On Automated Trust Computation in IoT with Multiple Attributes and Subjective Logic

Nuray Baltaci Akhuseyinoglu[1], Maryam Karimi[1], Mai Abdelhakim[2], and Prashant Krishnamurthy[1]
[1]Department of Informatics and Networked Systems, [2] Department of Electrical and Computer Engineering,
University of Pittsburgh, Pittsburgh, USA
{nub2, mak322, maia, prashk}@pitt.edu

***Abstract*—Developing automated trust mechanisms has become crucial for overcoming perceptions of uncertainty and risk by people using IoT services. Things are increasingly communicating with each other and trust in the data they deliver depends on several factors such as the links they use to communicate and the environment. This points to a need for a trust management method for "things" that considers the communication among them, environmental and security-related factors, and the network topology but without human intervention. To address these challenges, we propose a trust management framework that automatically computes the trust of "things". We use Multi-Attribute Decision Making (MADM) and Evidence-Based Subjective Logic (EBSL) in a trust network of "things" to take into account the uncertainty in trust values. We propose new normalization for non-monotonic attributes in MADM. We present an algorithm for automatic trust computation and evaluate its effectiveness using synthetic data and sampling from real datasets.**

***Index Terms*—IoT, trust management, multi-attribute decision making (MADM), subjective logic**

I. Introduction

With the vast increase in the number of "things" and physical objects that are equipped with sensing, actuating, storage, computation, and connectivity, the *trust* in the devices, the data they report, and the actions they take will have to be *automatically* determined. This is because human intervention is expensive, and many transactions occur primarily among things with support from computation and storage in the cloud. The potential for malicious activity (e.g., node compromise, man-in-the-middle of a link) adds to the importance of computing trust automatically. Even when things behave benignly, they may have inherent environmental limitations and variations in behavior (inexpensive vs robust to wear and tear) that impact trust in the reported information. Trust mechanisms are fundamental for people to overcome perceptions of uncertainty and risk in using IoT services and applications [7]. Further, when critical events occur, assessing trust in the things and their data is crucial for informed decision making or forensics of the event [32]. There are several challenges in determining trust in IoT. If the trust in reported measurements is a single number, it needs to capture a variety of potential conditions, including contextual/environmental conditions or technical/protocol-dependent features (key freshness, communication link characteristics). We ask: *how can we determine a temporal trust in things based on remote knowledge of the network topology, inherent limitations of sensors, generated data, and potential misuse of things or their connections, without continual human intervention?* The problem becomes more challenging when things belong to different administrative domains with diverse ownership.

There are *inherent* trust issues in the things sensing the environment due to several limitations. For instance, some sensors may operate correctly under some environmental conditions, but the trust in the data they sense may degrade under others. The characteristics of nodes may not, however, be easily quantifiable since the trust may not *monotonically* change with the characteristic. There are technical aspects of nodes that influence their trust. A thing whose keys used for cryptographic protocols have remained unchanged for a long time is more vulnerable to compromise than a thing that has its keys refreshed frequently. We consider both of these types as impacting what we call the *functional trust* (FT) of a thing. Trust needs to take into account the *beliefs* of things on each other (especially neighbors) and the belief of the cloud infrastructure on things based on topology, the delivered data, and even potential external data sources. We call this *referral trust* (RT). For example, a thing may reduce its trust in a neighbor if both sense the same physical phenomenon, e.g. the temperature, and there are significant differences. The cloud may adjust its trust on a node sensing temperature if the values are very dissimilar to the coarse-grained values reported by an external trusted source.

Major limitations in existing trust management systems include (i) lack of an automated approach for trust evaluation, for example due to their reliance on user feedback, such as in [4], [27], (ii) ignoring critical contextual and technical/protocol-dependent attributes (such as key freshness and communication link types) [11], [4], [2], [17], [16], [27], and (iii) adding computational burden on resource constrained IoT devices [28], [4], [17]. There have been trust management frameworks proposed for IoT [4], [27], [23], [17], wireless sensor networks [11], [16], and mobile ad hoc networks [28]. Most of the existing trust frameworks ([2], [20], [10], [4], [27], [23], [28]) either consider a social IoT (SIoT) paradigm based on social network theory and social relationships among things which makes them human dependent, or require feedback from a human user for trust quantification. This renders the automation of the trust computation difficult.

In this paper, we address the above challenges by developing a trust framework that captures FT and RT, both with multiple attributes (environmental aspects like time, location, temperature, and technical aspects like key freshness, link type). Our framework does not require trust computation on resource-constrained sensor nodes, but it offloads computations to resource-abundant cloud servers (which we assume have knowledge of the IoT topology and context). Moreover, it does not hinge on a feedback/recommendation mechanism that requires human intervention but automates computations based on collected observations by things. To account for multiple attributes, we employ Multi-Attribute Decision Making (MADM) [22]. To account for the uncertainty in trust values, we adopt the Evidence-Based Subjective Logic (EBSL) [25], which we also use to combine FT and RT values in a trust network (TN). We validate the performance of the proposed framework and trust evaluation mechanisms using simulations and real data. The results show that the trust framework can effectively capture nodes' behavior and limitations. In particular, the contributions of this paper are as follows:

- We propose a trust framework for IoT that utilizes EBSL for trust computations. Our trust computations account for FT and RT and uncertainty in trust evaluations.
- We propose a method for FT and RT computations, which is built upon the principles of MADM. It captures contextual as well as technical and protocol-dependent attributes. As some of the attributes are non-monotonic, we propose a normalization technique for such attributes. The method also combines real-time and historical values of both FT and RT.
- We present an algorithm for trust computations and evaluate the performance of our framework using synthetic data and sampling from real datasets.

II. Background

A. MADM

Decision making (DM) is the process of modeling a decision maker's preference on a set of competing *alternatives (a.k.a options)* concerning a set of *criteria*, which are usually in conflict. When alternatives are evaluated based on more than one criterion, the problem is referred to as Multi-Criteria Decision Making (MCDM) [15]. There are two classes of MCDM methods: MADM and Multi-Objective Decision Making (MODM) [12]. The former deals with a finite set of alternatives, while the latter is suitable for continuous DM problems where there is an infinite number of alternatives [15]. In our trust framework, we use the MADM approach since we have a finite set of alternatives and well-defined attributes. We present a detailed definition of our DM problem for trust computation in Section IV-B1. Note that there is no clear distinction between *criteria* and *attributes* in MCDM literature, so we use them interchangeably.

Concerning attribute information processing, MADM approaches are classified as compensatory and non-compensatory methods[13]. We adopt a compensatory method in our DM problem, which, unlike the non-compensatory methods, allows a trade-off between attributes [30]. In particular, we utilize the Simple Additive Weighting (SAW) as a widely accepted compensatory scoring model [30] in the proposed trust framework, which ranks alternatives and selects the ones with the highest score. It is one of the simplest models but produces results that are very close to more sophisticated DM methods [13], [19].

Attributes in MADM can be grouped under three categories based on the utility gained with respect to the attribute value: *benefit*, *cost*, and *non-monotonic* attributes [30]. Benefit attributes provide increasing monotonic utility, meaning that the higher attribute values are preferable. Cost attributes provide decreasing monotonic utility, so lower attribute values are preferable. Non-monotonic attributes do not have a monotonic utility change with respect to their values. The most preferred value of a non-monotonic attribute falls into a point between the range of the attribute values. Examples of these attributes are efficiency, cost, and room temperature, respectively [15]. Attributes can also be categorized into two classes as ordinal and cardinal attributes [15]. Ordinal attributes do not have a numerical scale such as ratio or interval, so their values are compared by their rank orders such as very high, average, low, etc. Cardinal attributes can have an interval or ratio scale and can be assigned with numerical values.

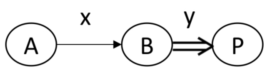


Fig. 1: A simple trust network consisting of two nodes

B. EBSL

Subjective Logic (SL) [14] is one of the prominent trust models. It considers the level of uncertainty in trust information. In SL, the trust of a node is represented by an opinion triplet $\omega = \{b, d, u\}$, where b, d, and u respectively represent the belief, disbelief, and uncertainty of trust information and $\forall b, d, u \in [0, 1]$, $b + d + u = 1$. For explaining the basic concepts, we present a toy example of a TN in Figure 1. In this TN, A and B represent two IoT devices: *thing A* and *thing B*. P is a proposition about which B forms an opinion. In our case, P is *"a data item d_t that is collected at time t has the value $v(d_t)$"*. In the figure, x represents the opinion that A has about the trustworthiness of B in providing recommendations, which is known as *referral trust (RT)* in SL. y represents the opinion that B has about the trustworthiness of the data items it measures, which corresponds to *functional trust (FT)* in SL. Opinions on the path from a node to another node form a *trust chain*. In a valid trust chain, there must be an FT on the last edge in addition to RTs on previous edges [25].

SL provides operators for combining opinions. Here we utilize *consensus operator* and *discounting operator*. In a TN, the trust of a target node is computed by aggregating trust opinions of paths (trust chains) between the target node and a source node. This operation is referred to as *aggregation of trust* and performed using *consensus operator* ($\oplus$):

$$x \oplus y = \frac{(x_u y_b + y_u x_b, x_u y_d + y_u x_d, x_u y_u)}{x_u + y_u - x_u y_u} \tag{1}$$

where x, y are two opinion triplets, x_b, x_d, x_u, y_b, y_d, y_u denote belief, disbelief, and uncertainty components of opinion x and y, respectively.

The trust opinion of a chain is updated every time each node is visited along the path. This update operation is referred to as the *propagation of referral trust* and performed using the *discounting operator* ($\otimes$). *Opinion discounting* represents the flow of information from one node to another in a TN. EBSL extends SL with a new *opinion discounting* operation ($\boxtimes$) as Škorić et al. [25] identify a number of basic problems of the discounting operation in SL. Namely, it has a *double-counting* problem which requires converting a TN into a "canonical form". Adopting this proposed discounting operation results in a consistent SL algebra as it resolves the double-counting problem of the original operator. We use EBSL [25] to compute the trust opinion for a measurement provided by an IoT device in a TN. The discounting operator of EBSL ($\boxtimes$) is expressed as follows:

$$x \boxtimes y = g(x).y = \frac{(g(x)y_b, g(x)y_d, y_u)}{(y_b + y_d)g(x) + y_u} \tag{2}$$

where x, y are two opinion triplets, y_b, y_d, y_u are belief, disbelief, and uncertainty components of opinion y, and $g(x) \geq 0$ is a scalar which denotes the proportion of evidence that is transferred from a node to another node. $g(x)$ can be selected arbitrarily, and suggested options are x_b or $\sqrt{x_b}$ where x_b is the belief for opinion x [25]. We use $g(x) = x_b$ for our computations.

III. System Model

The main components of our proposed trust management framework are resource-constrained "things" and more sophisticated devices such as a gateway. These components form a TN. The Gateway collects measurements from "things" and sends them to the cloud server quasi-periodically. The transferred data include measurements of interest, such as humidity, wind, water velocity, etc., and the data used for trust computations. We elaborate on the data used for trust computation in Section IV. The transmission of the data between IoT devices and from IoT devices to the cloud server can be realized by using different wireless technologies (see Table VII). Note that secure protocols are used in the proposed system model, so the security of communicated data depends on the deployed protocols and infrastructure.

When the cloud server receives data from the gateway, it records the data items and trust-related measures into the database together with a randomly assigned (but known) ID for each IoT device. Note that IDs are generated by the cloud, so an IoT device cannot imitate this number during the data collection process. From these periodic measurements, the cloud server computes a *real-time functional trust (RFT)* score for a node (referred to as destination node) and a *real-time referral trust (RRT)* score for each pair of nodes (or each edge) in the TN. These real-time scores are then converted into SL opinions by applying a transformation that we explain in Section IV-B3. The cloud server keeps historical records for each IoT device and fuses the RFT of a device with its historical FT (HFT). The result of this fusion is an SL opinion triplet that we use to represent FT (opinion y in Figure 1). Similarly, the RRT of each edge is fused with its historical RT (HRT) to obtain an RT opinion triplet. The *Overall trust (OT)* for a measurement reported by a target IoT device (destination node) is then computed by combining FT and RT values on the path from a source device to the target device in the TN [1].

TABLE I: Notations

Notation	Meaning
MT	$= \{mt_1, mt_2, ..., mt_p\}$: a set of measurement types
IM_{mt}, IM_{AB}	An initial matrix constructed for mt and for the edge between node A and B
D	A decision matrix obtained from IM_{mt}
$A_i =< ID, d_t^{ID}) >$	An alternative in D,which is a pair of ID of a "thing" and a data item d_t^{ID} collected by it at time t
X_j, w_j	An attribute in D and its weight
x_{ij}	Value of attribute j for alternative i
r_{ij}	Normalized value of x_{ij}
TN	Trust network
n_s, n_d	Source and destination nodes in TN
tc	A trust chain in TN
$< n_A, n_B >$	An edge between node A and B
T_{tc}	The trust opinion on a tc
OT_{dest}	Overall trust opinion for n_d
$RFT(d_t^{ID})$	Real-time FT score of d_t^{ID}
$RFD(d_t^{ID})$	Real-time functional distrust score for d_t^{ID}
DI_{ID}	Distrust interval of a device with given ID
RFT_t, HFT_t, HFT_{t-1}	Real-time and historical FT values of a device at time t and t-1, respectively
ω_{P,RF_t}^{ID}	Opinion triplet of a device with given ID on proposition P, representing RFT_t
$\omega_{P,HF_t}^{ID}, \omega_{P,HF_{t-1}}^{ID}$	Opinion triplet of a device with given ID on proposition P, representing HFT_t and HFT_{t-1}, respectively
RRT_t, HRT_t, HRT_{t-1}	Real-time and historical RT value of an edge $< n_A, n_B >$ at time t and t-1, respectively
ω_{B,RR_t}^{A}	Opinion triplet of node A on node B, representing RRT_t
$\omega_{B,HR_t}^{A}, \omega_{B,HR_{t-1}}^{A}$	Opinion triplet of node A on node B, representing $HRT_t$and HRT_{t-1}, respectively

IV. Proposed Trust Computation Mechanism

In this section, we present our proposed trust computation mechanism. Table I provides a list of notations and abbreviations we use. We first explain the overall process of trust computation (Figure 2). Next, we present an algorithm for computing the OT opinion for a measurement reported by a target "thing". In the sequel, we explain how we obtain FT and RT components of the OT. Finally, we explain how these two are combined using the EBSL.

A. Overall Trust Computation Algorithm

An IoT device (Figure 2) can be embedded with multiple sensors, each measuring different *measurement types (mt)*,

[1]The trust computation module may be accessed by end-users through a web application interface to query the trust of IoT devices in the TN.

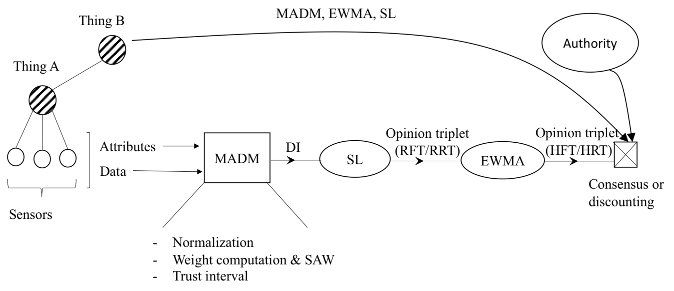


Fig. 2: Overall trust computation process

such as humidity, water velocity, or precipitation. A *measurement, data item,* or *observation*, refers to a single measurement collected by a single sensor of an IoT device. In our trust computation algorithm, we assume that the OT of an IoT device in a TN is queried for a single mt. Hence, we compute OT_{dest} as an opinion triplet for a single IoT device and a single mt. If we would like to let the user query the *trust of an IoT device* for all the mts, we can combine each opinion triplet for each mt using the *consensus* operator of SL[14]. Similarly, there may be multiple IoT devices measuring the same mt. In this case, if a user wants to query the *trust of an* mt, we can combine opinion triplets from the devices providing observations for the same mt.

Using Figure 2 and Algorithm 1 for computing OT_{dest} we explain the process at high-level and present details in Section IV-B and IV-C. The data items and trust-related attributes are collected to measure both FT (sensors embedded to thing A) and RT (between thing A and B). We use these attributes and data as input to the MADM method and obtain RFT and RRT scores. Steps of MADM include the construction of a decision matrix D, normalization of the values in it, computing weights that reflect the relative importance of attributes, and applying SAW on the normalized matrix using the weights. Then, we convert scores from MADM into a distrust interval DI_{ID} that is then used to get SL opinion triplets that represent RFT and RRT. We combine RFTs (or RRTs) using an exponentially weighted moving average (EWMA) method to obtain the HFT of a measurement as an opinion triplet (or HRT opinion for referral trust). Finally, we combine the opinion triplets on the paths from a source node to a destination node for obtaining OT_{dest} in a given TN.

The first step in Algorithm 1 is the initialization of $RFT(d_t^{ID})$. Next, the algorithm constructs a table for real-time data items collected by all the “things” (bigTable in line 2). Then, it finds and groups together d_t^{ID}'s collected for mt by all “things” measuring mt (line 3) to obtain an initial matrix (IM_{mt}) as a representation of the decision making problem (DMP). The mt of interest is given as an input to the algorithm, amongst the ones provided by the destination node n_d. Next, the algorithm constructs D from IM_{mt} (line 4), normalizes attributes in D (lines 5-6), and computes an RFT score for each d_t^{ID} in D (lines 8-11). After this, a DI_{ID} is computed from the list of $RFT(d_t^{ID})$'s for the n_d

Algorithm 1 Compute Trust

```
Input: $TN$, $n_s$, $n_d$, $mt$
Output: $OT_{dest}$
 1: Initialize: $RFT(d_t^{ID}) = 0$
 2: $bigTable \leftarrow$ generateBigTable()
 3: $IM_{mt} \leftarrow$ groupDataItems($bigTable$, $mt$)
 4: $D \leftarrow$ constructD($IM_{mt}$)
 5: for all $X_j \in D$ do
 6:     normalize($X_j$)
 7: end for
 8: for all $d_t^{ID} \in D$ do
 9:     for all $X_j \in D$ do
10:         $RFT(d_t^{ID}) \leftarrow RFT(d_t^{ID}) + w_j \times r_{ij}$
11:     end for
12:     $RFTList(ID) \leftarrow$ add $RFT(d_t^{ID})$
13: end for
14: $DI_{ID} \leftarrow$ computeDI($RFTList(ID)$)
15: $\omega_{P,RF_t}^{ID} \leftarrow$ convertToOpinion($DI_{ID}$)
16: $\omega_{P,HF_t}^{ID} \leftarrow \alpha(\omega_{P,RF_t}^{ID}) + (1-\alpha)(\omega_{P,HF_{t-1}}^{ID})$
17: for all $tc$ in $TN$ do
18:     for all $< n_A, n_B >$ on $tc$ do
19:         $\omega_{B,RR_t}^{A} \leftarrow$ computeRT($n_A$, $n_B$)
20:         $\omega_{B,HR_t}^{A} \leftarrow \alpha(\omega_{B,RR_t}^{A}) + (1-\alpha)(\omega_{B,HR_{t-1}}^{A})$
21:         $T_{tc} \leftarrow T_{tc} \boxtimes (\omega_{B,HR_t}^{A})$
22:         if $n_B = n_d$ then
23:             $T_{tc} = T_{tc} \boxtimes (\omega_{P,HF_t}^{ID})$
24:         end if
25:     end for
26:     $OT_{dest} \leftarrow OT_{dest} \oplus T_{tc}$
27: end for
28: return $OT_{dest}$
```

represented by the ID (line 14). DI_{ID} is then converted to an SL opinion triplet (ω_{P,RF_t}^{ID}) that represents the opinion of n_d on the trustworthiness of its measurements (line 15). Next, real-time and historical opinions for FT are combined using the EWMA method (line 16).

Lines 17-27 are for TN-related operations. First, every edge on every trust chain from n_s to n_d is assigned with an opinion triplet ω_{B,RR_t}^{A} (line 19). Then, similar to RFT, the EWMA method is used for combining the real-time and historical RT opinion for each edge (line 20). The next step is the propagation of referral trust (i.e. HRT_t) using the discounting operator proposed in [25] and updating T_{tc} (line 21). Note that when n_d is reached, a last discounting operation is performed using the FT opinion of n_d (lines 22-24). Finally, T_{tc} of every trust chain is aggregated using the consensus operator[14] (line 26) and OT_{dest} is returned (line 28).

B. Functional Trust

We compute the RFT_t of an IoT device based on the MADM approach. We take all $RFT(d_t^{ID})$ scores of a device, generate a DI_{ID} from them, convert the DI_{ID} into an opinion ω_{P,RF_t}^{ID}, and obtain the opinion reflecting HFT (i.e., ω_{P,HF_t}^{ID}) by combining ω_{P,RF_t}^{ID} and $\omega_{P,HF_{t-1}}^{ID}$. We use ω_{P,HF_t}^{ID} as the

FT label on the last edge of the TN. Next, we describe how MADM is used by considering a "water level sensor" as an example scenario. Note that this is a representative example to illustrate our approach and *not* the actual evaluation.

1) MADM Problem Definition for Trust Computations: We model the derivation of trust of the data collected by IoT devices as a DMP. In our framework, an alternative A_i is a single data item d_t^{ID} collected by a "thing". The set of alternatives is all of the d_t^{ID}s collected by all "things" measuring a specific type of measurement in a predetermined time window, such as d_t^{ID}s from all "water level sensors" for "water level" mt in 2 hours. Next, we describe the criteria we use in MADM formulation.

a) Selected Criteria: Table II shows the criteria we use in MADM for trust computations. We use the first two of them for computing RFT scores and the last four for RRT scores. We explain the first two FT-related criteria here and the remaining RT-related criteria in Section IV-C. Key freshness is inversely proportional to the time elapsed since the last time a cryptographic key was established. As mentioned earlier, trust in sensed data may be affected by several environmental factors. We select temperature among these factors as an example criterion in the MADM formulation of RFT. The indicator is the ambient temperature under which the data item is collected. Here, we assume that temperature is a non-monotonic attribute, which means that its utility function does not show a regular increase or decrease [2].

b) Initial Representation of the Decision Problem: In Table III, we present IM_{mt}, a simplified, yet carefully selected example for a water level sensor as a data source for which we want to compute the RFT. As shown in the table, the A_i's are from a single IoT device (ID=1). There may of course be more than one water level sensor collecting the water level measure, so Table III and the corresponding D may include alternatives from multiple sensors. However, a single IoT device collecting a specific measure is a special case concerning normalization as we explain below. For this example, we suppose there are five data items collected in five-minute intervals. The key was established a day before data is collected (yesterday in the table) and it has been refreshed again (today in the table). The ambient temperature of the sensor remains stable.

c) Construction of the Decision Matrix: We present the D corresponding to the IM_{mt} in Table IV. The task in this step is to transform raw data, i.e., criteria values in IM_{mt}, into trust indicator values in D. The difference between D and IM_{mt} is that D has *trust indicators* of the selected criteria (see Table II) as column labels and indicator values in cells, rather than raw data. The columns labeled as "O" are of interest to this step. They represent the original values of attributes transformed from IM_{mt}. To construct D, we assume that we are looking at the data in Table III today at 12 pm.

d) Normalization of the Decision Matrix: The next step after constructing D is normalizing the values in it. The normalization method is important to get a single trust value and depends on the utility type [24] (see Table II). We present the normalization methods that we use in Table V. We first experimented with normalization methods presented in seminal works of MADM [13], [30]. We discovered that they may lead to several issues in the trust computation – *zero denominator, zero normalized value, identical normalized value,* and *zero variance*. These usually occur when D includes identical values for d_t^{ID}s from a single IoT device[3]. A zero denominator leads to division by zero problems. Zero variance is a similar issue as the variance of attribute values is a denominator of the non-monotonic normalization formula in [13], [30]. A zero normalized value would cause discarding an attribute from MADM formulation. Identical normalized attribute values occur when the values of d_t^{ID}'s are identical in two different D's regardless of the actual attribute values x_{ij}.

The *zero denominator*, *zero normalized value*, and *identical normalized value* are related to linear normalization of monotonic attributes. To address them, we propose to adjust one of the linear normalization formulae proposed in [13], [30] with the *global maximum* and *global minimum* values for an attribute. The *zero variance* issue is related to the normalization of non-monotonic attributes. We address this by proposing a new normalization formula in Table V that adjusts the non-monotonic normalization formula in [13], [30]. The non-monotonic normalization formula in [13], [30] considers a single most favorable value for an attribute. We argue that using a *preferred interval* is a better choice as attributes may not always be represented by a single most favorable value. For example, the ambient temperature attribute may have a preferred interval for a water level sensor where the accuracy is best for sensed data. We can consider the compensated temperature range (see [3]) of a water level sensor as the preferred interval for the temperature attribute. We represent the *preferred interval* of attribute j as $PI_j = [LPV_j, UPV_j]$, where LPV_j and UPV_j are lower and upper preference value of PI_j, respectively. We treat attribute values less than LPV_j as benefit attributes and attribute values greater than UPV_j as cost attributes (like the utility types in Table II). Attribute values within PI_j are taken as the most favorable value and their normalized value is 1. This means that a non-monotonic attribute value yields the highest utility within PI_j. Its utility decreases as it moves away from PI_j.

For this example, we apply our normalization methods on the original values of D given in Table IV and present normalized values in the same table under columns labeled with "N". We note the global maximum, global minimum, and preference values that we use for normalization in the table. We set $PI_j = [32, 70]°F$ for temperature attribute, which is an acceptable range according to [9], [8].

2) Weight Elicitation and Overall Score Calculation: As noted in Section II-A, we use SAW to calculate the overall

[2]For an ultrasonic sensor such as a water level sensor in our scenario, the measurement accuracy shows a non-linear behavior with the temperature[1]

[3]For a detailed explanation of how each issue is relevant to which normalization method and for sample scenarios that they occur, please refer to: https://github.com/nbaltaci/trust-framework-for-IoT

TABLE II: Criteria Used in MADM for Trust Computations

Criteria	Indicator	Unit	Type	Utility type
Time of key establishment	Key freshness[a]	Days	Quantitative	Benefit
Temperature	Ambient temperature of the "thing"	Fahrenheit	Quantitative	Non-monotonic
Time of data collection	Data freshness[a]	Seconds	Quantitative	Benefit
Measurement deviation	Absolute measurement deviation [b]	unit of mt	Quantitative	Cost
Link type	Data reliability due to link type	-	Qualitative	Benefit
Location	Distance between two nodes	Meters	Quantitative	Cost

[a] Formula respectively for key freshness and data freshness are: $1/(T_{key\_est}+1)$ and $1/(T_{data\_col}+1)$, where T_{key_est} and T_{data_col} are time elapsed since last time a cryptographic key was established and a data item was collected, respectively.

[b] Formula for absolute measurement deviation is: $|d_t(A) - d_t(B)|$, $mt_j \in MT$.

TABLE III: IM_{mt} for Water Level Measure

Alternative	Time of key establishment	Time of data collection	Temperature
ID=1 d_1	Yesterday	09:05 am	75
ID=1 d_2	Yesterday	09:10 am	75
ID=1 d_3	Yesterday	09:15 am	75
ID=1 d_4	Now	09:20 am	74
ID=1 d_5	Today	09:25 am	74

TABLE IV: Decision Matrix for RFT Score Computation

Alternative	Key freshness		Temperature		RFT
	O	N	O	N	
ID=1 d_1	0.5000	0.5000	75	0.9460	0.6115
ID=1 d_2	0.5000	0.5000	75	0.9460	0.6115
ID=1 d_3	0.5000	0.5000	75	0.9460	0.6115
ID=1 d_4	1.0000	1.0000	74	0.9651	0.9913
ID=1 d_5	1.0000	1.0000	74	0.9651	0.9913
Glob_min	0		0		
Glob_max	1		100		
LPV_j	-		32		
UPV_j	-		70		

O: original value (x_{ij}), N: normalized value (r_{ij})

trust score of each item for our scenario. Weights assigned to attributes reflect their relative importance and are used to calculate the overall score for alternatives in D. Identifying the weights is known as the weight elicitation process. There are two ways to obtain attribute weights: recruiting human decision-makers who can assign a weight for each attribute or using a weight approximation method. A weight approximation method uses a ranked list of the attributes based on their relative importance and finds the surrogate weights, which are expected to be close to the "true" weights to be assigned by decision-makers [22]. In this paper, we adopt a weight approximation approach. There are three widely used methods, which are Rank Order Centroid (ROC), Rank Sum (RS), and Rank Reciprocal (RR) [26]. We use ROC in our trust computations as it has been shown to be superior to other methods (in terms of deviation from actual weights assigned by real decision-makers) [22]. The formula for calculating a ROC weight of an attribute X_j is: $w_j = 1/n \sum_{k=R_j}^{n} 1/k$, where n and R_j are the total number of attributes and the rank of X_j (the rank of an attribute indicates its importance), respectively[4].

The RFT score of each data item is a weighted sum of normalized attribute values ($RFT(d_t^{ID}) = \sum_{j=1}^{n} w_j r_{ij}$). The last column of Table IV shows the trust score of each data item.

3) Converting Trust Scores into Opinion Triplets: We next convert the scores ($RFT(d_t^{ID})$) obtained from MADM into opinion triplets that reflect FT (ω_{P,RF_t}^{ID}). This is performed in two steps. First, for each IoT device, we convert trust scores into distrust scores and compute a distrust interval (DI_{ID}). Next, we convert DI_{ID} into ω_{P,RF_t}^{ID}. We adopt methods presented in [32] for these conversions. In [32], local distrust values of a data report is converted into a distrust interval. Similarly, we convert *real-time functional distrust* scores ($RFD(d_t^{ID})$) into a $DI_{ID} = [LDB, UDB]$, where LDB and UDB respectively correspond to a lower distrust bound and upper distrust bound and $RFD(d_t^{ID}) = 1 - RFT(d_t^{ID})$. Then, computations are performed as follows:

$$LDB = max\{0, \overline{RFD(d_t^{ID})} - SD_{RFD(d_t^{ID})}/2\} \quad (3)$$

$$UDB = min\{1, \overline{RFD(d_t^{ID})} + SD_{RFD(d_t^{ID})}/2\} \quad (4)$$

where $\overline{RFD(d_t^{ID})}$ and $SD_{RFD(d_t^{ID})}$ represent the mean and the standard deviation of distrust scores, respectively. Next, we compute an opinion triplet from DI_{ID} as follows:

$$\omega_{P,RF_t}^{ID} = (1 - \frac{LDB + UDB}{2} - \frac{UDB - LDB}{2}, \frac{LDB + UDB}{2}, \frac{UDB - LDB}{2}) \quad (5)$$

Note that the proposed method in [32] converts a **distrust interval** into an opinion triplet for **data reliability**. Similarly here, we convert a distrust interval into an opinion that reflects the RFT of an IoT device (ω_{P,RF_t}^{ID}). Using RFT scores in Table IV and (3) and (4), we compute $DI_{ID} = [0.1326, 0.3406]$. For this DI_{ID} value, we obtain $\omega_{P,RF_t}^{ID} = (0.6594, 0.2366, 0.1040)$ using (5).

4) Historic Reputation of an IoT Device: We combine the RFT of an IoT device with its HFT to take into account the *history* of a device (ω_{P,HF_t}^{ID}) in OT computation. This is performed over opinion triplets that represent RFT (ω_{P,RF_t}^{ID}) and HFT ($\omega_{P,HF_{t-1}}^{ID}$). $\omega_{P,HF_{t-1}}^{ID}$ is a cumulative value of opinion triplets generated from DIs of a device at different time points in the past. We use the EWMA method to combine RFT and HFT. This leads to a case where weights of observations decrease exponentially from the recent observations to

[4]We compute weights by applying all the three methods and present them in supplemental material: https://github.com/nbaltaci/trust-framework-for-IoT

TABLE V: Normalization Methods Used to Normalize Criteria

Utility Type	Attribute type	Normalization formula [a, b]	
Monotonic	Benefit	$r_{ij} = \frac{x_{ij}}{x_j^*}$	$x_j^* = glob\_max(x_j)$
	Cost	$r_{ij} = 1 - \frac{x_{ij}}{x_j^*}$	$x_j^* = glob\_max(x_j)$
Non-monotonic	Benefit	$r_{ij} = e^{(-2z^2)}$	$z = \frac{LPV_j - x_{ij}}{LPV_j - glob\_min(x_j)}$; $x_{ij} < LPV_j$
	Cost		$z = \frac{x_{ij} - UPV_j}{glob\_max(x_j) - UPV_j}$; $x_{ij} > UPV_j$
	Preferred values		$z = 0$; $LPV_j \leq x_{ij} \leq UPV_j$

[a] $glob\_max(x_j), glob\_min(x_j)$: Functions returning predefined global maximum and global minimum values of the jth attribute

[b] LPV_j, UPV_j : Lower and upper preference values of the preference interval PI of the jth attribute

earlier ones. As a result, recent trust values have more impact on ω^{ID}_{P,HF_t}. Also, as presented in Algorithm 1, reputation is computed by using the equation below, where $\alpha \in [0, 1]$. Note that $\omega^{ID}_{P,HF_{t=0}} = 0$, so **for** $t = 1$, $\omega^{ID}_{P,HF_{t=1}} = \omega^{ID}_{P,RF_{t=1}}$.

$$\omega^{ID}_{P,HF_t} = \alpha\omega^{ID}_{P,RF_t} + (1-\alpha)\omega^{ID}_{P,HF_{t-1}} \tag{6}$$

We can tune α to weigh historical values differently.

C. Referral Trust

In this section, we explain how we obtain RT (ω^{A}_{B,HR_t}), as labels on the edges of the TN. Like the RFT scores, we compute RRT scores using MADM with the same steps. We convert RRT scores into DI, obtain opinion triplets from DI, and use historical values with EWMA. We do not delve into details of these steps again here, but explain points specific to RRT and present the results of those steps below.

First, the initial matrix IM is constructed for a pair of "things", e.g., thing A and thing B, for RRT computation in the cloud server. An edge in a TN is directed between one-hop neighbor nodes from a source node to a destination node (see Figure 1). We assume that thing A is the source and the trust it has on thing B is to be computed, so we represent the matrix as IM_{AB}. We show the initial representation in Table VI under the columns labeled as "IM". We include *measurement deviation*, *link type*, *location*, and *data freshness* as criteria (see Table II) in the MADM formulation of RRT. Here, measurement deviation corresponds to the deviation between the values of data items reported by two nodes for a specific mt and a time interval t. If two nodes do not report data items with the same mt, then the deviation is set to zero. In our sample scenario, we consider that thing A and thing B both report water level measurements. Hence, the *measurement deviation* refers to the deviation in water level reported by the two "things" with a measurement unit of meters. In the example given in Table VI, measurement deviation changes from 0 to 0.0020 meters after two data exchanges. The *location* criterion corresponds to the geographical coordinates of an IoT device. In Table VI, we arbitrarily represent the location of thing B as (x,y) and assume no change in it. *Link type* refers to the communication protocol that can be used to send data between devices in the IoT network. It covers all types of links among devices, between a pair of IoT devices, IoT device and a gateway or a gateway and a cloud server. The most commonly used communication protocols in IoT are presented in Table VII (such as in a flood early warning system [1] or for device pairing in general [6]). We use the list in Table VII as a set of possible values for link type criterion in our RRT computations. *Data freshness* is the recency of the data sent by an IoT device. The data freshness is a benefit attribute, since we trust information when data is more recent/fresh and trust decreases as data become outdated. It is inversely proportional to the time elapsed after a data item has been collected by a device and until it reaches a destination point. The elapsed time is the sum of propagation, processing, transmission delay, etc.

Second, we construct D from IM_{AB}. We show the values of D in Table VI under the column labeled as "O". D is constructed by converting the raw data of IM using the indicators of attributes presented in Table II. The measurement deviation attribute in D is $|d^B_t - d^A_t|$. For the location[5] criterion, we use the distance between the IoT nodes. Even when the distance between two nodes does not change, it is still a distinguishing factor for RT computation. The link type criterion is represented by a qualitative indicator: *data reliability due to the link type* as seen in Table II. We assume that this indicator is an ordinal variable and takes fuzzy values in Table VII: very high, high, average, low, and very low. This criterion reflects how reliable the link is in carrying the data. To convert the link type to *data reliability due to link type*, we assume that data reliability is affected by the typical coverage of a link. For example, in Table VII, NFC has the shortest range so it has very high reliability. For data freshness criterion, we use a quantitative indicator that is the inverse proportion of the elapsed time, as we explained earlier. It should be emphasized that other factors can be used or included in RT calculations in a similar manner.

Third, we need to normalize D. We present the normalized values in Table VI under the column "N". As seen in Table II, measurement deviation, distance, and data freshness are monotonic quantitative attributes. Hence, these attributes are normalized by using the formula presented in Table V. On the other hand, the link type attribute has a qualitative indicator. We map the values of a qualitative indicator to numerical values using a 10-point Bipolar scale [13]. Since data reliability due to link type is a benefit attribute as seen

[5]We note that it is unlikely that the location of a sensor will change unless it is embedded in a mobile device.

TABLE VI: Decision Matrix for RT Computation

	Measurement deviation			Link type			Location			Data freshness					
										IM			O[a]	N	
Alternative	IM	O	N	IM	O	N	IM	O	N	PD [a]	DD [a]	TE [a]			RRT
ID=B $d1$	0.0000	0.0000	1.0000	Bluetooth	Average	0.5556	(x,y)	100	0.8000	33.3333	881	914.3333	0.0011	0.0011	0.7879
ID=B $d2$	0.0000	0.0000	1.0000	Bluetooth	Average	0.5556	(x,y)	100	0.8000	33.3333	709	742.3333	0.0013	0.0013	0.7879
ID=B $d3$	0.0020	0.0020	0.9600	WiFi	Low	0.3333	(x,y)	100	0.8000	33.3333	218	251.3333	0.0040	0.0040	0.7069
ID=B $d4$	0.0020	0.0020	0.9600	WiFi	Low	0.3333	(x,y)	100	0.8000	33.3333	54	87.33333	0.0113	0.0113	0.7069
ID=B $d5$	0.0020	0.0020	0.9600	WiFi	Low	0.3333	(x,y)	100	0.8000	33.3333	796	829.3333	0.0012	0.0012	0.7069
Glob_min		0			1			0					0		
Glob_max		0.05			9			500					1		

IM:value in IM_{AB}, O: value in decision matrix (x_{ij}), N: normalized value (r_{ij})
PD: propagation delay, DD: data delay, TE: time elapsed since the data item was sent, [a] In units of $10^{-8}s$, [b] In units of $10^{8}s^{-1}$

TABLE VII: Link Types

Link type	Typical Coverage	Data Reliability due to Link Type	Attribute value
Zigbee	10-1000 m	Very low	1
WiFi	30-250 m	Low	3
Bluetooth	1-100 m	Average	5
RFID	10 m	High	7
NFC	10 cm	Very High	9

in Table II, we map the value of very high to a numerical value of 9. Table VII shows the mapping of remaining values for all link types under the column *Attribute value*. Then, we normalize attribute values using the linear normalization formula for monotonic benefit attributes in Table V. Note that we present the global maximum and minimum values used in normalization formulae at the bottom of Table VI.

Finally, we use SAW to obtain overall trust scores. Like the FT computation, we compute approximated weights by using the ROC method. Overall scores are presented in Table VI, under the column “RRT”. In weight computation, we assume the attributes are ranked from the left to the right in Table VI concerning the order of importance in decision making. At the next step, we apply (3) and (4) on these scores and obtain $DI = [0.2385, 0.2829]$. We obtain an opinion triplet $\omega^{A}_{B,RR_t} = (0.7171, 0.2607, 0.0222)$ from the DI by applying (5). Finally, we combine ω^{A}_{B,RR_t} with $\omega^{A}_{B,HR_{t-1}}$ using the EWMA method to obtain ω^{A}_{B,HR_t}. In the next section, we explain how to combine opinions ω^{A}_{B,HF_t} and ω^{A}_{B,HR_t} to compute OT_{dest}.

D. Overall Trust Opinion

As we explain in Section IV-A, the last step of trust computation for finding OT_{dest} is traversing through the trust chains (tc’s) between n_s and n_d in a TN. The trust opinion of a chain (T_{tc}) is updated as each node along the chain is visited. This update operation is performed using the discounting operator ($\boxtimes$) of EBSL. After the visit of each tc is completed, OT_{dest} is updated with the recently computed T_{tc}. This update operation is performed using the consensus operator ($\oplus$) of SL – see (1) and (2) in Section II-B. In our framework, we replace x and y in (1) and (2) with relevant opinion triplets in a TN of “things”. For example, we replace x with ω^{A}_{B,HR_t} and y with ω^{B}_{P,HF_t} for applying $\boxtimes$ on the sample TN in Figure 1. In the supplemental material[4], we present sample trust computations using a sample trust network.

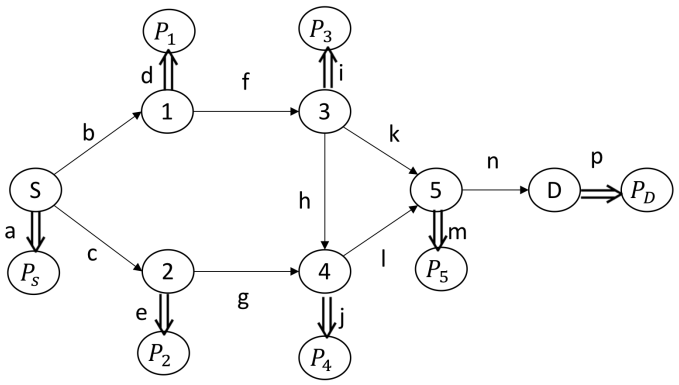


Fig. 3: Sample TN

TABLE VIII: RFT and HFT of Destination Node D

Time	DI_{ID}		ω^{D}_{P,RF_t}			ω^{D}_{P,HF_t}		
	LDB	UDB	b	d	u	b	d	u
t_1	0.1326	0.3406	0.6594	0.2366	0.1040	0.6594	0.2366	0.1040
t_2	0.4318	0.5651	0.4349	0.4985	0.0667	0.5471	0.3675	0.0853
t_3	0.3335	0.6189	0.3811	0.4762	0.1427	0.4641	0.4219	0.1140
t_4	0.3859	0.5008	0.4992	0.4434	0.0575	0.4817	0.4326	0.0857
t_5	0.3549	0.5759	0.4241	0.4654	0.1105	0.4529	0.4490	0.0981

E. Sample Trust Network

We demonstrate trust computations using our toy example and a sample trust network. Fig 3 shows the sample network. We introduced notation for trust network earlier in Section II-B. As an example, b is an opinion triplet, which represents the opinion of node S on the trustworthiness of node 1 providing recommendations. All P_is are for representing FT, i.e. opinions of nodes on trustworthiness of the measurement they collect. As explained earlier, we compute propagated trust on each path (tc) from node S to D and then aggregate those values to find OT_D. In our sample TN, we have the following paths:

$S-1-3-5-D-P_D$, $S-1-3-4-5-D-P_D$, $S-2-4-5-D-P_D$

We assume that Algorithm 1 is used to query the trust of water level measurement collected by the destination node D (OT_D). Table VIII shows LDB and UDB, associated real-time and historical opinion triplets of node D (ω^{D}_{P,RF_t} and ω^{D}_{P,HF_t}) at different times. LDB and UDB on the first row come from our earlier MADM formulation. We generate LDB and UDB for the following rows randomly from interval [0.30,0.45] and [0.50, 0.65], respectively. ω^{D}_{P,RF_t} and ω^{D}_{P,HF_t} are computed respectively using (5) and (6) (We set $\alpha = 0.5$). Note that the historical value on the last row of Table VIII (ω^{D}_{P,HF_t}) is used as the FT component of OT_D in Table IX, i.e. it is the label of the edge p in sample TN.

For other edges in the sample TN, we randomly generate trust intervals and obtain ω^{A}_{B,HR_t}. These opinion triplets of links are used to compute RT. Table IX shows ω^{A}_{B,HR_t} values we use. It also displays the propagated trust value for each tc, computed by visiting each link in the given order and by using (2). For demonstration purposes, we select $g(x) = x_b$

TABLE IX: OT Computation for the Sample TN

Path	Link	Historical Opinion			T_{tc}		
		b	d	u	b	d	u
1	b	0.5376	0.3298	0.1327	0.5376	0.3298	0.1327
1	f	0.4079	0.5604	0.0318	0.4508	0.2765	0.2727
1	k	0.1024	0.7793	0.1183	0.1330	0.0816	0.7855
1	n	0.0137	0.7873	0.1991	0.0023	0.0014	0.9963
				OT_D:	0.0023	0.0014	0.9963
2	b	0.5376	0.3298	0.1327	0.5376	0.3298	0.1327
2	f	0.4079	0.5604	0.0318	0.4508	0.2765	0.2727
2	h	0.6148	0.1927	0.1925	0.3850	0.2362	0.3788
2	l	0.2069	0.7065	0.0867	0.1570	0.0963	0.7467
2	n	0.0137	0.7873	0.1991	0.0029	0.0018	0.9954
				OT_D:	0.0052	0.0032	0.9917
3	c	0.0625	0.6019	0.3356	0.0625	0.6019	0.3356
3	g	0.4694	0.3330	0.1976	0.0453	0.4364	0.5183
3	l	0.2069	0.7065	0.0867	0.0152	0.1461	0.8387
3	n	0.0137	0.7873	0.1991	0.0002	0.0024	0.9974
				OT_D:	0.0054	0.0055	0.9891
				$\boldsymbol{\omega^{D}_{P,HF_t}}$	0.4529	0.4490	0.0981
				OT_D:	**0.4529**	**0.4491**	**0.0980**

in ⊠ operation. For example, T_{tc} on the second row of path 2 is obtained by: $b \boxtimes f = \omega^{S}_{1,HR_t} \boxtimes \omega^{1}_{3,HR_t}$. At the end of the section for each path, OT_D is updated with final T_{tc} using (1). Finally, the table shows the $OT_D = (0.4529, 0.4491, 0.0980)$ as an aggregation of all T_{tc}s and ω^{D}_{P,HF_t}, highlighted as bold in the table. As seen from the table, when trust values are propagated through a tc, uncertainty increases. On the other hand, when we aggregate trust values from different tcs, uncertainty decreases, which means that more evidence is gathered by aggregation from alternative sources.

V. Evaluation

In this section, we first examine how MADM attributes affect trust scores. Next, we validate our proposed trust computation method using real data.

A. *Effect of Attribute Values on Trust Scores*

We assume that there are 20 data items measured by a "thing" and *synthetically* generate attribute values for each. We change the value of "attribute of interest" while keeping other attributes constant at their most favorable values. These are global maximum, global minimum, and preference interval values in Table IV and VI, respectively for a benefit, cost, and non-monotonic attribute. We assume the importance of the attributes is as per their order in Table IV and VI. We use the normalization methods in Table V and use ROC weights to compute RFT and RRT scores.

Figure 4 shows the effect of change in individual MADM attributes on RFT scores for key freshness, data freshness, link type, and temperature. Key and data freshness attributes fall as (1/time elapsed) – the computed scores show a rapid decrease when the elapsed time increases. RFT scores for the link type attribute is a step-wise function. The reason is that we keep the link type constant for each group of 4 data items and then change it for the next block of 4 data items. The preference interval for temperature impacts the RFT scores in Figure 4b. In this figure, $PI1 = [32, 70]$ and $PI2 = [45, 55]$. A narrower PI has a smaller plateau – a shorter range for preferred temperature values that yield the highest RFT score.

B. *Validation of the Proposed Method*

We conducted two sets of experiments using synthetic data sampled from real datasets to validate our trust computation framework. Below, we present datasets we use in the evaluation, the method of evaluation, and the results we obtain.

1) Dataset and simulation setup: We use two different datasets in our experiments. The first, by Dataport [21], contains weather data collected from 3 cities in Texas, the USA from June 2012 to November 2012. We assume that these data represent weather-related data collected and reported by IoT nodes and call it a *local sensor dataset*. The second dataset is provided by the National Centers for Environmental Information (NCEI) [18], which contains local climatological data for different cities. We assume that NCEI is an authority such that their reported data can be used as a coarse-grained check of the reliability of the data collected by IoT nodes and call it the *authority dataset*. We match the authority data to the local sensor data reported within the same time frame. We consider *temperature* as the *measurement type* mt of the measurement reported by sensors.

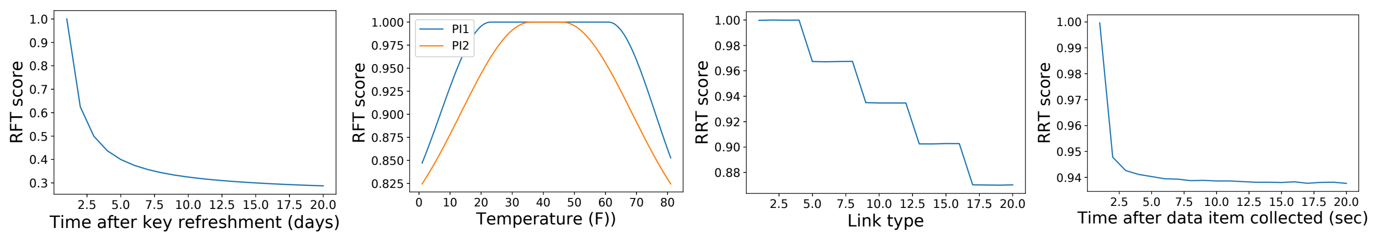
We assume an IoT network with nodes distributed according to a Poisson point process in a rectangular area of 10000 m^2. The Poisson distribution has parameter $\lambda = 0.001$ (intensity)[6]; hence, there are 16 nodes in the network. We consider Dijkstra's shortest path algorithm [5] for routing. We generated 300 data items for each node (explained below). We randomly selected 10% of nodes to be faulty (corresponds to 2 nodes in our random network) – they report incorrect observations (temperature values) during the entire simulation. We also consider possible benign nodes being compromised modeled as follows. A randomly selected node is assumed to be compromised after the first 200 observations. The last 100 observations are drawn from a distribution that has a different mean (smaller mean value in the simulation) compared to the mean value of the benign observations.

[6]This value is obtained by experimenting with different values, the goal being able to have different types of links between sensors (nodes within a distance less than 10 cm are connected by NFC and more than 46 meters are connected by Zigbee.

To apply MADM for FT computation, we used the attributes discussed in Section IV-B, i.e. *key freshness and temperature* in that order of importance. We generated key freshness values randomly between 0 and 1. For the temperature attribute, we used statistical distributions fitting best to the real temperature values in the local sensor and the authority datasets. Using the parameters of fitted distributions, we generated temperature values as measurements by each local sensor and the authority. The attributes we use to apply MADM for RT computation are the same as the attributes we discussed in Section IV-C, i.e., *measurement deviation, distance, link type, and data freshness*. Note, measurement deviation corresponds to the absolute difference of the temperature value measured by two sensors. We use the Euclidean distance between nodes. The link type attribute was decided based on the distance[6]. For data freshness, we computed propagation time between each pair of nodes based on their distance as explained earlier and added random delay values from a Gaussian distribution $N(5, 1)$ (sec) to model possible processing delays. We computed data freshness as the inverse of the sum of the delays.

2) Evaluation Method: For the first set of experiments, we set each node as a destination node in the random TN we generated and compute trust scores and opinions for each. To do so, we implemented the steps presented in Algorithm 1. We set initial $\omega^{ID}_{P,RF_{t-1}} = (0, 0, 1)$, $\omega^{A}_{B,HR_{t-1}} = (0, 0, 1)$, and $\alpha = 0.5$. We compute distrust scores for each of 300 data items by applying MADM for both FT and RT. Then, we assign a DI to each row in the dataset that takes into account distrust scores starting from the first row until the row of interest. We then convert the DI for each row into a real-time trust opinion (both for FT and RT) and combine the real-time opinion with historical opinions. We obtain a series of RFT scores, RFT and HFT opinions, RRT and HRT opinions for each node in the TN.

For the second set of experiments, we use a randomly selected destination node in the TN and compute trust scores and opinions for this destination node. In these experiments, we compare trust values generated by our approach for the destination node to the trust that the cloud generates from the authority data. We use Algorithm 1 to compute trust scores and opinions for the destination node. To find the trust based on authority data, we use a measure that is the absolute difference of the temperature values reported by the destination node and the authority for the same time instances (we recall that these temperature values were generated for nodes and the



(a) (b) (c) (d)

Fig. 4: Effect of changing MADM attributes on trust scores. Effect of: (a) key freshness (b) temperature preference interval (c) link type (d) data freshness

authority based on the best-fit distributions to the real data for the same time interval – so they will be different). We normalize this measure and use it as a real-time distrust score for each reported temperature value by the destination node. The remaining steps are the same as explained above, i.e., converting scores into a DI, then from DI's into real-time and historical opinions.

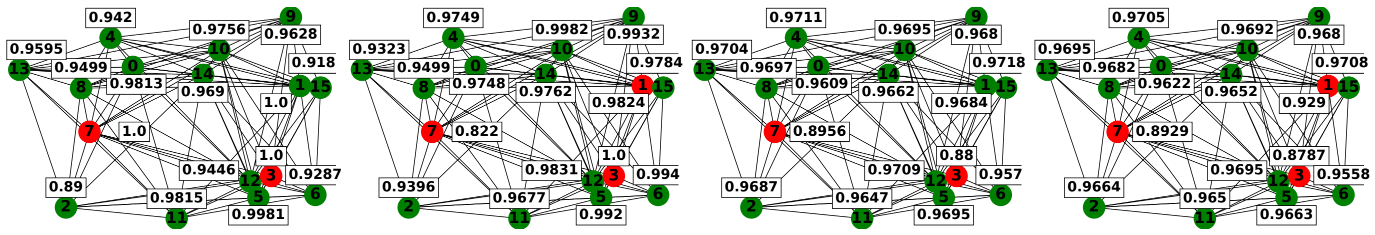
3) Effect of Change in Reported Temperature Values on Trust Scores and Opinions: Recall that one of the nodes was randomly selected to model node compromise. Here, node 1 in Figure 5 is a compromised node. It reports temperature values sampled from the best-fit distribution until node compromise (200th data item) and then temperature values from a distribution that has *significantly lower* mean temperature after node compromise; we gradually decrease the mean temperature, with the lowest being $60°F$ less than the "honest" mean. Nodes 3 and 7 in Figure 5 are assumed to be faulty from the beginning of the simulation.

We computed RFT/RRT scores, RFT/HFT and RRT/HRT opinion triplets for each node by keeping all attributes constant at their best normalized values except for the reported temperature. In other words, key freshness (FT attribute), distance, link type, data freshness (RT attributes) were set to 1. Note that the measurement deviation (RT attribute) was random as it is computed as the difference in reported temperature values between each pair of nodes. We then took two snapshots of the network and visualize RFT scores and OT opinions for every node: one network snapshot before the node compromise (200th data item) and one snapshot afterward (250th data item). Figure 5 shows the two network snapshots for RFT scores and OT opinions. The table in the figure shows the RFT scores, HRT and OT opinion values of nodes 1 and 7 for the 200 and 250 data items. As shown in Figures 5a and 5b, node 1 has a lower RFT score after being compromised and is marked by red. Node 7 also has a lower RFT score for data item=250 whereas node 3 has the same RFT score in both snapshots. The reason for these two faulty nodes to have different patterns in RFT scores is that their reported temperature values show some fluctuations around a mean value and do not necessarily decrease/increase constantly. Recall that the ambient temperature impacts the measurement accuracy, which is reflected in the functional trust. Here, node 7 has a lower normalized value w.r.t. temperature in the second snapshot, so its RFT score decreases whereas node 3 has the same normalized temperature value, so its RFT score stays the same. The same observation applies to the other healthy nodes, i.e., some of them have higher and some have lower RFT scores depending on the ambient temperature.

Figures 5c and 5d show the overall trust (OT) beliefs in the two snapshots. As expected, there is a decrease in OT (belief) for the compromised node, node 1. Note that faulty nodes (3 and 7) have lower OT beliefs in Figure 5d as they continue to report false data. It should be emphasized that neighboring nodes are affected if one of their neighbors is faulty or compromised, as the referral trust will also be impacted (in this evaluation, this is reflected in the measure deviation attribute). We note this in the slight decrease in the benign nodes' belief in the second snapshot (Figure 5d).

4) Comparing the Proposed Solution to an Authority data-based Approach: In these experiments, we compare the trust computation with a baseline method that computes trust scores and opinions using the authority dataset as explained in Section V-B2. Note that the authority data-based approach only uses a single attribute, which is the measurement deviation, whereas our proposed method considers multiple attributes discussed earlier. We randomly select a destination node and model node compromise for this destination node as in the previous experiment.

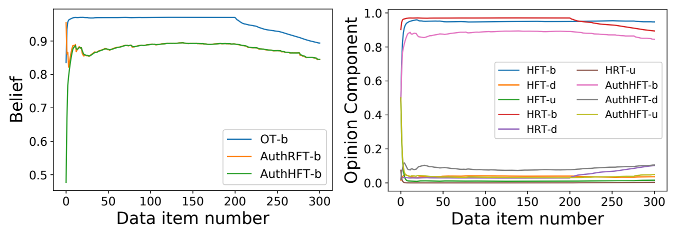
We compare overall trust opinion for the destination node generated by our trust computation method to the opinion based on authority data in Figure 6a. In this experiment, we keep all FT and RT attributes fixed at their maximum value except temperature and measurement deviation attributes. We focus on belief components in opinions. *OT-b* refers to the belief component of OT opinion obtained by applying our trust computation. *ARFT-b* and *AHFT-b* are the belief components of the opinion for functional trust, generated by considering the authority data. Our method has an immediate and dramatic decrease in its OT belief on the destination node after node compromise whereas with only the authority data, this happens, but with a smaller decrease. It is also observed that historical belief values align with the real-time belief values after a short period. In Figure 6b we present a more detailed comparison. Here we compare our method to the authority data-based method concerning all components of historical opinions. All FT and RT attributes are fixed except temperature



(a) (b) (c) (d)

	node 1 key freshness	node 1 temp	trust score node 1	HRT belief node 1	HRT disbelief node 1	HRT uncertainty node 1	OT belief node 1	OT disbelief node 1	OT uncertainty node 1	node 7 key freshness	node 7 temp	trust score node 7	HRT belief node 7	HRT disbelief node 7	HRT uncertainty node 7	OT belief node 7	OT disbelief node 7	OT uncertainty node 7
200	1	1.0000	1.0000	0.9684	0.0310	0.0006	0.9684	0.0310	0.0006	1	1.0000	1.000	0.8956	0.1037	0.0007	0.8956	0.1037	0.0007
250	1	0.9295	0.9824	0.9291	0.0685	0.0024	0.9290	0.0685	0.0025	1	0.2881	0.822	0.8929	0.1063	0.0008	0.8929	0.1063	0.0008

Fig. 5: Effect of temperature values on trust scores and opinions. Network snapshots showing: (a) FT scores before node compromise (b) FT scores after node compromise (c) OT beliefs before node compromise (d) OT beliefs after node compromise



(a) (b)

Fig. 6: Comparison of our trust computation method to an authority data-based method: (a) beliefs (b) components of historical opinions for the destination node.

and measurement deviation attributes. We observe that our method has a decreasing HRT belief for the destination node as does the authority data-based approach. We also see that decrease in HFT belief is not as apparent as the decrease in HRT belief. This is because pairwise measurement deviation between the destination node and all other nodes in the TN magnifies the effect of node compromise in RT compared to FT.

VI. Related work

A. Trust Schemes for IoT

Chen et al. [4] propose a distributed trust management protocol for SOA (Service-oriented architecture)-based SIoT systems. IoT users here are connected through social networks and relationships, so do the IoT devices users own. Trust comprises two components: direct trust from user satisfaction experiences on services provided by IoT devices and recommendations from other users. Truong et al. [27] propose an architecture of a trust platform as a service (TaaS) for the SIoT. They use a semi-centralized approach to alleviate the drawbacks of centralized and distributed architectures. Their trust model imitates human information processing and incorporates recommendation, reputation, and knowledge as trust metrics. A fuzzy-based approach is used for computing the knowledge metric but the computation of reputation and recommendation metrics are out of the scope there. The proposed trust scheme is not evaluated. Both [4] and [27] depend on the social relationship among things and user feedback from the user, trust computations may not be automated easily.

Saied et al. [23] propose a centralized context-aware trust management system for IoT. They compute the trust level of an IoT node based on service quality recommendations from other nodes. Recommendations are evaluated by a trust management system based on the contextual variables (the type of the service consumed, the time, and the capabilities of the node – resource capacity, memory/battery level). The drawback here is that there is no method presented for how service quality recommendations are obtained from IoT nodes. Perhaps, it requires manual interventions from users during the active operation. Based on the multi-service paradigm of the trust scheme proposed in [23], Mendoza et al. [17] propose a distributed trust management scheme for IoT. Their scheme relies on direct observations and communication among nodes for trust evaluation. Each node has a trust score incremented (or decremented) if it provides (or does not provide) service to its neighbor node which requests the service. Since trust computations are performed by IoT nodes, resource constraints may be an issue. The trust value reduces if a node only does not provide the requested service. A node may misuse this by providing a requested measurement, but with a bogus value.

B. Trust Schemes Similar to this Paper

He et al.[11] propose a trust management scheme for medical sensor networks with direct and indirect trust components. They use the number of "successful" and "failed" interactions between two nodes to compute direct trust. They consider historical values in trust computations, using a "sliding window" concept. We also use direct and indirect evidence of trust measures. However, unlike their method, we consider the context of "things" in trust computations, which is suggested as crucial [31], [29]. We also consider historical trust values, but we use EWMA instead. Thus we consider all past trust records as opposed to the sliding window method. Wang et al. [28] propose a trust model for service-oriented MANETs. Trust is defined as the probability of a service provider (SP) to provide satisfactory service in response to a request. They use logit regression to predict this probability concerning behavior patterns of an SP, such as energy-sensitivity (# neighbors sharing the channel), capability-limitation (packet traffic to SP), and profit-awareness (price for the service). The limitation of the framework is that nodes store service satisfaction history and contextual variables (for each interaction). As time progresses, the size of the historical records will increase. A limited IoT node may not be able to store them. Konwar et al. [16] propose a trust model for wireless mesh networks based on the MCDM approach, specifically the Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) model [13]. MCDM is used to quantify trust values of nodes primarily to support routing protocols by selecting only trustworthy nodes in secure route establishment. Selected criteria are the probability that an agent will perform a particular action (packet forwarding and recommendation exchange), the number of packets to be forwarded, the number of packets successfully forwarded, and Delivery Ratio Efficiency (DRE) of the agent. One limitation of their model is that each criterion in MCDM formulation has equal importance and thus equal weight. The total trust of a node is computed by simply adding individual and recommendation trust values without weighing them. This results in a trust value greater than allowed limits set in the model.

Unlike these approaches, we incorporate both direct and indirect evidence into trust computations with multiple contextual attributes. Our framework does not require trust computation on resource-constrained sensor nodes and can offload it to resource-abundant cloud servers. Furthermore, it automates trust computations instead of using feedback/recommendations that may require human intervention. We also account for uncertainty in trust values by adopting EBSL.

VII. Conclusion

In this paper, we propose an automated trust computation framework for IoT. We present a method for computing nodes' trust, which is based on EBSL and MADM. Our approach combines FT and RT components within a trust network (TN) taking into account uncertainty in trust evaluations. A significant contribution of our framework is that we provide a method to quantify trust scores and convert them into opinions to be used in a TN. The trust scores can account for multiple contextual and technical attributes affecting both FT and RT using the MADM approach. We validate the proposed trust evaluation mechanism using sampling from real data. The results show that the trust framework can effectively capture nodes' behavior and limitations, where faulty and compromised nodes can be identified. It is also observed that the trust of nodes would be impacted by their neighbors in the network. We note that the attributes we include in MADM are *design decisions that can change depending on the system and/or application*. Thus, the investigation of additional attributes in MADM formulation for trust computation in application-specific IoT networks could be a candidate future research direction. Yet, our proposed approach can be taken as a guideline example as it addresses various attribute types.

References

[1] M. Acosta-Coll, F. Ballester-Merelo, M. Martinez-Peiró, D. la Hoz-Franco *et al.*, "Real-time early warning system design for pluvial flash floods—a review," *Sensors*, vol. 18, no. 7, p. 2255, 2018.

[2] F. Bao and R. Chen, "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)*. IEEE, 2012, pp. 1–6.

[3] M. Beyer, "What does temperature compensation or compensated temperature range for pressure sensors mean?" https://blog.wika.com/knowhow/temperature-compensation-pressure-sensors/, accessed: 2019-02-11.

[4] R. Chen, J. Guo, and F. Bao, "Trust management for soa-based iot and its application to service composition," *IEEE Transactions on Services Computing*, vol. 9, no. 3, pp. 482–495, 2016.

[5] E. W. Dijkstra *et al.*, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[6] M. Fomichev, F. Álvarez, D. Steinmetzer, P. Gardner-Stephen, and M. Hollick, "Survey and systematization of secure device pairing," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 1, pp. 517–550, 2017.

[7] M. Frustaci, P. Pace, G. Aloi, and G. Fortino, "Evaluating critical security issues of the iot world: present and future challenges," *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 2483–2495, 2018.

[8] J. M. Fulford, "Guidance for selecting and conducting evaluations of hydrologic instruments and equipment at the usgs hydrologic instrumentation facility," https://water.usgs.gov/hif/services/evaluations/HIFEvaluationGuidance.pdf, accessed: 2019-02-11.

[9] I. Global Water Instrumentation, "Water level sensor user manual," http://www.globalw.com/downloads/wl400/wl400manual.pdf, accessed: 2019-02-11.

[10] J. Guo, R. Chen, and F. Bao, "Scalable, adaptive and survivable trust management for community of interest based internet of things systems," in *2013 IEEE eleventh international symposium on autonomous decentralized systems (ISADS)*. IEEE, 2013, pp. 1–7.

[11] D. He, C. Chen, S. Chan, J. Bu, and A. V. Vasilakos, "Retrust: Attack-resistant and lightweight trust management for medical sensor networks," *IEEE transactions on information technology in biomedicine*, vol. 16, no. 4, pp. 623–632, 2012.

[12] C.-L. Hwang and A. S. M. Masud, *Multiple Objective Decision Making–Methods and Applications: A State-of-the-Art Survey*. Springer Science & Business Media, 2012, vol. 164.

[13] C.-L. Hwang and K. Yoon, "Methods for multiple attribute decision making," in *Multiple attribute decision making*. Springer, 1981, pp. 58–191.

[14] A. Jøsang, "A logic for uncertain probabilities," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 9, no. 03, pp. 279–311, 2001.

[15] P. P. Kalbar, S. Karmakar, and S. R. Asolekar, "Selection of an appropriate wastewater treatment technology: A scenario-based multiple-attribute decision-making approach," *Journal of environmental management*, vol. 113, pp. 158–169, 2012.

[16] S. Konwar, A. B. Paul, S. Nandi, and S. Biswas, "Mcdm based trust model for secure routing in wireless mesh networks," in *2011 World Congress on Information and Communication Technologies*. IEEE, 2011, pp. 910–915.

[17] C. V. Mendoza and J. H. Kleinschmidt, "Mitigating on-off attacks in the internet of things using a distributed trust management scheme," *International Journal of Distributed Sensor Networks*, vol. 11, no. 11, p. 859731, 2015.

[18] National Centers for Environmental Information, "Climate data online," https://www.ncdc.noaa.gov/cdo-web/datasets#LCD, accessed: 2019-03-08.

[19] F. Naumann, "Data fusion and data quality," 1998.

[20] M. Nitti, R. Girau, L. Atzori, A. Iera, and G. Morabito, “A subjective model for trustworthiness evaluation in the social internet of things,” in *2012 IEEE 23rd international symposium on personal, indoor and mobile radio communications-(PIMRC)*. IEEE, 2012, pp. 18–23.

[21] Pecan Street Inc. Dataport, “Weather data,” https://dataport.cloud, accessed: 2019-02-18.

[22] R. Roberts and P. Goodwin, “Weight approximations in multi-attribute decision models,” *Journal of Multi-Criteria Decision Analysis*, vol. 11, no. 6, pp. 291–303, 2002.

[23] Y. B. Saied, A. Olivereau, D. Zeghlache, and M. Laurent, “Trust management system design for the internet of things: A context-aware and multi-service approach,” *Computers & Security*, vol. 39, pp. 351–365, 2013.

[24] H.-S. Shih, H.-J. Shyur, and E. S. Lee, “An extension of topsis for group decision making,” *Mathematical and Computer Modelling*, vol. 45, no. 7-8, pp. 801–813, 2007.

[25] B. Škorić, S. J. de Hoogh, and N. Zannone, “Flow-based reputation with uncertainty: evidence-based subjective logic,” *International Journal of Information Security*, vol. 15, no. 4, pp. 381–402, 2016.

[26] W. G. Stillwell, D. A. Seaver, and W. Edwards, “A comparison of weight approximation techniques in multiattribute utility decision making,” *Organizational behavior and human performance*, vol. 28, no. 1, pp. 62–77, 1981.

[27] N. B. Truong, T.-W. Um, and G. M. Lee, “A reputation and knowledge based trust service platform for trustworthy social internet of things,” *Innovations in clouds, internet and networks (ICIN), Paris, France*, 2016.

[28] Y. Wang, Y.-C. Lu, I.-R. Chen, J.-H. Cho, A. Swami, and C.-T. Lu, “Logittrust: A logit regression-based trust model for mobile ad hoc networks,” in *6th ASE International Conference on Privacy, Security, Risk and Trust, Boston, MA*, 2014, pp. 1–10.

[29] Z. Yan, P. Zhang, and A. V. Vasilakos, “A survey on trust management for internet of things,” *Journal of network and computer applications*, vol. 42, pp. 120–134, 2014.

[30] K. P. Yoon and C.-L. Hwang, *Multiple attribute decision making: an introduction*. Sage publications, 1995, vol. 104.

[31] H. Yu, Z. Shen, C. Miao, C. Leung, and D. Niyato, “A survey of trust and reputation management systems in wireless communications,” *Proceedings of the IEEE*, vol. 98, no. 10, pp. 1755–1772, 2010.

[32] V. Zadorozhny, P. Krishnamurthy, M. Abdelhakim, K. Pelechrinis, and J. Xu, “Data credence in iot: Vision and challenges,” in *Open Journal of Internet of Things (OJIOT), v. 3, N. 1, 114-126, 2017. Special Issue: Proceedings of the International Workshop on Very Large Internet of Things (VLIoT 2017) in conjunction with the VLDB 2017 Conference.*, vol. 3, no. 1, 2017, pp. 114–126.