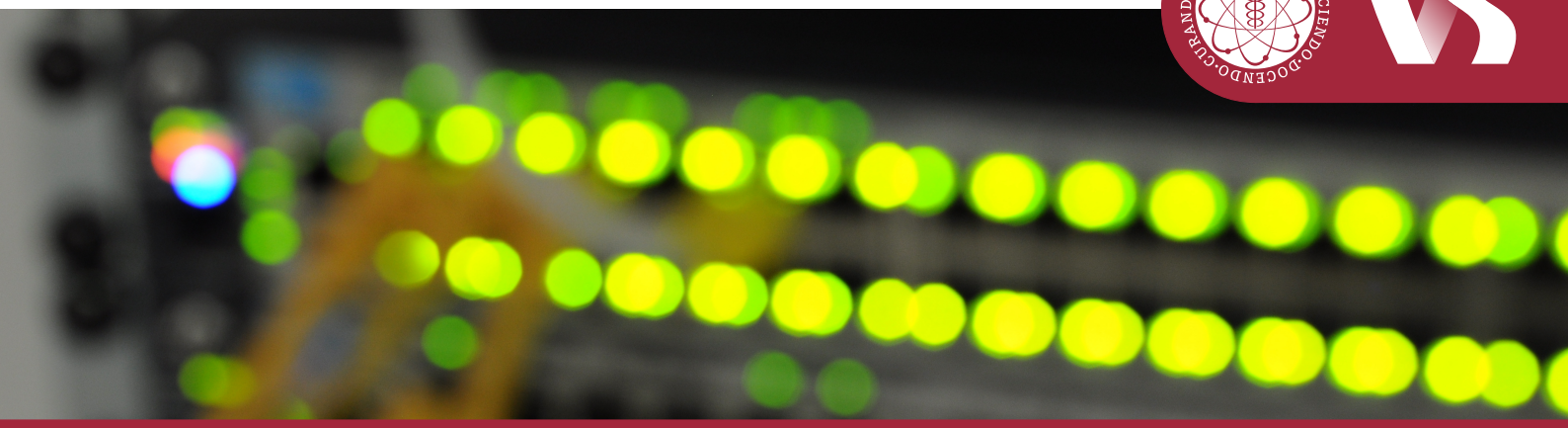

Security-Aware Privacy of Scalable Distributed Systems in High-Speed Software Defined Vehicular Networks in Node.js as a Service

Your Name
Master's Thesis

Supervisor: Dr.-Ing. Mike Slackenerny
Examiner: Prof. Dr.-Ing. Brian F. Smith
Co-examiner: Prof. Rivera
VS Number: VS-MA42-1970
Submission Date: August 2, 2017

Issued: August 2, 2017

I hereby declare that this thesis titled:

Security-Aware Privacy of Scalable Distributed Systems in High-Speed Software Defined Vehicular Networks in Node.js as a Service

is the product of my own independent work and that I have used no sources or materials other than those specified. The passages taken from other works, either verbatim or paraphrased in the spirit of the original quote, are identified in each individual case by indicating the source.
I further declare that all my academic work was written in line with the principles of proper academic research according to the official "Satzung der Universität Ulm zur Sicherung guter wissenschaftlicher Praxis" (University Statute for the Safeguarding of Proper Academic Practice).

Ulm, August 2, 2017

Your Name, student number 000000

Abstract

Contents

1 Introduction

2 Preliminaries

2.1 Subjective logic

2.1.1 Definition

Subjective logic is a logical mathematical framework that extends the probabilistic logic. Subjective logic preserves the capability of probabilistic logic to form logical statements and arguments while adding the capability to represent second-degree uncertainty. For further elaboration in probabilistic logic, we can represent the possibility of each outcome of an experiment as a value $\epsilon[0, 1]$. However, it ignores the reliability of the sources and sufficiency of the information we had before drawing the conclusion about these possibilities. The point where subjective logic extends probabilistic logic to handle is this second degree of uncertainty when we are not just lacking the certainty in the outcome of the experiment we also lack the certainty in the possibilities of the outcome of our experiment. In subjective logic, we can represent our belief in those possibilities making it very powerful tool in representing systems with uncertainty such as trust networks which we are using in our work.

2.1.2 Opinions

Subjectivity in the real world sprouts from having incomplete information, for the purpose of this work we will refer to information as evidence from now on, and draw opinions based on this evidence. Naturally, the chosen building block of subjective logic is the opinion. Opinions can be either binomial having only two possible outcomes, multinomial having multiple possible outcomes or hypernomial which having composite output consisting of many single simple possible outputs. In the work done in this thesis, we are only using binomial opinions. A binomial opinion is represented by a 4-tuple $\omega_X^A = (b, d, u, a)$. Where ω is the opinion statement. A is the opinion holder. X is the statement the opinion subject. The b is the belief in the opinion statement. The d which is the disbelief in the opinion statement. Then we have u the uncertainty about the statement. And a which is the base rate of our the probability of the statement being true without regard to the amount of information we have. The additivity requirement for an opinion is.

$$b + d + u = 1 \tag{2.1}$$

As clear from this from the chosen components of the opinion especially the belief, disbelief and uncertainty we can represent the main factors that affect subjectivity specifically

2.1.3 Projected Probability

In the context of binomial opinions, we can simply define projected probability as the probability that the opinion statement is true by taking into consideration our belief and the uncertainty in our the opinion statement. This clear from the equation to calculate his probability.

$$P(x) = b_x + a_x u_x \tag{2.2}$$

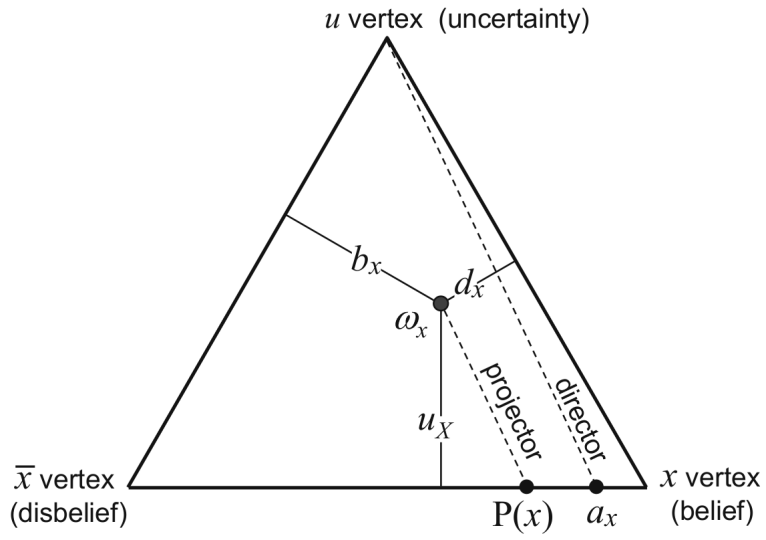

Figure 2.1: Barycentric triangle visualisation of binomial opinion [13]

2.1.4 Belief

While in literature belief calculation can differ according to the system according to the cause and the intended use, There are some formal methods of calculating belief including using evidence for and against the truth of our statement. Where r_x represents evidence for.s_x represents evidence against and W represents the non-informative weight; which is used to represent the element of uncertainty by putting a weight for it against the weight of the evidence [13].

$$b_x = \frac{r_x}{r_x + s_x + W} \tag{2.3}$$

2.1.5 Cumulative operator

There are many operators in subjective logic to be able to be able to operate on opinions and help make decisions. One essential operator for our work is the cumulative operator which is used when we have multiple opinions about the same statement from different independent sources to fuse these opinions and form one new more trusted opinion. The symbol for the cumulative operator is $\oplus$.

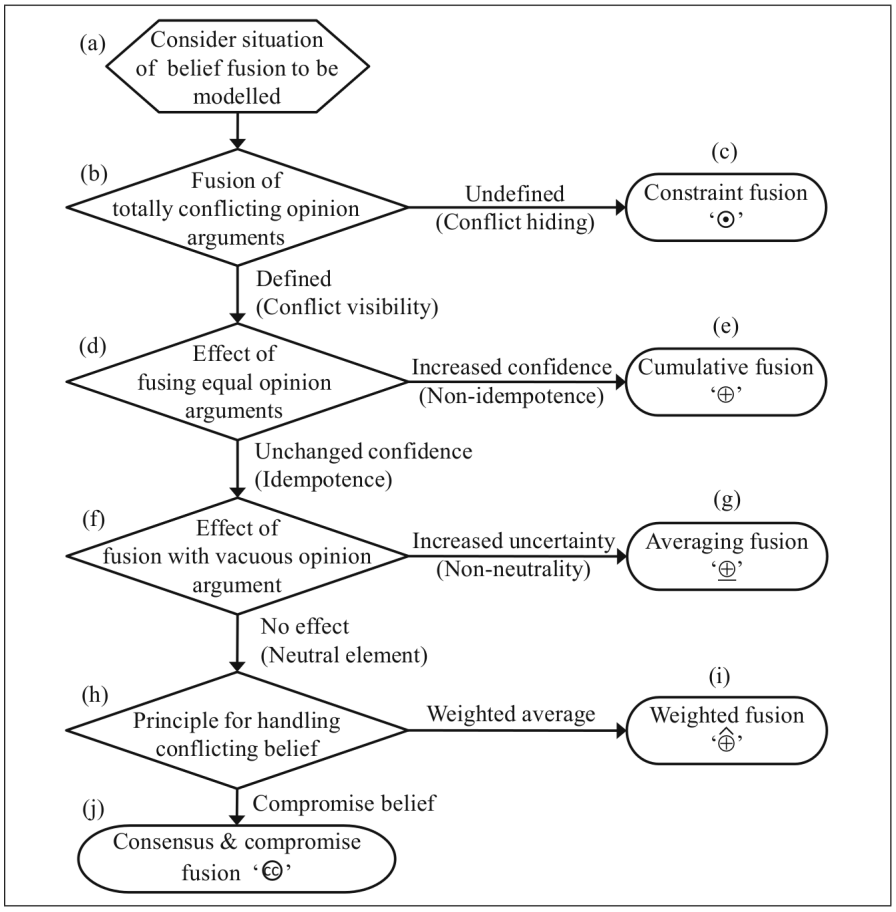

Figure 2.2: Procedure for selecting the most adequate fusion operator [13]

2.1.6 Transitivity operator

Transitivity operator is used for trust discounting. First, we explain two types of trust functional trust and referral trust. Functional trust is due to direct interaction with the subject of trust while referral trust comes from the subject of trust being referred to the trusting entity by an intermediate entity. so we use the transitivity operator to discount

the intermediate entity's trust in the subject of trust according to the end entity's trust in the intermediate entity to create a derived functional trust which is a direct trust link between the end entity and the subject of trust.

One good example to explain this using by [13], is if Alice just moved to town and needs to go to the mechanic but she does not know any good mechanics. However, her co-worker Bob has his car in a good condition. So, she asks Bob for a good mechanic and he refers her to Eric. Now, there are two people who trust Eric with different levels. The first is Bob's functional trust in Eric because he maintains his car. And, the second is Alice's referral trust in Eric because Bob referred him to Alice so normally this trust is affected by how much Alice trust Bob.

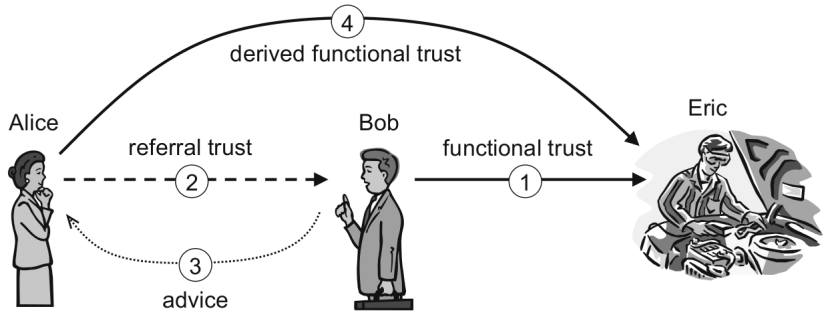

Figure 2.3: Transitive trust principle [13]

2.2 Particulate matter

2.2.1 Definition

Particulate matter or more commonly known as fine dust, are small particles of dust that exists in the air. The existence of particulate matter in high concentrations is considered polluting to the air. As people being exposed to high concentration of particulate matter causes many health complications over time. Particulate matter is classified into three categories according to their size being $PM_{0.5}$, $PM_{2.5}$, PM_{10} where the suffix number signifies the particle having a diameter of the suffix or less. In 2008, the European Union issued a directive stating the standards of air quality and the allowed concentrations of air pollutants [7]. In the mentioned directive, it was stated that $PM_{2.5}$ should be limited to an average concentration of $25\mu g/m^3$ annually later reduced to $20\mu g/m^3$. And in the extension for this directive in 2011, it was stated that PM_{10} should be limited to an average concentration of $48\mu g/m^3$ annually and $75\mu g/m^3$ daily permitted to be exceeded only 35 times per year. In this thesis, we work on forming an estimation of particulate

matter level within a certain area. There is a body of work that tried to develop a mathematical model to predict levels of particulate matter. Some examples are the work done by Tian et al. in [19] and that done by Yanosky et al. in [20]. However, Sensors are more interactive and can provide information about real-time unexpected events that can not be included in a mathematical such as construction work.

2.2.2 Sensor

We decided to use gp2y1010au0f sensors because it is the best cheap option according to [2]. First, we have to clear that this a sensor that reads the amount of dust in the air. The sensor does not have the ability to distinguish between different particle sizes [3]. However, this sensor can be calibrated to read both PM2.5 and PM10 by deriving calibration coefficients [3]. The results of testing the sensor were meaningful in terms of accuracy and precision. However, there are some problems that should still be fixed such as in cases of high-level humidity or any other light scattering problems which leads to reading bigger particle diameter than the one measured. In conclusion, noting in our prototype we only care about how much an attacker can change the actual readings and assume normal weather normal conditions and with careful deployment and careful calibration, this sensor can give us readings with high levels of accuracy and precision enough to test our security system on.

2.3 Open Networks

2.3.1 Definition

Our Definition for the term open network here is a network where anyone can participate and make use of its resources, for example in the work done in this thesis we develop an application that monitors fine dust concentration in the air by taking sensor reading at different points and produces a reliable estimation of the fine dust in the area. The term opens means that if anyone wants to deploy a sensor and send readings to the application to make the application's output more reliable they can easily do that without having to contact anyone or going through any kind of procedures.

2.3.2 LoRa and LoRaWan

LoRa and LoRaWan make us capable of creating our open network easily. According to the LoRa alliance in [1]. LoRa is a physical layer technology that is capable of long range communication. One LoRa gateway can cover hundreds of square kilometers. While LoRaWan is the communication protocol built on top of it to optimize battery lifetime of Internet of things nodes, usage of network capacity and maximize the quality of service. To summarize, using few gateways we could create low power wide area network

-LPWAN- which is a perfect for wireless sensor networks as the sensors operate with low power and send small chunks of data at a time.

2.3.3 TheThingsNetwork

TheThingsNetwork utilized all of this to create an actual open network [17]. On TheThingsNetwork, anyone can use the existing network resources from gateways and backend servers. This is done by deploying your own sensors and send them to your own application without needing to deploy network resources. While this approaches our definition for open networks it does not completely fulfill it. Because you can not simply deploy a sensor and extend the application to your area by registering the sensor. However, this can be simply achieved by simply publicly sharing the application key so anyone can register their sensor to the application. We also have to mention that security measures such as an end to end encryption are taken within TheThingsNetwork so that the network or the data traveling in it are not comprised [18]. In conclusion, the open network as we explained is already implemented and we are currently looking in how to secure the application functions within it.

2.4 Attacker Model

We consider that our attacker has full access to all information sent by the sensor and can change it freely according to his will. This data includes the position of the sensor. However, we do not consider any case where the attacker changes the sensor's position because we consider sensor readings as an estimation of particulate matter level within a certain area. Accordingly, where does the sensor position exactly within this area is irrelevant to our work. Noting that determining that the best way to divide large landscapes into smaller pieces will be left to future work. Also, attacks where the attacker changes the time-stamp are not considered because the data has to be real-time data. In conclusion, the work done in this thesis only consider attacks where the attacker the actual level of particulate matter or the battery level of the sensor.

We have two classifications for the attacks done. According to the first classification, the attacker could be either random attacker or clever attacker. Random attackers inject random readings. Clever attackers try to change to the final reading in a certain direction by constantly increasing or decreasing the reading. According to the second classification, the attacker could be either Continuous or periodic attacker. Continuous attacker when they start injecting wrong readings they do it consistently for every reading. However, periodic attackers inject false data in a certain period then stops for a while to regain some trust and then perform the attack again.

3 Related Work

As mentioned in the introduction in chapter 1, the work done in this thesis has two connected goals. The first is detection and mitigation of false data injection attack in our system. The second is investigating the capability of subjective logic to deal with the previously explained conditions of the system to achieve the first goal. Our literature review focused on the work done to detect false data with that have similar features and those who tried to use subjective logic to guarantee the data quality.

Our literature review started with looking into work done in the detection of false data injection attack with some similarities in the system. A lot of work has been done in the mitigating the effect of false data injection attack, especially in Smart grids. Smart grids systems are similar as some of the hardware sending the measurements could be completely controlled by an outside party. However, some differences exist such. As mentioned by Hao et al. [10] there are two approaches to deal with false data injection attack in general. The first is the protection against it by protecting some of the data sources. Protecting the data sources is completely not feasible in our open network. The reason for that is we need our application to be extendable easily in new areas. However, requiring some nodes to be secured first will hinder the extension and complicate the procedures. The second is to detect the injected false data then choose a way to handle them either correcting them or choosing to exclude them from the system analysis. The main method of detection in smart grids according to Rahman et al. [16] and Hao et al. [10] is residue test. However, the used method is inapplicable in our case because of the amount of data about the system needed to implement them and the usage of state estimation. As mentioned in the work proposed by Rahman et al. in [16] they use the information from both directions of the buses which are the input and the output. And in the work done by Cui et al. [6] they mentioned using state estimation to perform the residue test. Though, we try to use prediction model to estimate some readings as a reference we do not want it to be the only one because prediction can not consider unusual event which is why we went with the sensing network in the first place.

While in the work done by Sencun Zhu et al. [22], they developed a scheme to make sure that you need to compromise more than t+1 nodes, where t is a design parameter, to be able to inject false data. As the t+1 nodes would have to agree on the report so it could be taken into consideration. However, this will not work in our network model because their network is not an open one where anyone can join. So, having a design parameter related to the number of nodes for the scheme to work is not a realistic ap-

proach. And also our network is a sparse network, while the nodes here try to reach a consensus. However, our nodes report estimations for fine dust in an area but we do not expect them to send the exact same readings as each other.

In the work presented by Przydatek et al. [15], they have aggregator nodes that construct Binary Merkel hash trees where the leaves are the actual sensors. In the cluster, the hashing of the children node in the tree should get you the parent value. In this work, they pointed out that they can not work on every value. In our system that would be possible because we only view readings every hour. In this work, they consider the median of the random samples they get. And though the median value has a high robustness against false data, it has a high robustness cost as it does not use most of the available data. We adopted the concept of fusing the data but we decided to use subjective logic instead of normal statistics as subjective logic is more powerful tool as explained in the previous chapter 2.

In the paper written by Huang et al. [12], they have a noise mapping participatory sensing application in an office. In this paper, they chose an another approach to deal with false data injection. The approach is to minimize the effect of the false data in the system behavior using robust statistics. We chose this paper because participatory sensing is similar to our concept of having an open network. Their System is divided into three stages. The first is the watchdog module which accepts the readings and feedback from the reputation module, that will be explained later. The watchdog module uses the input and an outlier detection algorithm to calculate cooperative ratings for the readings. The second stage is the reputation module which uses the previously calculated cooperative readings as an input to build long term trustworthiness using Gompertz function. While in the third stage they use these reputations to get statistics summary about the noise level in the office where the usage of the reputations eliminates the effect of outliers. In this work contrary to most of the other reviewed work did expect the sensors to provide different data. However, they still used consensus based outlier detection algorithm because noise level within a closed area such as an office should nearly agree. In conclusion, we adopted the architecture of the detection system in this work which consists of three components that work sequentially. The outlier detection in a single time step. A module to build trustworthiness over a long time. Using the calculated trust to produce the final output.

Also in the work done by Burke et al. [4] to ensure data integrity in participatory sensing application, though they did offer some novel problems to ensure the security of the data and make it feasible to protect nodes. By offering special hardware with hardware-based cryptography hardware for the sensing output so it can not be manipulated. And having in accessible parts of the applications software vouch that no tampering happened to the software. However, we chose to try a different approach because we see that this might slow down the expansion of our open network.

Some works decided to use subjective logic such as work done by Gomez et al. [8]. They decided to have a broader look at the wireless sensor networks systems. They considered intentional and unintentional injected false data. They also considered bogus data and problems with injecting during data processing. However, we did not consider problems with data processing in our work. The reason is they defined different states for their data and defined the processing to be happening during the routing. While as explained in our network the data goes from the sensor to the gateway to the backend directly without any kind of processing and all the processing is done in the application. The part we focused on is bogus sensor data. We adopted from this work their approach to consider unintentional data due to exhausted batteries. They decided to test their system on a herd control application to monitor the health and well being of the cows. Their test case is very different from our system so we can not adopt their methods of calculating belief in the data values. They also had some related work in [9].

4 Reputation System

Recalling that one of the main purposes of this work is testing how effective is subjective logic in creating a reputation system. The reputations of nodes are kept as a list of opinions. Where the server has an opinion about every node that represents the trust of the server in the node.

4.1 Concept

4.1.1 Outlier detection in a single timestep

As mentioned in chapter **3**, when explaining the work proposed by Huang et al. [12]. The architecture of their system was adopted. This consisted of three stages. The first was detecting outlier in single time steps. The second is forming the trustworthiness over time using these single time steps results. The third is forming a statistical summary where the low trust nodes have minimal effect on the system. The third stage is the output of the system after forming the reputation and it is not a part of the reputation system. Thus in this chapter, we will focus on the first two stages.

For the first stage, We are aiming to detect outliers in a single time step. To achieve our goal, we had to research the properties of particulate matter to be able to distinguish illogical readings. We decided to use the statistical distribution of particulate matter to determine the outliers.

According to the work done by Henderson et al. [11] they mentioned that PM2.5 in the 25 sites they tested at were normally distributed. This was supported also by the paper published by Cameletti et al. [5] where they were trying to develop a spatiotemporal model for particulate matter levels prediction. It was also backed by the work done by Tian et al. [19] as they mentioned that the data from the Moderate Resolution Imaging Spectroradiometer (MODIS) for PM2.5 showed to be normally distributed.

To verify this we chose a random sample of 50 hours of readings. Then at each time step, we fitted the readings of all of the sensors into a normal distribution. Next, we performed the Kolmogorov–Smirnov test of 5% significance level to make sure that the fitted distribution does represent the data. Our hypothesis was only refused once out of

fifty.

Further investigation proved that this occurred at an instance where one of the sensors sent very high readings during this time step as it raised the average of the readings to nearly 244% of its value when we exclude this sensor. Such readings are possible but they indicate unusual events happening near the sensor such as fireworks, actual large fire or construction work.

To summarize, there are two main properties that we use which are particulate matter is normally distributed over space and unusual events distort that normal distribution. In figure 4.1, we can find a visual representation of the comparison between the empirical CDF of the data and the CDF of normal distribution resulting from fitting the data in one of the time steps.

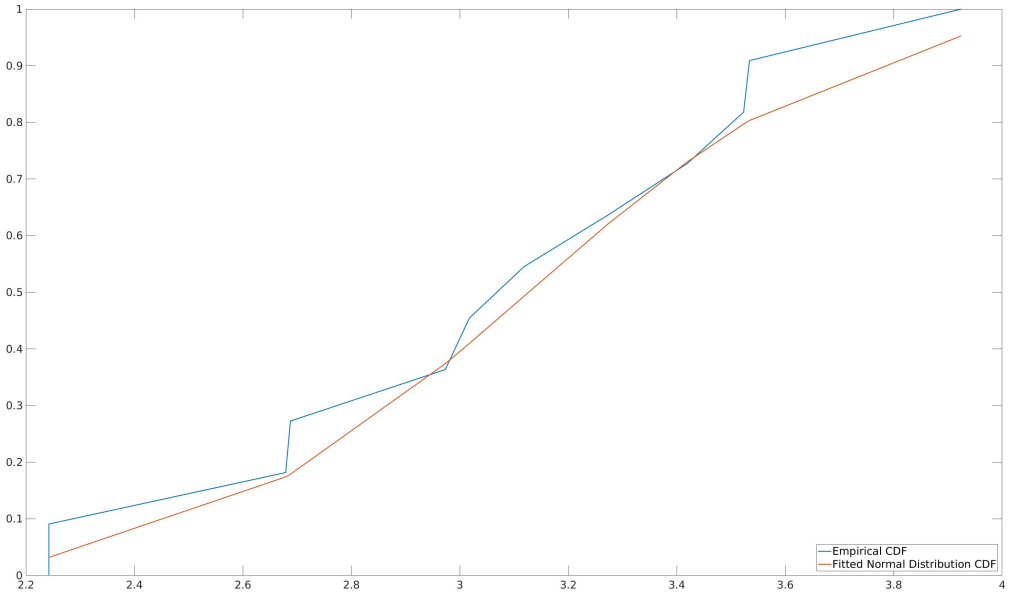

Figure 4.1: Visual comparison between Empirical and Fitted distributions

We tried to use the aforementioned information that we found to detect outliers in a single time step. Firstly, as previously mentioned we consider the reading of any of the nodes as an estimation of the PM level in the area, not as the exact reading at node's location. In addition to that, we suppose that we have a mathematical environmental model that is able to provide a prediction for the particulate matter level in the area.

For testing the honesty of a certain node, we fit all of the other nodes readings and the prediction of the environmental model into a normal distribution. Then we judge if the reading of the node belongs to the same distribution or not. This system uses both of the aforementioned properties.The normal distribution property which is clear in fitting the data to the distribution.

The unusual events property here is dealt with by considering as an attack because they lead to a false estimation of particulate matter within the area. So the effectiveness comes is that every distribution with false data will be distorted which makes it harder to get evidence from it, this will be explained later. While the better-fitted distributions result in more evidence.

Assuming we have one attacker, we will have one completely correct distribution that we gather evidence from. This will result in quickly completely distrusting the attacker. While the number of attackers increases all of the distributions will be ruined. However, the more attackers while forming one distribution the less evidence taken from that distribution and vice versa.

In conclusion, we try to figure out does a reading comes from the same distribution as all of the other nodes or not. An example of this in figure 4.2. Where R_A is the reading of node A. $\omega_X^{S_A}$ is the server's opinion about the reading of node A. While M represents the environmental mathematical model.

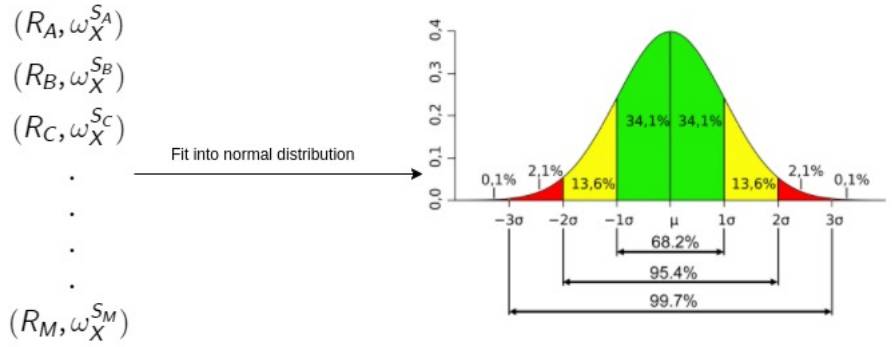

Figure 4.2: Outlier detection test for node Y

4.1.2 Trustworthiness over time

The method we devised for implementing the first stage proved perfect to use in the second stage. As for the second stage, we try to form trustworthiness over time. For this, we use subjective logic. Recalling equation 2.3 for calculating belief. This equation needs a

numeric value to be plugged into the equation as the value for the evidence for and evidence against. The method for getting the numeric values will be elaborated later. The evidence of each time step is aggregated with previous time steps to update the opinions. The more recent the evidence is the more weight they hold.

4.2 Implementation

4.2.1 Outlier detection in a single timestep

When fitting the data into normal distribution we faced a problem. As explained in 4.1, when we have unusual readings that do not belong to the system due to unexpected events or due to attacks the normal distribution is distorted. As shown in figure 4.2 we use the readings and the opinions about the readings to perform the distribution fitting.

Noting that a normal distribution has only two defining parameters. The first is its location parameter which is the mean. The second is its scale parameter which is the standard deviation. So, we use the expected probabilities of the opinions as weights for the readings to calculate a weighted average and a weighted deviation. This helps us avoid the distortion of our distribution. Thus, it ensures that we have a more accurate distribution for judging the node currently under test.

The next step would be to extract numeric data to plug into the belief calculating equation. For performing that the simplest method would have been to calculate the distance from the mean or the median value and calculating a threshold for the allowed distance from the mean. The problem with this is its naivety as only through careful studying of the relation between PM levels variation with space can this threshold be determined.

Even then the problem mentioned earlier with the environmental model will exist. The problem was that such an approach would not take into account the unusual events that may cause juristic changes in the PM levels readings. However, this possible in our case because the data follows a normal distribution. By using the property mentioned in figure 4.2 that a reading belonging to this distribution should lie at a maximum distance of two standard deviations from the mean with a probability of 0.954. And in the case of the unusual events, both the mean and the deviation will change appropriately.

So, what we did at first is determine the number of deviations allowed by fine tuning using some elementary tests results and we found out the best value to be 1.5 of the deviation. The amount of data included within a distance of 1.5 deviations is around 86% of the data belonging to the distribution.

An example of the method is explained in figure 4.3. The figure includes two scenarios. The yellow markers represent the reading in each of the scenarios. In the first scenario, The reading is between the 1.5 deviations away from the mean and the actual mean. So, the distance between the 1.5 deviations away from the mean and the reading itself is calculated as evidence for. In the second scenario, the reading is even further than 1.5 deviations away from the mean. So, the distance between the reading and the 1.5 deviations away from the mean is calculated as evidence against.

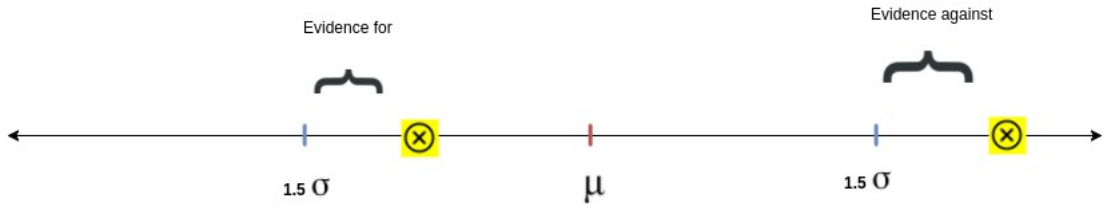

Figure 4.3: Evidence calculation in a single time step

During the full testing of the system, this method did not yield the best results. So, we decided to tweak the approach a little bit.

First, We point to the fact mentioned before that the existence of attackers distorts the distribution. Even if we managed to mitigate the effect of this from greatly distorting the distribution, the deviation value among our data was very high. So, We decided against having a constant marker at 1.5 deviations.

Instead, we implemented a moving marker that ranges from 2 deviations to 0.5 deviations. The decision of where to put the marker is based on the ratio between the deviation and the mean as explained in equation 4.1. Noting that if the equation produced a number less than 0.5, we will take 0.5 instead. In the work done by Monn et al. [14], they said that the PM2.5 levels might be uniformly distributed as they were discussing small scale spatial variations. This meant that we have to be more tolerant with our markers in case of the readings were more concise and nodes were in an agreement. In the aforementioned case, one sensor that lies on a road with high traffic would be distrusted quickly because of slight disagreement. That is why we increased our upper limit to 2. And now since the marker is a moving one. We can afford to do that because we can be more strict whenever we have a wider distribution where the nodes are not agreeing. The lower limit was obtained through fine tuning as lower numbers just ended up distrusting everyone.

Noting that, with this approach the larger the readings the larger the evidence. This could be changed by standardizing the normal distribution. However, we decided against standardization and decided to keep it as it is. Because this makes the system more sensitive when we have higher levels of particulate matter in general which is a more dangerous situation for people involved. Thus, it makes sense to keep the system as is.

To conclude this part, evidence collected in a single time step is collected according to the position of a reading with respect to the distribution of the other readings. If it is within the area between the markers, then the distance between it and the nearest marker is considered evidence for. If it lies outside the nearest marker the distance is considered evidence against. The markers are moving according to how wide the distribution is. And we call evidence for collected in one time step r_{x_t} and evidence against collected s_{x_t}.A comparison between the constant marker approach and the moving marker approach is can be found in the testing and evaluation in chapter 7.

$$marker = 2\sigma * (1 - \frac{\sigma}{\mu}) \pm \mu \tag{4.1}$$

4.2.2 Trustworthiness over time

For forming long term trustworthiness with the more recent data holding more weight for the system to be adaptive and interactive. Before starting to explain how our system work, we define our starting point. We are designing an adaptive system. The system starts from the point of complete uncertainty about the node whether it is an attacker or not. This kind of opinion is called a vacuous opinion [13]. A vacuous opinion with 0 belief, 0 disbelief and 1 uncertainty as in equation 4.2. We do that with every new node joining the system.

$$\omega_X^{S_A} = (0, 0, 1, 0.5) \tag{4.2}$$

As mentioned before, we use equation 2.3. In this equation, there are 3 variables non-informative weight W, evidence for r_x and evidence against s_x. The non-informative W is set to 2 as this the convention when you need to create a vacuous binominal opinion at the beginning. While evidence for and evidence against are the aggregations of evidence for and evidence against in single time steps.

However, aggregating over all of the time the system has been running is not realistic because it will reach a point where new evidence is insignificant. This will greatly affect the response speed, one of the metrics used to evaluate the system performance, of the system.

To fix this we thought about implementing a sliding window approach. However, implementing a sliding window approach is more complicated, requires more storage and does not introduce a solution for the problem of the necessity of having more weight for more recent evidence. While all of this can be fixed with a decay factor approach. Where we just multiply the old evidence with a factor $\alpha < 1$ so old evidence will get smaller over time. This is easier to implement, requires less storage and fixes the necessity of having more weight for recent evidence. The equations to implement this are equations 4.3 and 4.4.

$$r_x = \alpha * r_x + r_{x_t} \tag{4.3}$$

$$s_x = \alpha * s_x + s_{x_t} \tag{4.4}$$

5 Implementation

5.1 System architecture

In figure 5.1, We explain the complete architecture of our system. As clear from the figure, our system consists of four modules, has one time of input and Some output. The only input type to the system is packets with node's ID, node's location, PM level Reading and battery percentage. Clearly, Some of the packets components are missing in the figure. The reason for that is, we only included the components that are constantly used in the figure to give more focus on them. The desired output from the system is a final estimation of fine dust in a certain area and a subjective opinion about it. However, we have some other different outputs that were added for testing results and will be discussed in the testing chapter 6. The modules of the system are the virtual sensors, Statistical summary module, outlier detection in a single time step module and long term trustworthiness module. The latter two modules were discussed in details in the Reputation System chapter 4. Thus, we will focus only on the first two modules and the overall system behavior in this chapter.

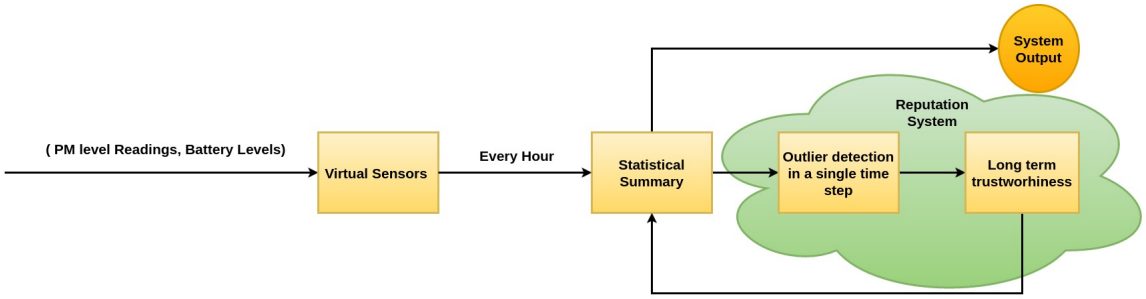

Figure 5.1: System architecture

5.2 Data Fusion

Before discussing the system modules, We need to explain the method operations are carried out on the data and the opinions. For this, we need to clarify That every reading is paired with an opinion of some entity about this opinion.

5.2.1 Data Operations

The operations that are done on the data is fusing data together when they represent the same thing. This is done by calculating a weighted average or weighted deviations. The weights for every reading in these operations is the expected probability of the opinion paired with these readings.

5.2.2 Opinions Operations

While there is more variety in operations done on opinions accompanying the data. The operations on opinions help update opinions during transition periods for the actual data. The two operations done are opinions cumulation together and opinions discounting by other opinions.

Cumulation of opinions is used when we fuse data together to get a weighted average, We at the same time cumulate the opinions using a cumulative operator to form an opinion about the weighted average.

Discounting the opinions when we have referral opinions this happens in two cases. The first case when the node has an opinion about its reading but it also has an opinion about itself, this will be explained later, we then discount the first opinion about the reading by the opinion reading about itself using the transitivity operator. The second case is when we have the node's opinion about its reading and the server's opinion about the node, and we need to know the server's opinion about the reading. In this case, we discount the node's opinion about the reading by the server's opinion about the node to get the server's opinion about the reading.

5.3 Virtual sensor

We delegate all of the data processing to the application to ensure a higher level of security. This was done by registering all of the real sensors in the application as corresponding virtual ones. The virtual sensors can save the readings and all useful information until we need to use the data.

5.3.1 Registration

Registering the node requires saving the node's ID and the node's location. In addition to that, the virtual sensor stores the real sensor reading and the opinion it has about this reading. Whenever a real sensor sends a reading, the application checks if the sensor already registered or not. If the sensor is not registered, the application registers

the sensor by creating virtual sensor save it in the list of sensors and create a vacuous opinion about it in the reputation list.

5.3.2 Self Opinion

The sensor forms an opinion about itself according to the battery level that is received with any reading. The purpose of this opinion is as mentioned in the work done by Gomez et al. [8] is to mitigate the effect of unintentional This opinion is referred to as $\omega_{battery}$.The components of the equation to form the opinions are explained in details in equation 5.1. This equation was taken directly from the work done by Gomez et al. in [8]. The $\alpha_{battery}$ mitigates the impact of the remaining battery on its own trust evaluation [8]. The *battery* is the remaining percentage of the battery's charge level from 0 to 100%. While *batteryuncertainty* represents the uncertainty we have about the battery level. We set the *batteryuncertainty* to 0 in our work. Next, we explain in detail how the input packets are handled.

$$\omega_{battery} = (b, d, u, a)$$

$$\text{where} \begin{cases} b = (battery/100) - \frac{100 * \alpha_{battery}}{battery} \\ d = 1 - b - u \\ u = battery\, uncertainty \end{cases}$$

(5.1)

5.3.3 Handling received readings

The received reading itself is handled the same whether it is the first every time or not. The reading is simply paired with a dogmatic opinion. We chose a dogmatic opinion because normally a sensor would be 100% of its own reading. Then this opinion is discounted by the self-opinion created using the battery reading.

After this, we check if this is the first reading received during this hour. If it is, we simply store the reading and the opinion associated with it as the sensor's reading and the opinion about it. If it is not, We fuse the reading with the previously stored one and cumulate the opinion with the previously stored one. The results of the fusion and cumulation are our new reading and opinion for the current hour.

Finally, every hour all of the virtual sensors sends their reading for the current hour to the Statistical summary module for processing. And after the module is done processing

the data the virtual sensor throws away its reading and opinion about the reading.

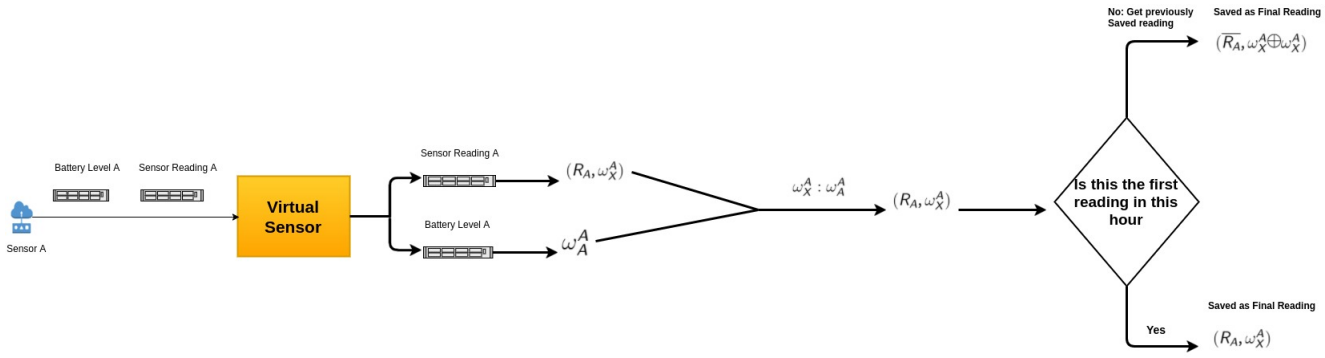

Figure 5.2: Input Readings Handling

5.4 Statistical summary module

The Statistical summary module is the module that produces the final output of the system. At first, when implementing this module, we thought about implementing an alarm that triggers when the PM level exceeds a certain level to simulate the real life application. However, since our implementation is a prototype designed with proof of concept as its goal, we decided to just produce a final estimate of the PM level in the area and an opinion about this estimation. We took this decision because it gives us more freedom to analyze the data and the behavior.

Every hour, the statistical summary module pulls the reading and opinion pairs from all of the virtual sensors. We take the opinion out of the sensors. Remembering from the reputation system chapter 4, the server keeps opinions about each sensor to represent their reputation.Then we discount the opinions of the virtual sensors about their readings by the opinions of the server about the virtual sensors. Next, we fuse data together and cumulate opinions. The result of the data fusion is the final estimate for fine dust in the area. And the result of this cumulation is the final opinion about this data.

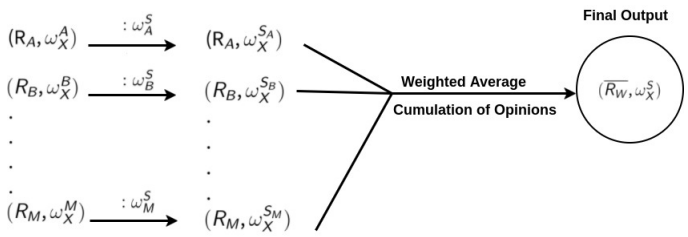

Figure 5.3: Input Readings Handling

5.5 Challenges

5.5.1 Environmental Model

The environmental model could not be implemented. Though, reviewing the literature revealed that a decent amount of work was done in the field of particulate matter levels. However, for their implementation by us is not possible because of missing parameters and lack of knowledge in the environmental sciences.

Such as in the work proposed by Yanosky et al. [21] they had the variable $g_t(s_i)$ accounting for residual monthly spatial variability, $g_t(s)$ accounting for time-invariant spatial variability and $Z_{i,t,1}$ and $Z_{i,t,P}$ for time varying covariates.

And in the work done by Henderson et al. [11], they developed a model for predicting the yearly average of PM2.5 level in the area. We thought of might for a way to manipulate the model to predict data in smaller intervals.However, the lack of variables such as the size of commercial and industrial areas in the city.And our lack of understanding for environmental science made us shy from using this approach.

5.5.2 Hardware

The hardware could not be implemented in the specified time frame for this project. The hardware implementation involved two components getting the backend to receive data on TheThingsNetwork servers. And having a microcontroller with a LoRaWan transmission module and the sensor of our choosing connected to it sending data to the backend application.

The first part we got the project to configure and run on our server making sure there is no problem with how it is written. However, the second part we were not able to get it running because of incompatible hardware. The problem is the developers of TheThingsNetwork provide a software development kit (SDK) for handling data sending when using Arduino microcontroller with few lines of code without handling any configurations in the network stuff. However, the available hardware for us was waspmote microcontroller.

Although, waspmote can understand Arduino code it still has its own integrated development environment (IDE) and few differences between it and the arduino code. This makes waspmote incompatible with some of the third party libraries. These incompatibilities included the SDK developed by TheThingsNetwork. The problem is the calling of *<Arduino.h>* file in one of the classes and this file does not exist in waspmote. So, we have two solutions.

The first solution is reprogramming the library and changing the functions called from the *<Arduino.h>* file by functions that exist in the waspmote. Before discussing the second solution, we should point out that waspmote contains a lot of previously implemented libraries to handle LoRaWan. So, the second solution would be to get the configuration for the LoRa channels used by the TheThingsNetwork. We performed a quick search for someone who already implemented this but none could be found. The solution was to do more research to be able to configure it. However, our research question could be answered without this. At the end, we decided to spend more time in analyzing the data rather than implementing this but we believe this can be surely implementable.

6 Testing Setup

7 Evaluation

8 Conclusion

8.1 Data

8.2

Bibliography

[1] L. Alliance. “A technical overview of LoRa and LoRaWAN”. In: *White paper, Nov* (2015).

[2] M. Budde, M. Busse, and M. Beigl. “Investigating the use of commodity dust sensors for the embedded measurement of particulate matter”. In: *2012 Ninth International Conference on Networked Sensing (INSS)*. June 2012, pp. 1–4.

[3] M. Budde, R. El Masri, T. Riedel, and M. Beigl. “Enabling low-cost particulate matter measurement for participatory sensing scenarios”. In: *Proceedings of the 12th international conference on mobile and ubiquitous multimedia*. ACM. 2013, p. 19.

[4] J. A. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. B. Srivastava. “Participatory sensing”. In: *Center for Embedded Network Sensing* (2006).

[5] M. Cameletti, F. Lindgren, D. Simpson, and H. Rue. “Spatio-temporal modeling of particulate matter concentration through the SPDE approach”. In: *AStA Advances in Statistical Analysis* 97.2 (2013), pp. 109–131.

[6] S. Cui, Z. Han, S. Kar, T. T. Kim, H. V. Poor, and A. Tajer. “Coordinated data-injection attack and detection in the smart grid: A detailed look at enriching detection solutions”. In: *IEEE Signal Processing Magazine* 29.5 (2012), pp. 106–115.

[7] E. Directive. “Council Directive 2008/50/EC on ambient air quality and cleaner air for Europe”. In: *Official Journal of the European Communities, L* 151 (2008), pp. 1–44.

[8] L. Gomez, X. Gentile, and M. Riveill. “A framework for trust assessment of sensor data”. In: *Wireless and Mobile Networking Conference (WMNC), 2011 4th Joint IFIP*. IEEE. 2011, pp. 1–7.

[9] L. Gomez, A. Laube, and A. Sorniotti. “Trustworthiness assessment of wireless sensor data for business applications”. In: *Advanced Information Networking and Applications, 2009. AINA’09. International Conference on*. IEEE. 2009, pp. 355–362.

[10] J. Hao, R. J. Piechocki, D. Kaleshi, W. H. Chin, and Z. Fan. “Sparse malicious false data injection attacks and defense mechanisms in smart grids”. In: *IEEE Transactions on Industrial Informatics* 11.5 (2015), pp. 1–12.

[11] S. B. Henderson, B. Beckerman, M. Jerrett, and M. Brauer. “Application of land use regression to estimate long-term concentrations of traffic-related nitrogen oxides and fine particulate matter”. In: *Environmental science & technology* 41.7 (2007), pp. 2422–2428.

[12] K. L. Huang, S. S. Kanhere, and W. Hu. “Are You Contributing Trustworthy Data?: The Case for a Reputation System in Participatory Sensing”. In: *Proceedings of the 13th ACM International Conference on Modeling, Analysis, and Simulation of Wireless and Mobile Systems*. MSWIM ’10. Bodrum, Turkey: ACM, 2010, pp. 14–22.

[13] A. Jøsang. *Subjective Logic: A Formalism for Reasoning Under Uncertainty*. Springer, 2016.

[14] C. Monn. “Exposure assessment of air pollutants: a review on spatial heterogeneity and indoor/outdoor/personal exposure to suspended particulate matter, nitrogen dioxide and ozone”. In: *Atmospheric environment* 35.1 (2001), pp. 1–32.

[15] B. Przydatek, D. Song, and A. Perrig. “SIA: Secure Information Aggregation in Sensor Networks”. In: *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*. SenSys ’03. Los Angeles, California, USA: ACM, 2003, pp. 255–265.

[16] M. A. Rahman and H. Mohsenian-Rad. “False data injection attacks with incomplete information against smart power grids”. In: *Global Communications Conference (GLOBECOM), 2012 IEEE*. IEEE. 2012, pp. 3153–3158.

[17] *TheThingsNetwork architecture*. `https://www.thethingsnetwork.org/docs/`. 2017.

[18] *TheThingsNetwork security*. `https://www.thethingsnetwork.org/wiki/Backend/Security`. 2017.

[19] J. Tian and D. Chen. “A semi-empirical model for predicting hourly ground-level fine particulate matter (PM 2.5) concentration in southern Ontario from satellite remote sensing and ground-based meteorological measurements”. In: *Remote Sensing of Environment* 114.2 (2010), pp. 221–229.

[20] J. D. Yanosky, C. J. Paciorek, and H. H. Suh. “Predicting chronic fine and coarse particulate exposures using spatiotemporal models for the Northeastern and Midwestern United States”. In: *Environmental health perspectives* 117.4 (2009), p. 522.

[21] J. D. Yanosky, C. J. Paciorek, and H. H. Suh. “Predicting chronic fine and coarse particulate exposures using spatiotemporal models for the Northeastern and Midwestern United States”. In: *Environmental health perspectives* 117.4 (2009), p. 522.

[22] S. Zhu, S. Setia, S. Jajodia, and P. Ning. “An interleaved hop-by-hop authentication scheme for filtering of injected false data in sensor networks”. In: *IEEE Symposium on Security and Privacy, 2004. Proceedings. 2004*. May 2004, pp. 259–271.