in which a central entity is responsible for making the necessary calculations for all the nodes of the network. The main disadvantage of the centralized method is that it does not allow the scalability while SIoT networks involve a large number of dynamic objects. The disadvantage of the decentralized method is that it is not suitable for SIoT devices, as they are constrained in storage and computing capacity, and unable to perform the complex computations required to manage the trust. Finally, for (iii) the Trust-Update phase, two methods are proposed in the literature. In the Time-Driven Method, updates are done periodically. The problem with this method is that it is difficult to set the granularity of the period. The majority of authors in the literature have opted for the Event-Driven method, in which updates are performed each time an event occurs (one node joins or leaves the network, two nodes interact, ... etc.). Its advantage is that it allows real-time Trust-Management. The disadvantage of this method is that it generates a large number of updates which does not comply with the constrained nature of IoT devices. For this reason, some authors propose to combine the Event-Driven update method with a Storage-Management Strategy.

Thus, the main objective of this work is to propose a TMM specifically designed for SIoT environments. To achieve this objective, the proposed TMM

must be able to (1) take into account the different Trust-Dimension involved (user trust-dimension, device trust-dimension and service trust-dimension), (2) ensure resilience against all types of Trust Attacks, (3) respect the different constraints specific to SIoT environments (scalability, dynamism, limited resources). The choice of appropriate methods for each phase of the TMM will enable the design of a Trust Management Mechanism that is appropriate and adapted to SIoT environments. The propagation, storage and update phases will ensure scalability, dynamism and resources-efficiency, while the Trust-Building phase will manage the different trust-dimension and guarantee Trust-Attacks resiliency.

Methodology and contributions

The preliminary of this work can be summarized as follows:

- (1) Definition of the "Trust" concept and its multiple properties, starting from sociological and psychological definitions, to its adoption in the Computer Science field in general and networking systems in particular. This allowed us to propose a Trust-definition specific to SIoT environments, based on the notions of "Intention" and "Ability".
- (2) Proposal of a conceptual SIoT model, allowing to depict the main types of actors and their different roles, to study the major constraints of such an environment and to study the risks incurred. This step enabled us to define our problem and set our objectives.
- (4) Building of a Trust-Attack model allowing to characterize the behaviour of malicious nodes in each type of Trust-Attack.

This preliminary work allowed us to implement "DSL-STM", a multidimensional, resilient, dynamic, resources-efficient and scalable TMM for the Social Internet of Things. Our contributions can be summarized as follows:

- For the (i) Trust-Building phase, we proposed a multidimensional TAM that manages the three trust-dimensions of SIoT and allows to detect

and counter the various Trust-Attacks, but also, to rank the different nodes according to a trust-score. To achieve these objectives :

  - In (i.1) the composition step, we propose Trust-Features that allows estimating the "Ability" of device and service type nodes, as well as the "Intention" of user type nodes.

  - In (i.2) the aggregation step, we propose a method based on supervised Machine-Learning and Deep-Learning that allows, on the one hand, to aggregate the proposed features to get a Trust-Score allowing to rank the nodes, but also to detect the different types of Trust-Attacks and to counter them.

- For the (ii) propagation and (iii) update phases, we propose a hybrid method that allows disseminating trust values in the network, while overcoming the drawbacks of centralized and distributed methods. This method allows to guarantee scalability and dynamism on the one hand, and to minimize resource consumption (computing and storage) on the other.

Overview of the structure

The plan of this thesis is as follows. The first chapter entitled presents, first, an overview of the Internet of Things paradigm to move to the Social Internet of Things, its history, its main advantages and constraints and its architecture to finish with a use case.

The second chapter deals with Trust-Management in SIoT environments. It presents different definitions and properties of this concept, starting from the social sciences fields and ending with a definition specific to SIoT environments. This chapter also presents a conceptual SIoT model that allows to specify the different types of actors and their different roles, to study the rules and constraints of such an environment and to present its risks and vulnerabilities. This chapter then details each component of the proposed conceptual model and focuses on the risks and vulnerabilities by propos-

ing a Trust-Attack model. The last section of this chapter represents the Trust-Management process and its different steps to conclude with a literature review and a comparison of the different related-works according to the elements presented upstream.

The next two chapters detail "DSL-STM", the proposed Trust Management Mechanism. The first chapter presents the Trust-Composition step of our Trust-Assessment Model and details the features proposed for each Trust-Dimension. The second chapter of the proposal details our contribution for the aggregation, propagation and update phases.

Finally, the last chapter of this manuscript is the experimental part. It has allowed us to test and validate the different contributions of this work, including the relevance of the proposed Trust-Features, the Resilience of the resulting Trust-Assessment Model and finally the scalability, dynamism and resources-efficiency of the Trust Management Mechanism that is built based on the proposed Trust-Assessment model. To conduct these experiments, we implemented a prototype that allowed us to conduct different simulations on synthetic data.

Part I

First part

Chapter 1

From Internet of Things to Social Internet of Things

Contents

1.1 Introduction

The social Internet of Things is the next level of the Internet of Things paradigm. It consists on the evolution of objects with a degree of smartness to social objects, able to interact with the surrounding, to autonomously establish relationships with other objects to join communities and build their own social networks.

As the purpose of this work is to propose a Trust-Management Mechanism dedicated to Social Internet of Things (SIoT) environments, it becomes important to understand and explain the SIoT paradigm, to understand its origin, its evolution, its characteristics, its advantages and its constraints. This chapter explains the evolution of the "Internet of Things paradigm (IoT)" into the "Social Internet of Things". It is divided into two main sections.

In the first section, we give an overview of the Internet of Things paradigm, the different visions of this paradigm, its main components, its architecture, its domains of application and its major challenges. In the second section, we present the SIoT. We present the evolution that allowed the emergence of this paradigm, its various advantages, the SIoT architecture, as well as a use case to better understand this paradigm and to demonstrate its effective interest. Finally, we outline the various constraints that characterize this type of environment and that must be taken into consideration when dealing with SIoT environments.

1.2 Overview on the Internet of Things

With the burgeoning technological evolution in computer science technology a new vision connecting the physical world with the digital space has emerged. Physical surrounding objects become embedded with computing technologies (such as sensors, actuators, communication devices, etc.,). This gives them a digital imprint and thus enable them to interact and to perceive the surrounding environment giving birth to the Internet of Things (IoT) paradigm.

1.2.1 The IoT paradigm

IoT is defined as a system of interrelated devices, objects, animals, or people that are provided with unique identifiers, and are able to collect, analyze and exchange data (Atzori et al. (2010)). Each such object in the IoT systems offers a particular service through which persuasive applications are designed. The main purpose of IoT is to allow better living for mankind, where the surrounding objects anticipate users needs and requirements and act accordingly without taking explicit instructions (Atzori et al. (2010), Dohr et al. (2010)). The Internet of Things thus makes it possible to connect the physical world, the virtual world and the digital world with the aim of making any service available to any object/person, anywhere and anytime (Figure 1.1).

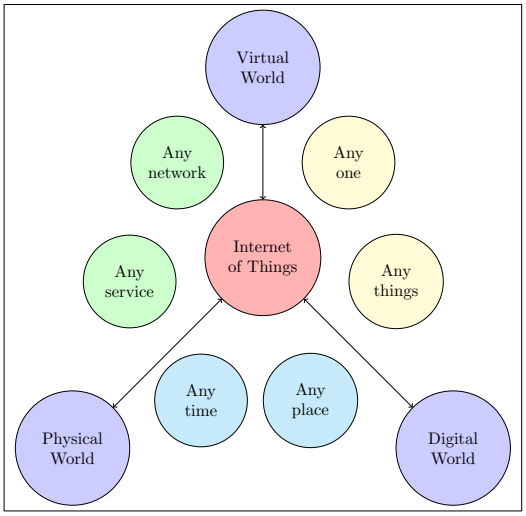


Figure 1.1: The Internet of Things paradigm.

1.2.2 The IoT components

To achieve this vision, the IoT relies on five major components (Figure 1.2). The most significant one is **the device** component which is composed of the internet and the devices also known as "Objects" or "Things" (sensors, actuators, ...). The prime purpose of these devices is to continuously collect information and transfer them to the subsequent layer. Most of IoT

devices have limited functionality due to low internal storage, memory, computational capability and limited power. Indeed, a part of the IoT devices are, basically, everyday objects not designed to perform computer processes but that was extended to be able to do so (example: a lamp, a coffee machine or a chair ...). These devices are said "smart" because they have been endowed with the ability to connect to the Internet and send/receive data. Nevertheless, these devices remain very limited in terms of storage capacity and computing power. They are unable to perform complex calculations, or sometimes, even simple ones. The majority of them cannot support standard communication protocols and have to use a specific protocol layer and pass by a gateway to be able to connect to the internet. **The gateway** is, therefore, the second component as it serves as the bridge between the cloud and the sensors or devices. Every data either towards or from the cloud has to pass through the gateway which can be in the form of a hardware device or a software program. **Cloud** is a network of sophisticated and high-performance servers, programmed to perform data processing at great speed, managing data traffic and delivering accurate analytics. **Analytics** involves the process of conversion of billions of data collected from the devices into meaningful insights which can easily be interpreted and further analyzed. Finally, the **User interfaces** component allows visualizing results for end-users.

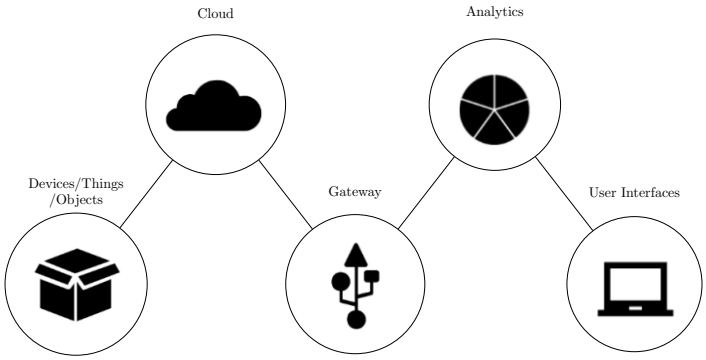


Figure 1.2: The Internet of Things main components.

1.2.3 The IoT architecture

To outline and specify the functional orientation of each IoT component, several architectures have been proposed in the literature. The figure 1.3 shows the best known of them. The first and basic architectures, known as IoT reference architectures are the three and five-layered ones which are derived from the ISO model (Figure 1.3.a). In the three-layered architecture, Perception layer deals with physical objects. The network layer is responsible for transmitting and processing the information between them. The application layer gives services and application to the final-user (Novo (2018)). In five-layer architecture, two more layers are there to give more abstraction (Guo et al. (2018)).

The figure 1.3.b presents the middleware architecture where the middleware layer is at the heart of the system and is not only responsible for managing it but also controls the flow of data. Perception layer, access layer and edge layer come under physical plane where sensor and actuators are present. Backbone Network Layer and Middleware layer are present in a virtualized plane which consists of clouds and servers (Gubbi et al. (2013)).

In service-oriented based architecture, the system functionalities are abstracted and exposed through interfaces (Figure 1.3.c). Objects and applications use these functionalities through services. The advantages are that API does not change even if the inner technology and code are changed as newly introduced functionality is simply used through a new service without impacting the existing system Chen et al. (2016b). Finally, in fog-based architecture, the monitoring layer observes and checks the data obtained from the sensors. The pre-processing layer performs operations on them. The storage layer gathers all the processed data and the security layer is responsible for the integrity and privacy of the data Sethi and Sarangi (2017).

1.2.4 The IoT application domain

The success of the IoT paradigm stems from its undeniable contribution in various fields of application. The figure 1.4 shows a ranking of the most im-

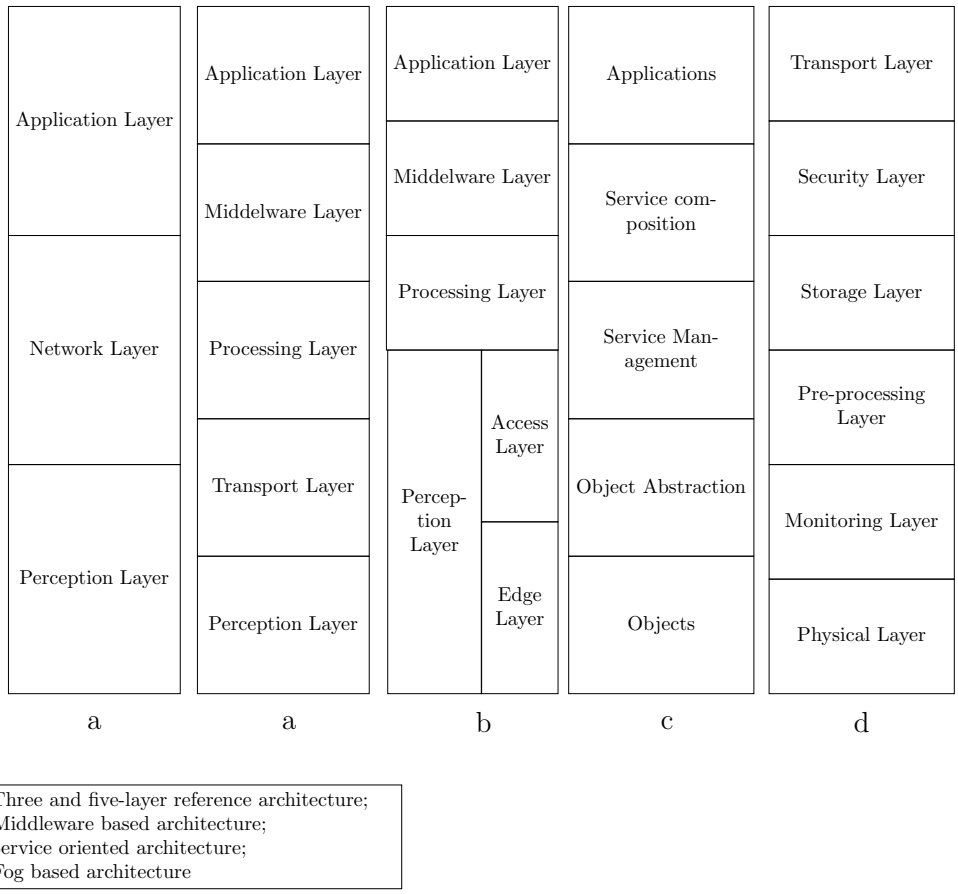


Figure 1.3: The Internet of Things architectures.

pacted application areas according to their popularity. Smart Home clearly stands out, ranking as highest Internet of Things application. Wearables remains a hot topic too. Smart city spans a wide variety of use cases, from traffic-management to water-distribution, to waste management, urban security and environmental monitoring. The industrial internet is also one of the special Internet of Things applications, however, its popularity currently does not reach the masses as smart-home or wearables do. The connected car is coming up slowly since the development cycles in the automotive industry is typically slow. Connected health remains the sleeping giant of the Internet of Things applications. The concept of a connected health care system and smart medical devices bears enormous potential. Prominent use-cases and

large-scale startup successes are still to be seen.

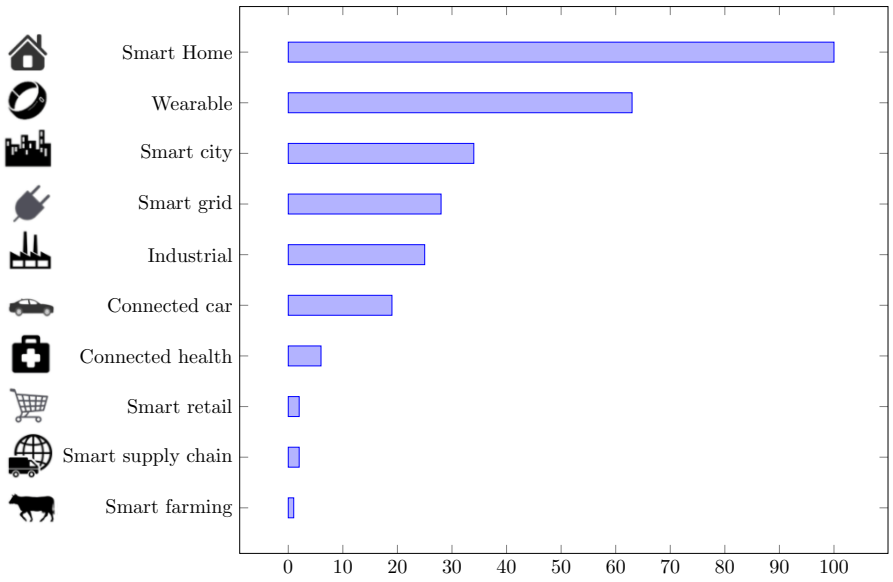


Figure 1.4: The percentage of the most popular IoT application domains.

1.2.5 The IoT challenges

Though IoT is of great benefit to the society, it raises a plethora of issues for the researchers, designers and developers.

Heterogeneity is a major issue where IoT devices are of varying nature at various levels. From devices level, Identifying, addressing, naming and managing heterogeneous devices presenting dissimilar capabilities from computational and communication standpoints in a standardized way is challenging. From a network-centric perspective, allowing those devices with various communication capabilities to communicate and interact through various networks and using different communication protocols is a tricky issue. From a data-centric vision, the lack of a standard data representation model prevents IoT applications to exchange and analyze massive amounts of data and there to support automated reasoning Singh et al. (2014). From a service-oriented vision, the main challenge relates to how to integrate and compose

functionality provided by smart objects into services Miorandi et al. (2012). This heterogeneity, occurring at different levels, prevents the IoT vision from becoming effective.

Scalability is a second primary concern where the number of devices that are getting connected to the IoT network is growing at a tremendous rate. As daily objects become connected to globally networked infrastructure, scalability issue arises at different levels, including (i) identifying, addressing and managing due to the size of the resulting system and to the constrained nature of typical IoT devices which do not enable quite a memory and computing capabilities; (ii) data communication and networking due to the high level of interactions, communications and data exchanges among involved entities; (iii) information and knowledge management due to the massive amount of data and information sensed, detected, generated and analyzed and (iv) service provisioning and management due to the high number of real-time services execution options that could be available and the need to handle heterogeneous resources Miorandi et al. (2012).

Both of these issues impact the Resources Discovery and the Navigability in IoT networks, preventing the IoT vision, where objects are supposed to communicate and interact autonomously and where people should have access to a range of services anytime and anywhere, from becoming effective.

Resources Discovery is, consequently, one of the major challenges in an environment like IoT, where objects are supposed to communicate, interact and exchange services autonomously. The term resource could mean physical objects and/or associated metadata or the services provided by the objects (Datta et al. (2015)). Several approaches have been proposed to ensure resource discovery in the literature. These approaches can be grouped into three categories:

(i) The centralized discovery where a central registry is the backbone of the architecture and is responsible for indexing smart resources. The searching of resources in a given domain can be done by simply connecting to the cen-

tral registry which provides a direct reference of the resource to the clients. Jara et al. (2013) presented a mechanism for the global resource discovery of devices and sensors across several scenarios. Fortino et al. (2013) proposed a service-oriented discovery framework integrated into a centralized architecture. The main limitation of the centralised approach is that it does not allow for scalability.

(ii) The distributed or peer-to-peer discovery: is recognized as one of the most prominent paradigms to achieve scalability. Paganelli and Parlanti (2012) propose a peer-to-peer (P2P) approach that adopts the distributed hash-table techniques and supports multi-attribute and range queries. Liu et al. (2013) presents an architecture for distributed resource discovery where the distributed resource peers communicate among each using peer-to-peer overlay protocol. Cirani et al. (2014) reports about a scalable and self-configurable peer-to-peer architecture for service discovery where an IoT gateway acts as a backbone of the Service Discovery architecture. In contrast to the centralized approach, in the distributed one, the Peers are both suppliers and consumers of resources. This method is efficient, but some IoT devices and objects are too limited in terms of computing and storage capacity to be able to perform these tasks.

(iii) The search-engine based Discovery is another alternative proposed in Ding et al. (2012). However, there is very limited work on the search engine for IoT and the existing ones do not support multimodal search like spatial-temporal, value-based and keyword-based criteria.

1.3 The Social Internet of Things

The Social Internet of Things (SIoT) is an extension of the IoT that has addressed the mentioned challenges through the integration of the social component. We present in this section, how the SIoT came into being. We quote some related paradigms. We then propose an architecture to represent the main functionality of the IoT and we detail each component of the architec-

ture. After, we give some examples of use-case. Then we finish by exposing the constraints specific to SIoT environment that need to be considered when we deal with such environment.

1.3.1 The emergence of the SIoT paradigm

In 2001, Holmquist et al. (2001) were the first to put forward the concept of socialization among the objects of an IoT system. The significant idea behind SIoT is that objects with identical profiles and features are able to share solutions to resolve problems encountered by other objects i.e., their interacting partners. The SIoT paradigm emerged as a result of a gradual process. The first step of this process consists of making objects "smart". The second step rests on the evolution of objects with a degree of smartness to pseudo-social objects Atzori et al. (2014), which can interact with the surrounding environment and perform a pseudo-social behaviour with other objects. The last step of this process relies upon the appearance of social objects which are able to autonomously establish relationships with other objects in order to join communities and build their own social networks, which can differ from those of the owners Atzori et al. (2014).

As shown in figure 1.5, the ideas behind the SIoT are (i) increasing sociality (or connectivity) and (ii) improving pervasiveness (or availability). Allied technologies such as ontology Perera et al. (2013), Machine Learning Ye et al. (2012), Commonsense reasoning Davis and Marcus (2015), Deep Learning Jiang et al. (2020) and Human–computer interfaces Kranz et al. (2009), among the many are increasingly contributing to the development of SIoT.

1.3.2 Related social and user-centric models

In this section, we quote some SIoT related models and paradigms aiming to give focus to the user.

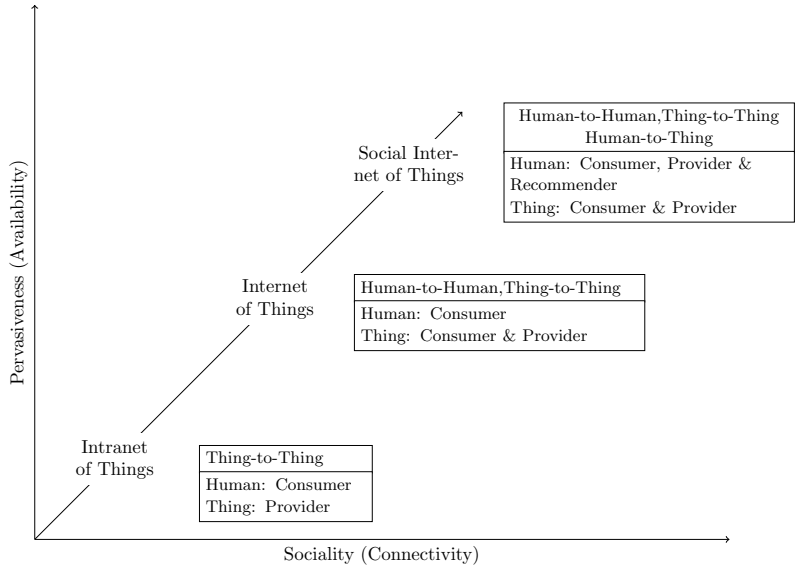


Figure 1.5: Evolution history of ubiquitous computing technologies.

Internet of People Miranda et al. (2015) define the Internet of People (IoP) as bringing the IoT closer to people in order to easily integrate into it and fully exploit its benefits. This paradigm aims to put people at the centre of innovation strategies and be able to make profit from the power of collective intelligence. Miranda et al. (2015) define a set of features they believe are essential foundations for any approach to the IoP: (i) being social; (ii) being personalized which mean that interactions must be personalized to users' sociological profiles and contexts; (iii) being proactive and not manually commanded by the user; (iv) being predictable which means that interactions must be triggered according to a predictable context. IoP includes numerous topics such as Biometric Sensors and Wearable Technology.

Physical-Cyber-Social computing Sheth and Anantharam (2013) propose Physical-Cyber-Social (PCS) computing, that takes a human-centric and holistic view of computing by analyzing observations, knowledge, and experiences from physical, cyber, and social worlds. Some of the main challenges in healthcare, sustainability, crime prevention and mitigation require a holistic approach to computing for providing actionable information. With

the increased digitization of the physical world culminating in a massive data generated from sensors, mobile devices, and personal/social observations has led to a deeper view into our physical, cyber, and social worlds. PCS computing is envisioned to derive insights from these observations to provide actionable information to humans.

People as a Service People as a Service (PeaaS) is a mobile-centric computing model that allows a user's sociological profile to be generated, kept, and securely provided as a service to third parties directly from a Smartphone. PeaaS emphasizes smart-phones' capabilities and relies on them for inferring and sharing sociological profiles. Unlike other mobile-centric models provide data, such as GPS localization and temperature, PeaaS allows a variety of information to be collected, such as moods, tendency, preferences, social statuses, daily habits and health habits of a group of peoples. Various techniques, including activity recognition approaches and affective computing, are used in PeaaS for building the richest sociological profile possible (Guillen et al. (2014)).

Social Devices Social Devices is an Internet of Things (IoT) model, introduced by Mäkitalo et al. (2012). The motivation behind the model was that Smart-phones have not only a lot of information about their owners, but also modalities that enable them to resemble humans. They can, for example, translate text into speech. At present, Social Devices concept is supported by a middle-ware platform. This allows proactive triggering of interactions between devices of co-located people. Additionally, it offers a complete set of Web-based tools to define interactions and their triggering contexts.

Social Sensing Social Sensing is an integral paradigm of the Internet of Things when objects being tracked are associated with individual people. Mobile phones, smartwatches, smart glasses and wearable sensors are good examples of sensing objects. Such paradigms have tremendous value in enabling social networking paradigms in conjunction with sensing (Aggarwal

et al. (2013)).

1.3.3 SIoT Architecture

The main functionalities of a SIoT environment have been addressed in various related works. Several works have focused on Relationship Management (Ali et al. (2018), Atzori et al. (2012), Chen et al. (2016a)), while some others have focused on Navigability in SIoT networks (Amin et al. (2019), Marche et al. (2017), Nitti et al. (2014a)). Many related-works have addressed the problem of Resource Discovery in this type of environment (Hussein et al. (2017), Li et al. (2015)), while other ones have addressed the Trust-Management issue (Huang et al. (2016), Jayasinghe et al. (2016), Militano et al. (2016), Truong et al. (2018)). Those functionalities are recognized and well defined in the literature, but there is no architecture, allowing to identify the responsible modules and to clarify the connection between them. We propose in figure 1.6 an architecture allowing to identify the main modules of a SIoT system and clarify their main functionalities.

Relationship Management As shown in the figure, the first module is about Relationship Management. Indeed, the main contribution of SIoT environments consists in the integration of different types of relationships and socialization that can link one or several types of actor nodes.

In Chen et al. (2016a), authors set up three types of relationships occurring between objects owners (users type node), namely; (i) the friendship relationship, which represents closeness and intimacy; (ii) the community of interest relationship, which designs common experiences; and (iii) the social contact relationship which represents proximity. Atzori et al. (2012) proposed various forms of socialization among objects. (i) The parental relationship is defined among objects belonging to the same category. (ii) The ownership relationship is defined for objects owned by the same user. (iii) The co-location relationship is established between objects in proximity. (iv) The co-worker relationship is defined for objects which collaborate to accomplish common tasks. Ali et al. (2018) propose the (i) STGOR — Stranger Ob-

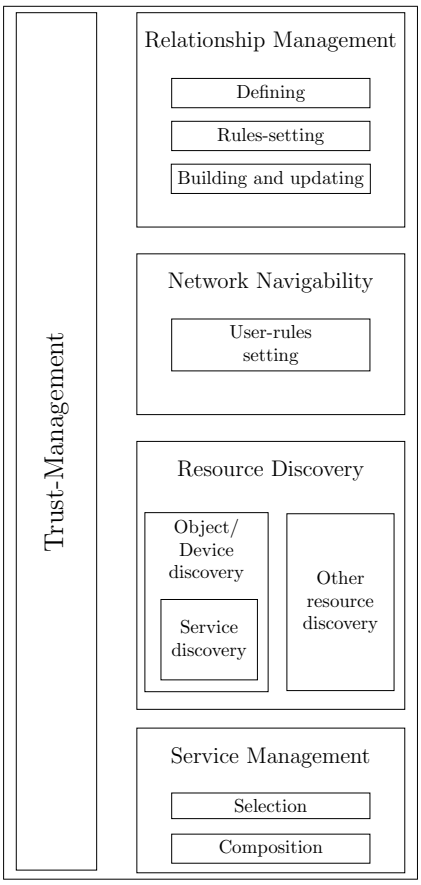


Figure 1.6: SIoT Architecture.

ject Relationship for objects that encounter the existence of each other in the public surroundings or on the go and the (ii) SVOR — Service Object Relationship, which is formed between objects that fulfil the service request by coordinating the same service composition.

According to the previous related works, relationships in a SIoT system can be classified into four categories as (1) User-to-User Relationship (UU Relationship); (2) User-to-Object Relationship (UO Relationship); (3) Object-Object Relationship (OO Relationship), where the physical objects are bound to each other through some relation; and (4) Object-Service Relationship (OS Relationship).

The first task of the Relationship Management module is to define the dif-

ferent types of relationships that are possible. The next task is to set the rules for establishing these relationships. For example, from how many interactions can object be considered as co-workers? From how proximity can two objects be considered as co-located? Finally, this module is responsible for creating and updating these relationships, which are generally dynamic and can vary over time.

Network Navigability Once the SIoT network is created, the idea behind the SIoT is to allow the different nodes to navigate within it. The navigation module is responsible for this task and allows the users to set their rules. Indeed, a user could indicate that he does not wish to interact with co-located nodes, but only with nodes with which he has a social or co-worker relationship.

Resource Discovery The next module is about Resource Discovery. Resource discovery includes, firstly, the discovery of objects and devices, and secondly, the discovery of the services they provide.

Service Management The last module is the Service Management module. The previous modules will allow the user to access different services. This module aims to help the user to select the most relevant or adequate services, and also to compose services.

Trust Management Finally, the Trust Management module, comes vertically because it operates on each of the mentioned modules. It can operate at the Relationship Management level to ensure that a node can only link relations with reliable other nodes. It can intervene at the Resources Discovery level to guarantee the relevance of the discovered resources. It can also operate at the Service-Management level to help users select the most reliable and trustworthy services. The figure 1.7 shows an example of the Service Discovery in SIoT.

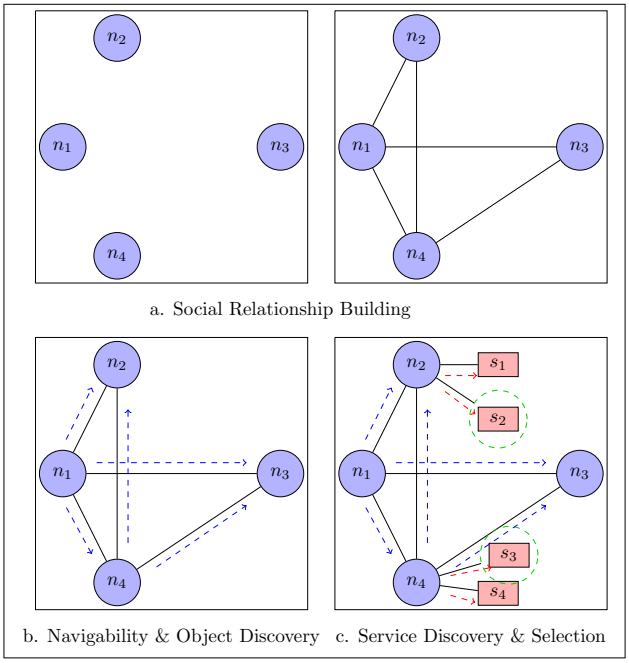


Figure 1.7: Service Discovery in SIoT.

1.3.4 SIoT use cases and scenarios

In order to demonstrate the value of integrating the social component and to explain how SIoT can be used, we cite here an example of a scenario in the context of Road Traffic Management. In this scenario, drivers will collaborate to learn about the current state of the road at their position. Information about the road conditions (accident, traffic jam, stalemate, works, floods, jolts, narrow road, ...) can be detected automatically through different types of intelligent objects (smart vehicles, smart-phones, Sensors, ...), or manually signalled by various conductors. In this type of scenario, users are the drivers. Devices are intelligent objects. And services are the information provided about the status of a given road at a given time.

The figure 1.8 shows an example. Bob will take the road. He sends a request through his smart-phone to get the road state. (i) At this step, the smart-phone will find Alice who has a social relationship with Bob. Alice has a smart vehicle equipped with intelligent sensors and able to predict the road state. (ii) Bob's smart-phone will also discover John who is in prox-

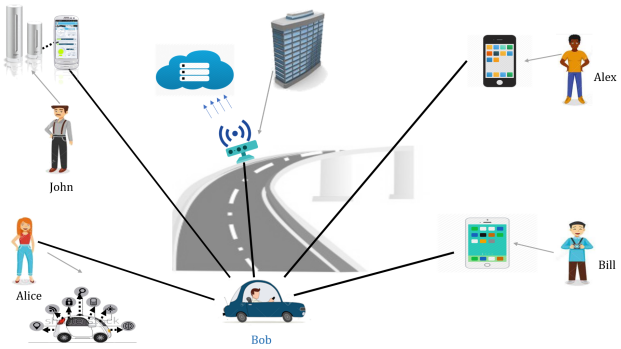


Figure 1.8: Road Traffic Management use case.

imity and has a co-location relationship with Bob. John has a connected weather station which enables him to get a notification if some incidents like fog, snow or flood will happen. (iii) Bob's smart-phone will also discover Alex through the parental relationship because Bob's smart-phone and Alex smart-phone belong to the same category. Alex smart-phone provides a collaborative traffic management service which bases on a global community and enables drivers to share road traffic information in real-time. (iv) Bob's smart-phone discover also Bill tablet which is equipped with GPS and which offers an online mapping service that uses satellite imagery to allow access to detailed information about different locations. (v) Finally, Bob's smart-phone discovers an installed ADI sensor (Automatic Detection of Incidents). The latter belongs to the Ministry and allows to send detected information to a server hosted on Cloud which will analyze them in real-time. Bob receive multiple information generated by multiple kinds of services, but he needs to estimate the trustworthiness of received information. Through this example, we can see how SIoT has enabled its users to discover a range of attractive services through the integration of different types of social relationships.

1.3.5 SIoT constraints and advantages

SIoT has enabled overcoming three major IoT challenges. The heterogeneity of devices, networks and communication protocols is solved Ali (2015) and scalability is guaranteed Atzori et al. (2012) by the use of social networks.

As explained in the previous sub-section, navigability, as well as resource discovery, are improved by reducing their scope to a manageable social network of anything. Besides, a larger data source becomes available because it comes from a set of users, a network of users or a community rather than from a single user. The continuous feeding of data from communities gives us a big data team Geetha (2016), and the quantity, as well as the variety of contextual data, are increased, allowing improved services intelligence and adaptability to users' situational needs Ali (2015).

However, SIoT despite having solved many IoT challenges has also inherited some of its constraints. Indeed, just as in the IoT, SIoT devices are numerous, dynamic and constrained in computing and storage capacity. Moreover, the structure of SIoT networks, which involves different types of nodes, makes them even more complex. In addition to this, the consumption of IoT/SIoT services remains below expectations as users remain suspicious and perplexed about the adoption of this new paradigm, making Trust Management one of its major challenge.

1.4 Conclusion

The Social Internet of Things is an extension of the IoT that has addressed some IoT challenges through the integration of the social component. In this chapter, we introduced the SIoT paradigm. The objective of this chapter is to understand the characteristics, specificity and constraints of SIoT environments. In the next chapter, we focus on one of the main SIoT challenges, which is Trust Management.

Chapter 2

Trust in the Social Internet of Things

Contents

2.1 Introduction

The study of SIoT environments has enabled us to understand the numerous advantages of this paradigm and its usefulness in the real world, but also to understand its constraints and specificity and to identify the different issues in this type of environment.

Trust Management is one of the most challenging issues raised in SIoT environments. We propose, in this second chapter, to study the "Trust" concept in SIoT environment. In the first three sections, we address the issue of Trust-Management in general. We propose an enlargement of the Trust definition, starting from the sociological and psychological fields to its incorporation in the world of computer sciences. Then, we present the Trust Management process and give a literature overview of trust in some collaborative and networked environments.

In the remainder of the chapter, we focus on Trust-Management in SIoT environments. We propose a conceptual trust model that allows us to define the different actors involved, to delimit the risks and vulnerabilities in this type

of environment, to establish a Trust-Attack model, to study the constraints and specificity to be considered and to deduce the purpose of the trust in SIoT environments. We end this chapter with a review of the literature. We compare related works that have dealt with Trust Management in SIoT, based on numerous criteria, extracted from the different definitions, explanations and models presented earlier in the chapter. Finally, we conclude with the conclusion.

2.2 Enlargement of Trust definition

Trust is a multifaceted concept essential for people to deal with uncertainty about the future and which represents an important aspect of decision making. The concept of Trust originally derives from social sciences. However, it has been studied in many disciplines including sociology (Helbing (1994), Möllering (2001), Molm et al. (2000)), psychology (Cook et al. (2005), Rotter (1967)) and economy (Granovetter (1985), Huang (2007)). Although its importance was already duly recognized, there was no consensus over "Trust" and its very definition varies greatly according to the disciplines and the perspectives.

2.2.1 General definitions of Trust Concept

To define the concept of trust, we start with its literary definition and its definition in the social and psychological fields, to arrive at its definition in the fields of computer science.

The Oxford Reference Dictionary states that Trust is “the firm belief in the reliability or truth or strength of a trustee”. In psychology, Trust is considered to be a psychological state of the individual, where trustor risks being vulnerable to a trustee, based on positive expectations of its intentions or behaviour (Rotter (1967)). In sociology, Zavlanos and Pappas (2007) defined Trust as the degree of subjective belief about the behaviours of a trustee. Luhmann (2000) defined trust as confidence in one’s expectations. Kini and Choobineh

(1998) defines Trust as a belief that is influenced by the individual's opinion about certain critical features. According to Beatty et al. (2011), trust is considered to have two aspects: cognitive and behavioral (Beatty et al. (2011)). Blomqvist and Ståhle (2000) proposed a similar vision and suggested that trust has two aspects of expectation: Ability which implies technical capability, skills, and know-how and Intention, which implies goodwill and moral responsibility towards the other.

There are several terms that are often associated with the term trust such as **(i) Confidence** which is the feeling or belief that one can have faith in or rely on someone or something; **(ii) Trustworthiness** which represents the ability to be relied on as honest or truthful; and **(iii) Belief** which represents the acceptance that something exists or is true, especially one without proof. Beliefs reflect the critical role that perceptions about the other party play in trust.

The important concepts that appear in most definitions are as follows:

Trustor and trustee Trust is usually specified in terms of a relationship between a Trustor, the subject who trusts and a trustee, the entity that is trusted. Trust forms the basis for allowing a trustee to use or manipulate resources owned by a trustor or may influence a trustor's decision to use a service provided by a trustee.

Expectation Expectancy reflects the future orientation of Trust. Trusting behaviours expected by trustors are identified as trust expectations. According to the previous definitions, the trustor's expectations of the trustee cover essentially two aspects: intention and ability.

Intention is central to the concept of voluntary action and includes cognitive choices and behavioural estimates.

Ability means that an entity is capable of performing the functions expected of it.

Risk The source of risk is the uncertainty regarding the intention of the other party. Many researchers recognized that trust is associated with risk Mayer et al. (1995).

2.2.2 Trust in Computer Sciences

In the computer science domain, Trust can be classified into two broad categories: "user" and "system". The standard notion of "system trust" is derived from the security domain. Grandison and Sloman (2000) define trust as "the firm belief in the competence of an entity to act dependably, securely and reliably within a specified context". Stump et al. (2008) introduced the term "Trust Management" and identified it as a separate component of security services in networks. Yao et al. (2010) define system trust as "the expectation that a device or system will faithfully behave in a particular manner to fulfil its intended purpose". For example, a computer is trustworthy if its software and hardware can be depended on to perform as expected, such that its services are still available today, unaltered, and behave in the same way as they did yesterday (Moreland et al. (2010)). The notion of “system” trust is supported by both software- and hardware-based solutions.

The notion of "user trust" is derived from psychology and sociology (Marsh (1994)) and has been integrated especially into collaborative environments implying interactions and exchanges. From this perspective, a large number of research groups have been working on trust-management in Networking Environments such as peer-to-peer (P2P) networks, Wireless Sensor Networks (WSN), Social Networks, and IoT/ SIoT networks; varying in many applications and services from access control (Zhou et al. (2015)) to e-Commerce (Kim and Peterson (2017), Oliveira et al. (2017)).

In this work, we, mainly, focus on "user trust", which is a crucial issue in a networked and collaborative environment such as the SIoT, although we also address some aspects of "system trust".

2.2.3 Trust properties

Before building a trust system, it is also important to identify the trust properties. These properties have been identified and presented in several works (Abdelghani et al. (2016)).

Subjective. The subjective nature of trust occurs because of differences in trustors' perceptions, opinions, beliefs, and expectations. The same trustor is likely to have different levels of trust regardless of a trustee.

Asymmetric. Trust is typically asymmetric. A trustor may trust a trustee more than s/he is trusted back. However, when both parties are trustworthy, they will converge to high mutual trust after repeated interactions.

Dynamic. Trust can increase or decrease with new experiences (interactions or observations). It may also decay with time. New experiences are more important than older ones, since old experiences may become obsolete or irrelevant with time.

Context or domain-specific. Trust is context-specific in its scope. For example, John trusts Bob as his doctor, but he does not trust John as a mechanic to fix his car.

2.2.4 Generic Conceptual Trust Modeling

Based on the various definitions and clarifications presented above, we propose in figure 2.1, a conceptual trust model. Indeed, whatever the system in which we want to establish a Trust Management Mechanism, it is necessary, before embarking on its implementation, to study the following points:

(1) The trust actors: as explained above, a trust relationship mainly involves two actors: a trustor and a trustee. The nature and characteristics of these actors depend on the system in which a Trust Management Mechanism is to be established. In a wireless sensor network, the actors are the sensors.

In a social network, the actors are the network users. In a recommendation system, the trustor is a user while the trustee is an item (article, movie, ..). Once we have defined the actors, it is important to study and formalize the trustor's expectations on the one hand and the behaviour and competence of the trustee on the other hand. Indeed, a Trust Management Mechanism must, in the first place, make it possible to measure the matching between them.

(2) The system: it is essential to study the system in which we want to set up a Trust Management Mechanism (TMM). A recommendation system, a social network or an e-commerce system are examples of systems, in which it is possible (and interesting) to integrate a TMM. The characteristics and purpose of this TMM depend on the nature of the system and its rules. Indeed, each system has a set of rules that must be respected by its users. Non-compliance with these rules influences, as well as non-compliance with the trustor's expectations, the degree of trust that can be placed in the trustee. For example, in an e-commerce system, a seller has to give his identity, give a correct description of his products, ...etc. In a recommendation system, the recommender has to give honest ratings representing his personal experience with the item. A Trust Management Mechanism must, therefore, allow measuring the degree of matching between the trustee's behaviour and the system rules.

(3) The incurred risk: the notion of risk goes hand in hand with the notion of trust. It is essential to define the risk incurred by the trustor if he takes the decision to interact with a trustee. The risks of such a decision may come either from a trustee who, for example, behaves maliciously or from the system itself. For example, an unsecured e-commerce site where a buyer risks having his bank details stolen.

(4) The environment: It is useful to study the environment and context in which the TMM must operate. The specificities and constraints (such as the number of actors (scalability), the number of interactions (dynamism),

the type of hardware used (resource capacity)) must be considered. This study helps to guide in the choices to be adopted for the creation of an effective and functional TMM.

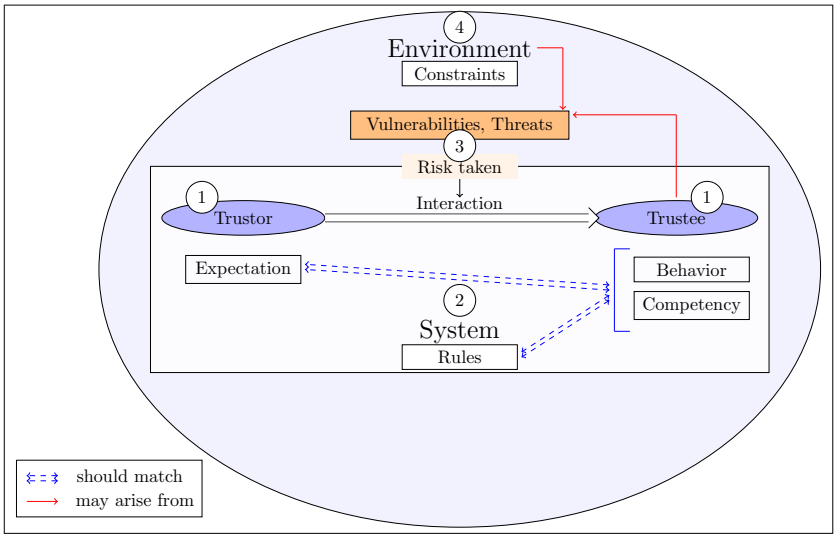


Figure 2.1: Conceptual Trust Modeling

2.3 Trust-Management process

Once the various conceptual aspects have been dealt with, it becomes possible to implement the Trust Management Mechanism. There are three main phases in the trust management process: (i) The Trust-Building phase, (ii) The Trust-Propagation phase and (iii) The Trust-Update phase (see figure 2.2). The performance of the proposed TMM depends on the methods selected for each of these phases (Abdelghani et al. (2016), Yan et al. (2014)).

2.3.1 Trust-Building phase

The Trust-Building phase is the main phase of a Trust Management Mechanism. It consists of implementing a Trust-Assessment Model (TAM). A TAM allows to assess and estimate the degree of trust that can be placed

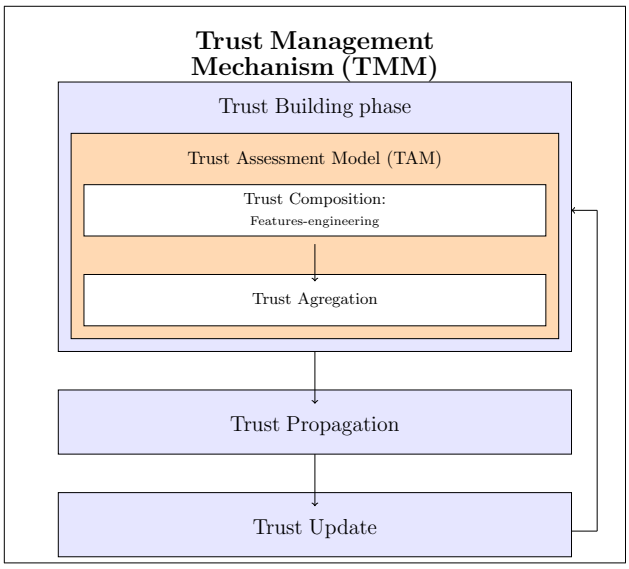


Figure 2.2: Trust-Management process

on a given entity and therefore assists in deciding to interact or not with that entity. The Trust-Building Phase consists of two steps, namely, (i) the Trust-Composition step and (ii) the Trust-Aggregation steps.

2.3.1.1 Trust-Composition step

The objective of Trust-Composition step is to gradually move from the abstract concept of Trust to concrete numerical values. As shown in figure 2.3, this gradual shift is done over three or more iterations. Usually, the first iteration consists of moving to more precise and less abstract sub-concepts, based on the adopted trust definition, for example, "Ability", "Goodwill", "Affinity", ... etc.

The second iteration consists of moving from these less abstract sub-concepts to features that describe each of them. In the literature, the term "feature" is also referred to as "parameter", "factor" or even "indicator". Depending on the purposes of the Trust Management Mechanism, the Trust-Features may make it possible either to (i) depict the trustee behaviour, or (ii) estimate his competence for the task for which he is invoked, or (iii) measure the matching between the trustor's expectations and the trustee's behaviour and

competence, or (iv) estimate the percentage of risk incurred by the trustor if he engages with the trustee, or (v) check whether the trustee complies with the system rules, ..., or even all of them.

Finally, the last iteration consists in fixing the metrics, formula or equations enabling to compute these features. Note that the number of iterations depends on the complexity of the system, the behaviour of the actors, the harmfulness of the risks ...etc. And we can, therefore, find models with one or two iterations that go directly from the abstract "Trust concepts" to measures.

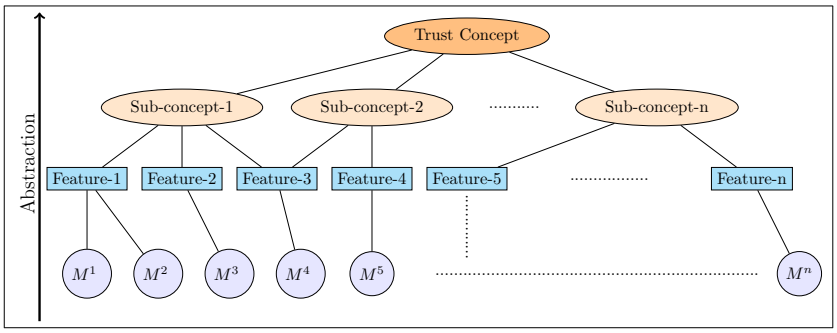


Figure 2.3: From Trust concept to concrete values

2.3.1.2 Trust-Aggregation phase

The composition phase returns as output a set of numerical values corresponding to the different selected Trust-Features. The Trust-Aggregation step comes afterwards, to allow aggregating these different values into a single final trust-value allowing the trustor to make the decision to trust or not a trustee. This final value can be binary (malicious/benevolent, trust/distrust, ...) or numerical allowing to rank the trustees according to a score. To obtain this final trust value, authors in the literature have used different methods such as Fuzzy Logic or Combinatorial Logic. But the most used one is the weighted mean which consists in assigning weights to the different features.

2.3.2 Trust-Propagation phase

The Trust-Propagation phase consists of choosing a method to propagate in the network the overall trust values returned by the Trust Assessment Model. Depending on the system in which we set up our TMM and on the characteristics and specificity of the environment in which the interactions are taking place, it is possible to opt for either the (i) centralized or the (ii) decentralized propagation method.

In the centralized propagation method, a central entity (a server, for example) does the various computations for all the nodes of the network. The main drawback of the centralized method is that it does not allow scaling up and is not suitable for environments involving a large number of nodes. It remains a reasonably robust solution for environments with a moderate number of nodes.

In the decentralized propagation method, each node makes its own calculations. This method solves the drawback of the centralized method and allows scalability, but it requires powerful nodes able to performing complex trust-calculations. To solve this issue, it is possible to create lightweight Trust-Assessment-Models to simplify such calculations. However, lightweight models may be less robust. Another potential solution is to combine this method with Storage Management Strategies to minimize resources consumption.

2.3.3 Trust-Updating step

The trust-Updating step consists of choosing a method for updating trust values, either the (i) time-driven method or the (ii) event-driven method. In the time-driven method, updates are done periodically, whereas, in the event-driven method, updates are done each time an event occurs. The event-driven method is the most robust as it allows fast detection of any malevolent behaviour. However, it is resources-consuming as it involves a huge amount of updating, and is, thereby, not suitable for highly-dynamic environments or resources constrained environments.

The time-driven method is resource-efficient, but it is difficult to set the granularity of the time-frame between updates. Besides, this method is less robust and can be very risky in critical and sensitive environments, as a dangerous incident may be detected too late.

To summarize, the Trust Building phase is the main phase of a TMM as it defines how Trust is computed and on which basis. It is also this phase that allows risk-management, system resiliency and qualified interactions. The propagation and update phases, come in the second position, to ensure the non-functional properties of the TMM. The choice of the method (centralized or decentralized) made for the propagation phase makes it possible to give priority either to the scalability of the TMM or to its efficiency in resources consumption. In some systems where the scalability problem does not arise, or where the hardware infrastructure is powerful, the choice of the propagation method does not really matter and becomes optional. For the Trust-Update phase, the choice of the update method (time-driven/event-driven) allows giving priority either to system dynamism and robustness or to resource consumption efficiency. As with the propagation phase, this choice is of little importance for systems with no resource constraints. These two phases are, however, of primer importance in the case of systems that require scaling up, are vulnerable and dynamic, or have constraints related to the consumption of storage and computing resources, since failure to take these properties into account is likely to generate a robust TMM that cannot be applied in real field conditions. Table 2.1 present the impacted properties for each TMM phase.

Table 2.1: The impacted properties for each TMM phase

TMM phase	Impacted property				
	Interactions quality	Resiliency	Scalability	Resources efficiency	Dynamism
Trust-building phase	✓	✓			
Trust-propagation phase			✓	✓	
Trust-updating phase		✓		✓	✓

2.4 Trust Management in networked and collaborative systems: A literature review

In this section, we present a review of some related-works that have focused on Trust Management in different collaborative and networked systems. We use the conceptual model of trust and the trust management process, explained above, to explain and compare the different related works. We present, therefore, for each environment, its actors, its characteristics and constraints, the risks and vulnerabilities and the purpose of Trust. We chose to study social networks, peer-to-peer networks and recommender systems because SIoT can be seen as a combination of these three systems.

2.4.1 Trust in social networks

A social network is described as a social structure made of nodes connected by edges that represent one or more specific types of inter-dependency (Barnes (1954)). Nodes represent individuals, groups, or organizations, while the connecting edges are relations like values, ideas, friendship, trade. Trust plays an important role in the formation of coalitions in social networks, in assessing the quality and credibility of information as well as in determining how information flows through the network.

We can trust the user or his publications. Some works focus on mutual user trust and try to study the relationship between users to determine similarity, affinity and community of interests and to perform pertinent recommendation. Some other works study single user trust and try to identify, for example, fake profile. Others are interested in items-trust (posts, publication, information...), and try to detect buzz, spam and rumour (Washha et al. (2017)).

2.4.2 Trust in peer-to-peer networks

The purpose of Trust-Management Mechanism, in a peer-to-peer network, is to allow two sides of a transaction to judge the reliability and/or quality

of transaction by studying the peer's history behaviour. Several reputation systems have been proposed in the research community to enhance the robustness, scalability, and efficiency of peer-to-peer computing models, such as EigenTrust (Kamvar et al. (2003)) and PeerTrust (Xiong and Liu (2004)). The main risks incurred in peer-to-peer systems are free-riding, virus, worms and Trojan horses spreading, fake files dissemination, index poisoning, etc (Chu et al. (2010).

To evaluate reputation values many TMMs where proposed. We can cite, for example, Credence which is a subjective, independent and local reputation mechanism based on Gnutella which is based on a polling mechanism (Walsh and Sirer (2006)).

TrustGuard is a secure reputation mechanism framework based on PeerTrust aiming to counter three types of threats, namely strategic oscillations, fake transaction, and dishonest feedback, and provide corresponding countermeasures (Xiong and Liu (2004)). In this framework, each peer has a transaction management unit, a reputation evaluation engine and a feedback data storage unit. LIP is an objective, global reputation mechanism which gives statistics automatically about file's remaining time in each user's computer, and then computes the number of holders of each file and also the file's average remaining time in user's computer (Feng and Dai (2007)).

2.4.3 Trust in Recommender Systems

The actors involved in Recommender Systems (RS) are the items and the recommenders. An item can be a book, a movie, a web service, a computer, ... Recommenders are users who have tested the items and given explicit feedback. A user who is looking for a given item will use other user's feedback to choose the right item. Nevertheless, two major problems make it impossible to retrieve feedback on some items. First, the cold start problem where a new item that has just been added and that has never been tested, has not been evaluated yet and thus, may never be tested. To solve this problem, Recommender Systems will try to predict the score of this new item (Kalaï

et al. (2018)).

To do this, they will rely on feedback given by recommenders who are similar to the user who is looking for an item. To calculate the similarity between users, recommender systems will apply similarity measures (Cosine measure, Jaccard index, Pearson correlation coefficient ..) on their rating vectors. However, in practice, this is often not possible because it raises the second issue: the data sparsity (Kalaï et al. (2018)). Indeed, rating matrices are often very scattered and it is not easy to find rates assigned to the same items. It is in this context that researchers have proposed to integrate the trust notion in the Recommender System. The idea is to find trustworthy recommenders and use their ratings to predict the score that can be assigned to a given item. To measure trust in RSs, authors in the literature have proposed various features based on profile/interests similarity, past interaction history and relationship strength/age. These factors are generally aggregated by a weighted mean. The problems of propagation, updating and resources management do not arise in this type of environment, as unlike SIoT environment, RS environments are centralized and unconstrained.

2.5 Conceptual trust model for SIoT

After having explained and studied the notion of trust in general, then in computer systems in particular, and more specifically in collaborative and networked systems, the rest of this chapter focuses on the Trust Management in the Social Internet of Things. We reproduce each of the elements presented before and explain it in the case of the SIoT.

In the sub-section 2.2.4, we have presented the structure of a generic conceptual trust-model. In this sub-section, we detail each component of this model, for the case of SIoT environments.

2.5.1 Actors and object of trust

In SIoT networks, we have three types of actors involved in each interaction: Users, Devices and Services. Indeed, interaction is initiated by a "user" who will use his "device" to provide or invoke a given "service". The quality of an interaction depends on the three types of actors mentioned. The user actor plays three roles in SIoT environments. He is both a "Service Provider" and a "Service Consumer". As in all collaborative environments, the service consumer is invited to provide feedback after each interaction and in this case, plays the role of "Recommender". A legitimate and benign recommender is a user who returns feedback that reflects his or her true opinion about the interaction he or she had with another user provider. A legitimate and benign provider is a user who offers qualified services.

The figure 2.4 shows the classic pattern of interaction in a SIoT environment. User u_1 launches a request for a given service. The service s_2 provided by user u_2 via the d_2 device responds to his request. Before deciding whether to invoke this service or not, user u_1 consults the feedback given on u_2. Users u_3 and u_4 have already interacted with u_2 and recommend it. u_1 considers that u_3 and u_4 are honest well-intentioned recommenders and therefore decides to trust them (2). When making the decision to invoke the s_2 service, u_1 trusts the u_2 provider and assumes that he is reliable and well-intentioned (3). He also trusts the d_2 device and assumes that he is able to execute the s_2 service according to his expectations (3) (computing power, security, availability ...). Finally, he trusts the s_2 service and assumes that the service is qualified (3) (response time, availability, ...). This figure illustrates the different trust-dimensions involved in a SIoT environment and gives an overview of the purpose of trust (expectations of u_1) for each trust-dimension.

2.5.2 Environment constraints

In the previous chapter, we mentioned in section 1.3.5, the various constraints of SIoT environments. These constraints are to be taken into consideration for the implementation of an applicable TMM and can be summarized as

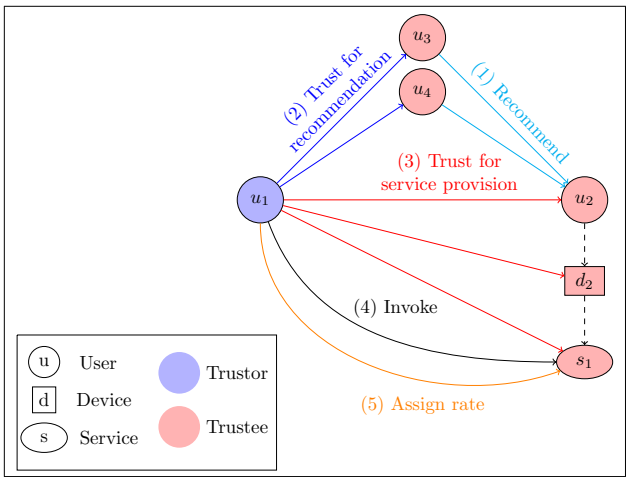


Figure 2.4: Example of interaction scheme.

follows:

- Multi-party networks
- Dynamic Networks
- a Large number of nodes
- Most nodes constrained in terms of storage capacity
- Most nodes constrained in terms of computing capacity

2.5.3 Risks and vulnerability

The main risk incurred by a user who is going to make the decision to engage with another user is to obtain a poor-quality service. To avoid this risk, collaborative environments such as SIoT allow access to feedback from other users. This feedback, expressed in the form of ratings, gives an idea about the reputation of each user in the network and can be used to discriminate a good Service Provider (SP) from a bad one. That said, bad Service Providers can behave maliciously and distort or dupe the Reputation System. To do this, they use two categories of Trust-Attacks (TA): (i) Malicious-Recommender Attacks and (ii) Malicious-Provider Attacks:

(i) Malicious Recommender Attacks aim at distorting reputation values by enhancing the reputation of a malicious user or by spoiling the reputation of a legitimate user through the attribution of erroneous ratings. Increasing a malicious user's reputation allows him to spread his poor quality services. Decreasing the reputation of other legitimate users, allows a malicious user to move up in the ranking and increases his chance of being selected as a service provider, which also allows him to propagate poor quality services. This category of attacks is launched by Recommender Users, either in groups or individually. (ii) Malicious provider attacks are launched by Service Provider Users individually. They aim to dupe the reputation system by propagating bad services while maintaining a good reputation in the network. To do so, these malicious users may provide good quality services until they reach a good reputation and then start providing poor quality services. They can, also, alternately provide good/bad services to still keep an acceptable reputation value.

Note that a Trust-Attack is always launched by a human user-driven by his bad "Intention". Indeed, services and devices have no intention and act under the control of their owners. In other words, it is their owner who decides to make good/ or bad use of them. We, therefore, define a Trust-Attack as "a set of interactions carried out by a malicious user to break and mislead the reputation system". We define a malicious user as "a bad-intentioned user" who launches Trust-Attacks intending to propagate poor quality services in the network".

Beyond Trust-Attacks, other risks can threaten the quality of interactions between users and influence the degree of trust that can be granted. Indeed, (a) a user may have no malicious Intention, and distort reputation system by attributing unrepresentative ratings to other users, because of his lack of expertise, which leads to his "Inability" to properly judge the quality of the invoked service. (b) The device used may, for example, lack the computing capacity required to complete the task assigned to it. It may also be vulnerable to security attacks (Intrusion, ...) and threaten the security of the entire network. The invoked (c) service, for example, may not be available at the

time of its invocation or need a long run-time. These various risks threaten the quality of interactions, but, are not due to a bad intention and are not considered as Trust Attacks. They are related to the "Ability" of each type of entity involved in SIoT networks and it is necessary to also consider them.

2.6 Trust-Management in SIoT: Literature review

In this sub-section, we rely on all the concepts presented above to compare related-works. This comparison is made according to the choices adopted for each phase of the Trust-Management process presented in section 2.3 and according to the impact of each choice on the resulting TMM characteristics (see Table 2.1).

2.6.1 Trust-establishment: Attacks-resiliency and Trust-dimensions

The Trust-Building phase is the main step of a Trust-Management Mechanism, as it consists in implementing a Trust-Assessment Model that will allow estimating the Trust-degree that can be granted to a given node. As explained above, a Trust-Assessment Model proceeds in two steps: (i) The Trust-composition step and (ii) The Trust-Aggregation step. For aggregating trust in SIoT, Truong et al. (2016,0) propose to base on fuzzy-logic. Chen et al. (2016a) opt for combinatorial-logic based aggregation method. That said, the vast majority of related work opt for the weighted-mean based aggregation method (Bernabe et al. (2016), Chen et al. (2016c), Huang et al. (2016), Jayasinghe et al. (2016), Militano et al. (2016), Nitti et al. (2014b), Truong et al. (2018)). We propose, in the following, to compare the Trust-Assessment Models proposed for SIoT in the literature according to three criteria. (i) The definition of trust, as we can find in the literature, those who, when proposing Trust-Features, have focused only on the "Intention" aspect, and those who consider only the "Ability" aspect; (ii) The consid-

ered trust-dimensions, as there are related-works that have proposed Trust-Features only for the user dimension, and those that have also dealt with the device and/or the service dimension; and (iii) The purpose of the Trust-Assessment Model, since some related-works has implemented TAM to detect Trust-Attacks, while many others aimed solely at Ranking nodes.

2.6.1.1 Feature engineering for trust-composition: Intention or Ability

When defining the Trust concept, Truong et al. (2017) and Huang et al. (2016) are the only who have considered both the Intention and the Ability aspects. Truong et al. (2017) propose the Reputation feature to estimate the Intention of a user and the Knowledge and Experience features to estimate users Ability. Bernabe et al. (2016) propose the "QoS" dimension which allows estimating service's Ability, and Chen et al. (2016c) proposed the "Current energy status" feature to estimate the device's ability.

The majority of other related-works focus only on the Intention perspective. Jayasinghe et al. (2016) propose the Recommendation and Reputation features. Nitti et al. (2014b) propose the Centrality, the Direct Experience and the Indirect Experience features. Chen et al. (2016a) propose the Honesty, Cooperativeness and Community-Interest features, and Militano et al. (2016) propose the Reliability and the Reputation features. All the mentioned features are designed to estimate the user's Intention.

2.6.1.2 User trust-dimension vs Multi trust-dimensions

The majority of the literature consider only the user trust-dimension (Huang et al. (2016), Jayasinghe et al. (2016), Militano et al. (2016), Truong et al. (2017,0)) and propose features such as: Reputation (Jayasinghe et al. (2016), Truong et al. (2018)), community-Interest (Chen et al. (2016a)), Social relationship (Chen et al. (2016c)) or centrality (Nitti et al. (2014b)), which are related only to user-type nodes . Bernabe et al. (2016) are the only to have considered the service trust-dimension and proposed the "QoS" dimen-

sion which refers to the evaluation of the overall quality of service provided by a device. This dimension is comprised of four main indicators: (i) The "Successful-Interactions property" recaps the percentage of successful interactions over the total amount of previous interactions within the device. (ii) "The Availability property" indicates the proportion of time that the IoT device is operating. (iii) The network Throughput is measured in bits per seconds and is defined as the rate of successful packets delivery over a communication channel. (iv) The average network Delay indicates how long it takes for a bit of data to travel across the network from one device to another.

Chen et al. (2016c) are the only who are interested in the device trust-dimension and propose the "Current energy status" feature. According to Chen et al. (2016c), the remaining energy factor and consumption rate factor (reflected by running time) are denoted to determine the energy status of a node in the present. Nodes with less energy will have a restrictive trustworthiness value, and their opportunity for cooperation will decrease to make the network stable

2.6.1.3 Ranking vs Attack-Detection

We can classify related works according to the purpose of the Trust-Assessment Model into two categories. Those who are interested in the detection of trust-related attacks (Chen et al. (2016a) and Chen et al. (2016c)) and those who are interested in the nodes-ranking (Bernabe et al. (2016), Huang et al. (2016), Jayasinghe et al. (2016), Militano et al. (2016), Truong et al. (2017,0)). The related works who focus on attack-detection aim at capturing suspected nodes. It is generally binary models that return as output classification of the node (benevolent/malicious, trust/distrust). The advantage of this approach is that it provides the ability to isolate malicious nodes and bypass the damage they cause. However, this approach remains restrictive, as it does not provide scores allowing to compare service providers and select the best one. Moreover, when we look closely at the related-works that has focused on Attack-Detection (Chen et al. (2016a) and Bao and Chen (2012b)), we notice that: (1) They do not deal with all types of Trust-Attacks, (2) We

can not see logical or semantic relationships between the proposed features and the description of attacks, and (3) They do not prove the resilience of the model against the treated Trust-Attacks through experiments.

For example, Bao and Chen (2012b) are only interested in BSA, BMA and SPA attacks. They suggest the "Honesty", "Community-of-interests" and "Co-operativeness" features, where, Cooperativeness is calculated as the number of common friends between two users, Community-of-Interest is calculated as the number of common interests, and Honesty is calculated as the number of negative received rating. The "Honesty" factor is, in this case; particularly sensitive to Trust-Attacks, as the rating on which this calculation is based can be erroneous.

The related works that focus on nodes ranking returns as output a value, generally, in the interval of [0,1] that represents the degree of trust that can be placed in a service provider and therefore allows to rank providers according to this score and helps the trustor to choose the best among them. This method maximizes the chances of good service providers being selected but does not prevent malicious nodes from acting. Besides, by launching BMA or BSA attacks, for example, malicious nodes can skew trust-scores.

2.6.2 Trust propagation: Scalability vs Resources-efficiency

The vast majority of related work has followed the centralized propagation method (Huang et al. (2016), Jayasinghe et al. (2016), Militano et al. (2016), Truong et al. (2017,0)). While this method is certainly robust and easy to implement, it is not appropriate in networks with up to billions of devices and an equally large number of exchanges and interactions per second. Thus, the TMMs they propose do not allow for scalability. Bao et al. (2013), Nitti et al. (2014b) are the only authors to have opted for the decentralized propagation method. This approach allows scalability to be ensured, but is very resources-consuming, especially if the proposed trust model is complex. It is therefore as inapplicable as the centralized method, given that most of the IoT devices

are characterized by limited computing and storage capacities. To overcome the drawbacks of this method, Nitti et al. (2014b) suggests combining it with a storage-management strategy. They propose that only special nodes, that called Pre-Trusted Objects (PTOs), can store the data about feedback or trustworthiness values. PTOs do not provide any service and are integrated into the architecture; their number is decided based on the number of nodes in the SIoT so that there is always a PTO available to manage the data.

Bao et al. (2013) propose to combine the distributed propagation approach with a Storage Management Strategy which consists of considering only nodes with the highest trust values and recent interacting nodes. The assumption behind it is that authors consider that these nodes are most likely to share common interests. That said, this strategy, although it saves storage and computation consumption, makes all traces of the interactions made by malicious users disappear.

2.6.3 Trust updating: Dynamism vs Resources-efficiency

The trust updating step consists of choosing a method for updating trust values, either the time-driven method or the event-driven method. The problem with the time-driven updating is that it is hard to fix the period. Indeed, it may be possible that no attack is carried out for a month, and that one hundred attacks are made in 20 minutes. The majority of authors in literature opted for the event-driven method Chen et al. (2016a), in which updates are done each time an event occurs. Its merit is that it allows having a real-time trust management system. The drawback of this method is that it will generate a huge number of updates and is very resources consuming.

2.6.4 Synthesis

The majority of related works focus on the user trust-dimension and propose only features asses the Intention of user type nodes while neglecting the Ability of device and service type nodes. To aggregate these factors, they

opt for the weighted mean, to classify network users according to trust scores and are not interested in Trust-Attacks detection. Those who deal with this problem are not interested in all the types of Trust-Attacks cited in the literature. To propagate trust values, the authors are divided between the centralized method which does not allow scalability and the decentralized method which is not adapted to limited and constrained SIoT nodes. Finally, to manage and update the trust values, the majority of the works opt for the event-driven method. This method is more robust, but it involves a large number of updates, which is not feasible for constrained SIoT nodes.

In our work, we propose a multidimensional Trust Assessment Model that encompasses all three trust-dimensions and propose features for assessing both Intention and Ability. Our model allows the classification of nodes but also the detection and countering of all types of Trust-Attacks, cited in the literature, thanks to a mechanism based on Machine Learning and Deep Learning. To propagate trust values in the network, we propose a hybrid aggregation method, which overcomes the drawbacks of centralized and decentralized methods. This hybrid propagation method consists in inserting trust nodes which base on distributed hash tables. Each trust node supports a set of nodes in the network and performs the necessary calculations for it. This method allows us to apply the event-driven update method.

2.7 Conclusion

In this chapter, we presented the background and the main concepts that need to be assimilated before setting up a Trust Management Mechanism. Then, we presented a review of the literature that allowed us to identify gaps and set the objectives of our work.

In the next two chapters, we will present our contribution, including the methods chosen to implement DSL-STM , a "Dynamic Scalable Lightweight - Social Trust- Mechanism", for Social Internet of Things.

Part II

Second part

Chapter 3

The Trust-Composition: Multidimensional Trust-Features engineering

Contents

3.1 Introduction

As explained in section 4 of the previous chapter, the process of trust management consists of three main phases. The first phase being the "Trust-building phase" allows the implementation of a Trust-Assessment model to estimate the trust-degree that can be granted on each node in the network. This phase is carried out in two steps. The (i) Trust-Composition step which consists in selecting the Trust-Features, and (ii) the Trust-Aggregation steps which focus on Attack-detection, node-ranking or both of them. As in our work we are interested in both attack-detection and node ranking, we start this chapter by providing the attack-model. The latter allows us to depict and identify the different malicious behaviours that may occur in SIoT and thus to propose the appropriate Trust-Features allowing to detect them. Then we expose the architecture of our Multidimensional Trust-Assessment Model, which deals with the three trust-dimensions involved in SIoT environment, namely, (i) Users-trust dimension, (ii) Devices-trust dimension and (iii) Services-trust dimension. We focus in this chapter, only, on the first part of the architecture which concerns the Trust-Composition step, as the Trust-Aggregation steps will be the topic of the following chapter.

The rest of the chapter is divided into three sections, each devoted to a different trust-dimension. The first section presents the features allowing to estimate the intention and the ability of user type nodes. The second and third sections present, respectively, the features allowing to estimate the ability of devices and services type-nodes.

3.2 Trust-Attacks model

A trust attack is a set of interactions conducted by a malicious user, to bias and mislead the reputation system. By launching an attack, the attacker seeks to propagate poor quality services in the network. We focus, in this section, on Trust-Attacks. We detail each type to build an intelligible Attack-Model. We end the section by a comparison between all the detailed attacks.

3.2.1 Bad Mouthing Attack (BMA)

Bad Mouthing Attack (BMA) belongs to the Malicious Recommender Attacks category. In BMA, malicious nodes collude to destroy the reputation of well-behaved ones. They collaborate by assigning low rates values to the targeted nodes. Attackers are generally those who fail to propagate their services in the network, either because of their bad reputation or because of the poor quality of the services they provide. In both cases, these attackers are not well ranked and their chances of being selected as a service provider are reduced. Destroying the reputation of other benevolent nodes allows them to move up in the ranking and increases the likelihood of the invocation of their services. These attacks are therefore launched against nodes having a good reputation in the network and providing qualified services. The BMA attacks are, generally, colluding attacks carried out in groups, and requiring a large number of interactions. Indeed, numerous low ratings are needed to effectively smash the reputation of the targeted node(s).

Figure 3.1 shows an example of a benevolent node u_i that has a good reputation (0.8/1) and offers three good-quality services ($s_{1.1}$, $s_{1.2}$ and $s_{1.3}$). Nodes u_2 and u_3, having a bad reputation and providing poor quality services, are bent on giving negative ratings to u_1. They, thus, manage to lower the reputation of u_1 from 0.8 to 0.3 and become ranked in the same position as him.

To summarize, in this type of attack, the attacker is a node with a low reputation value. The target is a node with a high reputation value. We

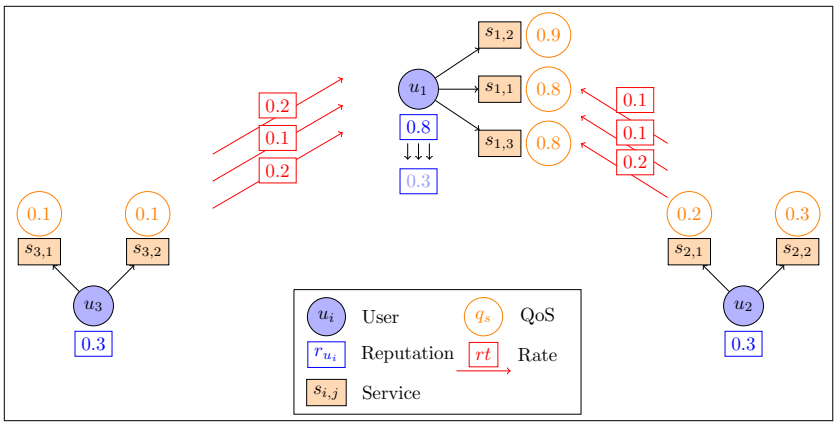


Figure 3.1: Example of Bad-Mouthing Attack.

can notice a large number of rating attributed by the attacker to the target, which reflects his persistence. These ratings are mostly negative and are therefore different from the rating assigned by the majority of other nodes in the network to the target.

3.2.2 Ballot Stuffing Attack (BSA)

Ballot Stuffing Attack is a colluding attack where malicious disreputable nodes collaborate to promote their mutual reputation. To do so, each of the nodes will assign positive ratings to the other. The figure 3.2 shows an example of a BSA attack. Nodes u_1 and u_2 with low reputation values of 0.2 and 0.3 respectively and offering poor quality services $s_1 1$, $s_1 2$, $s_1 3$, $s_2 1$ and $s_2 2$ are no longer able to propagate their services in the network and are no longer solicited. Node u_1 assigns several positive ratings to u_2 and node u_2 assigns several positive ratings to u_1. These dishonest and unrepresentative ratings will increase the reputation values of both malicious nodes to 0.8. They will therefore be considered as good providers in the network and will have a free hand to propagate their poor quality services. To summarize, in this type of attack, the attacker is a node with a low reputation value. The target is also a node with a low reputation value. We can notice a large number of ratings attributed by the attacker to the target, which reflects his persistence. These ratings are for the majority positive and are therefore

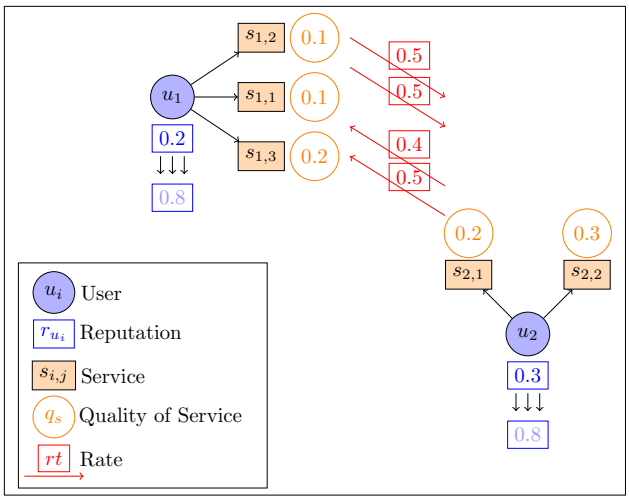


Figure 3.2: Example of Ballot-Stuffing Attack.

different from the rating attributed by the majority of other nodes in the network to the target.

3.2.3 Self Promoting Attack (SPA)

Self Promoting Attack (SPA) is an individual malicious recommender attack. In this attack, malicious nodes providing poor quality services and having a low reputation value try to boost their self-reputation. To do so, they are replicating themselves and self-assigning positives ratings. This malicious behaviour will allow them to obtain a good reputation value and allows them to spread poor quality services in the network. The figure 3.3 shows the example of a node u_1 who has a bad reputation value equal to 0.3 and who provides bad quality services $s_{1,1}$, $s_{1,2}$, $s_{2,1}$ and $s_{2,2}$. The node u_1 used his both devices d_1 and d_2 to assign positives rates to himself. As a result, the reputation of u_1 increase and reach 0.8. The other nodes in the network will take him for a good service provider and will therefore invoke his services.

As in the case of the BSA attack, in this attack, the attacker is a node with a bad reputation and offering bad services. The target (which, in this case, is the same node) is therefore also a node with a bad reputation and offering bad services. We also notice a large number of ratings which are positive

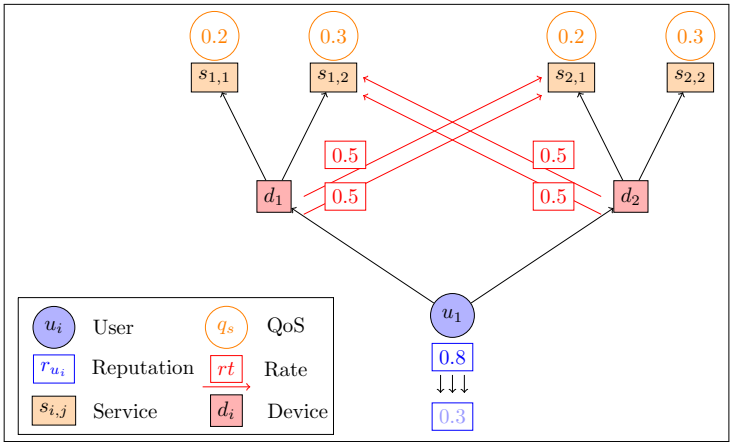


Figure 3.3: Example of Self-Promoting Attack.

and different from the majority of the other ratings attributed by the other nodes in the network. That said, contrary to the BSA attack where the target and the attacker are different nodes, in the SPA attack it is the same one, which will be reflected by a great similarity between the attacker node and the target node.

3.2.4 Discriminatory Attack (DA)

Discriminatory Attack (DA) is an individual malicious recommender attack where malicious nodes attack discriminatory other nodes, without a strong social relationship with them. As with all malicious recommender attacks, the malicious node will use erroneous and unrepresentative rating to attack other nodes. However, in the DA attack, the attacker has no specific criteria to choose his target. He does not target well-reputed nodes as in BMA or bad-reputed ones as in BSA but assigns negative rates to all nodes with which he does not have a strong relationship. Figure 3.4 shows an example of a node u_1 with a bad reputation that assigns negative ratings to other nodes regardless of their reputation or the quality of services they provide.

In this type of attack, the attacker is a node with a bad reputation. The target can be any other node and has no special characteristics and can be either a good provider or a bad one. We can not notice any harshness on

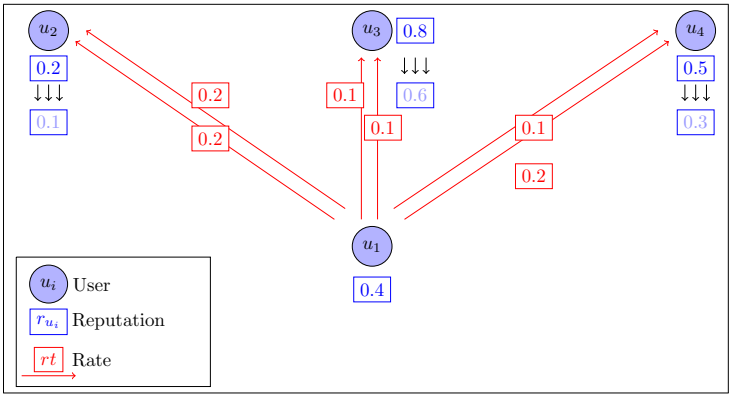


Figure 3.4: Example of Discriminatory Attack.

a particular node, nor a clear difference between the rating assigned by the attacker and those assigned by the other nodes. Indeed, the target can also be a bad provider who received negative ratings from other legitimate nodes. We notice, however, that the vast majority of the rating provided by the attacker are negative. We can also notice that the targets and the attacker do not have a strong relationship.

3.2.5 On-Off-Attack (OOA)

On-Off Attack (OOA) is an individual malicious provider attack which means that the attacker is not using ratings to skew the reputation system. In OOA the malicious node performs bad services randomly instead of always performing best services. It aims to avoid being labelled as a low trusted node by the trust system, to be selected as a service provider. In this case, the misbehaving node is identified as a trustee with medium old and new reputation values (neither high nor low compared to the selected threshold by the trust system).

Figure 3.5 shows an example of the OOA attack. The user who alternatively provides good/bad services, also alternatively receives high/low rates. The blue curve shows the variation of the received rates. The red curve shows the reputation of the node which is calculated classically as the average of received rates. The figure shows that the reputation is always greater than

or equal to the threshold (set here to 2.5). This gives this malicious provider the hand to propagate its poor quality services.

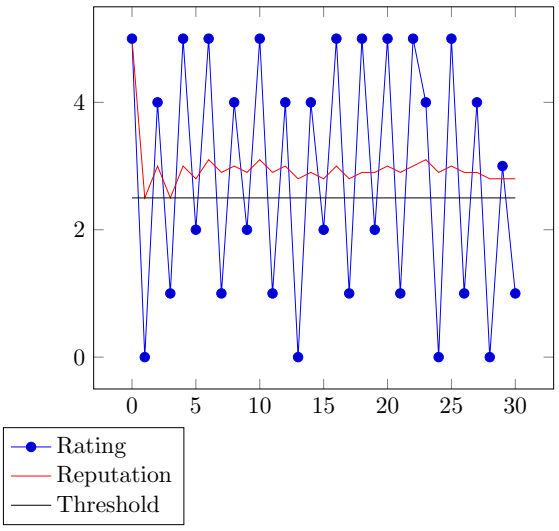


Figure 3.5: Example of On-Off Attack.

3.2.5.1 Opportunistic Service Attack (OSA)

The malicious user starts to provide good quality services until he reaches a high reputation value. Then he starts providing poor quality services. His reputation will gradually decline, but he will be able to spread a large number of poor quality services before his reputation value falls below the threshold.

The figure 3.6 shows an example of the OSA attack. The blue curve, which represents the ratings received by the user, shows how he provided good services up to the sixteenth interaction and started providing bad services from the seventeenth interaction onward. The red curve which represents the reputation of this user and which is calculated as the average of the received rates remains stable and then starts to decrease gradually. The figure shows that the reputation curve (red curve) remains above the threshold curve (black curve) even after the user has provided 15 poor services. This means that he can still be solicited despite the poor quality of the services

he provides.

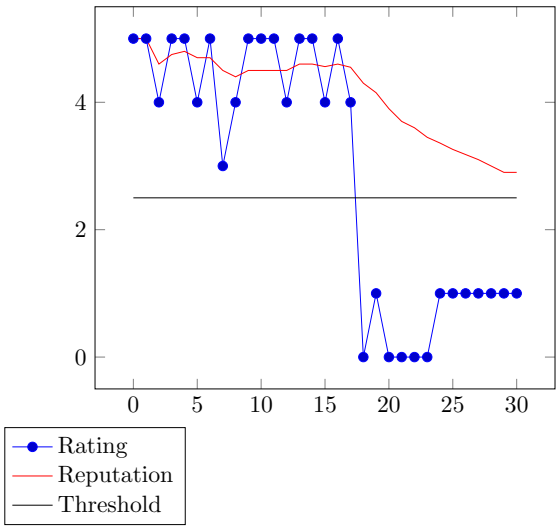


Figure 3.6: Example of Opportunistic Service Attack.

3.2.6 Trust-attacks comparison

According to the different descriptions, explanations and diagrams presented above, we propose in the following, comparisons of the attacks, according to different criteria. We compare in the table 3.1 the malicious recommender attacks according to (i) the characteristics of the attacker, especially his reputation and the quality of his services; (ii) the characteristics of the target, including his reputation and the quality of his services, but also his legitimacy, as in some colluding attacks (BSA for example) the target is also a malicious user; (iii) characteristics related to the attacker/target relationship, including frequency of interactions, strength of the relationship, similarity and rating trend; and finally (iv) characteristics related to the attacker's relationship with other nodes in the network, which differs, generally and depending on the type of attack, from its relationship with the target. The literature that has focused on Trust-Attacks detection has not considered these different aspects and has focused only on the characteristics related to the attacker

himself. We believe that a more complete and contextual description will allow us to find more relevant features.

Table 3.1: Malicious recommender attacks comparison

		BMA	BSA	SPA	DA
Attacker	Reputation	↘	↘	↘	↘
	Service quality	↘	↘	↘	↘
Target	Malicious/ Legitimate	L	M	M	NC
	Reputation	↗	↘	↘	NC
	Service quality	↗	↘	↘	NC
Attacker - Target	Rating frequency	↗	↗	↗	NC
	Rating value	↘	↗	↗	↘
	Relationship strength	NC	↗	NC	↘
	Similarity	NC	NC	↗	NC
Attacker - Other nodes	Rating similarity	↘	↘	↘	NC

We then compare, in the table 3.2, all Trust-Attacks, including Malicious Recommender Attacks and Malicious Provider attacks according to the following criteria: (i) The role of the attacker, because, as explained before, a user can be a recommender and/or a service provider; (ii) The behavior during the attack, especially if it is a colluding or individual attack; and finally (iii) The targeting, because in some attacks the attacker has a target while in others he does not. These comparison criteria help guide us in the choice of features that will enable us to detect the different types of attacks. For collusion attacks, the features that allow us to study the relationship between the attackers may be of interest. In targeted attacks, local features may be more interesting, while in non-targeted attacks, global features are sufficient. A local feature is a feature that involves two users $ft(u_i, u_j)$ and that for the same u_i varies in value according to the identity of the u_j. The "Similarity" is, for example, a local feature because $sim(u_i, u_j) \neq sim(u_i, u_k)$. A global feature is relative to a single user $ft(u_i)$ and keeps the same value independently of other users on the network. "Reputation" (average of received rating) or "Quality of Service" are global features. The dependency or not of a chronological order, also allows to avoid a bad choice of feature. Indeed, in BMA, BSA or other attacks, the order of interactions has no ef-

fect. However, in Malicious Provider Attacks (OOA and OSA) the same set of interactions in another order are no longer considered as an attack. This type of attack involves the integration of temporal features.

Table 3.2: Trust-attacks comparison

	Attackers role		Attackers behaviour		Targeting	
	Recommendation	Service Provision	Colluding	Individual	Target	No target
BMA	✓		✓		✓	
BSA	✓		✓		✓	
SPA	✓			✓	✓	
DA	✓			✓		✓
OOA		✓		✓		✓
OSA		✓		✓		✓

3.3 The DSL-STM Trust-Assessment Model architecture

The figure 3.10 illustrates the architecture of the proposed Trust-Assessment model. For the trust-composition step, we can distinguish the three trust-dimensions and the proposed features for each one. The mentioned trust-dimensions are based on the trust definition proposed in the Background part, and thus allow us to assess either the "Intention" of the node, or its "Ability", or both according to the purpose of the trust. Indeed, we trust a "Service-type node" for its "Ability" to meet our request, to be available when we need it, to process in a reasonable run-time... etc. We trust a "Device-type node" for its "Ability" to execute the required service (computing capacity, storage capacity, ..), for its availability when needed (connectivity, battery level, ...) and for its respect of minimum security measures (this device must not present security flaws that may endanger the whole network). Finally, a user node is trusted for his "Ability" to provide qualified services and for his "Intention" when providing honest and representative feedback and recommendation.

The Trust-Aggregation step takes as input those different Trust-Features and is composed of two main modules. (1) **The Attack-Detection module** concerns only user-type nodes, because as explained in the background chapter, Trust-Attacks are performed by the user-type node. This module proceeds in two steps to detect and distinguish the different type of launched attacks. By this way, this module will divide the network users into three classes: (i) Legitimate users; (ii) Malicious Recommender users; and (iii) Malicious Providers users. This distinction is essential to clarify the behaviour of the TAM to counter the effect of those attacks according to their type.

The node ranking model allows computing a Multidimensional Trust-Score based on the three Trust-Dimensions.

3.4 User Trust-Dimension

In this section, we focus on the user-trust dimension. A user node is trusted for his "Ability" to provide qualified services and for his "Intention" when providing honest and representative feedback and recommendation. We set-up a set of Trust-Features enabling to guess the user intention by depicting if he tries to perform a trust-attack.

3.4.1 Intention

The intention applies only to user type entities. Indeed, device-type or service-type entities act under the control of their owners or suppliers and have no intention. A user's intention indicates his willingness to comply with the various rules imposed by the system, the environment, and other users. To quantify the intention of a user, we propose three features, including (i) honesty, (ii) compliance with system rules, and (iii) compliance with users' requirements.

The honesty feature indicates that a user gives rates, which represent his real opinion about service and is measured by the trust indicator Credibility. The compliance with system rules indicates that a user does not try

to perform any of the cited trust-related attacks and is measured by different trust indicators, namely Credibility, Reputation, Similarity, Direct-Experience, Rating-Frequency, and Quality of Provider. The compliance with other users' requirements is a subjective feature indicating whether a user u_i respects the preferences of a user u_j or not. It is measured by the trust indicator Similarity, which represents affinity between users.

Reputation The term Reputation is widely used in literature to designate the renown, the notoriety and the prominence of a node in a network. It is necessary to differentiate between the concept "Reputation" and the indicator "Reputation", which consists in setting a measure to estimate this concept. The trust-indicator "Reputation" is denoted as $Rep(u_i)$ and represents the global renown of a user u_i in the overall network. It is computed as the quotient between the number of positive interactions and the total number of interactions (eq.3.1). Positive interactions are the ones that received a high rating value. Users with a high reputation value are more likely to be attacked by other users. Users with a low reputation value are more likely to perform trust-attacks. The reputation trust-indicator, combined with other trust-indicators, can help to reveal BMA, BSA, SPA, and DA attacks.

$$Rep(u_i) = \frac{1}{N^i} \sum_{k=0, r_k > m}^{N^i} r_k \tag{3.1}$$

(where N^i is the number of rates attributed to the user u_i, r_k is the rating value and m is the threshold, above which a rating value is considered as positive.)

Credibility The term Credibility is often associated with the term trust. It is widely used in literature. Nevertheless, it does not admit a consensual definition and is measured in different ways in each related-work. In our work, a user is considered credible if his ratings reflect his opinion, which means that he does not try to give wrong rating values to promote or decrease other users' reputation.

The Credibility feature is, a key factor as it may reveal when associated with other features, different types of attacks. Indeed, in the BMA attack, the malicious user gives bad ratings to a user providing good quality services, to ruin his reputation. In the BSA attack, the malicious user gives good ratings to another malicious user to promote his reputation. In the SPA attack, the malicious user tries to increase his own reputation by giving himself good ratings though his services are of poor quality. To measure the Credibility, we use the Cosine Similarity in order to compare the user rating vector with the overall rating matrix. If the user rating vector is far from the majority of other users' ratings, he is considered as incredible. We admit that a user may have different rating just because he has a different opinion. Nevertheless, in this case, his rates remain unrepresentative. Note that it is the combination of this indicator with all the other indicators that will decide if the user is malicious or not.

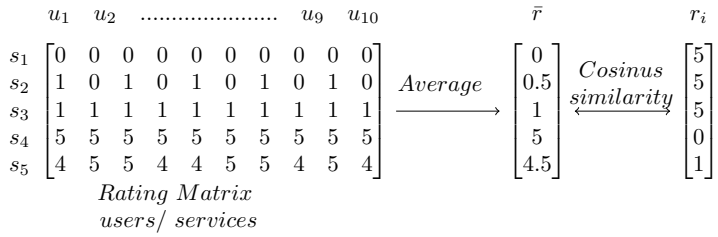


Figure 3.7: Measuring credibility trust-indicator.

To measure the similarity between a user's rating vector and the rating matrix, Cosine Similarity Measure is used (eq.3.2). The figure 3.7 shows an example of a rating matrix where we find the rating assigned by all the users to all the services they invoked. The first step is to extract $\bar{r}$, a column vector of dimension 1, containing the mean of ratings assigned to each service. The second step is to apply the cosine similarity, in order to compare this mean rating vector with u_i user rating vector. We can note, in figure 3.7, from the rating matrix, as well as from the mean rating vector $\bar{r}$ that services s_1, s_2 and s_3 are judged to be of good quality by the majority of users, while services s_4 and s_5 are judged to be of poor quality. The comparison of the

u_i vector with the mean rating vector shows that the ratings of u_i are far from the majority of other users rates. It is, therefore, possible that u_i is launching BMA and BSA attacks.

$$Cred(u_i) = MAX_r - \frac{1}{S}\sum_{j=0}^{S}\sqrt{(x_{i,j} - \bar{x}_j)^2} \tag{3.2}$$

(Where $x_{i,j}$ is the rate of the user i on the service j, $\bar{x}_j$ is the average of rates given by all network nodes on service j, S is the total number of services in the system and MAX_r is a static variable indicating the highest rating value.)

Direct-Experience This feature refers to the opinion of a user u_i about his past interactions with another user u_j and is denoted as $dExp(u_i, u_j)$. Unlike reputation and credibility, which are global indicators, direct experience is a local indicator. This means that its value varies from one pair of users to another. For example: $dExp(u_i, u_j) \neq dExp(u_i, u_k)$. This indicator is also asymmetrical as $dExp(u_i, u_j) \neq dExp(u_j, u_i)$, whereas for global indicators, u_i will have the same value of Reputation $Rep(u_i)$ and Credibility $Cre(u_i)$ for all the other users in the network.

The direct-experience indicator is computed as the quotient of successful interactions between user u_i and user u_j, divided by the total number of interactions between them. The direct experience feature can not, therefore, directly reveal an attack. However, combined with other features, it helps distinguish what kind of attack it is. Indeed, in the BMA attack, the node u_i aims to smash the reputation of u_j by assigning many low rates to him, which will result in a low $dExp(u_i, u_j)$ value. However, in the BSA attack, the node u_i aims to increase the reputation of u_j by giving him high rate values, which will lead to a high $dExp(u_i, u_j)$ value.

$$dExp(u_i, u_j) = \frac{1}{N^{ij}}\sum_{k=0}^{N^{ij}} r_k(u_i, u_j) \tag{3.3}$$

(where N^i is the number of rates attributed by the user u_i to the user u_j and $r_k(u_i, u_j)$ is the rating value assigned by the user u_i to the user u_j.)

Rating-Frequency The Rating-Frequency is another local and asymmetric indicator which relies on rating frequency to estimate if a user u_i target another user u_j and is denoted as $RateF(u_i, u_j)$. The idea behind this indicator comes from the description of the different types of attacks. In the BMA attack, a user aims to ruin the reputation of another well known legitimate user in the network. He will therefore try to give him a large number of negative rates in order to decrease his reputation. In the BSA attack, a user seeks to promote the reputation of another malicious user with a bad reputation in the network. He will, therefore, assign him a large number of positive rates, in order to increase his reputation's value. In the SPA attack, a user that provides bad services and has a bad reputation will attribute to himself a large number of positive rates in order to increase his own reputation value. In all three types of attacks, the large number of rates attributed is revealing, because a small number of rates will not allow modifying the reputation of the targeted node.

Figure 3.8 shows an example of an interaction history for three types of behaviour. In each interaction history, we have s_{ij} the invoked service, p_i the provider of the invoked service and rt the assigned rating value. The first table shows the normal behaviour of a user who invokes services from different providers and assigns positive/negative rates regardless of the identity of the provider. The other two tables show the behaviour of users that are bent on invoking services provided by a particular provider which is the u_1 user. In the second table, the attacker launches a BMA attack and assigns a large number of negative rating to u_1. In the third table, the attacker launches a BSA attack and assigns a large number of positive rates to u_1. To calculate $RateF(u_i, u_j)$ the rating-frequency of u_i to u_j, we divide the number of rates assigned by u_i to u_j by the total number of rates assigned by u_i (3.4).

$$RateF(u_i, u_j) = \frac{\sum_{p_z^{ij} \in I_i} z}{\sum_{p_y^{ik} \in I_i, u_k \in U} y} \tag{3.4}$$

(where I_i the set of interactions of node u_i with all network nodes, p_z^{ij} The z-th interaction between the user u_i and the user u_j, p_y^{ik} The y-th interaction

Normal behaviour

Rt	S_{ij}	P_i
4/5	s_{11}	u_1
5/5	s_{42}	u_4
0/5	s_{21}	u_2
4/5	s_{22}	u_2
1/5	s_{32}	u_3
5/5	s_{41}	u_4
4/5	s_{33}	u_3
0/5	s_{13}	u_1
4/5	s_{15}	u_5
2/5	s_{25}	u_5

BMA behaviour

Rt	S_{ij}	P_i
0/5	s_{11}	u_1
1/5	s_{12}	u_1
3/5	s_{21}	u_2
1/5	s_{13}	u_1
0/5	s_{14}	u_1
0/5	s_{15}	u_1
4/5	s_{16}	u_1
0/5	s_{31}	u_3
1/5	s_{17}	u_1
0/5	s_{18}	u_1

BSA behaviour

Rt	S_{ij}	P_i
4/5	s_{11}	u_1
5/5	s_{12}	u_1
3/5	s_{21}	u_2
4/5	s_{41}	u_4
4/5	s_{14}	u_1
5/5	s_{15}	u_1
4/5	s_{16}	u_1
0/5	s_{31}	u_3
4/5	s_{17}	u_1
4/5	s_{18}	u_1

Figure 3.8: Examples of rates targeting distribution for different behaviours.

between the user u_i and another user u_k, with u_k belonging to U the set of all the users in the network.)

Rating-Trend The Rating-Frequency is a global and asymmetric indicator used to understand a user's general behavior in the network and to detect whether he is mainly optimistic or pessimistic. This feature allows to estimate whether the user's opinion is representative. Indeed, the opinion of a user who displays an unmitigated opinion, and who gives a majority of negative rates to everyone, or conversely positive rates to everyone, is representative of his extremist attitude and not of the quality of the interactions he has had. The weight of this type of rates must therefore be lower than the weight of the other rates. In addition, this feature makes it possible to reveal DA-type attacks, in which the malicious user randomly and discriminatory assigns negative rates to all other users, regardless of the quality of the services they provide.

The rating-trend feature is denoted as $RateT(u_i)$ and is and is calculated as the number of positive rates assigned by the user u_i divided by the total number of rates assigned by the user u_i.

Similarity The similarity indicator aims to measure the degree of similarity between two users. This indicator is based on all the data that can be collected about a user from his profile, his list of interests, his social

network, his interaction history, his locations, etc....This indicator is local and symmetrical. This means that $SimU(u_i, u_j) \neq SimU(u_i, u_k)$ and $SimU(u_i, u_j) = SimU(u_i, u_k)$. To estimate this similitude we use Jaccard Index.

The similarity indicator can reveal affinities between users, but it is mainly used to reveal that it is the same user under another identity. This indicator is not interesting in the case of certain attacks such as the BMA or BSA attack. But, combined with other indicators, it allows detecting a SPA type attack, where the user hides under a false identity to self-promote his reputation.

Relationship strength In order to detect colluding attacks, it is interesting to study the relationships that occur between users. Indeed, users that collaborate to launch an attack have, potentially, a strong relationship. We, therefore, propose the Relationship-Strength indicator, denoted as $RelStren(u_i, u_j)$ which makes it possible to use the type as well as the number of relationships that links two users, to estimate the strength of this relation. Indeed, two users can be linked by several relations of different types. A user can, for example, have both a social relationship and a co-location or co-working relationship with another user.

In the table 3.3 we present the weights assigned according to the type of the relation. These weights are set according to the semantics of each type of relationship. The parental relationship is estimated as the weakest one because it just links objects belonging to the same category. The ownership relationship that links two devices belonging to the same user is the strongest.

To calculate this indicator, we calculate, two by two, the strength of the relationship between the devices of user 1 and the devices of user 2, based on the weights presented in the table. The equation 3.5 explains this calculation.

$$RelStren(u_i, u_j) = \frac{1}{|D_i||D_j| \times T_w} \sum_{d_k \in D_i} \sum_{d_y \in D_j} \sum_{r_z \in R_{ky}} f^w(r_z) \qquad (3.5)$$

With D_i the set of devices belonging to the user u_i, D_j the set of devices

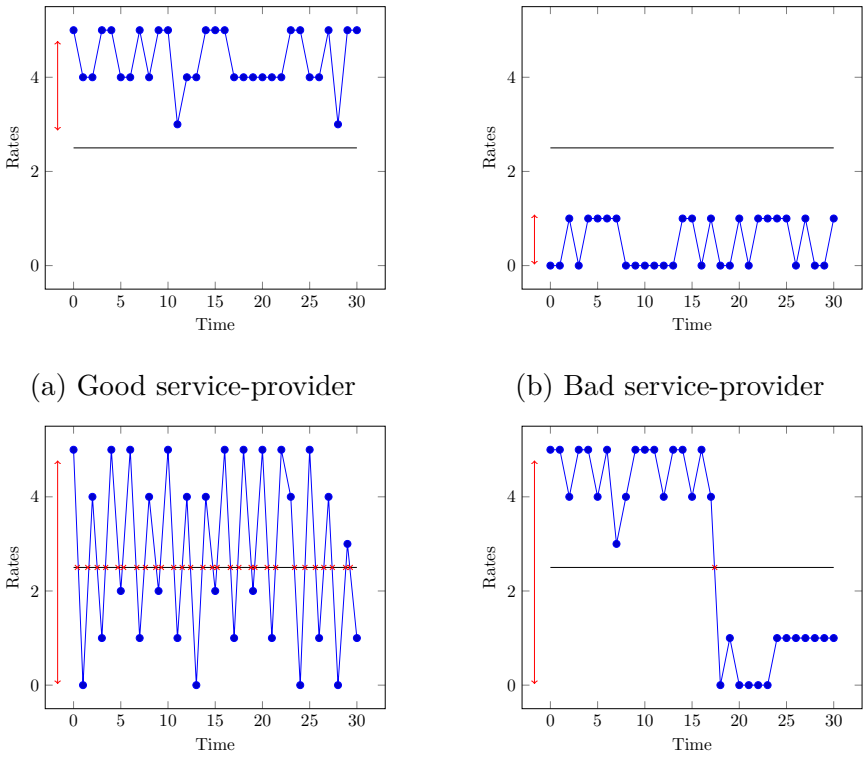
Type of relationship	Weight
or	1
sr	0.8
wr	0.8
lr	0.5
pr	0.0

Table 3.3: Weights according to relationship type

belonging to the user u_j, $T_w = 2.2$ *the sum of the weights for all types of relations,* R_{ky} *the set of relations linking the device* d_k *and the device* d_y, f^w *the function assigning to each type of relation* r_z *the corresponding weight according to the table.*

Fluctuation Figure 3.9 shows an example of the votes received by four users who have four different types of behaviour. The sub-figure 3.9.a shows the case of a good service provider, which always receives votes above the threshold (black horizontal line). The sub-figure 3.9.b shows the case of a bad service provider who always receives votes below the threshold. Despite the poor quality of the services he offers, this user is not a problem. Indeed, he does not launch an attack to improve his reputation and he will no longer be solicited because of his bad reputation. The subfigures 3.9.c and 3.9.d show the cases of a user launching the OOA attack and a user launching the OSA attack, respectively. The latter two have oscillating behaviour and alternate between good and bad services to keep a reputation above the threshold.

To capture this kind of behaviour, we propose the feature called Fluctuation e denoted as $Fluct(u_i)$. It consists in counting the number of passes through the threshold and dividing it by the total number of interactions. The higher this value is, the more the user is estimated as having an oscillatory and not stable behaviour that reveals an OOA or OSA attack.



(a) Good service-provider

(b) Bad service-provider

(c) OOA attack

(d) OSA attack

Figure 3.9: Example of rating variation in Malicious provider attacks.

3.4.2 Ability

A user's ability concerns only the user as a recommender. If a user chooses to provide poor quality services (provider), it has no relation to his ability and instead shows his bad intention. If a user gives rates that are not representative (recommender role), this may be due to two different reasons. Either the user in question intends and tries to perform an attack, or this user is a beginner and is, therefore, unable to provide adequate rates. It is, in this case, the user's ability. To quantify the user's ability, we propose the Expertise trust-indicator.

User Expertise The Expertise indicator evaluates the user as a recommender. It allows judging the user's ability to give representative rates after soliciting a given service. Services can be grouped according to their domains (for example, weather, traffic management, ...). The more a user has used services belonging to a given domain, the more his rates can be useful and representative. The expertise is therefore defined according to the domain and is denoted as $Expt(u_i, d)$ where u_i represents the user and d the service's domain. It is calculated by the ratio of the number of services invoked by u_i belonging to the d domain, divided by the total number of services invoked by u_i.

$$Expt(u_i, d) = \frac{\sum_{p_z^{ij} \in I_i} z}{\sum_{p_y^{ik} \in I_i, u_k \in U} y} \tag{3.6}$$

3.5 Device Trust-Dimension

Devices node are not concerned with Intention, as they act under the control of their owners. In this section, we are interested in quantifying the Ability of device type-node. We present in the first sub-section the criteria we choose for classifying Devices nodes. Then we introduce the Device-Trust score indicator, which bases on the cited criteria to assign a score allowing to estimate the ability of each device.

3.5.1 IoT devices classification criteria

A "Device-type node" is trusted for its "Ability" to :

- Execute the required service: which mean that the device has enough storage and computing capacity to achieve the needed task. This criterion is important for IoT devices which can be constrained and have limited resource.

- Be available when needed. The availability of an IoT device is threatened by two factors. Firstly, IoT devices are generally mobile and ubiquitous devices and are therefore not powered via mains but rather by a battery. Ensuring that the device has the power level required to perform a task is an important requirement. Secondly, to perform the requested task, an IoT device must, necessarily, be connected to the Internet.

- Respect the minimum security requirements. Although this condition has no direct effect on the execution of the requested task, it is critical because a single device with security vulnerabilities can threaten the entire network and allow malicious entities to intrude.

On the basis of the above-mentioned assumptions and in order to be able to assess the Ability of device-nodes, we have been guided by a standard proposed by the Internet Engineering Task Force (IETF) (Bormann et al. (2014)) that was set to classify IoT devices. The criteria for comparison are as follows: (i) The constrained device capability criterion; (ii) The energy limitation criterion; and (iii) The security requirements criterion.

3.5.1.1 The constrained device capability criterion

The constrained device capability criterion (CDC) permits to classify devices according to their storage, processing, and communication capabilities into three classes.

CDC.0: The Very Constrained Devices Class: gathers IoT devices which are very constrained sensor-like motes. They are so severely constrained in memory and processing capabilities that most likely they will not

have the resources required to communicate directly with the Internet without the help of larger devices acting as proxies, gateways, or servers.
CDC.1: The Quite Constrained Devices Class contains devices which are quite constrained in code space and processing capabilities. They are capable enough to use a protocol stack specifically designed for constrained nodes (such as the Constrained Application Protocol (CoAP) or the UDP) and participate in meaningful conversations without the help of a gateway node. However, when using this kind of devices, we need to be parsimonious with state memory, code space, and often power expenditure for protocol and application usage.
CDC.2: The Not Constrained Devices Class groups less constrained devices capable of supporting most of the same protocol stacks as used on notebooks or servers. Using this class of devices might reduce development costs and increase interoperability.

3.5.1.2 The energy limitation criterion

The energy limitation (EL) criterion allows classifying devices according to (i) their power terminology and (ii) their strategies for using power for communication.

Considering (i) the power terminology we can distinguish: (a) **EL.0.0: Devices with energy limited by events** (i.e. event-based harvesting); (b) **EL.0.1: Devices with energy limited by period** (i.e. replaceable battery); (c) **EL.0.2: Devices with energy limited by lifetime** (i.e. primary battery); and (d) **ELC.0.3: Devices without energy limitation** (i.e. mains-powered).

Considering (ii) the strategies for using power for communication, we can categorize IoT devices into three classes:
EL.1.0: The Devices using normally-off communication strategy They need specific strategies to minimize energy consumption during the reattachment process.
EL.1.1: The Devices power-limited which must communicate frequently.

For that, low-power solutions must be used for the hardware and the link-layer mechanisms.
EL.1.2: The Devices using always-on communication strategy when there is no need for power-saving measures.

3.5.1.3 The minimal security requirements criterion

IoT devices are exposed to a variety of security threats and vulnerabilities. Typical security threats and vulnerabilities of IoT devices include unauthorized access, loss or theft, physical destruction, information leakage, illegal data modification, and denial of service attacks.

Many research proposes to classify IoT devices according to their ability to face different security threats and vulnerability based on four elements of information security which consist of confidentiality, integrity, availability and Authentication/Authorization. IETF propose to classify IoT devices according to security requirements but don't give a specific classification. International Telecommunication Union (ITU) classified into four different types of IoT devices according to type and functionality as follows: data-carrying device, data-capturing device, sensing and actuating device, and general device.

In this work, as the security of IoT devices is a vast and wide field that is beyond the scope of this work, we are just interested in minimal security requirements (MSR). We propose to reuse the classification proposed by Seungyong et al. (2017) who propose four categories of minimal security requirements, namely (i) Confidentiality, (ii) Integrity, (iii) Availability and (iv) Authentication/ Authorization.

The (i) Confidentiality consists of preventing sensitive information from reaching the wrong people. In other words, access must be restricted to those authorized to view the data in question. To estimate if an IoT device is able to preserve confidentiality, we can check the existence of the following functionality Seungyong et al. (2017):

- Transmitted message encryption: Messages exchanged between IoT devices should be transmitted in an encrypted format.
- Data encryption: IoT devices should encrypt sensitive data such as private information and cryptographic keys.
- Tamper resistance: IoT devices should provide a function ensuring the safety from physical attacks.
- Device ID management: IoT device should have unique device identification information.

The (ii) Integrity involves maintaining the consistency, accuracy, and trustworthiness of data over its entire life cycle. Data must not be changed in transit, and steps must be taken to ensure that data cannot be altered by unauthorized people. To assess if an IoT device is capable to ensure integrity, we can check the existence of the following functionality Seungyong et al. (2017):

- Data integrity: IoT device should provide data integrity verification function to prevent forgery of data.
- Platform integrity: IoT devices should provide a platform integrity verification function of system level.
- Secure booting: IoT devices should provide secure booting function to ensure the reliability of the device.

The (iii) Availability guarantees that systems, applications and data are available to users when they need them. The most common attack that impacts availability is denial-of-service in which the attacker interrupts access to information, system, devices or other network resources. To estimate if an IoT device is able to guarantee availability, we should check the presence of the following functionality Seungyong et al. (2017):

- Logging: IoT device should provide the adequate log function for the user, the system and the security event.

- State Information Transmission: IoT device should provide a periodic keep-alive or device state messages to prevent from physical removal/destruction.
- Security patch: IoT device should provide a safe and secure software update and patch function.
- Security policy setting: IoT device should provide the ability to set an appropriate security policy on the different types of devices.
- Software safety: IoT devices should ensure software safety, with features such as appropriate module separation or removal, and access restrictions.

The (iv) Authentication/ Authorization: Authentication provides a way of identifying a user before access is granted. The process of authentication is based on each user having a unique set of criteria for gaining access. Following authentication, a user must gain authorization for doing certain tasks. To estimate if an IoT device is able to preserve minimal Authentication/ Authorization requirement, we can check the existence of the following functionality Seungyong et al. (2017):

- User authentication: IoT device should provide a user authentication function to prevent unauthorized access.
- Device authentication: IoT device should provide a device authentication function to block the access of illegitimate devices.
- Password management: IoT device should set a robust password and ensure periodic update.
- Mutual authentication: IoT device should provide mutual authentication between the devices to establish secure, autonomous communication environment.
- Identification information verification: IoT device should provide a function to verify unique device identification and prevent device repli-

cation or appropriation.

Based on the previous definitions for each security requirements, we can categorize IoT devices in four classes.

MSQ.0: Devices unable to provide minimal security requirements This category of devices are unable to provide integrity, availability, and authentication/ authorization mechanisms. However, they are able to identify themselves.

MSQ.1: Devices able to ensure basics security requirements These devices are capable of ensuring confidentiality and authentication requirements.

MSQ.2: Devices supporting almost security requirements These devices are able to support the majority of security requirements.

MSQ.3: Devices supporting all security requirements These devices are capable to ensure confidentiality, availability, integrity and authentication/ authorization requirements.

3.5.2 The Devices-Trust score

Based on the above classification, we propose a new trust-indicator called Quality of Devices ($T_d(d_i)$) allowing to assign a score to each device implied in SIoT networks. The table 3.4 summarizes the classes of each classification criterion and present the scores corresponding to each class.

The final $T_dD(d_i)$ value is computed according to the following equation (Eq.3.7):

$$T_d(d_i) = \frac{1}{4} \sum_{cc_i \in C} score_{cc_i}(c(d_i) \tag{3.7}$$

(Where ($score_{cc_i}(c(d_i))$ represents the score of the class c to which the device belongs for each comparison criterion cc_i in C the set of criteria.)

Note that the characteristics of a device can be deduced from the type and category of the device (sensor, actuator, smartphone, tablet, ...) or explicitly entered by its owner. A trustful device should accept sharing his characteristic. If it is not the case, this characteristic can be deduced with the parental

Table 3.4: Scores for each IoT device class

Classification Criterion		Class	Class Denomi -nation	Score
Constrained Device Capability		Very constrained capability	CDC.0	0.0
		Quite constrained capability	CDC.1	0.5
		Not constrained capability	CDC.2	1.0
Energy Limitation	Power Terminology	Energy limited by events	EL.0.0	0.0
		Energy limited by period	EL.0.1	0.25
		Energy limited by lifetime	EL.0.2	0.5
		Unlimited energy	EL.0.3	1
	Strategy for Using Power	Normally-off strategy	EL.1.0	0.0
		Low-power solutions	EL.1.1	0.5
		Always-on strategy	EL.1.2	1.0
Minimal Security Requirements		Insufficient security requirements	MSQ.0	0.0
		Basic security requirements	MSQ.1	0.25
		Almost security requirements	MSQ.2	0.5
		All security requirements	MSQ.3	1.0

relationship. If it is not possible, clustering techniques allow to affect the device to a category and deduce his quality-score.

3.6 Service Trust-Dimension

Like device nodes, service nodes have no intention. We trust a "Service-type node" for its "Ability" to meet our request, in other words for its functional characteristics, but also for other non-functional characteristics such as its availability when we need it or its response-time. Functional characteristics $fc(s_k)$ are quantified as the average rate given to a service s_k. To quantify non-functional characteristics, the method proposed by Al-Masri and Mahmoud (2007) consisting of ranking services based on the characteristics cited below is reused.

- Response Time: refers to the time taken by service to respond to input and is measured in millisecond (ms).
- Availability: aims to estimate the degree to which the service is in an operable state and is computed by dividing the number of successful invocations by total ones.
- Throughput: refers to the maximum rate of production of service and

is computed as the number of invocations of service in a given interval of time.

- Succesfulness: represents the number of successfully accomplished queries.
- Reliability: refers to the number of failures committed by the service and is computed as the number of erroneous messages divided by the total number of messages.
- Latency: refers to the delay between request and response and is measured in millisecond (ms).

A score h_i is attributed to each service based on equation 3.8 and a rank $WsRf(s_k)$ is given based on equation 3.9.

$$h_j(s_k) = w_j \frac{q_{s_k,j}}{Max(q_{s_m,j})} \tag{3.8}$$

$$WsRf(s_k) = \sum h_j(s_k) \tag{3.9}$$

We propose the service trust score feature denoted as $T_s(s_k)$, which combines functional and non-functional characteristics as follows (Eq.3.10).

$$T_s(s_k) = \frac{\alpha * fc(s_k) + \beta * nfc(s_k)}{\alpha + \beta} \tag{3.10}$$

$$fc(s_k) = \frac{1}{N} \sum_{j=0}^{N} rt(u_j, s_k) \tag{3.11}$$

$$nfc(s_k) = \frac{1}{WsRf(s_k)} \tag{3.12}$$

3.7 Conclusion

In this chapter, we have set up the features proposed for the Trust-Composition phase. These features are aligned on three dimensions and

allow to estimate, on one hand, the "Intention" of user nodes, and on the other hand, to quantify the "Ability" of the device and service nodes. The features relative to the user trust-dimension are deducted from the proposed Attack-Model and aim at detecting if a user has launched a Trust-Attack. The features related to the device trust-dimension allow to check if a device is able to perform the task assigned to it, if it is available and if it does not present a security flaw that could compromise the security of the whole network. The features proposed for the service trust-dimension allow to check the non-functional characteristics of a service such as its response time or its availability.

In the following chapter we describe the methods proposed to aggregate the values of the proposed features into a trust-score that allows to take the decision to interact or not with a node, to propagate these trust-scores in the network and to manage them, while respecting the specific constraints of SIoT environments.

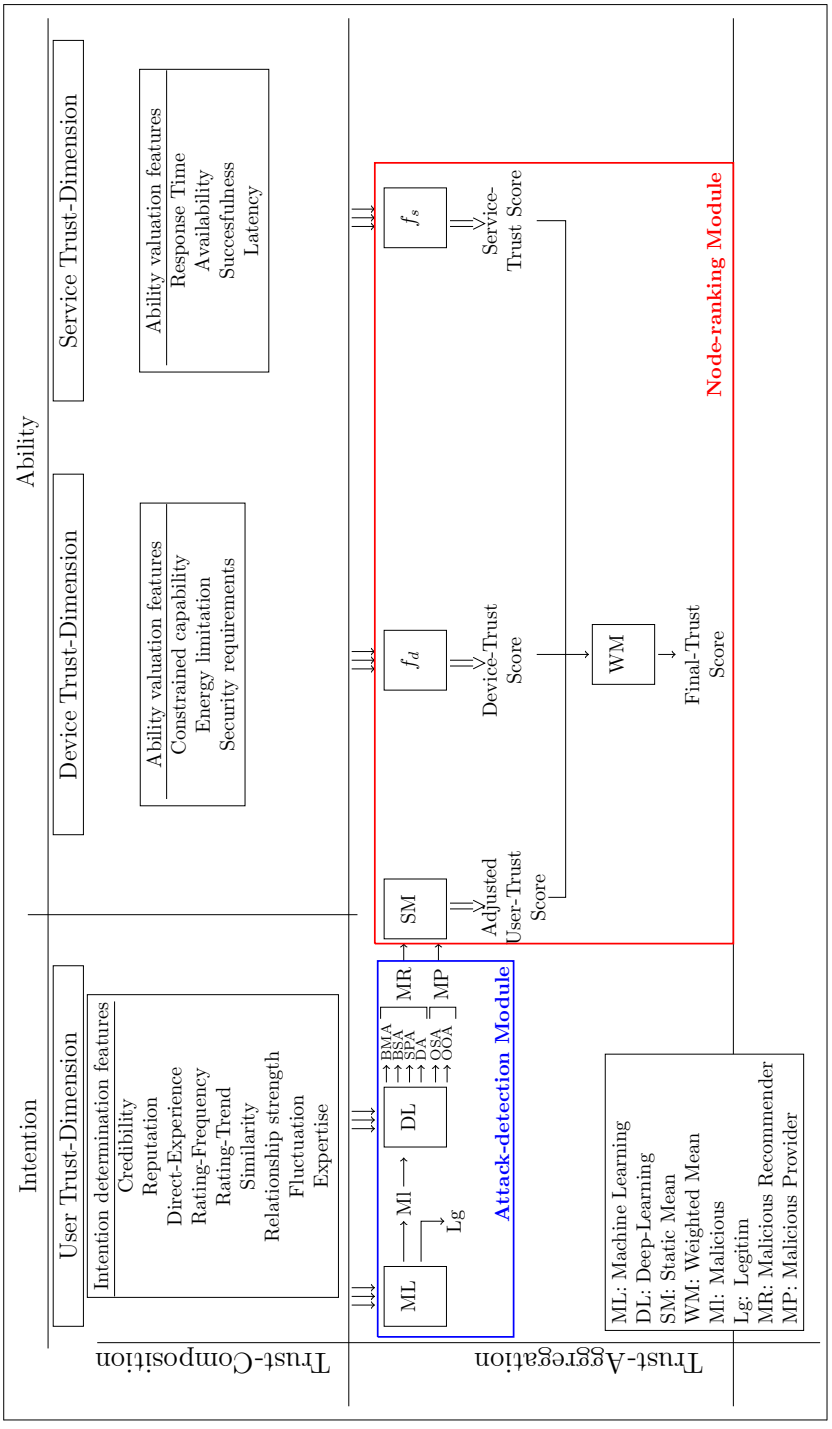


Figure 3.10: The DSL-STM Trust-Assessment Model architecture.

Chapter 4

Trust-aggregation, propagation and update for resilient, dynamic, scalable and resource-efficient TMM

Contents

4.1 Introduction

In the previous chapter, we presented our contribution to the Trust-Composition step and detailed the different Trust-Features proposed for each Trust-Dimension. In this chapter, we focus on the Trust-Aggregation, Propagation and Updating.

We start by emphasizing on the proposed aggregation method, which permits, on the one hand, to isolate malicious nodes and counter their attacks and, on the other hand, to rank legitimate nodes according to a trust score. To achieve this result, the proposed method bases on artificial intelligence techniques, including Supervised Machine-Learning and Deep-Learning. Then we outline our proposition for the propagation and update phases, to guarantee the scalability, dynamism and resources-efficiency of the proposed TMM.

4.2 Trust-Aggregation

Trust-Aggregation step consists of selecting a method to combine Trust-Features values into a single final value allowing to make the decision to trust or not. We presented in section 3.3 the architecture proposed for the Trust-Assessment Model. As can be seen in figure 3.10, the propagation phase of our TAM is composed of two main modules: (i) The Trust-Attack detection module and (ii) The Node-Ranking module. (i) The Trust-Attack detection module proceeds in three steps. (a) The first step consists in detecting malicious users. (b) The second step consists in distinguishing the type of launched attack. (c) The last step consists in countering these attacks. (ii) The Node-Ranking module comes afterwards to estimate the degree of trust that can be granted to a given service. This trust-degree is used to rank services and assists services requester in selecting the most appropriate service. Note that this degree is based on the three mentioned trust-dimensions, as it includes the quality of the Service-Provider (user type node), the quality of the device used to operate the service and the quality of the service provided.

This section summarizes the different steps of the Trust-Aggregation method including the steps performed by **(i) The Trust-Attack detection module** and the one performed by **(ii) The Node-Ranking module**, as follows: ((i).a) Detection of malicious users; ((i).b) Detection of Trust-Attack types; ((i).c) Trust-Attacks countering; ((ii).d) Trust-scores computing.

4.2.1 Machine-learning for Malicious-users detection

We used supervised machine learning techniques to identify malicious users from legitimate ones. We report in this subsection, the motivation of the choice of this method as well as its implementation and its setting.

4.2.1.1 Motivations

Since the problem of the detection of malicious nodes is considered as a complex problem requiring an in-depth analysis of nodes behaviour, we propose to use Machine-Learning technique. From this perspective, we consider our system as a classification problem. Indeed, our objective is to detect if a user is a malicious or a legitimate one. A user is considered as malicious if he/she performed BMA, BSA, SPA, DA, OOA and/or OSA attack. If the user did not perform any of the cited attacks, he is considered as legitimate. Thus, for each couple of users (u_i, u_j), we recover all of the past interactions. We compute on the basis of these interactions the value of the different features related to u_i, u_j and (u_i, u_j) (see Table 4.1). The input to the algorithm is the set of these values. The analysis of these values will allow detecting if an attack has taken place. According to this, the user u_i will be judged as malicious/benign (see Figure 4.1).

4.2.1.2 Implementation and setting of the Deep Learning Model

In order to select the most suitable Machine Learning algorithm for our problem, we have tested Naive Bayes, Multi-Layer Perceptron, and Random Tree algorithms on a sample of 1000 observations. A 10-fold cross-validation algorithm was used to evaluate the average classification performances.

Table 4.1: The input of the Machine-Learning algorithm.

Users	Features	Denomination
u_i	$Rep(u_i)$	Reputation
	$Cred(u_i)$	Credibility
	$Fluct(u_i)$	Fluctuation
	$rTren(u_i)$	Rating-Trend
u_j	$Rep(u_j)$	Reputation
	$Cred(u_j)$	Credibility
	$Fluct(u_j)$	Fluctuation
	$rTren(u_j)$	Rating-Trend
(u_i, u_j)	$dExp(u_i, u_j)$	Direct-Experience
	$rFreq(u_i, u_j)$	Rating-Frequency
	$Sim(u_i, u_j)$	Similarity
	$RelStren(u_i, u_j)$	Relationship-Strength

The average recall, precision, and F-measure obtained for each Machine-Learning algorithm are reported in Figure 4.2. Overall, we observed good classification performances for every learning algorithms, with F-measures ranging from 0.9 to 0.95. Multi-Layer Perceptron Network has shown the best-average results, with a recall of 0.935, precision of 0.955, and an F-measure of 0.95. We, therefore, apply Multi-Layer Perceptron for this step.

4.2.2 Deep-learning for type-attack detection

The purpose of this step is to distinguish the type of launched Trust-Attacks and will, therefore, only apply to users judged as malicious by the previous step. The identification of the type of attack will allow us to choose the effective method to counter it in the next step. To achieve this goal we have opted for supervised Deep-Learning. We report in this sub-section a restrained state of the art on Deep-Learning, as it is a relatively recent method. We also present the motivations that led us to select this method, as well as its implementation and setting.

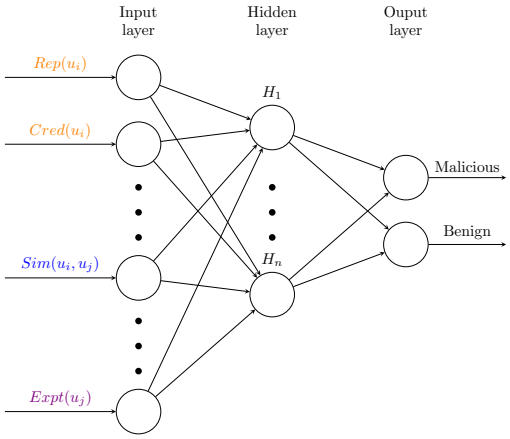


Figure 4.1: Trust-Aggregation based on Machine Learning.

4.2.2.1 Background and Motivations

Deep Learning is the new generation of Artificial Intelligence that takes Machine Learning to the next level through the use of artificial neural networks. The performance of Deep-Learning is revealed not only in the task of classifying objects but also in the recognition of images, videos, texts and other types of content.

According to their topology deep neural networks can be classified into (i) Convolutional Neural Network (CNN) which is a special type of 2 Dimensional (2D) convolutional layer network, well suited for image and video recognition; (ii) Recurrent Neural Network which is a neural network whose connection graph contains at least one cycle and is well suited for tasks that involve sequential inputs, such as speech and text; and (iii) The Multi-Layer Perceptron (MLP) that can be applied to digital and numerical data and is, therefore, best suited to the context of our work.

The Multi layer-Perceptron or deep neural network is an acyclic neural network (Feed-Forward Neural Network) structured in several successive connected layers. The higher the number of layers, the "deeper" the network is. These layers are intermediate (hidden) layers between the input layers and the output layers all having the same function. Indeed, in a perceptron, the

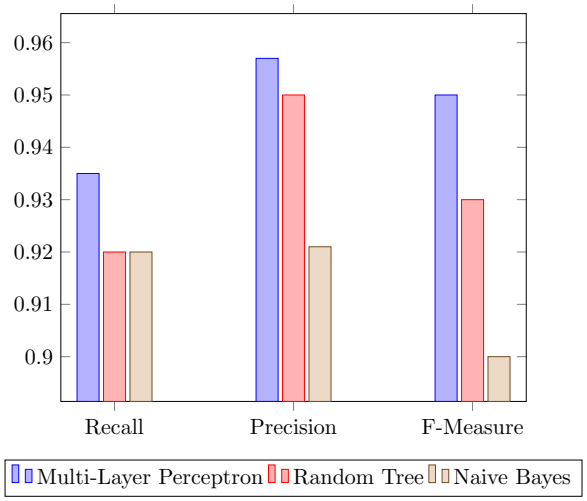


Figure 4.2: Classification results obtained with machine-learning algorithms trained with the proposed features.

outputs of one layer are the inputs of the next. The inputs of the MLP are associated with synaptic weights. Thus, for each hidden layer and the output layer, there is a bias term (threshold) whose value is equal to 1.

For each layer, the perceptron calculates a weighted average of its inputs taking into account the bias, called the total input. This total input will be transmitted to an activation (or transfer or thresholding) function f, in order to produce the output. This function can extremely affect the performance of the network.

Activation functions include :

- The identity function: the input signals are not changed.
- The Softmax function: generally used for classification tasks. It allows the construction of neural networks with several probabilistic outputs.
- The hyperbolic tangent function: the most used.
- Etc.

To determine the best synaptic weights applied to each of the inter-neuronal

connections, the perceptron uses an algorithm called Error Gradient Back-propagation. This algorithm is repeated thousands of times for each neuron in a neural network from the last layer to the first one. The purpose is to adjust the weights according to the average error committed by the network over the training set until the error can no longer be reduced. The error is calculated by taking the difference between the desired output of the model and the output calculated by the neural network.

The gradient back-propagation algorithm is summarized in the following four steps:

1. The input signal propagates forward in the network layers using the activation function.
2. When forward propagation is complete, the output is delivered.
3. The algorithm calculates the error between the desired output and the output given by the network.
4. The algorithm propagates the error backwards by adjusting the weights in all layers to decrease the error.

Recent advances in Deep-Learning, have improved the state of the art in several areas such as image recognition, speech recognition, natural language processing, and so on. Therefore, in this part, we will review some examples of related work that have exploited Deep-Learning in social networks and recommendation systems. Our choice is justified by the fact that these two areas are closest to our field of study, especially since no work has used Deep-Learning for Trust-Management in SIoT environments.

Yuan et al. (2014) propose in their work a method for detecting malicious applications. The proposed method based on two fundamental steps. In the first one, features are extracted from Android applications. In the second one, an MLP Deep-Learning model, proceeding in two steps, is applied. The first step is the pre-processing one, during which unlabeled data are trained. The second step is the back-propagation step which allows refining the pre-

trained neural network by classes.

Deng et al. (2016) addressed the problem of data-sparsity in Social Recommendation Systems (SRS). AS SRS bases on users rating matrix to measure similarity, matching and affinity between users, the fact that these matrices are generally very scattered is a great challenge. [Deng et al., 2017] propose to use what they call a deep auto-encoder which is an unsupervised Deep-Learning method that allows initializing the rating matrix more efficiently and accurately.

Jaradat et al. (2016) propose an adaptive method for dynamic privacy customization for users of online social networks (OSNs). This is done in three steps. (i) The first step is to calculate the trust value assigned by one user to another over time. (ii) The second step is to introduce a brain/long-term memory component that allows calculating the benefit of sharing information with each user. (iii) The final step is the generation of a component allowing the calculation of the final Trust. [Jaradat et al., 2016] use deep learning in the second stage. Indeed, the proposed brain/long-term memory component is based on LSTM neural networks.

To summarize, Deep-Learning has been successful in Trust & Privacy Management in Social Networks and Social Recommender Systems. We propose to use Deep-Learning and specifically the MLP neural networks in order to aggregate the Trust-Features proposed in the composition steps and detect the type of Trust-Attack launched by each Malicious user.

4.2.2.2 Implementation and setting of the Deep-Learning Model

We used the MLP algorithm implemented in the PyCharm4 tool based on Python, in order to build our classification function. To parameterize the MLP, we tried different values for each parameter. All the results obtained for each configuration are reported in the appendix . Table 4.2 reports the settings that give us the best performance. The figure 4.3 shows the resulting deep model.

Table 4.2: The MLP Deep-Learning algorithm setting

Designation		Value
Input Layer	Number of neurons	12
	Activation function	Hyperbolic tangent
Hidden Layers	Number of hidden layers	1
	Number of neurons	8
	Activation function	Rectified Linear Units (ReLu)
Output Layer	Number of neurons	6
	Activation function	Sigmoid
Optimizer		Adam
Error calculation function		Categorical crossEntropy
Epoch		25
Batch-size		1

4.2.3 Attack Countermeasures

Contrary to a lot of related work, in this work we focused on the detection of trust attacks. Our goal is to recognize malicious users, identify the types of attacks they have launched so that we can counter them and reverse their harmful effects on the system. We distinguish two methods to counter attacks, depending on the type of attack.

Malicious recommendation type attacks harm the system by propagating erroneous votes. To counteract this effect, once a user is deemed malicious by the system and once the attacks he has performed are identified as malicious recommender attacks, his votes are eliminated and no longer count towards calculating trust scores.

Malicious vendor attackers do not give false votes. However, they mislead the system by alternating between good and bad services and are therefore harder to spot. These attackers may not have a high reputation value but may have average values, allowing them to continue propagating their services. To counter this type of attack, the reputation of users deemed to be malicious providers is divided by 1.5.

These two methods are explained in more detail and through scenarios in the next two subsections.

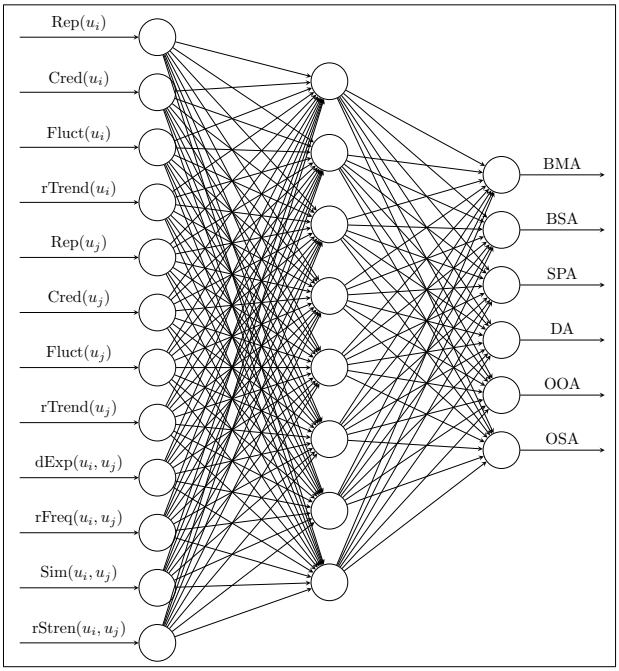


Figure 4.3: Deep-Learning Model for Type-Attack detection.

4.2.3.1 Countering Malicious Recommender Attacks

Figure 4.4.a shows an initial configuration of a SIoT network with a set of seven user nodes ($u_1, u_2, u_3, u_4, u_5, u_6$ and u_7) each offering a set of services. These users have already interacted with some other users and received feedback. We see on the figure, for each user, the list of his past interactions. Nevertheless, this configuration is said to be initial, because there is no attack yet. In other words, a good quality service received a high rate and bad quality service received a low rate. The reputation of each user is calculated classically, as the average of the received ratings and is displayed on the history table of each user.

Figure 4.4.b shows a second configuration of the same network after several trust-recommender attacks have been launched:

- The user u_1 did not invoke any service and did not assign any rating.
- User u_2 invoked a service $s_1 2$ provided by user u_1. This service is of

good quality and u_2 has acted legitimately by assigning it a positive rating. u_2 has also invoked the service $s_6 2$ provided by user u_6. This service is of poor quality and u_2 acted legitimately by assigning it a negative rating.

- The user u_3 invoked the service $s_2 2$ provided by the user u_2. This service is of good quality and u_3 has acted legitimately by assigning it a positive rating.
- The u_4 user who has a bad reputation launched a BMA attack against the user u_1 to decrease his reputation. He assigned him two negative ratings despite the good quality of the services he provides. As a consequence of this attack, the reputation of u_1 goes from 4.5/5 to 2.8/5. User u_4 also launched a BSA attack in favour of user u_5 to increase his reputation. He assigned him two positive ratings despite the poor quality of the services he offers. As a consequence of this attack, the reputation of u_5 went from 0.5/5 to 2.75/5.
- The user u_5 did not invoke any service and did not assign any rating.
- The user u_6 has a bad reputation (1/5) and has launched a SPA attack to self-promote his reputation. To do so, he duplicated himself (node u'_6) and gave himself two positive ratings despite the poor quality of his services. As a consequence of this attack, the reputation of u'_6 went from 0.5/5 to 3/5.
- The user u_7 has a bad reputation (1/5) and launched a DA type attack against users u_2, u_3 and u_6. As a result of this attack, the reputation of u_2 goes from (5/5) to (3.5/5), the reputation of u_3 goes from (4/5) to (2/5) and the reputation of u_6 goes from 3/5 to 2/5.

As a result of these different trust-recommender attacks, u_1, who is a legitimate user offering two good quality services $s_1 1$ and $s_1 2$, has almost the same reputation value as the user u_5 who offers two poor quality services $s_5 1$ and $s_5 2$. Similarly, user u_3, who is a legitimate user offering two good quality services $s_3 1$ and $s_3 2$, has almost the same reputation value as user u_6, who

offers two poor quality services $s_6 1$ and $s_6 2$. Reputation values, in this case, are erroneous and unrepresentative and may lead network users to solicit a bad service provider rather than a good one.

The application of the machine-learning on the proposed trust-features makes it possible to detect malicious users. The application of deep-learning on the same features makes it possible to identify the type of attack and to classify the malicious user as a malicious-recommender or malicious provider. The rating attributed by the malicious providers biases the reputation system. Our Trust Rating Model allows us to identify these erroneous votes (coming from malicious-recommender users) and to exclude them in order to counter these attacks and to ensure correct and representative reputation calculations. We can see on figure 4.4.b the red rows in the interaction histories that represent the non-legitimate interactions (attacks). The adjusted reputation values returned by our Trust Assessment Model are displayed in green. We can see that, using our Trust Assessment Model, good service providers have reputation values greater than or equal to (4/5) and bad service providers have reputation values less than or equal to (1/5).

4.2.3.2 Countering Malicious Provider Attacks

Figure 4.5 shows the example of two malicious nodes u_1 and u_2, each offering 4 services, respectively u_{11}, u_{12}, u_{13} and u_{14} for u_1, u_{21}, u_{22}, u_{23} and u_{24} for u_2. These two nodes interact with the rest of the nodes in the network. The table 4.3 reports all the interactions. Unlike other trust attacks, for malicious provider attacks, the order of the interactions is important. These interactions are therefore presented in their chronological (numbered) order. We see in the table, that the node u_1 launches an OOA attack, and alternates between good services (interactions t_1, t_3 and t_5) and bad services (interactions t_2 and t_3) each time. The node u_2 launches an OSA attack. It started providing good quality services (t_6, t_7 and t_8 interactions), then it went on to provide poor quality services (t_8 and t_9 interactions).

The last two columns of the table show respectively the trust scores obtained

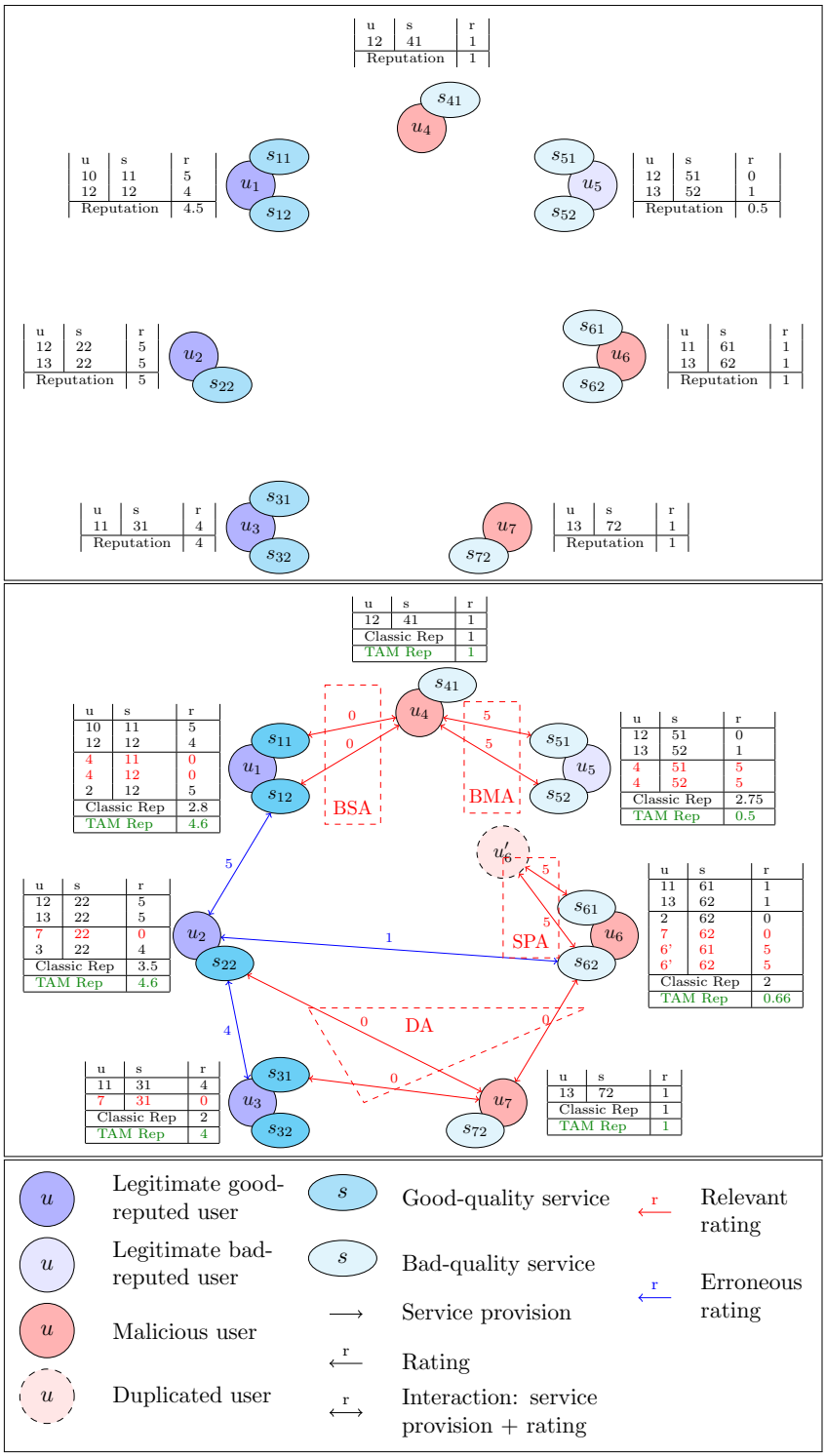


Figure 4.4: Malicious Recommender Attacks Coutering.

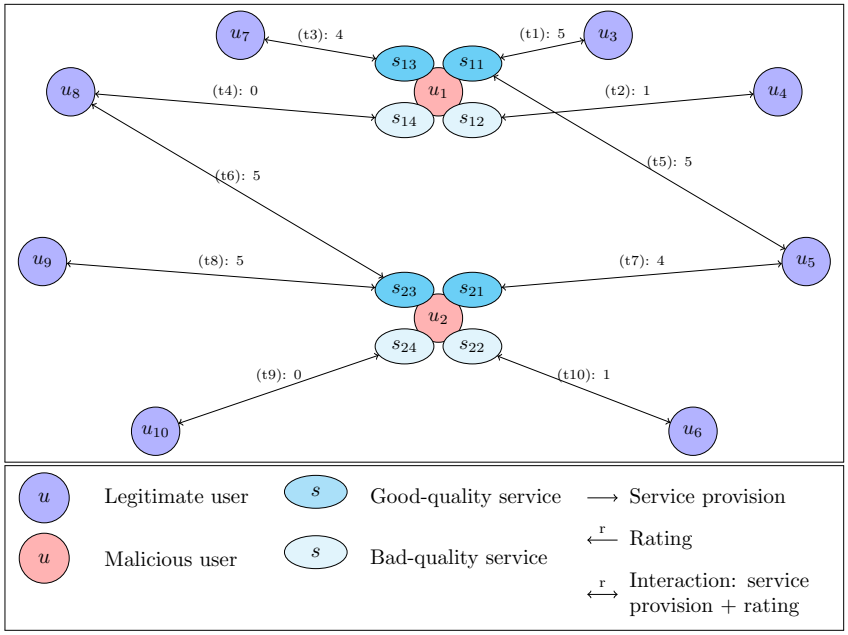


Figure 4.5: Malicious Provider Attacks Countering.

by a classical reputation system (average of votes) and the scores obtained by DLS-STM. The values below the threshold are in red. The table shows how the values of the two malicious users quickly fall below the threshold in the case of DLS-STM, while they remain above the threshold in the case of the classical reputation system.

4.2.4 Node-ranking

The objective of the node ranking module is to quantify the degree of trust that can be placed in a given node into a numerical value or score. This score will allow to classify the services and will thus assist service requester in selecting the most appropriate service for their needs. In this work, we propose a multidimensional score that allows to calculate the trust score according to the three mentioned trust dimensions and on the basis of the features proposed in the composition step for each of the dimensions.

For the user dimension, the trust score t_u is calculated according to the following equation, as the average of the reliable ratings received. The reliable

Table 4.3: List of Malicious Provider interactions.

time	prov	req	ser	rating	C-Rep	DLS-STM-Rep
t1	u_1	u_3	s_{11}	5	5	5
t2		u_4	s_{12}	1	3	2
t3		u_7	s_{13}	4	3.33	2.22
t4		u_8	s_{14}	0	3	2
t5		u_5	s_{11}	5	3	2
t6	u_2	u8	s_{23}	5	5	5
t7		u_5	s_{21}	4	4.5	3
t8		u_9	s_{11}	5	4.66	3.1
t9		u_{10}	s_{11}	0	3.5	2.33
t10		u_6	s_{11}	1	3	2

ratings are those assigned by legitimate users. The trust attack detection module fulfils this role by detecting and countering attacks that can bias this score.

For the device dimension, the trust score T_d is calculated according to the indicator presented in section 3.5 in the previous chapter. This factor is based on three criteria including (i) Constrained Device Capability, (ii) Energy Limitation and (iii) Minimal Security Requirement. This score checks whether a device has enough storage and computing capacity to achieve the needed task, whether it is available when needed and whether it doesn't compromise the network security by respecting the minimum security requirement.

For the service dimension, the service trust score T_s is calculated according to the indicator detailed in the section 3.6 of the previous chapter. This feature classifies services according to their non-functional properties, such as response time, latency, throughput and availability. The final trust-score is obtained by the mean of the three trust-scores.

4.3 Trust-Propagation and Update

The main objective of a Trust Management Mechanism is the provision of a representative and unbiased trust score. This objective is fulfilled by the trust

assessment model, in particular by the composition and aggregation steps presented previously. Propagation and update steps ensure non-functional properties, but which remain just as important in the case of a constrained environment such as SIoT. Indeed, the characteristics of SIoT networks, which concern a large number of nodes involved, the great dynamism and the limited computing and storage capacity of device type nodes, represent strong constraints. A reliable and robust trust management mechanism risks not being able to be effectively established in the real world if these various constraints are neglected.

4.3.1 Motivation

Trust propagation refers to how to propagate trust evidence to peers. In general, there are two trust propagation schemes, that is, centralized and decentralized.

Decentralized trust propagation refers to IoT devices autonomously propagating trust observations to other IoT devices they encounter or interact with without the use of a centralized entity. While this method is certainly robust and easy to implement, it is not appropriate in networks with up to billions of devices and an equally large number of exchanges and interactions per second, as it does not allows for **scalability**.

Centralized trust propagation requires the presence of a centralized entity, either a physical cloud or a virtual trust service implemented by participating in IoT devices. This approach allows for scalability, but is very resources-consuming, especially if the proposed trust model is complex. It is therefore as inapplicable as the centralized method, given that most of the IoT devices are characterized by limited computing and storage capacities.

The trust updating step consists of choosing a method for updating trust values, either the time-driven method or the event-driven method. The problem with the time-driven updating is that it is hard to fix the period, as, it may be possible that one hundred attacks are made in 1 minute. In the event-driven method, updates are done each time an event occurs. Its merit is

that it allows having a real-time trust management system. The drawback of this method is that it will generate a huge number of updates and is very resources consuming for the constrained IoT devices.

To overcome the drawbacks of those methods, we propose, at first, a hybrid propagation method, allowing to overcome the disadvantages of both centralized and decentralized ones. Then, we opt for the event-driven updating method. Indeed, combining it with the hybrid propagation method will allow overcoming its drawbacks.

4.3.2 The hybrid propagation

We propose a new hybrid propagation method that combines centralized and decentralized propagation methods, to ensure a good compromise between the two methods and to overcome their disadvantages. The hybrid propagation method consists of inserting into the network, a number of nodes called Trustful Nodes (TN). Their number depends on the number of nodes in the network and varies in a proportional way to the latter. These nodes are called Trustful Nodes because they do not provide or invoke services. Hence, they are not motivated to make trust-related attacks. Their main role is to compute and store the values and trust parameters of the other nodes. Each TN is responsible for a set of nodes. A hash function fh is used to assign each node to a TN. This function will take as an input, the identifier of the node and return the identifier of TN, which will manage it.

Take the simplified example of the following hash function:
$fh(x) = x \ mod \ 4$.
Assume that the SIoT network is composed of five nodes identified respectively as follows: $id_{Ni} \ in\{1, 2, 3, 4, 5\}$. And suppose that four TNs are inserted in the network and are identified as follows: idTN1 = 0, idTN2 = 1, $id_{TN3} = 2$ and $id_{TN4=3}$. By applying the hash function fh on the identifier of each node in the network, we will obtain a value belonging to {0,1,2,3}. The nodes {0,4} will be assigned to TN1, the nodes {1,5} will be assigned to TN2 and so on (Figure 4.6). A Distributed Hash Table (DHT) will be

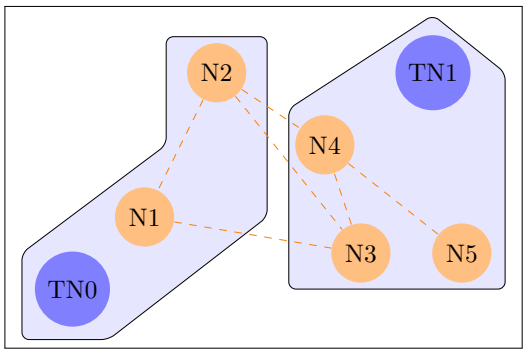


Figure 4.6: Network structure example.

responsible for mapping each node and the TN that is responsible for it.

Figure 4.7 presents a sequence diagram that represents the exchanges between nodes and trustful nodes. We have on the figure three nodes $N1$, $N2$ and $N3$ and two trustful nodes $TN1$ and $TN2$.

The figure shows two use cases. The first case concerns an interaction of a node u_1 managed by a TN TN_1, with another node u_2 managed by the same TN TN_1. The second case concerns an interaction of a node u_1 managed by a TN TN_1 with another node u_3 managed by another TN TN_2.

The advantages of this hybrid method are multiple. Unlike the centralized propagation method, the hybrid method makes it possible to deal with a huge number of nodes without fearing saturation and, thus, makes it possible to ensure **scalability**. Unlike the decentralized approach, the proposed hybrid method respects the constrained nature of IoT objects as TN will make the necessary calculations and storage in their place. This allows ensuring **Resources-efficiency**.

4.3.3 The event-driven update

The use of hybrid propagation will make the choice of the event-driven update method feasible. Indeed, unlike the time-based update approach, in which it is difficult or impossible to set the granularity of the period, the event-driven approach is more suitable for a system as unpredictable as the SIoT. However,

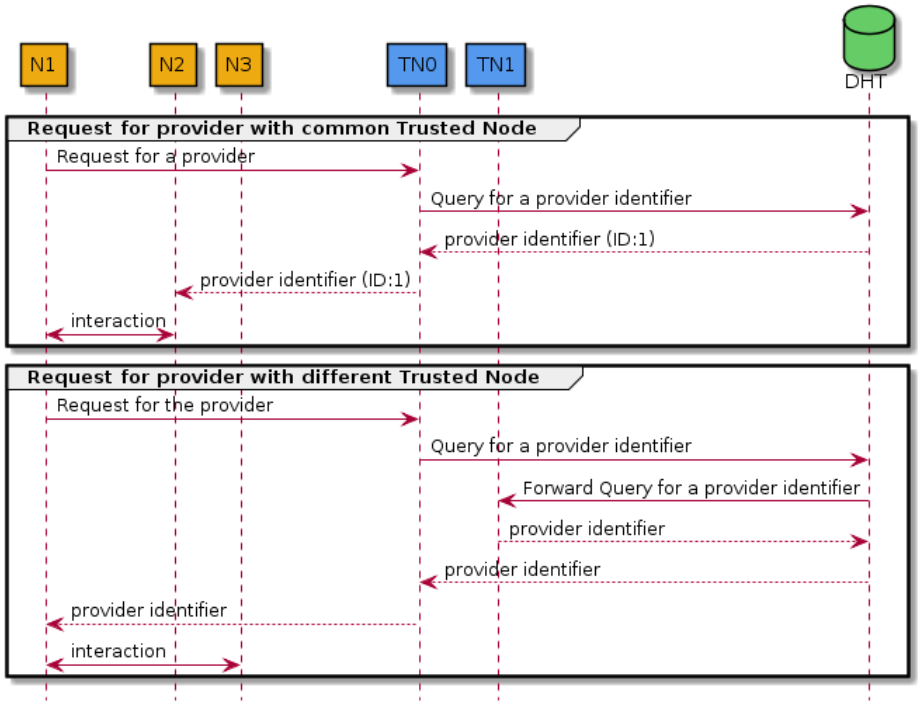


Figure 4.7: Sequence diagram of Hybrid propagation based on Distributed Hash Table.

given the size and dynamism of a SIoT network, the event-driven update approach is likely to be very expensive for both centralized and decentralized propagation. Therefore, a hybrid storage model is proposed in DSL-STM, allowing to apply the event-driven update method.

4.4 Conclusion

In this chapter, we focus on the Trust-Aggregation, Propagation and Updating. Our proposal for the Trust-Aggregation allows ensuring resilience against different Trust-Attacks. Our proposal for the Trust-Propagation and Updating allows ensuring scalability, dynamism and resources-efficiency.

The next chapter of this manuscript is the experimental part. It has allowed us to test and validate the different contributions of this work, including the relevance of the proposed Trust-Features, the Resilience of the resulting Trust-Assessment Model and finally the scalability, dynamism and resources-efficiency of the Trust Management Mechanism that is built based on the proposed Trust-Assessment model

Chapter 5

Experimentation

Contents

5.1 Introduction

Throughout previous chapters, we have presented DLS-STM, a trust management mechanism specifically designed for SIoT environments. The main contributions of this work concern the following aspects:

- **Resilience to trust attacks**: this property is ensured by the "Attack Detection" module in the Trust Aggregation phase.
- **Multidimensionality of the trust**: this property consists of the feature proposition for each SIoT trust dimension during the composition phase.
- **Scalability**: This property is ensured by the hybrid propagation method proposed in the aggregation phase.
- **Dynamism**: This property is ensured by the event-driven updating method during the updating phase.
- **Resource-efficiency**: This property is ensured by the hybrid propagation method proposed in the aggregation phase.

We propose, in this chapter, a series of experiments to validate the above-mentioned properties one by one.

5.2 The Trust-Assessment Model: Attack-detection and Node-Ranking

The Trust-Assessment Model is composed of the (i) Composition step: which consists in selecting the Trust-Features and the (ii) Aggregation step: which consists in aggregating the features proposed in the composition step into a final value allowing to take the decision to invoke or not to invoke a service.

In chapter 3, we proposed features for each trust dimension, notably features (honesty, ...) for the user trust-dimension, features (quality of device) for the device trust-dimension and features (quality of service) for the service trust-dimension.

In the first section of chapter 4, we proposed an (i) Attack-Detection module, that allowed to aggregate the features of the user trust-dimension using Deep-Learning and Machine-Learning and a (ii) Node-Ranking module, that allowed to aggregate the features of the three trust-dimensions using a

weighted Mean.

The objective of this sub-section is to validate the proposed Trust-Assessment Model, in other words, to validate the proposals made at the level of the composition and aggregation steps. To do this, we start by assessing the relevance of the proposed Trust-Features using the "information gain" metric Kent (1983). We, then, evaluate the Attack-Detection module by assessing each of its components to validate the **"Resilience"** property. Finally, we evaluated the Node Ranking module to validate the interest of considering the **"Multidimensional aspect of trust"**.

5.2.1 Simulations setting and nominal scenarios

Owing to the unavailability of real data, the majority of related works provided experiments based on simulations. In our work, we have also implemented simulations based on different scenarios. The scenarios differ according to the perimeter, the network dynamism, the frequency of interactions, and the number of users, devices and services. We can thus distinguish, in the table 5.1, the scenario "scena-1" which covers an area of 1 kilometre and represents a weakly dynamic network and a moderate interactions frequency. This scenario does not include many users/devices and services. and can represent the case of a Smart-House, a Smart-Restaurant or a Smart Gas-station,...

We can, also, distinguish in the table 5.1, the scenario "scena-2" which covers an area of 10 kilometres perimeter and represents a moderately dynamic network and a medium interaction frequency. This scenario can represent a Smart-Campus or a Smart-hospital for example. Finally, the scenario "scena-3" is a scenario involving a huge number of actors and which is highly dynamic and interactive. This scenario can represent a Smart-city or a Vehicular-Network.

For each of the mentioned scenarios, we launched simulations over 3 different durations, namely, one week, one month and six months. The objective here is to verify and monitor the impact of the temporary evolution on the

proposed TMM.

Finally, according to the purpose of the experimental series, we have launched on these same sub-mentioned scenarios:

- Random simulations: users select and invoke services randomly, without considering the reputation of the service provider and without providing rating.
- Simulations with only legitimate nodes: users select and invoke services based on the reputation of the service provider. The reputation is computed as the mean of received rating. All the users acts legitimately by affecting high rating value for good quality services and bad ratings value for bad quality services.
- Simulations with malicious nodes without the integration of a TMM: users select and invoke services based on the reputation of the service provider. The reputation is computed as the mean of received rating. The users act maliciously by launching different trust-attacks in each interaction.
- Simulations with malicious nodes and with the integration "DLS-STM" TMM: users select and invoke services based on the trust values given by DLS-STM. The users act maliciously by launching different trust-attacks in each interaction, but DLS-STM is introduced in order to detect and counter each attack.
- Simulations with malicious nodes and with the integration of some TMMs proposed in the literature: users select and invoke services based on the trust values given by TMMs proposed in the literature and act maliciously by launching different trust-attacks.

5.2.2 Features selection

To demonstrate the relevance of each feature proposed in the composition step, we used the Information Gain metric Kent (1983) to estimate the dis-

Nominal scenarios for experimental series

Denomination-1	Perimeter	Network Dynamicity	Interaction Frequency	Number of users	Average device/ user	Average service/ device	Duration	**Denomination-2**
Scena-1	1 km	25 %	25 %	200	2	4	1 week	Scena-1-D1
							1 month	Scena-1-D2
							6 months	Scena-1-D3
Scena-2	10 km	50 %	50 %	1000	3	10	1 week	Scena-2-D1
							1 month	Scena-2-D2
							6 months	Scena-2-D3
Scena-3	100 km	100 %	100 %	10000	5	30	1 week	Scena-3-D1
							1 month	Scena-3-D2
							6 months	Scena-3-D3

criminating character of each feature used as input to a machine learning algorithm.

Figure 5.1 depicts the information gain when using one single feature in the learning process. The *similarity* feature has the highest value of information gain. This can be explained by the fact that it is the only feature that enable to reveal Self Promoting Attacks (SPA). *Rating frequency*, *quality of provider*, *rating trend*, *honesty* and *reputation* features present almost equal information gain values. Indeed, they are equally discriminative for the detection of BMA, BSA, and DA attack types. The direct experience attribute has the lowest information gain value. This attribute does not actually detect attacks. However, it allows, as explained previously, to make the difference between BMA and BSA attack.

5.2.3 Experimenting the Attack-Detection Module

The Attack-detection module represents the first module in the Trust-Aggregation phase. Its role consists of detecting and countering the different trust-attacks. To do so, the Attack-Detection module proceeds in three steps: (i) Malicious users detection, (ii) Type-attack detection and (iii) attacks countermeasures.

In this section, we assess the performance of each step. We use the Recall, Precision and F-measure metrics to assess the pertinence of the first and the second step, as they are based on Machine Learning and Deep-Learning. For

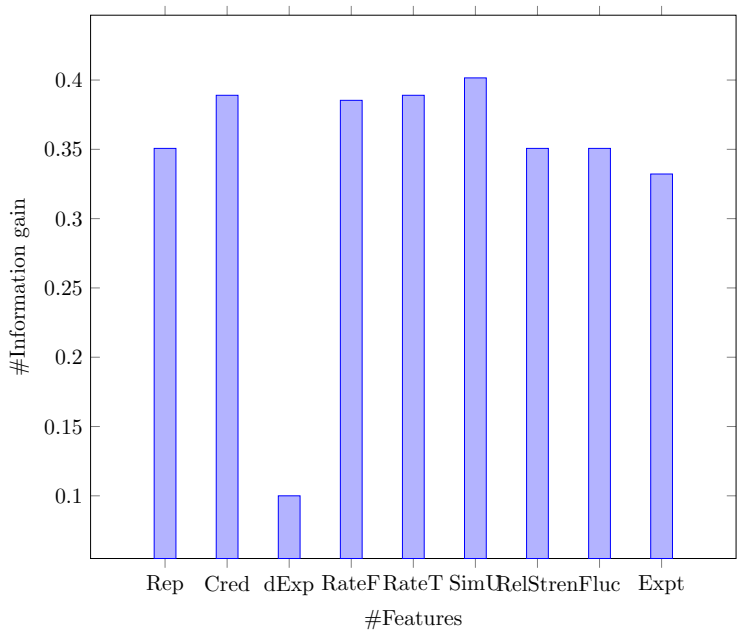


Figure 5.1: Information gain for single feature performance.

evaluating the attack countermeasure steps, we measure the rate of propagation of bad-quality services in the network.

5.2.3.1 Malicious users detection

The malicious user detection step is performed using the application of supervised machine learning (more precisely the multi-layer perceptron algorithm) on the features proposed for the user dimension during the composition step. This step thus allows classifying users as malicious/legitimate based on trust features. To validate the proposed learning model, we conducted different series of experiments for each of the scenarios scena-1, scena-2 and scena-3, mentioned above.

The first series represents a network with 50% of malicious nodes all performing only the BMA attack. The second series represents a network with 50% of malicious nodes, performing only the BSA attack. The third series represents a network with 50% malicious nodes, performing only the SPA attack.

Similarly for the fourth, fifth and sixth series, which present networks with 50% malicious nodes performing, respectively, DA, OOA and OSA attacks. The last series represents three networks with percentages of 10%, 25% and 30%, respectively, of malicious nodes performing all types of attacks. The objective of these experiments is to evaluate the relevance of the model to detect malicious users, regardless of the type of attack they have launched.

A 10-fold cross-validation algorithm was used to evaluate the average classification performances. Experiments were implemented using the WEKA API.

The obtained averaged recall, precision, and F-measure are reported in the table 5.2. We can notice that the Recall values are better in the case of BMA, BSA, OOA and OSA attacks compared to those of DA and SPA, for all three scenarios. We can also notice that the Recall, Precision (and consequently F-Measure) values are better for the case of the scena-3 scenario. Indeed, this scenario implies a higher number of interactions. And it is true that OOA and OSA attacks can only be discovered after a certain number of interactions. Finally, we notice that the model gives higher recall, accuracy and F-Measurement values when dealing with all mixed attacks. Knowing that this module is not supposed to distinguish between the different types of attacks, and considers a user as malicious as soon as he has performed any attack. We also notice that the recall, precision and f-measurement values remain, more or less, invariable in the face of a much higher number of nodes and interactions.

5.2.3.2 Attack-type detection

The step of detecting the type of attack will take as input all the users judged as malicious by the previous step, and will return the attack(s) launched as well as its/their type(s). To make this classification, this step is based on deep learning. To validate this step, we proceed in the same way as for the previous step and we apply the simulations on the three scenarios mentioned before with each time 50% of malicious users for each type of attack, then

Malicious Users Detection

Series			Evaluation Metric		
			R	P	F-M
Scena-1	BMA (50%)		0,85	0,80	0,82
	BSA (50%)		0,89	0,82	0,85
	SPA (50 %)		0,70	0,80	0,75
	DA (50%)		0,79	0,73	0,76
	OOA (50%)		0,80	0,80	0,80
	OSA (50%)		0,81	0,89	0,85
	ALL-Attack	10%	0,81	0,86	0,83
		25%	0,84	0,89	0,86
		50%	0,83	0,91	0,87
Scena-2	BMA (50%)		0,90	0,90	0,90
	BSA (50%)		0,91	0,91	0,91
	SPA (50 %)		0,89	0,89	0,89
	DA (50%)		0,70	0,70	0,70
	OOA (50%)		0,85	0,85	0,85
	OSA (50%)		0,79	0,79	0,79
	ALL-Attack	10%	0,98	0,98	0,98
		25%	0,92	0,92	0,92
		50%	0,90	0,90	0,90
Scena-2	BMA (50%)		0,95	0,93	0,94
	BSA (50%)		0,97	0,93	0,95
	SPA (50 %)		0,94	0,91	0,92
	DA (50%)		0,89	0,75	0,81
	OOA (50%)		0,93	0,82	0,87
	OSA (50%)		0,90	0,87	0,88
	ALL-Attack	10%	0,98	0,98	0,98
		20%	0,98	0,92	0,95
		50%	0,97	0,90	0,93

with 10%, 25% and 50% of malicious users for all types of attacks mixed. We also use Recall, Precision and F-Measure evaluation metrics to assess the relevance of a classifier.

The obtained averaged recall, precision, and F-measure are reported in table 5.3. We can notice that the greater the number of interactions, the higher the recall values increase. They reach up to 0.98. The precision values decrease slightly when the number of interactions is huge, but they remain above 0.8. The resulting F-Measurement values are above 0.85 in all cases and reach 0.95 when the number of interactions is large. We notice, also, that the values for the detection of the type of attack are better for the case of BMA, BSA, OOA and OSA attacks. It is average good for SPA and DA attacks but remains above 0.7.

5.2.3.3 Attack-countermeasure

After detecting the type of attacks, the Attack-countermeasure step consists of adopting, depending on the type of attack, the necessary measures to counter them and prevent them from biasing the reputation system. It is not possible to evaluate the relevance of this step with the classical recall and precision metrics. We, therefore, stipulate to study the effect of this step on the network, by depicting the network evolution. We propose to measure the Propagation Rate of Poor-Quality Services (prpqs). Indeed, this step is supposed to prevent this type of services from propagating and limit their invocation.

We apply the *prpqs* metric to each of the scenarios $scena-1$, $scena-2$ and $scena-3$, but over different duration. The purpose is to capture the evolution over time. The table 5.4 reports the results obtained for different network configurations. We can visualize in the second column the total number of interactions for each sub-scenario $scena-i-Dj$. These interactions are established over different durations. We can see that for the scenario $scena-1$, for example, $scena-1-D1$ which represents a capture of the scenario $scena-1$ over a duration of "one week" (see the table 5.1 for the duration)

Type Attack Detection.

Series			Evaluation Metric		
			R	P	F-M
Scena-1	BMA (50%)		0,9	0,8	0,85
	BSA (50%)		0,89	0,82	0,85
	SPA (50 %)		0,7	0,8	0,75
	DA (50%)		0,79	0,73	0,76
	OOA (50%)		0,8	0,8	0,80
	OSA (50%)		0,81	0,89	0,85
	ALL-Attack	10%	0,81	0,86	0,83
		25%	0,84	0,89	0,86
		50%	0,83	0,91	0,87
Scena-2	BMA (50%)		0,9	0,9	0,90
	BSA (50%)		0,91	0,91	0,91
	SPA (50 %)		0,89	0,89	0,89
	DA (50%)		0,7	0,7	0,70
	OOA (50%)		0,85	0,85	0,85
	OSA (50%)		0,79	0,79	0,79
	ALL-Attack	10%	0,98	0,98	0,98
		25%	0,92	0,92	0,92
		50%	0,9	0,9	0,90
Scena-2	BMA (50%)		0,98	0,89	0,93
	BSA (50%)		0,97	0,88	0,92
	SPA (50 %)		0,9	0,85	0,87
	DA (50%)		0,8	0,79	0,79
	OOA (50%)		0,85	0,85	0,85
	OSA (50%)		0,87	0,82	0,84
	ALL-Attack	10%	0,99	0,91	0,95
		25%	0,99	0,91	0,95
		50%	0,98	0,9	0,94

Table 5.4: Rate of propagation of bad-quality service.

Series	Nb Total Interactions	Random		Legitimate		All Attack		DSL-STM	
		NBQIS	%	NBQIS	%	NBQIS	%	NBQIS	%
Scena-1-D1	38 400	11096	28,90	1700	4,43	32200	83,85	3000	7,81
Scena-1-D2	144 000	102022	70,85	4989	3,46	138023	95,85	4050	2,81
Scena-1-D3	864 000	64005	7,41	5090	0,59	851233	98,52	8450	0,98
Scena-2-D1	1 440 000	1340065	93,06	100	0,01	1410353	97,94	45903	3,19
Scena-2-D2	5 400 000	2222330	41,15	1209	0,02	4897603	90,70	170034	3,15
Scena-2-D3	32 400 000	23578900	72,77	1000101	3,09	31899003	98,45	900380	2,78
Scena-3-D1	120 000 000	100000	0,08	1356030	1,13	103003000	85,84	10000767	8,33
Scena-3-D2	45 000 000 000	7000003	0,02	2340000000	5,20	41233300300	91,63	2500000203	5,56
Scena-3-D3	2,77E+15	7,E+09	0,00	7,E+12	0,26	2,E+15	83,63	7,E+13	2,35

includes 38 400 interactions. Whereas $scena - 1 - D2$ which represents a capture of the same scenario over a duration of "one month" includes 144,000 interactions.

Each of the sub-scenarios admits different configurations described and explained in the sub-section 5.2.1. In the "random", "legitimate" and "all attacks" configurations, no TMM is applied and no method to counter the attacks is used. In the "DLS-STM" configuration, our method to counter attacks is applied. We represent for each configuration the number of poor quality services invoked, as well as the percentage of propagation of these services compared to the total number of interactions. In the random configuration, the *prpqs* is very random and varies in a non-proportional way, which justifies the interest of implementing a reputation system in general. In the "legitimate" configuration, the *prpqs* is very low, while, on the contrary, it is always very high and varies between 83.85% and 98.45% in the "all attack" configuration. Indeed, the propagation of this type of services is mainly due to Trust-Attacks. Finally, we note that when applying our Attack-countermeasure method, the rate obtained is very low and is very close to the rates of the "legitimate" configuration whatever the size of the network and the duration. Moreover, we notice that the more the number of interactions increases, the lower the rate decreases. Thus this experiment allows us to validate **The Resilience** of the proposed model for different scenarios and different configurations.

Table 5.5: Parameters setting for comparison with related work

	Features	Weights	Threshold
Jayasinghe et al. (2016)	Recommendation Reputation	0,62 0,38	0,30
Truong et al. (2018)	Reputation Experience	0,84 0,16	0,22
Chen et al. (2016a)	Honesty Cooperativeness Community-Interests	0,74 0,12 0,14	0,22
Militano et al. (2016)	Reliability Reputation	0,37 0,63	0,35
DSL-STM	User Trust-score Device Trust-score Service Trust-score	0,33 0,33 0,33	0,50

5.2.4 Node-ranking

Since the majority of works in the literature do not address the problem of Trust-Attacks detection, we could not compare ourselves to them to validate the previous module. Indeed, these works are mainly interested in the ranking of nodes based on the weighted mean of the different features they propose. In our work, we also propose a node ranking module allowing to rank nodes on the basis of the features "user trust-score", "device trust-score" and "service trust-score".

In contrast to the majority of related works, our features are multidimensional as they allow to study the three trust-dimensions, generated by the three types of SIoT nodes. To validate the interest of this approach, we have re-implemented the models proposed by Jayasinghe et al. (2016),Truong et al. (2018), Chen et al. (2016a) and Militano et al. (2016) and we have applied them on our data. For each of the sub-mentioned related work, we conducted a series of tests to set the optimal parameters. The parameters are essentially the weights assigned to each feature and the threshold allowing to distinguish good nodes from bad ones. We report in the table 5.5 the parameters set for each of the related works, including ours.

Comparison of Node-Ranking module with related works

	Scena-1			Scena-2			Scena-3		
	R	P	F-M	R	P	F-M	R	P	F-M
Jayasinghe et al. (2016)	0.8	0.71	0.75	0.79	0.7	0.74	0.77	0.7	0.73
Truong et al. (2018)	0.91	0.82	0.86	0.91	0.82	0.86	0.87	0.79	0.83
Chen et al. (2016a)	0.78	0.92	0.84	0.67	0.82	0.74	0.67	0.82	0.74
Militano et al. (2016)	0.98	0.72	0.83	0.98	0.72	0.83	0.91	0.72	0.80
DSL-STM	0.98	0.88	0.93	0.96	0.88	0.92	0.95	0.9	0.92

In order to compare ourselves to the related work, we have applied each of the above-mentioned related work on the scena-1, scena-2 and scena-3 scenarios. We used recall, precision and f-measurement evaluation metrics. The obtained results are reported in table 5.6. Considering the three scenarios, our node-ranking module provide better results in terms of recall and precision (respectively (0.98;0,88) (0,96;0,88) and (0,95;0,9)). In scenario $Scena - 2$, Recall with features of Militano et al. (2016) reached respectively 0.98 and outperformed our set of features. However, with these configurations, their precision was significantly lower (0.72) than the precision obtained with our set of features (0.88). Overall, this shows the relevance of our features over those proposed in other works considering the same aggregation method.

5.3 Trust-propagation and update: Scalability, dynamism and resources-efficiency

In the previous section, we validated the functional properties of our TMM, in particular its resilience. In this section, we focus on the non-functional properties, which are equally important in a dynamic and constrained environment such as the SIoT. In what follows, we present a series of experiments conducted on the different scenarios mentioned at the beginning of the chapter, in order to prove the scalability, dynamism and resource minimization of DLS-STM.

Table 5.7: Average execution time for centralised, decentralised and hybrid propagation methods.

Series	Average execution time (second)		
	Centralised	Decentralised	DLS-STM (Hybrid)
Scena-1-D1	1100	1100	1100
Scena-1-D2	1200	1200	1100
Scena-1-D3	1300	1500	1100
Scena-2-D1	1500	1100	1300
Scena-2-D2	1600	1200	1300
Scena-2-D3	1600	1500	1300
Scena-3-D1	1800	1300	1400
Scena-3-D2	2100	1400	1400
Scena-3-D3	3100	1700	1400

5.3.1 Scalability vs Resources-efficiency

To estimate the behaviour of our TMM in front of a large number of nodes and its capacity to support scale-up, we applied simulations on the different scenarios scena1-D1, scena1-D2 up to scenario scena-3-D3. These scenarios differ based on two properties: the number of nodes and the dynamism. We used as evaluation metrics the average execution time and compared our hybrid propagation method to the centralized and decentralized methods. The results are reported in table 5.7.

We can deduce that the hybrid and decentralized methods are insensitive to the increase in the number of nodes. The hybrid propagation method is slightly sensitive to dynamism, whereas the decentralized method is sensitive to both the increase in the number of nodes and the dynamism of the network.

5.3.2 Dynamism vs Attacks-resiliency

To prove the interest of adopting the method of event-driven updating, we propose to compare, on the basis of different scenarios, the propagation rate of poor quality services ($NBQIS$) in the case of time-driven updating with the case of event-driven updating. We tested the time-driven method with

Table 5.8: Comparison between Time-driven and event-driven updating methods.

Series	Nb Total Interactions	DSL-STM + Time-driven update						DSL-STM + Event-driven update	
		Per 10 mn		Per 30 mn		Per 60 mn			
		NBQIS	%	NBQIS	%	NBQIS	%	NBQIS	%
Scena-1-D1	38 400	4002	10,42	7865	20,48	11333	29,51	3000	7,81
Scena-1-D2	144 000	8099	5,62	14506	10,07	30823	21,40	4050	2,81
Scena-1-D3	864 000	35100	4,06	95090	11,01	185715	21,49	8450	0,98
Scena-2-D1	1 440 000	301567	20,94	411111	28,55	453453	31,49	45903	3,19
Scena-2-D2	5 400 000	350233	6,49	733909	13,59	989603	18,33	170034	3,15
Scena-2-D3	32 400 000	3450300	10,65	7859003	24,26	7248897	22,37	900380	2,78
Scena-3-D1	120 000 000	12431030	10,36	17222080	14,35	33003000	27,50	10000767	8,33
Scena-3-D2	5,E+10	3540000000	7,87	6040000000	13,42	11236799321	24,97	2500000203	5,56
Scena-3-D3	2,77E+15	3,00E+14	10,83	6,11E+14	22,06	8,00E+14	28,90	7,00E+13	2,53

three different periods: updates every 10 minutes, updates every 30 minutes and updates every 60 minutes.

The results are reported in table 5.8. The results obtained with a 10-minute period on the scena-1 scenario are close to the results obtained by our TMM. Nevertheless, for the scena-2 and scena-3 scenarios, which have more nodes and more interactions, the number of poor quality services propagated is doubled. For the 30-minute and one-hour periods, they are almost tripled. The results obtained by the event-driven method are better.

5.4 Conclusion

We have presented, throughout this work, DSL-STM, a Multi-Dimensional Trust-Model for Dynamic, Scalable and Resources-efficient Trust-Management in Social Internet of Things. This chapter allowed us to test and validate the different contributions of DSL-STM, including the relevance of the proposed Trust-Features, the Attack-resiliency of the resulting Trust-Assessment Model and finally the scalability, dynamism and resources-efficiency of the Trust Management Mechanism that is built on the basis of the proposed Trust-Assessment model. To conduct these experiments, we used different evaluation metrics and tested on varying scenarios close to reality. We also compared ourselves to the work in the literature. The results presented in this chapter allow us to verify and validate the different

contributions.

Conclusion

With the burgeoning technological evolution in computer science technology, physical surrounding objects become embedded with computing technologies so as to give them a digital imprint and thus enabling them to interact and the surrounding environment giving birth to the Internet of Things (IoT) paradigm.

Though IoT is of great benefit to the society, it raises a plethora of issues for the researchers, designers and developers. Heterogeneity is a major issue where IoT devices are of varying nature at various levels. Scalability is a second primary concern where the number of devices that are getting connected to the IoT network is growing at a tremendous rate. Both of these issues impact the Resources Discovery and the Navigability in IoT networks, preventing the IoT vision, where objects are supposed to communicate and interact autonomously and where people should have access to a range of services anytime and anywhere, from becoming effective.

The Social Internet of Things (SIoT) is an extension of the IoT that has addressed the mentioned challenges through the integration of the social component. Intelligent objects in the IoT have evolved into social objects able to autonomously establish relationships with other objects, join communities and build their own social networks that may differ from those of their owner. The adoption of such a vision is a promising trend that offers

multiple advantages. However, in SIoT networks, the various players compete to propagate their services. Some of them resort to malicious behaviour and launch different types of attacks to spread poor quality services. These attacks, known as Trust-Attacks, can be dangerous in an environment like SIoT, where services can alter and act on the physical world. The implementation of a powerful and robust Trust Management Mechanism to counter these attacks and ensure reliable and qualified interactions was the first issue of this thesis.

On the other hand, SIoT networks are complex and multipartite, which generates different dimensions of trust to manage. These networks involve a large number of nodes that interact dynamically and in real-time. The majority of these nodes have limited storage and computing capacities. Ensuring that the proposed Trust Management Mechanism respects these different specificities and constraints was the second challenge of this thesis. The provision of a Trust Management Mechanism for the SIoT, therefore, requires the removal of barriers linked to scalability, dynamism and minimization of resource consumption.

Thus the contributions of this thesis cover the following points:

- Proposal of trust features relative to each of the trust dimensions involved.
- Proposal of a method for detecting and countering Trust-Attacks (based on Machine-Learning and Deep-Learning).
- Proposal of a hybrid propagation method that allows spreading trust values in the network, while overcoming the drawbacks of centralized and decentralized methods. This method thus ensures scalability and minimizes resource consumption.
- Proposal of a time-driven update method that respects the dynamic nature of SIoT networks.

The series of experiments based on simulations and carried out on different

scenarios have allowed validating the different points of this proposal.

As perspectives for this work, we plan to deepen our study of the device and service trust-dimensions. Indeed, contrary to the other works which were limited to the user trust-dimension, we proposed in this work to introduce these dimensions. Nevertheless, we have limited ourselves to the basic features. We can integrate other requirements related to GWPD, energy consumption or contextualization.

Our second perspective concerns the hybrid propagation stage. The method proposed in this work, although it overcomes the drawbacks of centralized and decentralized methods, involves the introduction of new nodes in the network. This can be costly when dealing with very large networks. With this in mind, we study two alternatives. The first concerns the introduction of Blockchain techniques, which go hand in hand with the notion of trust, and distributed systems. The second concerns distributed computing techniques, which is based on the MapReduce, and which allows complex computations to be split over the different nodes, in order to respect the limits of constrained nodes. The effective implementation of these methods will allow the validation or not of these ideas.

Finally, a last idea, consists in adapting and testing the proposed model in other social and collaborative environments. For example, for the detection of malicious nodes in Social Networks and Mobile Social Networks, for fraud detection in e-commerce sites or for cyber-security .

List of publications

Wafa Abdelghani, Florence Sèdes, Corinne Amel Zayani, Ikram Amous: Détection des attaques de confiance dans l'Internet des Objets Social. INFORSID 2020: 155-170.

Wafa Abdelghani, Florence Sèdes, Corinne Amel Zayani, Ikram Amous: Proposition d'un modèle d'évaluation de la confiance pour la détection des attaques dans l'Internet des Objets Social. EGC 2020: 471-472.

Mariam Masmoudi, Wafa Abdelghani, Ikram Amous, Florence Sèdes: Deep Learning for Trust-Related Attacks Detection in Social Internet of Things. ICEBE 2019: 389-404.

Wafa Abdelghani, Corinne Amel Zayani, Ikram Amous, Florence Sèdes: Trust Evaluation Model for Attack Detection in Social Internet of Things. CRiSIS 2018: 48-64.

Wafa Abdelghani, Corinne Amel Zayani, Ikram Amous, Florence Sèdes: User-centric iot: Challenges and perspectives. In: The Twelfth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (2018).

Ahlem Kalaï, Corinne Amel Zayani, Ikram Amous, Wafa Abdelghani, Florence Sèdes: Social collaborative service recommendation approach based on user's trust and domainspecific expertise. Future Gener. Comput. Syst:

355-367 (2018).

Wafa Abdelghani, Corinne Amel Zayani, Ikram Amous, Florence Sèdes: Trust Management in Social Internet of Things: A Survey. I3E 2016: 430-441.

Ahlem Kalaï, Wafa Abdelghani, Corinne Amel Zayani, Ikram Amous: LoTrust: A social Trust Level model based on time-aware social interactions and interests similarity. PST 2016: 428-436.

Appendix A

Overview of results for different deep-learning settings

In this appendix, we present a summary of the results for different deep-learning settings. This allowed us to choose the best configuration. The parameters used are as follows:

The batch size which is a hyperparameter of gradient descent that controls the number of training samples to work through before the model's internal parameters are updated.

The number of epochs which is a hyper-parameter of gradient descent that controls the number of complete passes through the training data-set.

The Neural network activation functions determine the output of a deep learning model, its accuracy, and also the computational efficiency of training a model and have a major effect on the neural network's ability to converge and the convergence speed.

Table A.1: Overview of results for different deep-learning settings

Batch-size	1									1								
Epoch	10									25								
Nb neur. Hid. L.	6			8			12			6			8			12		
A. f. Inp. L.	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
A. f. Hid. L.	R	T	T	R	T	T	R	T	T	R	T	T	R	T	T	R	T	T
A. f. Out. L.	R	R	S	R	R	S	R	R	S	R	R	S	R	R	S	R	R	S
Accuracy	0,73	0,75	0,79	0,82	0,87	0,91	0,72	0,81	0,88	0,83	0,85	0,84	0,89	0,91	0,98	0,82	0,88	0,88

Batch-size	1									2								
Epoch	50									10								
Nb neur. Hid. L.	6			8			12			6			8			12		
A. f. Inp. L.	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
A. f. Hid. L.	R	T	T	R	T	T	R	T	T	R	T	T	R	T	T	R	T	T
A. f. Out. L.	R	R	S	R	R	S	R	R	S	R	R	S	R	R	S	R	R	S
Accuracy	0,8	0,81	0,82	0,81	0,81	0,88	0,72	0,78	0,81	0,7	0,71	0,75	0,8	0,83	0,89	0,69	0,78	0,83

Batch-size	2									2								
Epoch	25									50								
Nb neur. Hid. L.	6			8			12			6			8			12		
A. f. Inp. L.	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
A. f. Hid. L.	R	T	T	R	T	T	R	T	T	R	T	T	R	T	T	R	T	T
A. f. Out. L.	R	R	S	R	R	S	R	R	S	R	R	S	R	R	S	R	R	S
Accuracy	0,79	0,8	0,8	0,83	0,85	0,93	0,76	0,79	0,8	0,71	0,78	0,79	0,78	0,81	0,80	0,72	0,74	0,8

References

Abdelghani, W., Zayani, C. A., Amous, I., and Sèdes, F. (2016). Trust management in social internet of things: A survey. In *Social Media: The Good, the Bad, and the Ugly - Conference on e-Business, e-Services, and e-Society*, pages 430–441, Swansea.

Aggarwal, C. C., Ashish, N., and Sheth, A. (2013). The internet of things: A survey from the data-centric perspective. In *Managing and mining sensor data*, pages 383–428. Springer.

Al-Masri, E. and Mahmoud, Q. H. (2007). Qos-based discovery and ranking of web services. In *Proceedings of the 16th International Conference on Computer Communications and Networks, IEEE ICCCN 2007, Turtle Bay Resort, Honolulu, Hawaii, USA, August 13-16, 2007*, pages 529–534, Honolulu, Hawaii, USA. IEEE.

Ali, D. H. (2015). *A social Internet of Things application architecture: applying semantic web technologies for achieving interoperability and automation between the cyber, physical and social worlds*. PhD thesis, Institut National des Télécommunications.

Ali, S., Kibria, M. G., Jarwar, M. A., Lee, H. K., and Chong, I. (2018). A model of socially connected web objects for iot applications. *Wireless Communications and Mobile Computing*, 2018.

Amin, F., Abbasi, R., Rehman, A., and Choi, G. S. (2019). An advanced algorithm for higher network navigation in social internet of things using small-world networks. *Sensors*, 19(9):2007.

Atzori, L., Iera, A., and Morabito, G. (2010). The internet of things: A survey. *Computer networks*, 54(15):2787–2805.

Atzori, L., Iera, A., and Morabito, G. (2011). Siot: Giving a social structure to the internet of things. *IEEE communications letters*, 15(11):1193–1195.

Atzori, L., Iera, A., and Morabito, G. (2014). From" smart objects" to" social objects": The next evolutionary step of the internet of things. *IEEE Communications Magazine*, 52(1):97–105.

Atzori, L., Iera, A., Morabito, G., and Nitti, M. (2012). The social internet of things (siot)–when social networks meet the internet of things: Concept, architecture and network characterization. *Computer networks*, 56(16):3594–3608.

Bao, F., Chen, I., and Guo, J. (2013). Scalable, adaptive and survivable trust management for community of interest based internet of things systems. In *International Symposium on Autonomous Decentralized Systems*, pages 1–7, Mexico City. IEEE Computer Society.

Bao, F. and Chen, I.-R. (2012a). Dynamic trust management for internet of things applications. In *Proceedings of the 2012 international workshop on Self-aware internet of things*, pages 1–6. ACM.

Bao, F. and Chen, R. (2012b). Trust management for the internet of things and its application to service composition. In *2012 IEEE international symposium on a world of wireless, mobile and multimedia networks (WoWMoM)*, pages 1–6. IEEE.

Barnes, J. A. (1954). Class and committees in a norwegian island parish. *Human relations*, 7(1):39–58.

Beatty, P., Reay, I., Dick, S., and Miller, J. (2011). Consumer trust in e-commerce web sites: A meta-study. *ACM Computing Surveys (CSUR)*, 43(3):1–46.

Bernabe, J. B., Ramos, J. L. H., and Gomez, A. F. S. (2016). Taciot: multidimensional trust-aware access control system for the internet of things. *Soft Computing*, 20(5):1763–1779.

Blomqvist, K. and Ståhle, P. (2000). Building organizational trust. In *16th Annual IMP Conference, Bath, UK*, pages 7–9. Citeseer.

Bormann, C., Ersue, M., and Keranen, A. (2014). Terminology for constrained-node networks. Technical report, IETF.

Chen, R., Bao, F., and Guo, J. (2016a). Trust-based service management for social internet of things systems. *IEEE transactions on dependable and secure computing*, 13(6):684–696.

Chen, R., Guo, J., and Bao, F. (2016b). Trust management for soa-based iot and its application to service composition. *IEEE Transactions on Services Computing*, 9(3):482–495.

Chen, Z., Ling, R., Huang, C., and Zhu, X. (2016c). A scheme of access service recommendation for the social internet of things. *Int. J. Communication Systems*, 29(4):694–706.

Chu, X., Chen, X., Zhao, K., and Liu, J. (2010). Reputation and trust management in heterogeneous peer-to-peer networks. *Telecommunication Systems*, 44(3-4):191–203.

Cirani, S., Davoli, L., Ferrari, G., Léone, R., Medagliani, P., Picone, M., and Veltri, L. (2014). A scalable and self-configuring architecture for service discovery in the internet of things. *IEEE internet of things journal*, 1(5):508–521.

Cook, K. S., Yamagishi, T., Cheshire, C., Cooper, R., Matsuda, M., and Mashima, R. (2005). Trust building via risk taking: A cross-societal experiment. *Social psychology quarterly*, 68(2):121–142.

Datta, S. K., Da Costa, R. P. F., and Bonnet, C. (2015). Resource discovery in internet of things: Current trends and future standardization aspects. In *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*, pages 542–547. IEEE.

Davis, E. and Marcus, G. (2015). Commonsense reasoning and commonsense knowledge in artificial intelligence. *Communications of the ACM*, 58(9):92–103.

Deng, S., Huang, L., Xu, G., Wu, X., and Wu, Z. (2016). On deep learning for trust-aware recommendations in social networks. *IEEE transactions on neural networks and learning systems*, 28(5):1164–1177.

Ding, Z., Gao, X., Guo, L., and Yang, Q. (2012). A hybrid search engine framework for the internet of things based on spatial-temporal, value-based, and keyword-based conditions. In *2012 IEEE International Conference on Green Computing and Communications*, pages 17–25. IEEE.

Dohr, A., Modre-Opsrian, R., Drobics, M., Hayn, D., and Schreier, G. (2010). The internet of things for ambient assisted living. In *2010 seventh international conference on information technology: new generations*, pages 804–809. Ieee.

Feng, Q. and Dai, Y. (2007). Lip: A lifetime and popularity based ranking approach to filter out fake files in p2p file sharing systems. In *IPTPS*.

Fortino, G., Lackovic, M., Russo, W., and Trunfio, P. (2013). A discovery service for smart objects over an agent-based middleware. In *International Conference on Internet and Distributed Computing Systems*, pages 281–293. Springer.

Geetha, S. (2016). Social internet of things. *World Scientific News*, 41:76–77.

Grandison, T. and Sloman, M. (2000). A survey of trust in internet applications. *IEEE Communications Surveys & Tutorials*, 3(4):2–16.

Granovetter, M. (1985). Economic action and social structure: The problem of embeddedness. *American journal of sociology*, 91(3):481–510.

Gubbi, J., Buyya, R., Marusic, S., and Palaniswami, M. (2013). Internet of things (iot): A vision, architectural elements, and future directions. *Future generation computer systems*, 29(7):1645–1660.

Guillen, J., Miranda, J., Berrocal, J., Garcia-Alonso, J., Murillo, J. M., and Canal, C. (2014). People as a service: a mobile-centric model for providing collective sociological profiles. *IEEE software*, 31(2):48–53.

Guo, H., Ren, J., Zhang, D., Zhang, Y., and Hu, J. (2018). A scalable and manageable iot architecture based on transparent computing. *Journal of Parallel and Distributed Computing*, 118:5–13.

Helbing, D. (1994). A mathematical model for the behavior of individuals in a social field. *Journal of Mathematical Sociology*, 19(3):189–219.

Holmquist, L. E., Mattern, F., Schiele, B., Alahuhta, P., Beigl, M., and Gellersen, H.-W. (2001). Smart-its friends: A technique for users to easily establish connections between smart artefacts. In *international conference on Ubiquitous Computing*, pages 116–122. Springer.

Huang, F. (2007). Building social trust: A human-capital approach. *Journal of Institutional and Theoretical Economics (JITE)/Zeitschrift für die gesamte Staatswissenschaft*, pages 552–573.

Huang, J., Seck, M. D., and Gheorghe, A. (2016). Towards trustworthy smart cyber-physical-social systems in the era of internet of things. In *System of Systems Engineering Conference (SoSE), 2016 11th*, pages 1–6. IEEE.

Hussein, D., Han, S. N., Lee, G. M., Crespi, N., and Bertin, E. (2017). Towards a dynamic discovery of smart services in the social internet of things. *Computers & Electrical Engineering*, 58:429–443.

Jara, A. J., Lopez, P., Fernandez, D., Castillo, J. F., Zamora, M. A., and Skarmeta, A. F. (2013). Mobile digcovery: A global service discovery for the internet of things. In *2013 27th International Conference on Advanced Information Networking and Applications Workshops*, pages 1325–1330. IEEE.

Jaradat, S., Dokoohaki, N., Matskin, M., and Ferrari, E. (2016). Trust and privacy correlations in social networks: a deep learning framework. In *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 203–206. IEEE.

Jayasinghe, U., Truong, N. B., Lee, G. M., and Um, T.-W. (2016). Rpr: A trust computation model for social internet of things. In *Ubiquitous Intelligence and Computing, Advanced and Trusted Computing*, pages 930–937. IEEE.

Jiang, N., Tian, F., Li, J., Yuan, X., and Zheng, J. (2020). Man: mutual attention neural networks model for aspect-level sentiment classification in siot. *IEEE Internet of Things Journal*, 7(4):2901–2913.

Kalaï, A., Zayani, C. A., Amous, I., Abdelghani, W., and Sèdes, F. (2018). Social collaborative service recommendation approach based on user's trust and domain-specific expertise. *Future Generation Computer Systems*, 80:355–367.

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

Kent, J. T. (1983). Information gain and a general measure of correlation. *Biometrika*, 70(1):163–173.

Kim, Y. and Peterson, R. A. (2017). A meta-analysis of online trust relationships in e-commerce. *Journal of Interactive Marketing*, 38:44–54.

Kini, A. and Choobineh, J. (1998). Trust in electronic commerce: definition and theoretical considerations. In *Proceedings of the thirty-first Hawaii International conference on System sciences*, volume 4, pages 51–61. IEEE.

Kranz, M., Holleis, P., and Schmidt, A. (2009). Embedded interaction: Interacting with the internet of things. *IEEE internet computing*, 14(2):46–53.

Li, Z., Chen, R., Liu, L., and Min, G. (2015). Dynamic resource discovery based on preference and movement pattern similarity for large-scale social internet of things. *IEEE Internet of Things Journal*, 3(4):581–589.

Liu, M., Leppänen, T., Harjula, E., Ou, Z., Ylianttila, M., and Ojala, T. (2013). Distributed resource discovery in the machine-to-machine applications. In *2013 IEEE 10th International Conference on Mobile Ad-Hoc and Sensor Systems*, pages 411–412. IEEE.

Luhmann, N. (2000). Familiarity, confidence, trust: Problems and alternatives. *Trust: Making and breaking cooperative relations*, 6:94–107.

Mäkitalo, N., Pääkkö, J., Raatikainen, M., Myllärniemi, V., Aaltonen, T., Leppänen, T., Männistö, T., and Mikkonen, T. (2012). Social devices: collaborative co-located interactions in a mobile cloud. In *Proceedings of the 11th International Conference on Mobile and Ubiquitous Multimedia*, page 10. ACM.

Marche, C., Atzori, L., Iera, A., Militano, L., and Nitti, M. (2017). Navigability in social networks of objects: The importance of friendship type and nodes' distance. In *2017 IEEE Globecom Workshops (GC Wkshps)*, pages 1–6. IEEE.

Marsh, S. P. (1994). Formalising trust as a computational concept. Master's thesis, University of Stirling.

Mayer, R. C., Davis, J. H., and Schoorman, F. D. (1995). An integrative model of organizational trust. *Academy of management review*, 20(3):709–734.

Militano, L., Orsino, A., Araniti, G., Nitti, M., Atzori, L., and Iera, A. (2016). Trusted d2d-based data uploading in in-band narrowband-iot with social awareness. In *Personal, Indoor, and Mobile Radio Communications (PIMRC), 2016 IEEE 27th Annual International Symposium on*, pages 1–6. IEEE.

Miorandi, D., Sicari, S., De Pellegrini, F., and Chlamtac, I. (2012). Internet of things: Vision, applications and research challenges. *Ad hoc networks*, 10(7):1497–1516.

Miranda, J., Mäkitalo, N., Garcia-Alonso, J., Berrocal, J., Mikkonen, T., Canal, C., and Murillo, J. M. (2015). From the internet of things to the internet of people. *IEEE Internet Computing*, 19(2):40–47.

Möllering, G. (2001). The nature of trust: From georg simmel to a theory of expectation, interpretation and suspension. *Sociology*, 35(2):403–420.

Molm, L. D., Takahashi, N., and Peterson, G. (2000). Risk and trust in social exchange: An experimental test of a classical proposition. *American Journal of Sociology*, 105(5):1396–1427.

Moreland, D., Nepal, S., Hwang, H., and Zic, J. (2010). A snapshot of trusted personal devices applicable to transaction processing. *Personal and Ubiquitous Computing*, 14(4):347–361.

Nitti, M., Atzori, L., and Cvijikj, I. P. (2014a). Network navigability in the social internet of things. In *2014 IEEE world forum on internet of things (WF-IoT)*, pages 405–410. IEEE.

Nitti, M., Girau, R., and Atzori, L. (2014b). Trustworthiness management in the social internet of things. *IEEE Transactions on knowledge and data engineering*, 26(5):1253–1266.

Novo, O. (2018). Blockchain meets iot: An architecture for scalable access management in iot. *IEEE Internet of Things Journal*, 5(2):1184–1195.

Oliveira, T., Alhinho, M., Rita, P., and Dhillon, G. (2017). Modelling and testing consumer trust dimensions in e-commerce. *Computers in Human Behavior*, 71:153–164.

Paganelli, F. and Parlanti, D. (2012). A dht-based discovery service for the internet of things. *Journal of Computer Networks and Communications*, 2012.

Perera, C., Zaslavsky, A., Christen, P., and Georgakopoulos, D. (2013). Context aware computing for the internet of things: A survey. *IEEE communications surveys & tutorials*, 16(1):414–454.

Rotter, J. B. (1967). A new scale for the measurement of interpersonal trust. *Journal of personality*.

Sethi, P. and Sarangi, S. R. (2017). Internet of things: architectures, protocols, and applications. *Journal of Electrical and Computer Engineering*, 2017.

Seungyong, Y., Jeongnyeo, K., and Yongsung, J. (2017). Security considerations based on classification of iot device capabilities. In *SERVICE COMPUTATION 2017*, pages 13–15, Athens, Greece. IARIA, 2017.

Sheth, A. and Anantharam, P. (2013). Physical cyber social computing for human experience. In *Proceedings of the 3rd International Conference on Web Intelligence, Mining and Semantics*, page 1. ACM.

Singh, D., Tripathi, G., and Jara, A. J. (2014). A survey of internet-of-things: Future vision, architecture, challenges and services. In *Internet of things (WF-IoT), 2014 IEEE world forum on*, pages 287–292. IEEE.

Stump, E., Jadbabaie, A., and Kumar, V. (2008). Connectivity management in mobile robot teams. In *2008 IEEE international conference on robotics and automation*, pages 1525–1530. IEEE.

Truong, N. B., Um, T.-W., and Lee, G. M. (2016). A reputation and knowledge based trust service platform for trustworthy social internet of things. *Innovations in Clouds, Internet and Networks (ICIN), Paris, France*, pages 104–111.

Truong, N. B., Um, T.-W., Zhou, B., and Lee, G. M. (2017). From personal experience to global reputation for trust evaluation in the social internet of things. In *GLOBECOM 2017-2017 IEEE Global Communications Conference*, pages 1–7. IEEE.

Truong, N. B., Um, T.-W., Zhou, B., and Lee, G. M. (2018). Strengthening the blockchain-based internet of value with trust. *2018 IEEE International Conference on Communications (ICC)*, pages 1–7.

Walsh, K. and Sirer, E. G. (2006). Experience with an object reputation system for peer-to-peer filesharing. In *NSDI*, volume 6, pages 1–1.

Washha, M., Qaroush, A., Mezghani, M., and Sèdes, F. (2017). Information quality in social networks: Predicting spammy naming patterns for retrieving twitter spam accounts. *INFORSID*.

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

Yan, Z., Zhang, P., and Vasilakos, A. V. (2014). A survey on trust management for internet of things. *Journal of network and computer applications*, 42:120–134.

Yao, J., Chen, S., Nepal, S., Levy, D., and Zic, J. (2010). Truststore: Making amazon s3 trustworthy with services composition. In *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pages 600–605. IEEE.

Ye, J., Dobson, S., and McKeever, S. (2012). Situation identification techniques in pervasive computing: A review. *Pervasive and mobile computing*, 8(1):36–66.

Yuan, Z., Lu, Y., Wang, Z., and Xue, Y. (2014). Droid-sec: deep learning in android malware detection. In *Proceedings of the 2014 ACM conference on SIGCOMM*, pages 371–372.

Zavlanos, M. M. and Pappas, G. J. (2007). Potential fields for maintaining connectivity of mobile networks. *IEEE Transactions on robotics*, 23(4):812–816.

Zhou, H., Liu, B., Hou, F., Luan, T. H., Zhang, N., Gui, L., Yu, Q., and Shen, X. S. (2015). Spatial coordinated medium sharing: Optimal access control management in drive-thru internet. *IEEE Transactions on Intelligent Transportation Systems*, 16(5):2673–2686.