Trust as a Pre-Defense Step for IoT Authorization

W R AL-Swed1,2, M A AL-Wahah1
1University of Thi-Qar – College of Education for Pure Sciences, Iraq.
2University of Thi-Qar – College of Nursing, Iraq.

wafaa_razaq@utq.edu.iq, mouiad-al@utq.edu.iq

Abstract. To order and provide services, IoT devices need to communicate and talk to many other devices before these services are provided. However, in complex and heterogeneous systems, certain devices need to trust each other. In this paper, we present an approach that is combined of two lines of defense to help build trust in IoT scenarios and to provide more secure interaction. The paper presents a precisely engineered ontology that serves as the main knowledge base for the definition and semantic registration for the entities within the IoT perimeter to facilitate their automated deployment. We show, in this paper, how trust can be used as a pre-defense step for authorization in flexible and an adaptive manner using semantic web technologies. We develop a proof of concept implementation and give the complexity analysis for our approach.

1. Introduction
The Internet of Things (IoT) is a concept and framework that considers the ubiquitous existence of several things/objects in the world that can communicate with each other and collaborate with other objects/things through wireless and wired connections and specific addressing schemes to build new services/applications and achieve common objectives. A world where the virtual, real, and digital are converging to build smart environments that support energy, transport, communities, and several other areas smarter [1]. The IoT is a modern paradigm where wired and wireless technologies such as RFID, WSN, Zigbee, Bluetooth, NFC, GPRS, and LTE are used to connect users, objects, and everything [2].

The IoT includes a technology that connects virtual world objects to the real world with the help of unique identifiers and interacts with each other through the network, allowing objects to be linked to anything with an ON and OFF switch at any time, anywhere [3].

IoT merges a vast quantity of everyday life devices of heterogeneous communication environments, making protection, and trust a major challenge. These devices in IoT are frequently exposed to public spaces and communicate by wireless networks and are, therefore, vulnerable to malicious attacks [4]. Another problem is that the IoT devices are work together to seek and provide services themselves. Hence, such devices must trust one another in such heterogeneous and complex environments [5].

In this paper, we propose a trust method as a proactive step for conducting IoT authorization using Semantic Web (SW) technologies. We present flexible and expandable connotations of trust between IoT entities. We have developed two OWL ontologies. The first one, Trust (TM) ontology, is used for representing and reasoning over the trusted/untrusted device to control other devices, the second ontology is a user access to the trusted device, it's called (TMA) ontology. It is used to represent and reasoner over the Knowledge Base (KB) to derive the access control decisions. The approach described in this paper is based on semantic web technologies, specifically, OWL ontologies, Jena inference rules, Pellet reasoner, and SPARQL (SPARQL Protocol and RDF Query Language) queries.
1.1. Ad-Hoc Trust in IoT

TM is a useful security service technology and its consequence is also utilized in many applications as Grid, P2P, etc. In IoT, TM plays a very important role in reliable data mining and fusion, qualified services with context-conscious knowledge, and optimized user privacy and security of information [3]. TM concept has been suggested to provide solutions to issues such as authentication mechanisms, key, and safe routing, where trust's basic idea is to create a trust relationship between two individual nodes. IoT trust is a term that includes analyzing the behavior of devices linked to the same network. The relationship of trust between connected devices helps to influence the future conduct of their interactions. To a certain degree, when devices trust one another, they tend to share resources and services. TM enables trust between devices to be computed and analyzed to make appropriate decisions to establish effective and reliable communication between devices. [4]. The TM in IoT takes different classifications according to the vision of each researcher. Generally speaking, TM models can be categorized as follows:

1. Context – aware - Based Trust: Assigns collaborating nodes with dynamic confidence scores according to various contexts (what was the state of the assisting node?) and different roles (which service needed the assistance of that node?). Each node is given a quality recommendation score representing its truthfulness when rating new nodes and adjusted also during the learning process after each interaction [6].

2. Fuzzy-Based Trust: A trust model for the IoT world based on fuzzy theory. Applied to enhance routing algorithm efficiency and to identify node behaviors in IoT for WSN networks [7].

3. Direct/Indirect – Based Trust: TM uses the direct knowledge created from direct node communication to determine trust between nodes. This distributed approach enables nodes to be fully autonomous in making important decisions about other nodes’ actions [4].

4. Machine Learning and an Elastic Slide Window Technique in Trust: A smart TM approach based on machine learning and an elastic slide window technique, which automatically assesses trust in the IoT resource. And determining service provider attributes solution to get a real-time method for the automated evaluation of OA resources assessing service attributes such as data availability or quality of service data [5].

5. Model HEXAGON Based Trust: TM System represents the human definition of trust throughout the computational algorithm with six main factors: Peer Advice, Operational Cost, Operational Risk, Credibility, Privacy, and Position and Identity, to establish the value of trust in the IoT-dependent context [8].

1.2. Semantic-Based Trust

The SW is a distributed model that uses network protocols to publish, use, and extend structured knowledge. One of the main aims of this technology is to simplify data collection and integration and to make inferences over the existing knowledge bases. This automation includes languages of logic and rules that render inferences, pick action plans, and answer questions. The open-mindedness web, therefore, contributes to many Issues, including handling for contradictions, the incorporation of different information, and the commitment of data quality and trustworthiness [9]. The SW provides machine-readable web data rendering technologies. These technologies include a standard data model, the Resource Definition Framework (RDF), the RDF Schema (RDFS), a language for defining vocabulary such as entity and property hierarchies utilizing RDF, and the Web Ontology Language (OWL 1 & 2), a language of ontology for communicating all kinds of domain information, including complex relationships between properties and classes [10].

For the SW, the central principle is the incorporation and use of data from various sources. When dealing with the data integration and TM process, there are many definitions related to trust. It is regarded variously in various fields, e.g. it is perceived as protection and access control in computer networks, reliability is in distributed systems, and the right decision-making with uncertainty is viewed in policies and game theory [11].
There are two types of trust we need to know before we can completely determine trust in the web operations: trust in data sources and trust in rules. These two confidence levels can be mixed to synthesize a degree of trust in the inferences a reasoner had made. Trust in sources of data forms the pillar of trust on the web because no confidence can be built in the data we use when making decisions without a firm base. Considering rules for which the trust of a source of data also plays a role in deciding whether a rule that has matched is worth considering, this data type, also known as linked data, is linked but is dispersed spatially. Laws must be able to navigate this data network dynamically to gain additional facts to support conclusions, if this is needed. However, given the transparency of the web, the reliability of the information depends on the reliableness of the source, the data maintainer, the last date it has been modified, etc. [10].

The remainder of this paper is organized as follows: in section 2, we present works related to our approach. Section 3 is ontology-based trust, and in section 4 we show our implementation results and the complexity analysis of the proposed approach. Section 5 concludes the paper and outlines future work.

2. Related Work

Kowshalya et al. [12] suggest a model for measuring trust between social IoT (SIoT) nodes that uses first-hand observation (Direct Trust), second-hand (Indirect Trust), centrality, and reliability of a node that makes recommendations for a SIoT network or group is not easy and involves the integration of heterogeneous data and communication solutions. In this way, secret codes are used to provide security and reliability, and this may make the system insecure and vulnerable to attack, while in our approach, we are using TM and authorization to provide reliability, and to maintain the security of the system.

Esposito et al. [13] propose tackling heterogeneity in complex distributed networks by using fuzzy logic, combining it with an ample means to handle heterogeneous fuzzy sets. They proposed a technique for combining qualitative and quantitative specifications of trust scores to calculate a new degree of confidence. Complex distributed networks are characterized by having heterogeneous technologies such as IoT. Hence, their technique intensifies existing threats to internet security since fuzzy logic suffers from so many inaccuracies and is not as capable as semantic technologies. In our work, the solution is provided by using one of the SW technologies via using ontologies in both TM and authorization.

In their research, AL-Wahah and Farkas [14] introduce an IoT context-aware authorization method. They employ ontologies and logic programming rules to formulate the authorization policies, but they do not take into consideration using TM.

Abdul-Rahman et al. [15] introduced a trust model based on consensus recommendations, proposing a solution to the TM issue using a distributed trust model and a recommendation protocol. However, their model ignores several issues such as anonymity and entity naming, while our framework provides naming entities and identification using the ontology.

3. Ontology-Based Trust

Ontologies are a key technique to solve the issue of automating the deployment for technologies in heterogeneous IoT environments, allows any IoT entity to convey the sense of data/information they 'bear' unambiguously. As an abstraction technology, the purpose of IoT ontology is to mask the heterogeneity of IoT entities, act as a mediator between providers of IoT applications and consumers, and encourage their semantic match making [16]. The data structure can be acquired semi-automatically with the use of text-mining strategies and assisted by ontology [11]. Ontology is an explicit, formal definition for a particular domain of a shared conceptualization [12].

In OWL, the basic element is a class (or concept in DL) that is used to specify the domain entities. Two types of properties are also described by OWL: Datatype properties and Object properties. Object properties specify the relationship between class instance pairs (individuals), whilst Datatype properties specify the relationship between the class instance and the value of Datatype [14].
In our approach, we demonstrate how semantic-based trust can be seamlessly integrated into IoT systems. And the ontology for authorization when doing so can serve as a safe choice key for the IoT application to choose among the available entities, the entity(ties) that the application must trust for its effective deployment in each context. As the main technology for the semantic abstraction and recording of these entities, to support their automated deployment. To reason over the ontology, we use Pellet reasoner for investigating concept’s hierarchy, consistency, satisfaction, subsumption, and the realization of instances.

3.1. Trust Comes First in IoT

In this section, we employ TM as the first line of defense and as a pre-defense step for IoT authorization. Where the requirements for managing trust between IoT entities will be represented using the ontology shown in figure 1, and we will show a short example of a smart company consists of several rooms and each room contains a group of smart devices. There is also a person who is conditional to be an employee with a manager grade, and this person is responsible for controlling the device (the trusted device) through which he controls other devices, this trusted device can be a smart phone or a smart laptop. We will also illustrate in this scenario how this person (manager) can control these devices through the device (smart phone or smart laptop) as it can operate these devices on working days and during specific hours, as well as her/his ability to turn off these devices at the end of working hours. And so on by (the smart trusted device) controlling operation and switching off the other smart devices, we have proved TM between IoT entities.

We developed the company's ontological model using Protégé. Below is a detailed explanation of the ontology used to explain the TM between IoT entities:

TM ontology defines eleven basic concepts: These concepts are Day, Trusted Devices, Features, Holiday, Location1, Administrator, Time, Trusted, UnTrusted, Device, and User. The class Features have five subclasses. These are MAC Address, Password, Serial Number, ID and Pin code and they represent the features of devices. The class User has sub class Employee. The class Administrator is the complex class that represents the user who owns or uses the trusted device that controls other devices according to a set of conditions to allow her/him to use the device as shown in formula 1 (written in OWL):

\[
\text{Administrator Equivalent To}
\]

\[(\text{isEmployee some Employee}) \text{and (has_Manager value true)}\]
Formula 1. Administrator definition.

According to our current example, when we're running the reasoner, the instances of the class Administrator is (User1). The class Trusted_Devices represents the trusted devices; in our example this class contains five instances from trusted smart devices. The class Location1 has subclassCompanyLoc, this class defines the spatial context of the devices that we want to use in controlling the trusted device. The class Trusted has a complex subclass that is Trusted1_device. The purpose of our work is illustrated in this complex class. Class Trusted1_device determines that the device is trusted/untrusted. Being trusted or not is determined by a set of features that distinguish an instance of a device from other devices, and these features are (MAC Address, Password, Serial Number, ID and Pin code) as shown in formula 2 (written in OWL):

```
Trusted1_device Equivalent To (belongsTo some
(Administrator and User and (hasUserID value 15)
and (has_Name value Rawan) and
(has_UserPW value "15R"))) or (has_ID value "135")
or (has_MACAddress value 12-23-34-45-56-67)
or (has_Password value 315765) or
(has_PinCode value 210455) or
(has_Serial_Number value "65")
```

Formula 2. Trusted1_device

After running the reasoner, it will determine if this device is trusted or not, whether it belongs to the group of trusted devices included in the class Trusted_Devices or not. For our current example, after running the reasoner, the inferred trusted device is (smartphone), and it is classified as (device2) one of the trusted devices and an instance of the class Trusted_Devices. We should note that the results of the reasoner are not only classification of these smartphone and device2 which appeared in the class Trusted1_device but also include consistency checking and realization of the instances of the whole ontology.

The class Device has four subclasses are DevicesContByTrDev_Operation, DevicesContByTrDev_Turn_Off, At_WorkingEnv, and At_HomeEnv. The complex subclass DevicesContByTrDev_Operation represents all the devices (objects) that we want to control (run it) by (the trusted device) and are identified by the feature that distinguishes each device from the other, and this feature is (MAC address) as shown in formula 3:

```
DevicesContByTrDev_Operation Equivalent To (Device
and (has_MAC_Address value "20:14:34:50:45:62"))
or (has_MAC_Address value "13:30:33:80:90:99")
```

Formula 3. DevicesContByTrDev_Operation.
After running the reasoner, the instances of this class are all devices that we want to run, and according to our example, the devices are (smart_lamp_on and smart_lock_door_on). The complex subclass DevicesContByTrDev_Turn_Off represents all the devices (objects) that we want to control (turn off) by (the trusted device) and are identified by the feature that distinguishes each device from the other, and this feature is (MAC address), and as shown in formula 4:

$$DevicesContByTrDev\_Turn\_Off \text{ Equivalent To } (Device$$

$$\text{and } (\text{has\_MAC\_Address \text{value } }"66:77:88:90:10:11")$$

$$\text{or } (\text{has\_MAC\_Address \text{value } }"11:13:15:17:19:21")$$

Formula 4. DevicesContByTrDev_Turn_Off.

After running the reasoner, the instances of this class are all devices that we want to turn off, and according to our example, the devices are (smart_lamp_off and smart_lock_door_off). The complex subclass At_WorkingEnv represents the specific context of the work, as shown in formula 5:

$$At\_WorkingEnv \text{ Equivalent To } (\text{has\_Location1 some Location1})$$

$$\text{and } (\text{has\_Day only (not (Holiday))) and (has\_Time \text{value } }"09:17")$$

Formula 5. At_WorkingEnv context.

This class is also equivalent to a class DevicesContByTrDev_Operation, after running the reasoner, the outputs of class At_WorkingEnv are the same as instances of class DevicesContByTrDev_Operation, and they denote the devices we want to run. Finally, the complex subclass At_HomeEnv, which represents the specific context of the work, as shown in formula 6:

$$At\_HomeEnv \text{ Equivalent To } (\text{has\_Location1 some Location1})$$

$$\text{and } (\text{has\_Day only (not (Holiday))) and (has\_Time \text{value } }"17:09")$$

Formula 6. At_HomeEnv context.

This class is also equivalent to the class DevicesContByTrDev_Turn_Off, after running the reasoner, the instances of class At_HomeEnv are the same as outputs class DevicesContByTrDev_Turn_Off, they are the devices we want to turn off.

3.2. Trust-Based Authorization in IoT
After we have proven in the first line of defense TM, we will move here to the second line of defense to increase the security of the system against external attacks, where we will determine who is authorized to use the trusted device. This is done through a set of conditions. In our approach, the access authorization depends on the of certain conditions that are to be fed to the access control engine. The access control engine, depending on the provided conditions and information, may accept or refuse such an access authorization request. Access control policy requirements are represented using the TM & Authorization (TMA) ontology shown in figure 2.
In our work, the access control model consists of a set of rules and each rule is given by the following definition:

(TMA Rule): Let Rule, Request, RequestType, RequestSubject, SubjectContext, RequestObject, ObjectContext, Request − Decision, DecisionEffect, be concepts defined in TMA ontology. Then the TMA rule is given as:

\[
\text{rule} = (r, rt, rs, sc, ro, oc, rd, de), \quad \text{where} \quad r \in \text{Rule}, \quad rt \in \text{RequestType}, \\
rs \in \text{RequestSubject}, \quad sc \in \text{SubjectContext}, \quad ro \in \text{RequestObject}, \quad oc \in \text{ObjectContext}, \\
rd \in \text{RequestDecision}, \quad de \in \text{DecisionEffect}.
\]

Each rule is triggered by an access request and is evaluated to a decision at runtime. The decision has an effect which, in our model, is either a "Refuse" or an "Accept". We represent the TMA policy as a set of conjunctive rules as follows:

\[
\text{TMA policy} = \text{rule1} \land \text{rule2} \land \text{rule3} \land \text{rule4} \land \text{rule5} \ldots
\]

The rule axiom becomes straightforward and is given as shown in formula 7:

\[
\text{Rule Equivalent To Request and (hasType some RequestType)}
\]

\[
\text{and (hasSubject some (Subject and (hasContext some Context)))}
\]

\[
\text{and (hasObject some (Object and (hasContext some Context)))}
\]

\[
\text{and (hasDecision some (Decision and (hasEffect some}}
\]

\[
\{"Refuse","Accept"\})
\]

Formula 7. Request rule axiom.

From our current example, we build access control authorization rule as:
Rule: A manager who is Subject can access the Trusted device.

This rule states that if a request \( r \) of type access is issued to the TMA policy engine by a subject \( s \) that has a context Subject on a resource \( rs \) and the request has a decision \( d \), then insert an accept effect for \( d \) into the KB. Using our example, a user (Rawan) is a Subject, as shown in formula 8:

\[
\begin{align*}
\text{Subject Equivalent To Administrator Equivalent To} \\
\quad \text{((is_Employee some Employee) and (has UserID some xsd:int)} \\
\quad \text{and (has UserName some xsd:string) and (has Manager value true)}
\end{align*}
\]

Formula 8. Rawan as a Subject.

Access authorization to a resource is given only if the context of the subject, the context of the resource, and/or any other context defined by the access control policy corresponds to a specific context rule of "accept".

In addition, if the TMA policy engine wants to understand the subject and resource identities, it simply asks its KB store to know that. If the TMA policy engine, for example, you need to know who the subject is in context and what \( rs \) can access resources; it can simply issue the SPARQL (Language of SPARQL Protocol and RDF Query [17]) query shown in figure 3 to its KB store.

```sparql
PREFIX appw1: <http://www.semanticweb.org/m/appw1#>
PREFIX A: <http://www.semanticweb.org/A#>
PREFIX CT: <http://www.semanticweb.org/CT#>
PREFIX TM: <http://www.semanticweb.org/TM#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xml: <http://www.w3.org/XML/1998/namespace#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

SELECT DISTINCT ?r ?rs ?s ?d ?ac
WHERE {
  ?r rdf:type TMA:Request.
  ?r TMA:hasResource ?rs.
  ?r TMA:hasSubject ?s.
  ?r TMA:hasDecision ?d.
  ?d TMA:hasImpact ?ac.
}
```

Figure 3. SPARQL query over TMA KB.
SPARQL precedes a string by? Symbol to denote a variable, so the? r, ?rs, ? s, ? d, and? ac are variables representing the request, the resource, the subject, the decision, and the Impact, respectively. The results of the SPARQL query are given in table 1 shown below:

Table 1. SPARQL query results.

| ?r  | ?rs | ? s  | ?d  | ?ac |
|-----|-----|------|-----|-----|
| r   | devices | user 1 | d   | accept |

Algorithm 1 is used to compute trust and authorization decisions.

Algorithm 1 (Ontology based TMA)

Input: Request (Pin Code, Password, MAC Address, Serial Number, ID),

TMA ontology, RQ is an Access Request; RS is Jena rule − set.

Output: semantic class classification {Trusted, UnTrusted}, Access decision, either"Accept" or "Refuse".

1. TT = Features (Pin Code, Password, MAC Address, Serial Number, ID)
2. if TT = UnTrusted then
3. exit()
4. Call_DL_Reasoner (Pellet, TMA)
5. if Trusted1_device ⊑ Trusted then
6. return (Trusted)
7. exit()
8. else
9. return (untrusted)
10. exit()
11. RT ← parse (RQ)       RT ← parse (RQ)  
12. if RT = access then
13. sc ← getContext(s); 
14. rc ← getContext(r); 
15. a ← evaluate(s, sc, r, rc, TMA, RS); 
16. if a = "Accept" AND noConflict (a, s, sc, r, rc, TMA, RS) then
17. return ("Accept"); 
18. exit()
19. end if
20. else
21. return ("Refuse"); 
22. exit()
23. end if
24. end

4. Implementation Results

Ten groups of data have been generated and processed to eliminate duplicates using an online data generator. Then we used Protégée's Cellfie plugin to claim these instances to the ontologies of the model. The sets are 10000, 20000, 30000, 40000 and 50000 triples in their size. By sending SPARQL queries to the populated samples of TMA ontologies in our application program, we performed tests
on the TMA engine. Only the approved data defined by the sample access control rules were returned successfully by the engine. Using the DL reasoner Pellet, the reasoning process is completed. DL reasoning has been used to verify the ontology consistency and satisfiability of, conceptual subsumption, and realization of instance. DL executes the majority of the reasoning process in this article. The process of reasoning has been replicated ten times for each of the defined data sets. Then the average time measured for these ten repetitions is taken to obtain particular values for the execution time of the reasoning process as shown in the table 2. Figure 4 also shows the results of DL reasoning time.

| Axiom | Pellet reasoner |
|-------|----------------|
| 331   | 260.6          |
| 1331  | 822.2          |
| 2331  | 825.5          |
| 3331  | 774.6          |
| 4331  | 1041.1         |
| 5331  | 1181.2         |

The Description Logic (DL) used in this work has an expressivity of $ALCO(D)$. This kind of DL’s expressivity has a computational complexity of ExpTime (exponential time) for the worst-case computations. We believe that this complexity is not an obstacle for the practicality of our approach since it rarely reached in real-world applications. Even if this happens, there are already existing methods to reduce this complexity or to overcome its implications such using incremental reasoning or engaging less expressive DL axioms.

Another advantage of using semantic web technologies with reasoning is that of trackability and explanation. We can traceback any action that is done previously such as access operations. For example, to understand why smart_lock_on which a Device is unlocked/locked, we just need to show the explanation facility of Protégé reasoning (or calling it from inside our Java application), as shown
in figure 5. This facility makes it possible to detect and traceback any penetration (if any) and provide an explanation about why it is happened.

![Image of Traceability facility in Protégé.](image)

Figure 5. Traceability facility in Protégé.

5. Conclusion and Future Work
In this paper, we propose a trust approach as a pre-step for authorization between IoT entities. We achieve this by using two lines of defense, the first line of defense is the TM and the second line of defense is represented by the authorization. We employ SW technologies to provide an approach that IoT devices can use to manage the trust and to provide authorization. We have developed two OWL ontologies. The first one, TM ontology is used for representing and reasoning over the trusted device to control other devices, the second ontology, TMA ontology is used to represent and reason over the KB to derive the access control decisions. As a reasoning approach, we adopted DL to demonstrate its applicability. We have shown how the SW techniques can be used in providing a more secure authorization in IoT environments.

Our ongoing work is to add a third line of defense using machine-learning algorithm to increase security of the protected resources.

6. References
[1] Patel KK, Patel SM, Scholar PG. Internet of Things-IOT: Definition, Characteristics, Architecture, Enabling Technologies, Application & Future Challenges. Int J Eng Sci Comput [Internet]. 2016;6(5):1–10. Available from: http://ijesc.org/
[2] Gubbi J, Buyya R, Marusic S, Palaniswami M. Internet of Things (IoT): A vision, architectural
elements, and future directions. Futur Gener Comput Syst. 2013;29(7):1645–60.

[3] Singh A. Internet of Things and Trust Management in IoT – Review. PdfsSemanticscholarOrg [Internet]. 2016;761–7. Available from: https://pdfs.semanticscholar.org/d351/97c59620aca36a29599cb5916cfe587fe743.pdf

[4] Mendoza CVL, Kleinschmidt JH. Mitigating on-off attacks in the internet of things using a distributed trust management scheme. Int J Distrib Sens Networks [Internet]. 2015;2015. Available from: https://doi.org/10.1155/2015/859731

[5] Caminha J, Perkusich A, Perkusich M. A smart middleware to detect on-off trust attacks in the Internet of Things. 2018 IEEE Int Conf Consum Electron ICCE 2018. 2018;2018-January:1–2.

[6] Ben Saied Y, Olivereau A, Zeghlache D, Laurent M. Trust management system design for the Internet of Things: A context-aware and multi-service approach. Comput Secur [Internet]. 2013;39(PART B):351–65. Available from: http://dx.doi.org/10.1016/j.cose.2013.09.001

[7] Chen D, Chang G, Sun D, Li J, Jia J, Wang X. TRM-IoT: A trust management model based on fuzzy reputation for internet of things. Comput Sci Inf Syst. 2011;8(4):1207–28.

[8] Al-johani AA, Youssef AE. a Framework F or. 2013;3(3):1–14.

[9] Kagal L, Jacobi I, Khandelwal A. Computer Science and Artificial Intelligence Laboratory Technical Report Gasping for AIR Why we need Linked Rules and Justifications on the Semantic Web Gasping for AIR — Why we need Linked Rules and Justifications on the Semantic Web *. MIT CSAIL Tech Reports. 2011;

[10] Kagal L, Jacobi I, Khandelwal A. From Truth Maintenance to Trust Management on the Semantic Web. MIT Technical Rep. 2010;(04281).

[11] Lavbič D, Žitnik S, Šubelj L, Kumer A, Zmec A, Bajec M. Traversal and relations discovery among business entities and people using semantic web technologies and trust management. Front Artif Intell Appl. 2013;249(May 2014):164–74.

[12] Meena Kowshalya A, Valarmathi ML. Dynamic trust management for secure communications in social internet of things (SIoT). Sadhana - Acad Proc Eng Sci [Internet]. 2018;43(9):1–8. Available from: https://doi.org/10.1007/s12046-018-0885-z

[13] Esposito C, Castiglione A, Palmieri F, Ficco M. Trust management for distributed heterogeneous systems by using linguistic term sets and hierarchies, aggregation operators and mechanism design. Futur Gener Comput Syst. 2017;74:325–36.

[14] 2018 ICITST | WorldCIS WCST | WCICSS.

[15] 2018 ICITST | WorldCIS WCST | WCICSS.

[16] Kotis K, Vouros GA. Trust Semantics in IoT Entities’ Deployment. Artif Intell Internet Things Hell Conf Artif Intell. 2016;

[17] Fernandez-Gago C, Moyano F, Lopez J. Modelling trust dynamics in the Internet of Things. Inf Sci (Ny) [Internet]. 2017;396:72–82. Available from: http://dx.doi.org/10.1016/j.ins.2017.02.039.