Formal Trust Architecture for Assuring Trusted Interactions in the Internet of Things

1st Milankumar Patel
Computer and Engineering Sciences
Florida Institute of Technology
Melbourne, USA
patelm2017@my.fit.edu

2nd Siddhartha Bhattacharyya
Computer and Engineering Sciences
Florida Institute of Technology
Melbourne, USA
s.bhattacharyya@fit.edu

3rd Ali Alfageeh
Computer and Engineering Sciences
Florida Institute of Technology
Melbourne, USA
aalfageeh2017@my.fit.edu

***Abstract*—The Internet of Things (IoT) has provided a flexible platform for a large number of heterogeneous devices to dynamically join and leave the network. This enhances the availability of a diverse range of services provided by a network. However, this dynamic expansion of the network with mobile IoT devices introduces a major challenge to security especially related to management of trust across the diverse IoT platforms. Thus, arises the necessity for a mechanism that can ensure the selection of secure and trusted devices as these devices try to join the network. In this work, an effective formal approach to trust modeling mechanism has been investigated to support dynamic trust development between devices based on two-tier trust architecture. We proposed the need to formally represent the requirement for trust among IoT devices by developing a well formulated trust ontology. Trust reasoning and management is a challenging aspect in this dynamic environment which we have addressed here by associating integrity with interactions among the device. Additionally, we have developed a method to select a recommender device among participant nodes who acts as a trusted leader within IoT device.**

***Index Terms*—Internet of Things, Trust Architecture, Trust Ontology, Trust Reasoning, Trust Semantics, IoT Trust Evaluation, Trusted Interactions, Semantic Web, Formal Approach.**

I. Introduction

The evolution of the Internet of Things (IoT) has increased the growth of deployment of various IoT devices for diverse applications, because it is an open platform that is distributed in nature. This allows many heterogeneous devices to communicate with each other based on their objectives or tasks that need to be accomplished. Currently, there are a lot of ongoing research in this domain to make it more adaptable, versatile and secure to help the future generation of IoT networks to function efficiently. For this to happen, it is essential to make devices smart enough in the distributed environment, about their selection process of other devices for data communication. Furthermore, a distributed network requires a mechanism that can predict the behavior of nodes based on the evolution of several factors such as personal attributes of that device, recommendation from other devices, and previous experiences that the device has had with other entities to intelligently make a decision about collaboration with other nodes in the IoT network as discussed in [6].

Developing a trustful relationship between devices plays a vital role in reliable and secure communication. Devices use the concept of digital trust to select and apply feasible measures to overcome potential risks. Digital trust considers more aspects compared to social trust to evaluate the relationship as mentioned by Li et. al. in [11]. Accumulating such accurate information for trust purposes in a distributed remote digital system can be a challenging task.

Many solutions have been proposed but one of the key limitations of present approaches is the lack of a structured software engineering approach to develop automated reasoning to compute trust among IoT devices as they join and leave a network. We propose to introduce such an approach by following more of an incremental design and build approach. In this method one of the initial challenges is in gathering the requirements, extracting the relevant attributes and then modeling them to perform analysis in regards to generating trust values among IoT devices.

In order to address the challenges in achieving digital trust among IoT devices a more formalized approach was investigated in [22]. Here, we have proposed a two-tier trust establishment architecture as discussed in section III. Once the architecture is defined the next step involves representing the knowledge within the defined architecture. In our case the knowledge is represented using a formal approach as provided by an ontological representation which is explained in section IV. This was followed by mapping of ontology and SWRL queries into a lower level implementation environment(Java) to as covered in V. Finally, a Java based trust reasoner to evaluate dynamic behaviour of device and recommendation calculation is covered in section VI.

II. Literature Review

The work proposed by Kini and Choobmeh [9] considered the theoretical framework of trust with the perspective of psychology and economics. They relied on the Merriam-Webster definition of "Trust" which is as follows: "A trust is the reliance on someone or things for specific context". Marsh's work [12] on formalizing trust as a computational concept gives an idea about how biology and sociology use trust to define the interaction between different agents in a distributed environment . Alternatively, it can be defined as the level of reliability imposed on a device or entity in terms of confidence as part of their conditional relationship. It is a

combination of many mandatory attributes such as reliance, dependency, honesty, trustfulness, and security. The absence of attributes mentioned above in a participating node leads to distrust. It also includes further scrutiny like authorization and authentication to verify node's authenticity. Authentication is a process where all the properties of a node in terms of the device's attributes as well as its security certificate related attributes would get verified by an authorized entity to make the device secure and trustworthy. The trust architecture proposed here is used to calculate the digital trust for a device in a distributed digital system. The amount of data collected and processed in a digital environment is large, so the trust architecture helps in decomposing the data into different levels, which makes the data manageable to process inorder to develop a trustful relationship between devices.

A. Trusted relationship

Before initiating any interaction between devices, there must be a some trusted relationships. The work proposed by Yefeng and Arjan about classifying different types of trusted relationships between devices and human helps us to the understand importance of trust in IoT [17]. Relationships are classified as human-to-human trust relationship, device-to-device trust relationship, and human-to-device trust relationship. Behavioral evidence, data similarities, profile information, and identity are the parameters that IoT devices look for from other devices. Behavioral evidence includes specific characteristics related to personal properties of the device. A device's profile information and list of attributes such as serial number, manufacturer information, owner information is used to calculate the authenticity of the device. Yan and MacLaverty proposed the following steps to establish trusted relationship [23]. This approach was based on relationship of the trustee with the trusted agent and based on the personal list of attributes. Here are the necessary four steps are as follows:

- **Trust establishment:** It monitors the behavior of the trustee and its task is to collect useful evidence to assign an initial trust level to the trustee.
- **Trust monitoring:** It keeps track of the behavior of a trustee with others and its task is to categorize interactions based on the trustee's behavior .
- **Trust assessment:** It assigns a new value of trust to the existing relationship after each successive interaction.
- **Trust control and re-establishment:** This part of the process takes care of the continuous relationship status and re-establishes the detached or broken connections between entities.

B. Trust Management Model

The trust modeling and management can be applied to a real-time distributed environment to enhance its security and to reduce the dependability of a system by increasing the number of trustful interactions between devices [10]. In a static environment where all the conditions can be controlled by the user, a device uses the initial trust level to establish a trust-oriented relationship with other devices. It is more secure in nature, but it is not useful to understand the behavior of the device and for trustworthiness calculation. Since we are proposing an architecture for distributed environment, we also have to consider the dynamics and randomness of the distributed environment. In order to make trust assessment more robust and trustworthy for devices, we have implemented a dynamic environment by invoking an inbuilt functionality in a Java environment. The concept of recommendation was mentioned by many researchers in the context of social trust. To make it applicable in a distributed environment to compute digital trust, we have proposed and implemented a strategy in the context of an IoT device. The distributed trust model proposed by Abdul-Rahman and Stephan is recommendation-based trust where they have proposed a way to achieve conditional transitivity of trust [1]. The distributed trust model uses direct and recommended trust to assign continuous values of trust to the nodes, which range from [0,1].
One of the renowned works done in the dynamic trust management protocol proposed in [3] has considered completely different trust properties in the protocol. Those are called honesty, cooperativeness, and community internet. These properties can also be calculated using the equations provided in his work. This protocol is event-driven and interaction-based in nature. It checks nodes correspondences with others, and keeps track of the community interest present in the network.

III. Two-Tier Trust Architecture

The architecture that we propose in this work is aimed to establish a trusted relationship between IoT devices. So, the creation of the architecture focuses on identifying and representing the essential elements required to have reliable communication among IoT devices in a distributed environment. Arbia Rahi [16] proposed that a trust architecture has four main responsibilities. These are initial trust establishment for all the interacting devices in the network, computation of trust for newly connected IoT device in a network, the modification of trust after successful interactions in an IoT environment, and computation of a recommendation for indirect trust assessment. Our architecture is also influenced by Flip Forsby's light-weight certificate [5] for secure and trusted IoT interactions. After extensive review of all the potential attributes [2], [21] that are key to enable automated digital trust generation we classified them broadly as static and dynamic. This led to the design of the two-tier architecture that can satisfy the above-mentioned responsibilities.

The top level of the architecture emphasizes on the requirements to establish initial trust among IoT devices. The initial trust assessment process includes an initial trust assignment to all the individual nodes that are present in the distributed environment. Initial trust establishment is done based on the list of personal traits some of which is as shown in Fig. 1, the full list is as documented by Patel et.al [13]. Thus, the trust establishment at this level focuses on static attributes associated with a device.

The second or the bottom level of the architecture captures the essential elements required to dynamically compute the

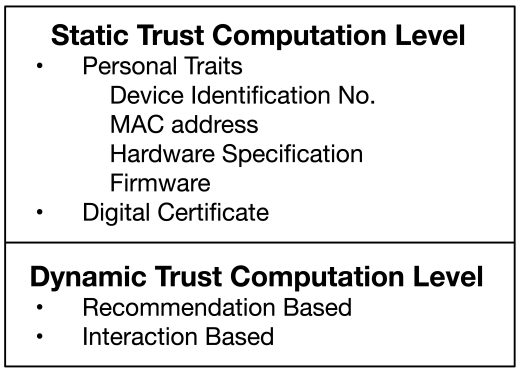

Figure 1. Two-Tier Trust Computation Architecture

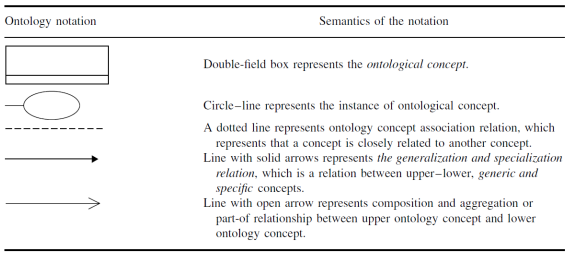

Figure 2. Ontology Notation

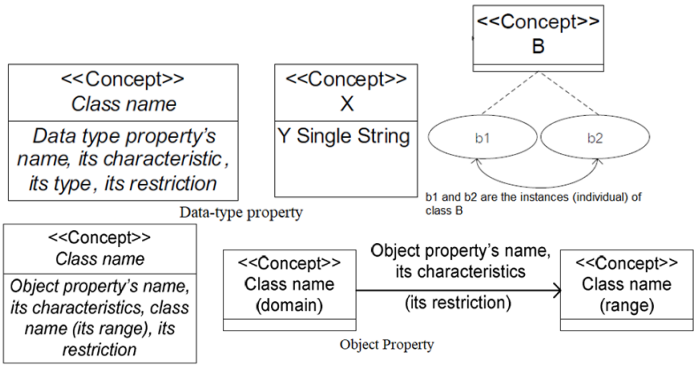

Figure 3. Ontology Semantics

trust value. These are computed based on recommendation generated by other nodes or evolving trust value based on interactions. A device is recommended by a recommender device to a requesting device when the requesting device wants to develop a trustful relation with the unknown device, which is known to the recommender device. The dynamic trust computation uses an initial trust level gained by a device or obtained from a recommender along with keeping track of all the interactions that are happening in between the devices. Finally, based on their honest number of interactions, it calculates the trustworthiness for individual devices. Trustworthiness calculation is a part of trust assessment. It is an assessment that uses the number of direct interactions that are taking place between two nodes. Based on their honest and dishonest behavior it calculates the trustworthiness factor for both devices. This is further explained in section VI.

The most relevant research work related to the computation of digital trust can be found in [4], [20]. These approaches model trust using relational axioms to model. But it fails to represent techniques to compute digital trust for IoT devices in a dynamic environment. In this research we propose an approach to compute dynamic trust as the interactions evolve.

IV. Ontology For Digital Trust

Metaphysics and the philosophical sciences are the origins of "Ontology." [7] "Ontology is considered as an explicit representation of a conceptualization," where the knowledge of a domain can be represented using relationships to describe specific context-oriented information. Ontology uses the term **"concept"** instead of **"object"** to represent relationships between conceptual classes and individuals. The ontology represents a shared understanding of domain knowledge and uses the concept classes to describe the domain knowledge and the relationships among them [15]. Ontology follows notation as mentioned in Fig. 2. Semantics of conceptual classes, instances of concepts (individuals), data and object property of ontology are shown in Fig. 3.

A. Proposed Trust Ontology

The trust ontology is based on the two-tier architecture described in section III to solve trust issues between the devices. The ontology consists of concept classes. All of the concept classes inherit the attributes mentioned in the first tier of the two-layer trust architecture. The ontology is divided into different concept classes to appropriately represent the knowledge domain. Object properties and data properties are being used to represent the IoT trust domain [19]. Object properties express the relations of a base concept class with its derived instances. The basic framework of the ontology is mentioned below: **O = (C, OP, DP)**

O = Name of ontology = Trust Ontology for IoT Devices.

C = OWL classes(concepts)= (Device_Attributes, Device_Certificate, IoT device, Trust_Mechanism).

OP = OWL object property = (Has_Attributes, Has_Certificate, Has_Trust).

DP = OWL datatype property = (Certificate_DateofIssue. Certificate_IssuerName , Certificate_IssuerSign , Certificate_PublicKey , Certificate_SerialNo, DeviceUID, FirmwareVendor, FirmwareVersion, HardwareVersion, HardwareVendor, MacAdd, Manufaturer, Name, SerialNumber, TrustLevel). Fig. 4 shows all the classes(C) and Fig. 5 shows the data property.

The ontology was built using a tool called Protègè [14].

B. Reasoning on Ontology in Semantic Web Rule Language (SWRL)

The design of the proposed ontology is validated using SWRL a semantic web rule language. The focus of the validation was to check if the results in regards to generation of trust levels met the requirements as shown in *Proposition*

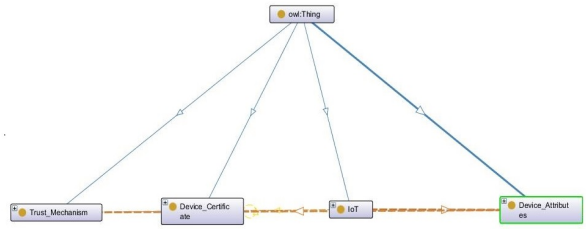

Figure 4. Ontology Graph of the Parent Classes in Protègè

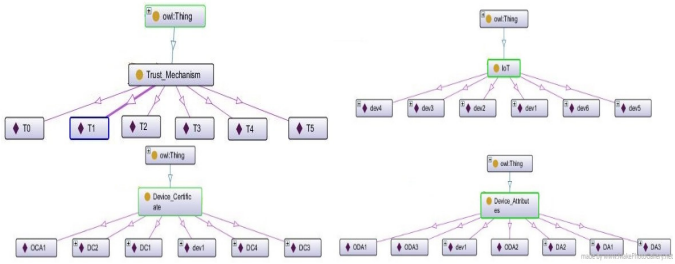

Figure 5. Ontology Data Property in Protègè

1. SWRL is a semantic language that allows representation of rules in a semantic way using propositional logic. It uses first-order predicate logic rules to combine the OWL knowledge base. The syntax of SWRL rule includes antecedent, consequent, and axioms to form a semantic rule.
Antecedent [axiom ⋀ axiom] ⇒ Consequent [axiom ⋀ axiom]
Pellet reasoner [18] was integrated within the Protègè environment to evaluate the outcomes of the SWRL rules. In the process of this reasoning the created ontology concepts (classes), data and object properties are used by Pellet. Thus, validating if the designed ontology supports the generation of trust level among the IoT devices. Finally, the reasoner assigns Trust level based on the satisfaction of the SWRL rule.
A SWRL rule and its implementation in protègè. is shown as *Proposition 1 Instance* .

Proposition 1: Satisfaction of Mandatory Device Attributes (MDA) AND Mandatory Certificate Attribute (MCA) ⇒ Trust Level 5

Proposition 1 Instance IoT(dev1) ⋀ Has_Certificate(dev1, DC1) ⋀ Has_Attributes(dev1, DA1) ⋀ Has_Attributes(dev1, DA2) ⋀ Has_Attributes(dev1, DA3) ⋀ Has_Certificate(dev1, DC2) ⋀ Has_Certificate(dev1, DC3) ⋀ Has_Certificate(dev1, DC4) ⇒ Has_Trust(dev1, T5)

Figure 6 shows the list of attributes associated with IoT device (Dev1) as a result of which the trust mechanism assigns Dev1 a trust level of 5. Reasoning about trust level is done based on the SWRL query (*Proposition 1 Instance*) In total there were 5 propositions created each for a trust level. There were 55 such *proposition instances* created in SWRL to test all the potential scenarios.

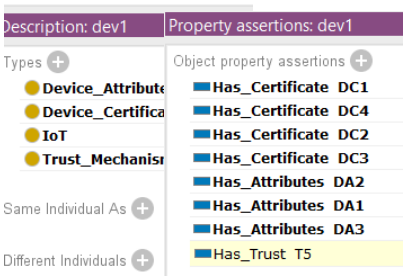

Figure 6. Reasoner output shows Trust Level 5 is generated

V. Ontology to Java programming language mapping

Once the requirements are validated we translate the knowledge expressed in the ontology into a Java implementation. The ontology has a repository that consists of various relationships with its elements. It could be a challenging task for a novice to work with all the functionality of ontology; also it is complicated to implement condition-based reasoning. To overcome such challenges, we mapped the ontology into Java. This conversion also helps to implement a dynamic environment and allows us to implement constraint oriented functionality. Here we have considered the Java programming language as an Object-Oriented Environment (OOE) which supports all the higher-abstraction functionality of OOPS (Object- Oriented Programming Paradigm). The mapping process helps to manipulate data stored in an ontology object into Java objects. Our approach shows how requirements can be modeled and validated before the actual implementation.

Table I shows all the constructs that are mapped into the Java implementation from the ontology. The mapping process considers the concepts, relations, data and object properties that map to Java. This is very similar to the process discussed in [8]. But, we have extended it with the mapping of the SWRL rules into the Java program.

Table I
Construct comparison between ontology and Java

Constructs in Ontology	Constructs in Java
Concepts classes	Interface and concrete classes
Data properties	Variables
Class range	Data types
Class domain	Access specifier
Object properties	Behavioral methods (Get / Set methods)
Individual	Instance of concrete class (Object)

The class properties represented within the ontology is as shown below:
Device_Attributes = (DeviceName, SerialNum, DeviceID, Mac-Add, FirmwareVersion, FirmwareName, HardwareInfo)
Device_Certificate_Attributes = (DateofIssue, IssuerSign, SerialNum, PublicKey, IssuerName)
The mapping of class properties related to device attributes and device certificate attributes are shown in Table II and Table III respectively. Algorithm 1 is the condition mapping pseudo algorithm that aids the rule transformation process from SWRL to Java.
A logical condition consists of all the data properties and object properties mentioned within **Device_Attributes** and

Table II
DEVICE ATTRIBUTE MAPPING

OWL class properties	Java class properties
ODA1= Name of device(str)	setName // getName
ODA2= Serial number (Int)	setSerialNum // getSerialNum
ODA3= Hardware info (TPM)(str)	setHardwareInfo // getHardwareInfo
DA1= Device ID (Int)	setDeviceID // getDeviceID
DA2= Mac-Add (Int)	setMacAdd // getMacAdd
DA3= Version of OS (Doule)	setFirmwareVersion // getFirmwareVersion
DA3= Name of the OS (str)	setFirmwareName // getFirmwareName

Table III
DEVICE CERTIFICATE ATTRIBUTE MAPPING

OWL class properties	Java class properties
ODA1= Certificate Issuername	setIssuerName // getIssuerName
DC1= Valid DateofIssue of cert.	setDateofIssue //getDateofIssue
DC2= IssuerSignature on Digital cert.	setIssuerSign //getIssuerSign
DC3= SerialNumber of cert.	setSerialNum //getSerialNum
DC4= Public Key of device for encryption	setPublicKey //getPublicKey

Device_Certificate_Attributes as the base attributes. Dev1 [Device 1] object extends the functionality of the base attributes.
"**IF** (dev1.**getDeviceID**()!= 0 AND dev1.**getMacadd**()!= null AND dev1.**getFirmNum**()!= 0 AND dev1.**getFirmVendor**()!= null AND dev1.**getDateOfIssue**()!= null AND dev1.**getValidity**()== true AND dev1.**getSignAlgo**()!= null AND dev1.**getSerialNumber**()!= 0 AND dev1.**getPublicKey**()!= null)
{**T1 = 5; System.out.println("Digital Trust level is 5"**}; "

Here we have illustrated the translation of one among the fifty five *proposition instances* that were considered to evaluate the potential scenarios in the trust assessment process.

VI. JAVA-BASED REASONER

This section covers the computation process of trustworthiness and recommendation for participating nodes that are present in the distributed network which is level two of our architecture. According to a work proposed in [3], calculation of trust is an event-driven process where factors such as honesty, cooperativeness, and malicious intent are considered for the calculation of trustworthiness and recommendations for the nodes. Here trust for all the nodes is calculated based on the direct interactions that a node has with one other node.

SWRL to Java relational rule mapping;
if *Proposition symbol is* $\rightarrow$ **then**
  Map it to ==
end
if *Proposition symbol is* $\bigwedge$ **then**
  Map it to &&
end
if *The relational is Has_Certificate* **then**
  Map ObjectName.SetCertificateAttribute
  Map ObjectName.GetCertificateAttribute
end
if *The relational is Has_Attributes* **then**
  Map ObjectName.SetAttributeName()
  Map ObjectName.GetAttributeName()
end

Algorithm 1: Mapping Algorithm for relation mapping from SWRL to Java

A. Dynamic Trust Computation Method

The process of initial trust level assignment was achieved by modeling the requirements in ontology followed by the successful execution of the mapping algorithm and the reasoning checks mentioned in section IV and V. Finally, both the static and dynamic attributes for trust assessment is implemented in the Java based reasoner. So, to assign an initial trust level to an IoT node, we have defined and implemented the logical Java rules that are used by the reasoner to evaluate attribute configuration of the device and then assign an initial trust level value to a node based on the satisfaction of the rules. To accurately measure trustworthiness, it becomes evident that behavioral check of a device is mandatory as the device interacts and performs its tasks in the network. A dynamic environment is implemented to keep track of the different interactions. Here we have classified interactions of a node with other nodes into four different categories. These categories are honest interactions with legal request, partially honest with legal request, dishonest interaction for legal request and dishonest interaction for illegal request. The dishonest interactions are considered as malicious interactions. Number of honest interactions are used to calculate a trustworthiness index. The recommendation calculation depends on the trustworthiness index. The recommendation is useful for a node when it asks for an opinion about an unknown entity to ease its selection process of a node.
The steps below illustrate the working of the trust level assignment process implemented in the Java reasoner. It is a completely user interactive process where attributes get assigned to participating nodes based on the choices made by the user.

1) Assigning a list of attributes to a device: An user is asked to enter different sets of attributes for an IoT device which includes device attributes and device digital certificate information. Here the user is permitted to enter any combination of attributes which is stored in an object Dev1 for further use.

2) Initial trust level assessment for device: Based on the different combinations of the ODA (Optional Data attribute), the MDA (Mandatory Data Attribute), the OCA (Optional Certificate Attribute), the MCA (Mandatory Certificate Attribute) , different devices fall under different categories of trust level. As shown in Figure 7 the initial input to the personal trust assessment are all the attributes obtained for each of the devices. The output is the trust level which is 4 for Device 1, 3 for Device 2 and 5 for Device 3.
After the completion of the initial trust level assessment, a device

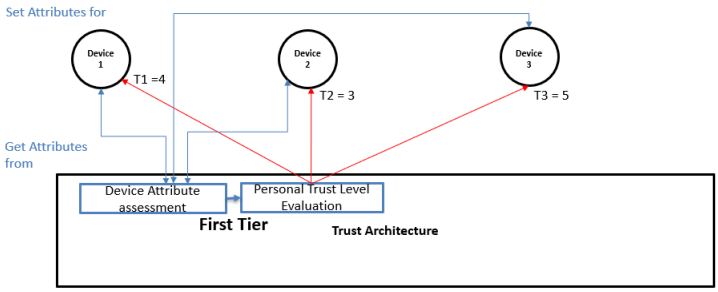

Figure 7. Initial Trust Level Assessment

begins interaction with neighboring devices with an assigned personal trust value. To create a dynamic virtual environment for the interaction, here we have used a built-in functionality of Java to implement the dynamic reasoner.

3) Implementation of a dynamic environment: A dynamic environment helps to observe the behavior of the nodes as the situation evolves. Based on their performance during interactions, the

calculations of trustworthiness and recommendation for individual nodes are being computed. To implement a dynamic environment here, we have used a random number generator function, which generates random numbers that we have considered as random trust levels, the pseudo code is as shown in Algorithm 2. The range for generating numbers is between [1 = Min, 5 =Max] in our scenario. When the two nodes begin their interaction with each other, one node acts as a trusting node (Device 2) and another node acts as a trusted node (Device 1), where trusting node (Device 2) makes a request to get information from the trusted node (Device 1). A request is made by the trusting node to a trusted node to get the device attribute information. **Each request and response are categorized in four categories based on their legitimacy and honesty.**

for *NumberOfInteractionsTimes* **do**
  Random Trust_Level will get generated for a Device Dev1 , Dev2, Dev3 using RandomTrustLevel function
  T1 = Actual personal trustlevel of Dev1 computed in step1.
  T2 = Actual personal trustlevel of Dev2 computed in step1
  T3 = Actual personal trustlevel of Dev3 computed in step1
  Range for random trust generation is 1 = Min value and 5 = Max value
  R1= RandomTrustLevel (1, 5);
  R2= RandomTrustLevel (1, 5);
  R3= RandomTrustLevel (1, 5);
  stores generated random Trustlevel value in Dev1 ,Dev2,Dev3 variable
  Dev1 = R1;
  Dev2 = R2;
  Dev3 = R3;
end

Algorithm 2: Random trust generation process

4) Evaluation of Interactions: Before we initiate our explanation about dynamically changing behavior and interactions in a dynamic environment, we need to introduce two important concepts called Honesty and Legitimate Request. These concepts provide a foundation for the dynamic analysis. **Here *i* and *j* identifies a device based on a number associated with the device represented as integers.**

Important conditional notation can be interpreted as follows:
Dev***i*** **HL** Dev***j*** = Dev***i*** and Dev***j*** both are being honest (H) with each other and Dev***j*** is making a legitimate (L) request.
Dev***i*** **DL** Dev***j*** = Dev***i*** is being dishonest (D) to respond to a legitimate (L) request made by Dev***j***.
Dev***i*** **DI** Dev***j*** = Dev***i*** and Dev***j*** both are being dishonest (D) with each other and Dev***j*** is making an illegitimate (I) request.
Dev***i*** **PL** Dev***j*** = Dev***i*** is being partially honest (P) to respond to a legitimate (L) request made by Dev***j***.

5) Trustworthiness calculation : To compute the trustworthiness factor of individual nodes towards each other based on their behavior during an interaction , we have used the following equation:
To compute **Y's Trustworthiness towards X = [X [HL] Y + *W* { X [PL] Y }] / [*Total number of interactions*]* 100**
Where **X, Y = denotes Device 1, Device 2, Device 3**
W= Weighting factor *(0 < W < 1)*
The meaning of "Y's Trustworthiness towards X" is : It is based on the number of honest, legitimate and partially honest interactions that occurred between device X and Y. Based on the responses sent by device x, device Y evaluate its trustworthiness, where, ***W*** is a weight factor, which can be computed based on the numbers of partially honest interaction. Higher occurrence of partially honest interactions generate lower value of weight factor and vice-versa . It is useful to compute precise trustworthiness of a device based on their request and responses.

6) Recommendation calculation:: In the recommendation process, the recommender node is being selected based on its highest number of honest interactions with others. So, whoever has the highest number of honest interactions with all the nodes is considered as the honest recommender node. This recommendation calculation uses the following equation: X **[HL]** Y + X **[PL]** Y interactions where X, Y = Dev1, Dev2, or Dev3. The highest value obtained after computing these two equations gives a recommender node. After the successful completion of the recommendation computation process, a recommender node is selected based on its highest honesty with

for *Device [i] and Device [j] is* **do**
  if *Actual personal trust level of device is* $Ti = Ri$ *which is a random trust level of device* **then**
    Device *i* is considered as Honest [H].
  end
  if *Actual personal trust level of device is* $Ti > Ri$ *which is a random trust level of device* **then**
    Device *i* is considered as Partial-honest [P].
  end
  if *Actual personal trust level of device is* $Ti < Ri$ *which is a random trust level of device* **then**
    Device *i* is considered as Dishonest [D].
  end
end

When a device ***i*** acts as a trusting agent making a request to get device information from a device ***j*** trusted agent, it follows the given condition.

for *When device [i] makes a request to device [j]* **do**
  if $Ri \leq Tj$ *And* $Ri \leq Ti$ **then**
    Request made by Device *i* to j is considered as Legitimate [L].
  end
  if $Ri \geq Tj$ *And* $Ri \geq Ti$ **then**
    Request made by Device *i* to j is considered as ill-legitimate [I].
  end
end

Algorithm 3: Classifying interactions among IoT devices

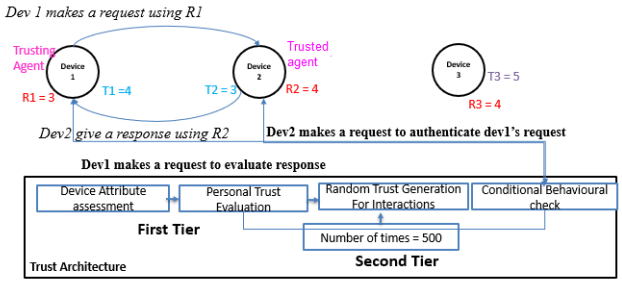

Figure 8. Request and response categorization between devices

other nodes during the interactions. A recommender node can help nodes that ask for a recommendation about the unknown entity that are known to the recommender. In such a situation, the recommender node has an ability to pass its opinion about the unknown node in term of trustworthiness values to a recommendation requester node. It helps the recommendation requester node in its selection of an honest node. It also helps it to establish a trustful relationship.
Figure 8 shows the process of trusted relationship establishment. This process includes trusting agent(Device 1) and trusted agent (Device 2). Each agent is assigned an individual trust level (T) and a random trust level(R). The process is divided into a four step. In the first step, trusting node [Device 1] makes a request using its R value to a trusted node [Device 2]. Trusted agent makes a request to conditional block, to check and verify the request. In second step, it will check the legitimacy of a request. In third step, trusted agent makes a response to a trusting agent. Trusting agent makes a request to conditional block check to verify the response. In the final step, based on the conditional check mentioned earlier for responses, it will check the honesty of a response. After successful execution of these steps, each relationship will be denoted with the appropriate notation mentioned above to categorize an interaction.

VII. Future work and Conclusion

In conclusion, we have shown a formal way of implementing trust for IoT devices by first designing an architecture to broadly classify the levels of computation we need to perform. Then, we represented and validated the trust requirements in an Ontology. Knowledge represented in ontology helped us to design semantics of trust reasoning rules. Finally, we translated the requirements and rules to develop the automated reasoner to generate both static and dynamic trust. The reasoner is capable to categorise interactions among IoT device based on the type of request and responses sent such as honest request gets legitimate response and dishonest request gets illegitimate response. Each response helps reasoner to compute device trustworthiness. In future work, to increase the camaraderie between devices in a distributed environment, we are planning to include a list of shared attributes into the third tier of trust architecture. Based on their common attribute, it will assign a level of friendliness between nodes. This way shared attributes will be helpful to establish trusted relationship. It can also be helpful to enhance trustworthiness factor and individual trust level for devices. One of the other issues that need to be addressed is scalability with increasing number of IoT devices. This can be accomplished by decomposing the interaction between IoT devices depending upon the objective.

Acknowledgment

Dr. William Allen's (Associate Professor, Computer and Engineering Sciences, Florida Institute of Technology) support is much appreciated in conducting this research.

References

[1] Alfarez Abdul-Rahman and Stephen Hailes. A distributed trust model. In *NSPW*, 1996.

[2] Amazon. amazon x.509 certificate for iot platform. https://docs.aws.amazon.com/iot/latest/developerguide/x509-certs.html.

[3] Fenye Bao and Ing-Ray Chen. Trust management for the internet of things and its application to service composition. *2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1–6, 2012.

[4] Juan Chen, Zhihong Tian, Xiang Cui, Lihua Yin, and Xianzhi Wang. Trust architecture and reputation evaluation for internet of things. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–9, 2018.

[5] Filip Forsby, Martin Furuhed, Panos Papadimitratos, and Shahid Raza. Lightweight x. 509 digital certificates for the internet of things. In *Interoperability, Safety and Security in IoT*, pages 123–133. Springer, 2017.

[6] Tyrone Grandison and Morris Sloman. A survey of trust in internet applications. *IEEE Communications Surveys & Tutorials*, 3(4):2–16, 2000.

[7] Maja Hadzic, Elizabeth Chang, Tharam Dillon, and Pornpit Wongthongtham. *Ontology-based multi-agent systems*. Springer, 2009.

[8] Aditya Kalyanpur, Daniel Jiménez Pastor, Steve Battle, and Julian Padget. Automatic mapping of owl ontologies into java. In *SEKE*, 2004.

[9] Anil Kini and Joobin Choobineh. Trust in electronic commerce: Definition and theoretical considerations. In *HICSS*, 1998.

[10] Konstantinos Kotis, Iraklis Athanasakis, and George A Vouros. Semantically enabling iot trust to ensure and secure deployment of iot entities. *International Journal of Internet of Things and Cyber-Assurance*, 1(1):3–21, 2018.

[11] Xin Li, Joseph S Valacich, and Traci J Hess. Predicting user trust in information systems: A comparison of competing trust models. In *37th Annual Hawaii International Conference on System Sciences, 2004. Proceedings of the*, pages 10–pp. IEEE, 2004.

[12] Stephen Paul Marsh. Formalising trust as a computational concept. Technical report, 1994.

[13] Patel Milankumar. Formal trust architecture for assuring trusted interactions in the internet of things. page 151. Florida Institute of Technology,Melbourne,FL, 2019.

[14] M.A. Musen. The protégé project: A look back and a look forward. *AI Matters, Association of Computing Machinery Specific Interest Group in Artificial Intelligence*, 2015.

[15] Natalya F. Noy and Deborah L. Mcguinness. Ontology development 101: A guide to creating your first ontology. Technical report, 2001.

[16] Arbia Riahi, Yacine Challal, Enrico Natalizio, Zied Chtourou, and Abdelmadjid Bouabdallah. A systemic approach for iot security. In *2013 IEEE international conference on distributed computing in sensor systems*, pages 351–355. IEEE, 2013.

[17] Yefeng Ruan, Arjan Durresi, and Lina Alfantoukh. Trust management framework for internet of things. *2016 IEEE 30th International Conference on Advanced Information Networking and Applications (AINA)*, pages 1013–1019, 2016.

[18] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A practical owl-dl reasoner. *Web Semantics: science, services and agents on the World Wide Web*, 5(2):51–53, 2007.

[19] Mohsen Taherian, Rasool Jalili, and Morteza Amini. Pto: A trust ontology for pervasive environments. In *in Proc. of IEEE International Conference on Advanced Information Networking and Applications*, pages 301–306, 2008.

[20] George Theodorakopoulos and John S Baras. On trust models and trust evaluation metrics for ad hoc networks. *IEEE Journal on selected areas in Communications*, 24(2):318–328, 2006.

[21] Lea Viljanen. Towards an ontology of trust. In *International Conference on Trust, Privacy and Security in Digital Business*, pages 175–184. Springer, 2005.

[22] Zheng Yan and Silke Holtmanns. Trust modeling and management: from social trust to digital trust. In *Computer security, privacy and politics: current issues, challenges and solutions*, pages 290–323. IGI Global, 2008.

[23] Zheng Yan and Ronan MacLaverty. Autonomic trust management in a component based software system. In *International Conference on Autonomic and Trusted Computing*, pages 279–292. Springer, 2006.