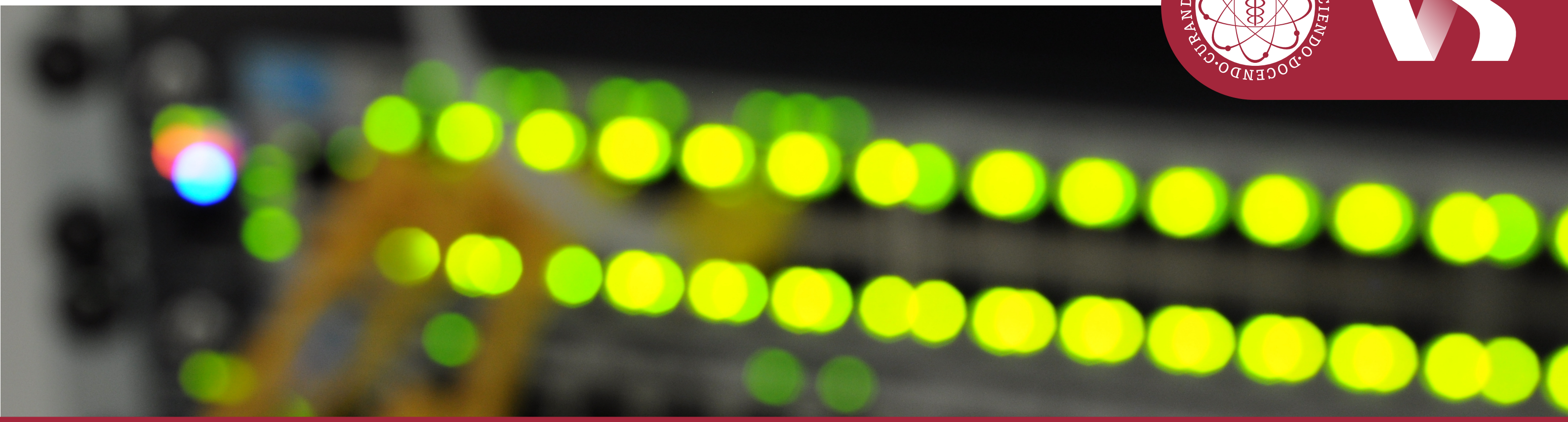


Detecting Data Injection attacks on Sensor Networks

Mahmoud Nabegh
Bachelor's Thesis

Supervisor: Rens van der Heijden, M. Sc.
Examiner: Prof. Dr. rer.nat. Frank Kargl
VS Number: VS-MA42-1970
Submission Date: August 29, 2017

Issued: August 29, 2017

🅭🅯 This work is licenced under a Creative Commons Attribution 4.0 International Licence.

To view a copy of this license, visit
`http://creativecommons.org/licenses/by/4.0/deed.en`
or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.

I hereby declare that this thesis titled:

Detecting Data Injection attacks on Sensor Networks

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 29, 2017

Mahmoud Nabegh, student number 000000

Abstract

This work tackles detection and mitigation of false data injection attack in open wireless sensor networks using subjective logic. Where open networks are networks where anyone can contribute their own data to the application. Through this work, we try to evaluate how efficient is using subjective logic for data fusion and reputation system. Our open networks concept is relatively new and the only close concept to it is participatory sensing which is also a relatively new research field. Not only there is a scarcity of work in detection of false data injection in such networks. W also have found no work that used subjective logic in such networks. We are using data analytics for the detection and mitigation of attacks to avoid implementing any restrictions on the openness of the network or its users. Next, we went by to define the details of the system, then develop an attacker model and evaluate the output of our test cases. Our research provides a building block to build a complete monitoring system for the users of any open network whatever the application with minimal changes to the system.

Acknowledgements

To Prof. Dr. rer.nat. Frank Kargl, I thank you for your help enabling me to set the right direction for the the project which was the key factor to deliver a successful meaningful output.

To my supervisor Rens Van der Heijden, I am deeply grateful for your support, for your constant sincere feedback, for encouraging me to find the optimum solution for every challenge, and for your constant availability to name a few. It was a pleasure having you as a mentor to guide me to the right way to do research. Thank you once more.

To the whole team of the Institut für Verteilte Systeme in Ulm University, thank you for always taking the time to respond to my questions and requests and for creating the prefect environment for productivity and effective work.

To Professor Amr Talaat and Professor Amr El Mougy, Thank you for giving me this opportunity.

To my professors in the German University in Cairo especially professor Tallal El Shabrawy and Professor Mohamed Ashour, for this thesis is the product of what you have taught me throughout my college journey so far, be it helping me to acquire the academic knowledge or the needed skills.

Last but not least, to the support system, the backbone, and to those whom I will never be able to thank enough, you have been my number one sponsors giving me the strength to bring the best in me, so thank you, my family and friends.

Contents

1 Introduction

For now, we will define open networks as networks where anyone can deploy a sensor and send a reading to the backend to provide more data. This makes the application more accurate and easily extendable. It is easily extendable because if new users started reporting data about new areas those new areas will be covered in the application's range. According to this definition of open wireless sensor networks, it is very easy for a malicious user to join the network and send wrong readings to the backend and disrupt the application's functionality. We decided to find a way to mitigate the effect of false data injection attacks while adding no restrictions to the behavior of the network and its users.

1.1 Motivation

1.1.1 Open networks

The concept of the open network offers a fertile environment for creativity. As this open network provides the chance for increasing the level of data openness. By reviewing history, we can find that the more people are able to access and use information, the more the civilizations grow. This ability of people to access and use information is what we call here the openness level. Multi-billion dollar businesses flourished because of the internet that opened the world and made information available with the click of a button. And later on because of the introduction of smartphones when internet and applications using it became easier to access and available to us in our pockets.

Moreover, the openness of the network gives the power to the individuals whose governments are not implementing services using the internet of things and they do not have these multi-billion dollar companies to start implementing and testing in the country. Therefore, the citizens of these countries most probably sit at the end of the queue for people who are receiving such services. However, increasing the level of openness gives those citizens the freedom to implement their own self-made services.

However, the openness does not come free of risks. As this openness is a perfect environment for malicious users to achieve their goals and ruin people's lives. Thus, we decided to start by trying to protect people's self-made services by researching a way to protect their applications from false data.

1.1.2 Particulate matter

We chose particulate matter (PM) as an application for our system due to the importance of controlling its levels in the air. Particulate matter is the scientific name for fine dust. As PM is toxic according to the work done by Harrison et al. in [11] and has negative effects on the respiratory and cardiovascular systems of humans according to the work done by Schwarze et al. in [19]. This is clear from the European Union's (EU) directive about air quality [7]. Where the EU instructed the countries to keep the PM levels under certain thresholds.

1.2 Problem Statement

Before explaining our problem statement, we first begin by explaining exactly how does the system behaves in normal cases and what are the threats faced by this system. Firstly, we have a wireless sensor network. Each sensor senses the particulate matter level in its specific location. Then, each sensor sends its reading directly to the gateway of the network. After that, the gateway sends the readings to the server. Our application pulls the data from the server and starts forming an estimation for the particulate matter level in the area.

Our main concern in this system is its openness of the network where anyone can join the network with a malicious intent or some people could leave their sensors unprotected making it easy for attackers to hijack the sensors. And since we do not want to restrict the openness of the network and the ability for someone to easily extend the application's reach by deploying sensors. Therefore, we do not want to resort to complicated deploying policies or nodes protection methods. Thus, our conclusion was to protect the system by data analysis, creating a reputation system and robust method for data aggregation. The data analysis will allow us to detect outliers in single time steps. The reputation system eases the process of keeping track of the nodes' behavior over time. Finally, the robust method for data aggregation allows us to eliminate the effect of maliciously behaving nodes over time.

We decided to use subjective logic to implement our reputation system and the robust data aggregation. Our decision was based on the fact that subjective logic has a high expressive power for data and opinions. We chose opinions to represent trust. Therefore, if we used these opinions that represent the trust to aggregate the data, we expect it to have a high level of robustness.

1.3 Research Focus

We focused on two main questions in our research:

1. What is the robustness level of data aggregation using subjective logic?
2. How efficient is subjective logic in creating a reputation system?

The efficiency in the second question refers to two main points. The first point is that subjective logic can express the system state. The second point is that using the operators and tools of subjective logic, we can accurately update those reputations.

1.4 Work Overview

After defining the problem clearly, we explain the thesis structure. In the next chapter 2, we explain the background knowledge needed to understand the work done in the thesis. After this comes the *Related Work* chapter 3 where we do a review of the literature and explain how similar problems were tackled in the past. While in the *System Design* chapter 4, we explain the system architecture in detail, the basic operations performed in the system and the concept behind some of the modules in the system. Then we pinpoint to our biggest contribution in this work in the *Reputation System* chapter 5. Next, we explain how the evaluation was approached and the scenarios we are testing on in the *Test Setup* chapter 6. While in the *Evaluation* chapter, we explain the metrics we judged the system performance based on and analyze the results of running the test scenarios we had. Finally in the *Conclusion* chapter 8, we explain the meaning of the results that came from the analysis, suggest some applications for our system and explain the direction of the future work that we are looking into to improve our system.

2 Preliminaries

2.1 Subjective logic

2.1.1 Definition

Subjective logic is a mathematical framework that extends 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 the 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 originates 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. Therefore, the chosen building block for 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. As our chosen opinion statement is *"does this reading represent the state of the real world"*. Clearly, this statement is either true or false and does not have any other possibilities. 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. Noting that $a, b, d, y\epsilon[0, 1]$. The additivity requirement for an opinion can be seen in equation 2.1.

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

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.

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 the opinion's projected probability.

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

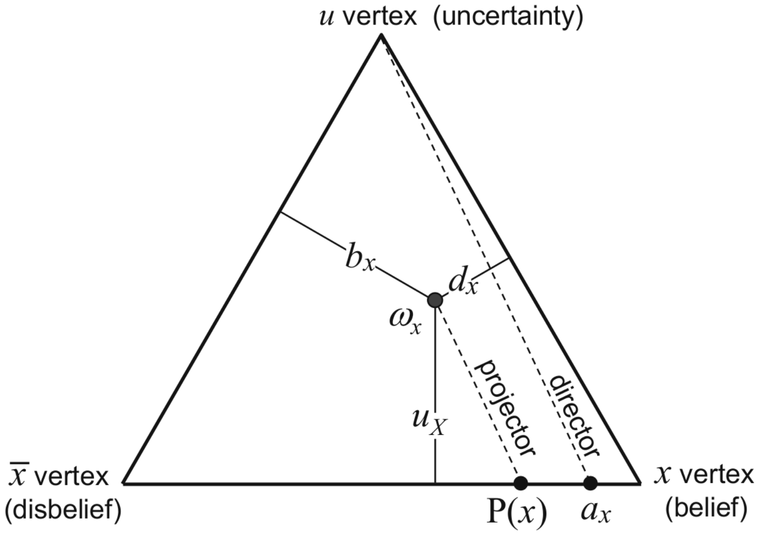


Figure 2.1: Barycentric triangle visualisation of binomial opinion [14]

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 [14].

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

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 $\oplus$.

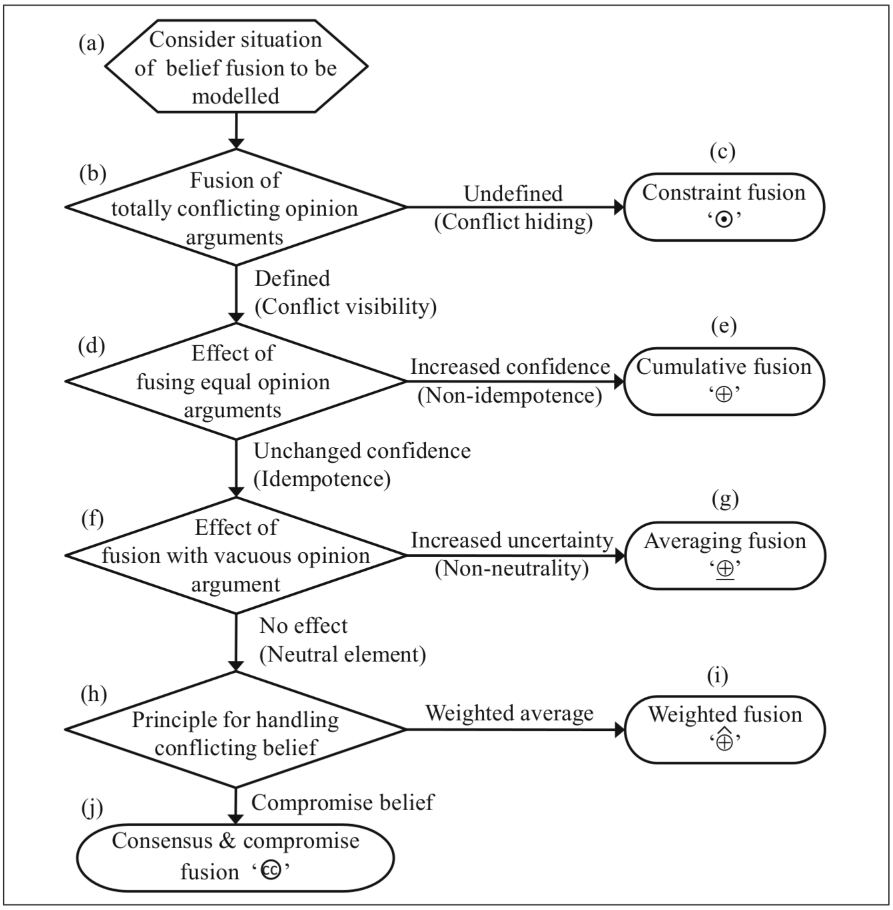

(a)
Consider situation of belief fusion to be modelled
(b)
Fusion of totally conflicting opinion arguments
Undefined (Conflict hiding)
(c)
Constraint fusion '⊙'
Defined (Conflict visibility)
(d)
Effect of fusing equal opinion arguments
Increased confidence (Non-idempotence)
(e)
Cumulative fusion '⊕'
Unchanged confidence (Idempotence)
(f)
Effect of fusion with vacuous opinion argument
Increased uncertainty (Non-neutrality)
(g)
Averaging fusion '⊕̲'
No effect (Neutral element)
(h)
Principle for handling conflicting belief
Weighted average
(i)
Weighted fusion '⊕̂'
Compromise belief
(j)
Consensus & compromise fusion '©'

Figure 2.2: Procedure for selecting the most adequate fusion operator [14]

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. 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 [14], 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. 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.

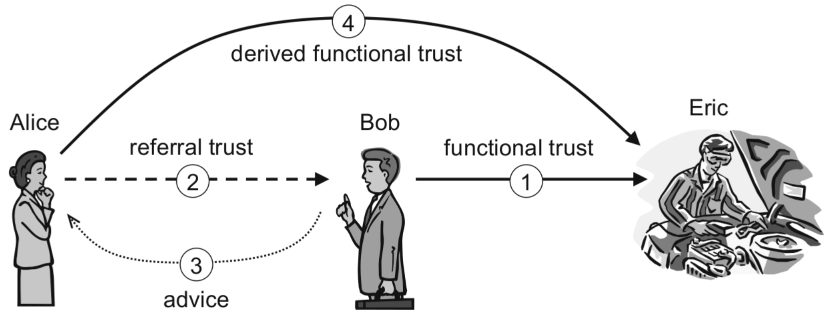


Figure 2.3: Transitive trust principle [14]

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 annual average concentration of $25\mu g/m^3$ 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 annual average concentration of $48\mu g/m^3$ and daily average concentration of $75\mu g/m^3$ 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 [22] and that done by Yanosky et al. in [23]. 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. 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 special 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 implemented most of the previously mentioned concepts of the open networks and created an actual open network [20]. 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 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 [21]. Also, there exists a registration process for the devices. In the registration process, the device is authenticated and given an ID making sure that Sybil attacks are not possible. 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 under the attacker's control 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.

3 Related Work

As mentioned in the introduction in chapter 1, the work done in this thesis we focus on detection and mitigation of false data injection attack in our system. Our literature review focused on the work done to detect false data with systems 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 having information on the two points of the bus, the generator and the consumer.

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 securely and easily in new areas. However, requiring some nodes to be secured first will hinder the secure extension and complicate the procedures. As we will have to choose between having secure systems or extendable systems. 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. [18] and Hao et al. [10] is residue test. However, the used method is inapplicable in our case because of the difference in the system information where they can sense production. However, we do not have consumption here and we are only monitoring matter production. As mentioned in the work proposed by Rahman et al. in [18] they use the information from both directions of the bus 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 the particulate matter level in the area as a reference. However, we do not want the model to be the only source 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. [25], 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 approach. And also our network is a sparse network, while the nodes here try to reach a consensus. Their implement by taking the readings of the nodes that send the exact same report. 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. [17], they have aggregator nodes that construct Binary Merkel Hash Trees (BKMHT) 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. [13], 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 System Design

4.1 System architecture

In figure 4.1, We explain the complete architecture of our system. As clear from the figure, our system consists of four modules, has one type 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. The desired output from the system is a final estimation of particulate matter level in a certain area for the last hour 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 will be discussed in details in the Reputation System chapter 5. Thus, we will focus only on the first two modules and the overall system behavior in this chapter.

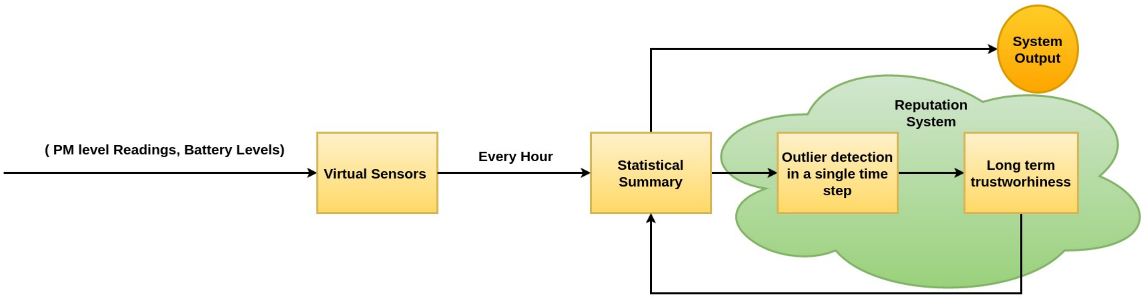


Figure 4.1: System architecture

4.2 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.

One example to that is equation 4.1, where R_A represents the reading of the sensor A. While ω_X^A represents the opinion of A about its reading.

$$(R_A, \omega_X^A) \tag{4.1}$$

4.2.1 Data Operations

The operations that are done on the data is fusing data together when they are the estimations of different sources for the same area. This is done by calculating a weighted average or weighted deviations. As the particulate matter is normally distributed, this will be proven later, thus we chose the mean as an estimation of the particulate matter level in the area. The weights for every reading in these operations is the expected probability of the opinion paired with these readings.

4.2.2 Opinions Operations

While there is more variety in operations done on opinions paired with the readings. The operations on opinions help update opinions during transition periods for the actual data. The two operations done are opinion cumulation 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.

In equation 4.2, we can see an example of the two parallel operations of data fusion and opinions cumulation to produce an average estimation and an opinion about it.

$$(\frac{R_A * P_A + R_B * P_B}{P_A + P_B}, \omega_X^A \circ \omega_X^B) \tag{4.2}$$

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.

In equation 4.3, we see an example of data transfer from one of the sensors A to the server. Where the server just takes the same exact reading. However, it discounts sensor's A opinion about the reading by its own opinion about sensor A.

$$(R_{S_A}, \omega_X^A : \omega_A^S) \tag{4.3}$$

4.3 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.

4.3.1 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.

4.3.2 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 errors in the reading such as having low power which is the only handled case in our work. The components of the equation to form the opinions are explained in details in equation 4.4. This equation was taken directly from the work done by Gomez et al. in [8]. This opinion is referred to as $\omega_{battery}$. 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.

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

$$\text{where} \begin{cases} b = (battery/100) - \frac{100*\alpha_{battery}}{battery} \\ d = 1 - b - u \\ u = \; battery\, uncertainty \end{cases} \tag{4.4}$$

4.3.3 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's trust would be 100% in 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. We chose to aggregate the data over each hour to have enough window so that no sudden event that produces a huge amount of particulate matter is able to affect our system.

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.

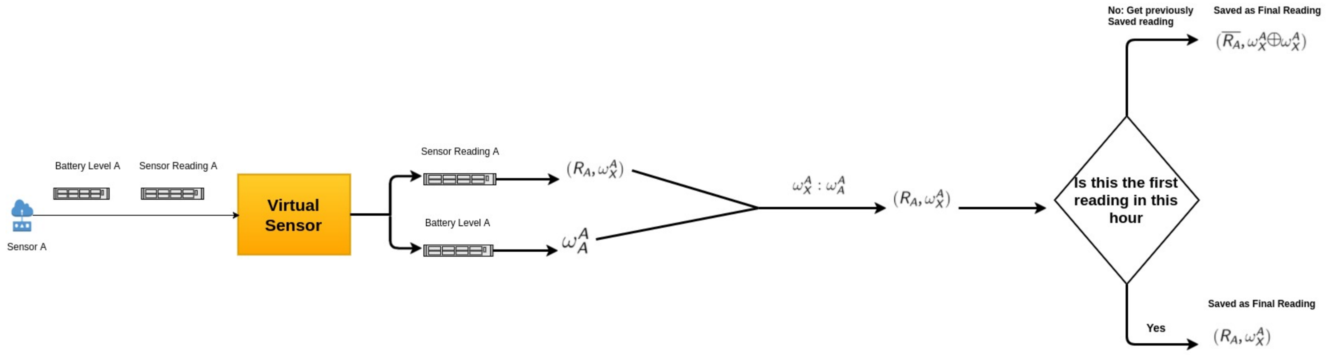


Figure 4.2: Input Readings Handling

4.4 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. Noting that the server keeps opinions about each sensor to represent their reputation. How these opinions are formed will be explained in the Reputation System chapter 5. We take the opinion of the sensors about their readings. 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.

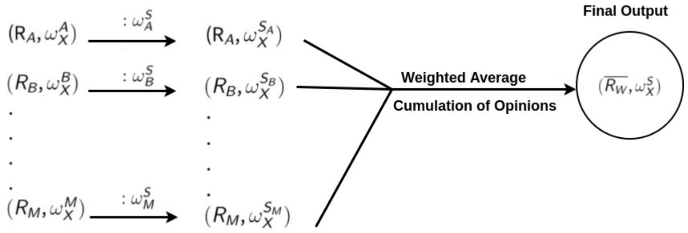


Figure 4.3: Input Readings Handling

5 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. The server keeps an opinion about every node that represents the trust of the server in the node in this list.

5.1 Concept

5.1.1 Outlier detection in a single timestep

As mentioned in chapter **3**, when explaining the work proposed by Huang et al. [13]. 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 and have already been explained in the *System Design* chapter 4. 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. [12] 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. [22] 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 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.

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 the 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 it as an attack because they lead to a false estimation of particulate matter within the area. So the effectiveness comes as every distribution with false data will be distorted which makes it harder to get evidence from it, this will be further elaborated 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 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 5.1. 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. Also, $\omega_X^{S_M}$
represents the accuracy of the model.

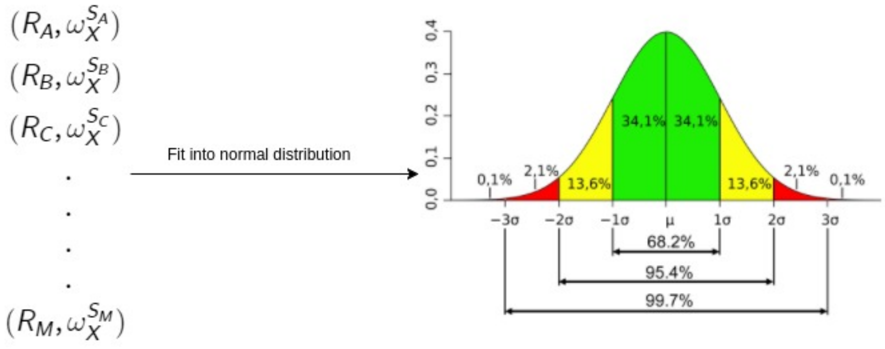


Figure 5.1: Outlier detection test for node Y

5.1.2 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 added to the evidence from the previous time steps to update the opinions. The more recent the evidence is the more weight it holds.

5.2 Implementation

5.2.1 Outlier detection in a single timestep

When fitting the data into normal distribution we faced a problem. As explained in 5.1, 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 5.1 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 enormous 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 5.1 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 initial tests results and we found out the value that showed the best mitigation as we tried values between 1 and 2 for the marker. 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 5.2. 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.

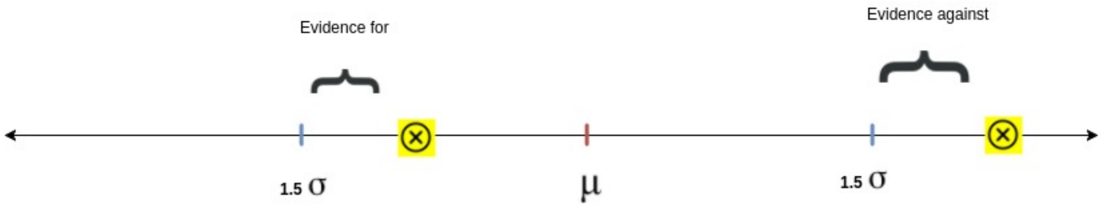


Figure 5.2: Evidence calculation in a single time step

We also had a second approach that we decided to test. 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 to have a moving marker.

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 5.1. 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. [16], 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 close to an explosion 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.

$$marker = 2\sigma * (1 - \frac{\sigma}{\mu}) \pm \mu \tag{5.1}$$

5.2.2 Trustworthiness over time

For forming long term trustworthiness with the more recent data holding more weight for the system to be adaptive and interactive. 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 [14]. A vacuous opinion with 0 belief, 0 disbelief and 1 uncertainty as in equation 5.2. Also, the base rate is 0.5 because with no available information, the possibility of the node being an attacker is exactly the same of it not being an attacker. We do that with every new node joining the system.

$$\omega_X^{S_A} = (0, 0, 1, 0.5) \tag{5.2}$$

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. These disadvantages do not exist when using 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 5.3 and 5.4. Noting that in all of this thesis the term decay factor represents the factor by which we multiply the old evidence at each time step. Hence, when we say later that the decay factor decrease, this means having smaller weight for the old evidence as the factor multiplied by the old evidence decrease.

$$r_x = \alpha * r_x + r_{x_t} \tag{5.3}$$

$$s_x = \alpha * s_x + s_{x_t} \tag{5.4}$$

5.2.3 Blocking attackers

When implementing the system, we faced a problem that an attacker node could still affect the system's output no matter how much we distrust it. This is done by sending very large readings with respect to other nodes. Thus, to keep our high robustness and low robustness cost. We decided to block nodes with an opinion about it having $belief < 0.1$ and $uncertainty > 0.5$. However, this blocking is done when calculating the final estimation only. While at the time of updating the reputations, its reputation is updated but it does not participate in updating the reputations of others.

This approach was perfect for our goal of having high robustness and low robustness cost. As when a node that is honest, but there are unusual events close to its location causing it to report readings that do not represent the PM level in the area will be blocked. However, it can come back automatically when it starts reporting more accurate readings. Also, the low belief threshold ensures that only very aggressive attackers are blocked.

5.3 Challenges

5.3.1 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 prediction. 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. [24] 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. [12], they developed a model for predicting the yearly average of PM2. 5 level in the area. We thought there might be 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.

5.3.2 Hardware

We planned to implement the hardware wireless sensor network, since most of it can already be hosted on TheThingsNetwork resources, and use it for testing. However, 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 of the hardware implementation is to configure and run our application on their backend. We got this part running and the application was ready. However, the second part which was to deploy the sensor and send data. 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 of the transmission technology configuration. However, the available hardware for us was a *waspmote PRO v1. 2* microcontroller. Our transmission module is RN2483 LoRa wireless transmission module.

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 substituting the functions called from the *<Arduino. h>* file with functions that exist in the waspmote. While the second solution would be to get the configuration for the LoRa channels used by the TheThingsNetwork. And configure the transmission channels ourselves. However, we could not find resources that explained the required configurations. And since this was not a problem of the highest priority because we had an alternative data source. Therefore, we decided to focus more on the experiment rather than implementing the hardware.

6 Testing Setup

6.1 Overview

The testing is done by simulation. We have data that represents the reading of a real network. The readings represent the particulate matter levels in the city of Ulm, Deutschland. The data is plugged into the application prototype. Thus, we only changed few lines in our code to handle the different input format from the actual sensors readings being sent in a message.

6.2 Data

The data is obtained from the archives of LUFTDATEN SELBER MESSEN [15]. It is collected by the people of Ulm. The application to collect the data is a project of the Open Knowledge Foundation Germany (specifically the OK Lab Stuttgart). They also link to the official daily average from the environmental office of the state government. We used their data collected starting from 8^{th} of June 2017 till 8^{th} of July 2017.

However, we did not have the battery readings for these sensors. Therefore, we assumed that all the sensors had full batteries at the time of testing.

The data is saved in CSV files. This data is saved as a CSV file for each sensor in each day. The file contains the longitude, latitude, timestamp of the reading, PM2. 5 level reading and PM10 level reading. We implemented our system using PM2. 5 level. We took this decision because we were still investigating the possibility of using the work done by Henderson et al. [12] as an implementation for the environmental model. And in their work, they focused on the PM2. 5 level. And we do not expect to need any modifications in the system for as the only required property we have is that the data follows a normal distribution which is true for both PM2. 5 and PM10 as explained in chapter 5.

6.3 Attackers Classification

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. Thus, we have four possible combinations and attacker would be one of them. The four types of attackers are continuous clever, continuous random, periodic clever and periodic random.

6.4 Output

As we previously discussed, the main outputs that we are keeping track of our final estimation of the particulate matter level in the area and the final subjective opinion about the estimation. We also keep track of the reputation list containing the trust of every node during the run of our system. We were recording this to help us better analyze the results. However, we do not discuss this extensively in our evaluation chapter 7. The reason for doing this is because the information we need to present is implicit in our final opinion. As we do not really focus on the blocking parameter. The output data are written in CSV files.

6.5 Attack Scenarios

6.5.1 Overview

At the beginning of the run, all of the nodes start reporting honest readings. No nodes join midway through the runs. All of the nodes stay honest for 50 hours after that the attacking nodes start attacking. All of the attacking nodes start injecting their false data at the same time. We took this case because that is the critical point where the attacking node is a trusted one at the beginning of the attack. And it is most effective when all of the nodes attack at the same time.

Every attack is tested for different attackers node to total number of nodes. The different ratios $\epsilon\{1/11, 2/11, 3/11, 4/11, 5/11, 6/11\}$.

6.5.2 Random Continuous attacks

In the random continuous attack, our attackers injected random false data instead of the actual readings. They did that continuously after the 50 hours at the beginning we use to stabilize the system, meaning they do it for every single reading. The tested random readings $\epsilon[0, 50]$. We chose 50 because it is double the permitted value for PM2. 5 by the European Union in their directive [7].

6.5.3 Clever Continuous attacks

Clever continuous attack behaves nearly in the same way as the random continuous attack. The difference is that the clever attacker tries to move the reading in a certain direction instead of just injecting random readings. They do that by taking the correct reading and adding a certain value to it.

The added values $\epsilon\{5, 10, 15, 50, 200\}$. The first three values in the previous set are chosen to see the effect of small changes. This is tested to view the effect when the particulate matter is already high and the person wants to push it over the threshold stated by the authorities. The latter two values in the set are used to test injecting greatly shifted readings where these values occasionally or rarely exist in a real system.

6.5.4 Periodic attacks

Periodic attacks have nearly the same implementation. The difference is that we do not inject false data every single time. Instead, it injects data for a certain period and repeats the same process with a certain frequency. Our frequency will be referred to as the number of attacks per day. The period length $\epsilon\{1, 2, 3, 4, 5\}$. For example, And the frequency $\epsilon\{4, 2, 4/3, 1\}$. These numbers represent attacking every 6, 12, 18 and 24 hours. For example, period length of 2 and frequency of 4 means we attack for 2 hours every 6 hours.

Though periodic attacks are nearly similar in implementation. However, we chose to only focus on the effect of their combination with significant shifting clever attacks. We have done this as we needed to focus more on something because of the output size that we got. Therefore, we needed to exclude some of the output to be able to perform a more thorough analysis. And marginal shifts with periodicity will hardly affect our system when injected in a periodic manner. Thus, we chose to focus on significant shifts which can affect the system.

6.5.5 Decay Factor

As explained in chapter 5, we have a decay factor for old evidence by which we let their effect on our opinions decay over time. On the time of testing, we were not able to argue choosing a certain decay factor. Thus, instead of choosing one decay factor in our implementation, we decided to test on more than one and observe our results. Therefore, the results that will be presented in the next chapter 7 will contain a comparison between the behavior of the system using different decay factors that $\epsilon\{0.975, 0.9, 0.8, 0.7, 0.6, 0.5\}$. Finally, as mentioned earlier in chapter 5 the term decay factor represents the factor by which we multiply the old evidence at each time step.

7 Evaluation

7.1 Metrics

We have three main metrics to evaluate our system based on them. Since we try to detect and mitigate the effect of false data injection attack our most important metric would be the impact of the attack on the final reading. We chose the root mean square error (RMSE) of the final when attackers exist compared to when they do not to represent the impact of the attack. However, we also care about the mitigation speed. Therefore, we chose the response speed as our second metric. We have two main indicators for the response speed. They are how fast are attackers distrusted and how fast do the RMSE stabilize. RMSE stabilizing means it returns to normal with the only difference in readings coming from the lack of information happening due to having a fewer number of information sources. Finally, we expected all of this to be reflected in our opinions, meaning whenever the RMSE is low, we will have low belief in our reading. Therefore in the case, that mitigation is not possible, we will know that we have a non-trustworthy reading. This means that our third metric is measuring the power of the opinions to represent the system state.

7.2 Injecting random data

Random attacks were used for the initial evaluation of the system. Note that the discussed results in the random attacks do not have the same seed. However, we depended on the fact that we have 50 possible values for the injected value. And we used the Random object on java which uses a *Linear Congregational* algorithm to produce 50 values. Since the choosing of the random values is according to a uniform distribution of all the possible values and we choose 50 times. Therefore the values chosen for each decay fa actor will not differ massively. In conclusion, we decided that for an initial evaluation the difference is not that huge that we need to go through the trouble of choosing our seeds manually.

As we suggested our first target is to evaluate the impact of the attack on our results. This going to be done mostly through graphs similar to figure 7.1. Thus, we decided to start discussing the evaluation by explaining Figure 7.1 first. First of all this graph is produced from running our system 5 times. Each time lasted for 50 hours unless stated

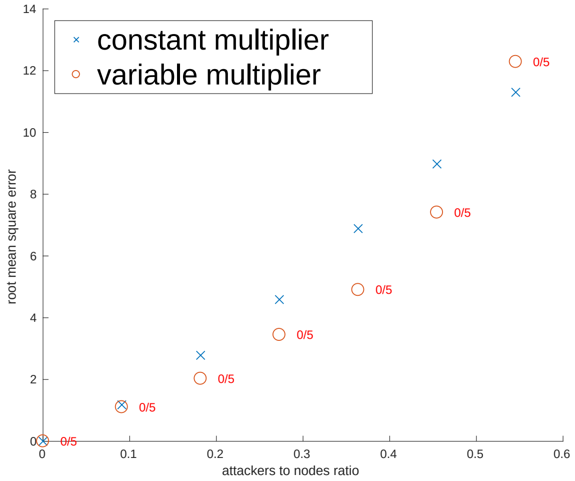


Figure 7.1: Random attack with decay factor of 0.975

otherwise. And each run started at the same point in time for all decay factors. However, each run started at a different random point in time. And we made sure that at each random point the ranges of particulate matter were different. As clear from the graph the x-axis represents the ratio of the number of attackers to the total number of nodes. And the y-axis the RMSE of the reading. The graph includes a comparison between the constant marker and variable marker approaches discussed in chapter 5. While the markers are called multipliers in the graph. The fractions besides the red scatter points represent the number of times the system completely broke and could not function properly out of the five performed runs. The cases where the system breaks down are not included in the graph and only represented in the text that states their number. If the system breaks in all the runs no red point will be drawn. Next, we explain what is meant by the system breaking down.

The system breaking down means that all the nodes are distrusted and they become blocked and no readings are accepted from any of the nodes. This is an observation we had when running the system multiple times. However, this observation was expected before implementing the moving marker approach. The explanation for this observation is that the measures taken to stop the distortion of the distribution are not enough. Therefore, when the distribution is too distorted the mean becomes in the middle between everyone and with very high deviations the markers just close up on the mean. Thus, all the nodes just start accumulating evidence against. This at the end leading to Therefore, the system breaks down completely.

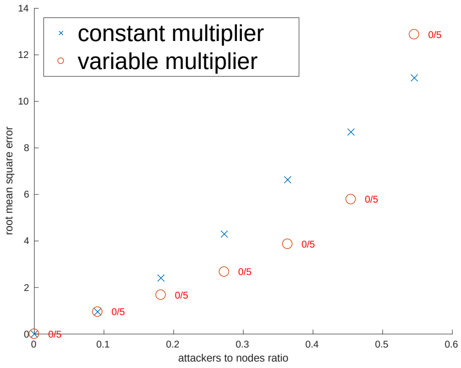


Figure 7.2: Random attack with decay factor of 0.9

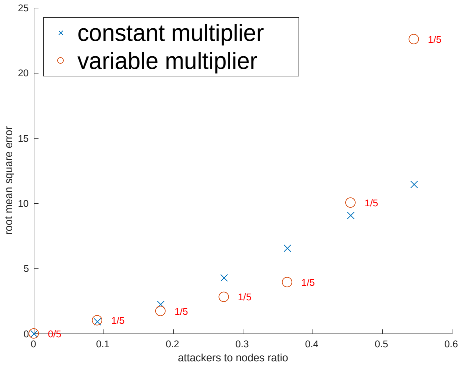
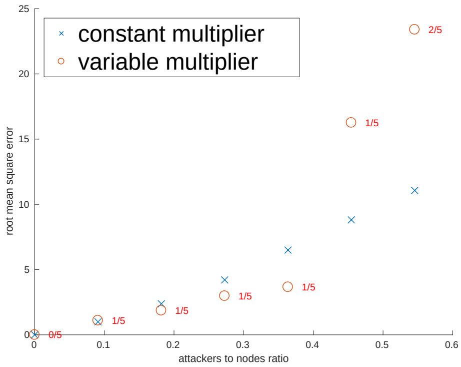


Figure 7.3: Random attack with decay factor of 0.8

The set of graphs shown in figures 7.2-7.5 help us prove and visualize the first three observed relations. The first relation is that increasing our decay factor helps the system becoming more stable and more resilient against being broken. The second relation is that the more added stability due to increasing our decay factor helps us deal with higher attacker ratios. The third relation is that the lower the decay factor the lower RMSE.



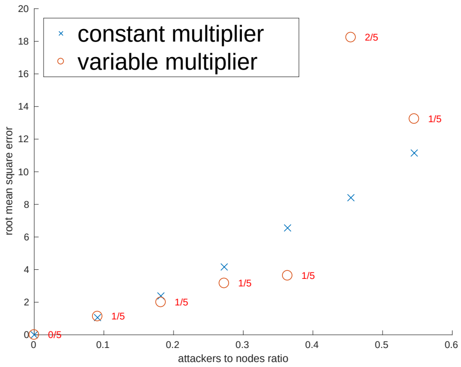


Figure 7.4: Random attack with decay factor of 0.7

Figure 7.5: Random attack with decay factor of 0.6

The first relation observed in the first set of graphs could be easily interpreted. As we explained before in chapter 5 we formed the system so that more evidence will accumulate against outliers even when all the distributions are distorted. Therefore, having a higher decay factor allows the old evidence to have more weight which gives the system more time to accumulate new evidence to even out in the weight against the old ones. This allows the system the required time to accumulate enough significantly larger evidence against the outliers to make their weight insignificant and even block them. Then the system starts reverting to normal with excluding the outliers and honest nodes start accumulating evidence for.

The third relation can be explained that the less the decay factor is, the less evidence retained from the past and the more weight given to recent observation which contains the false data. Making our system more responsive. This should be supported by our analysis of the response speed.

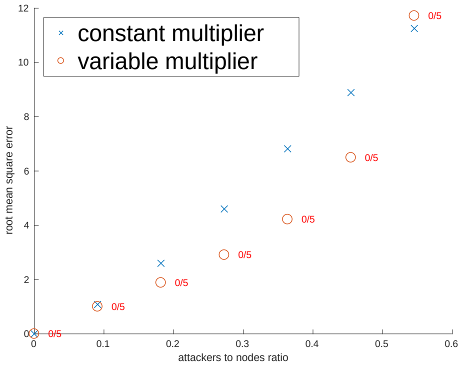


Figure 7.6: Random attack with decay factor of 0.9 in 30 hours

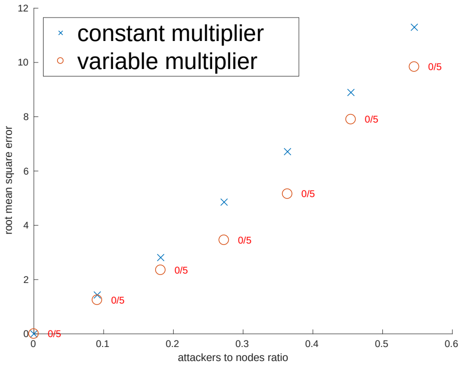


Figure 7.7: Random attack with decay factor of 0.9 in 10 hours

While in figures 7.6 and 7.7, we can observe how the system performance improve over time by viewing the RMSE for the system at different points in time. While the visible difference between the RMSE after 50 and 30 hours is very small and can be deemed as insignificant. There is a considerable difference when looking at the RMSE after only 10 hours. Thus, we can conclude that to some extent after 30 hours the system stabilizes meaning the RMSE after a certain point that precedes the 30 hour mark is generated from the difference of information source as in the attackers case, we have a significant weight for the honest nodes information, while the others are either excluded or have insignificant weight. However, in the *"attackers absent"* runs all of the 11 nodes we have report honestly. In conclusion, RMSE decreases as more time passes. However, it comes at a point where it stabilizes due to the difference in the number of information sources.

Also comparing the graph in figure 7.9 with the previously mentioned one in figure 7.7, we can find our fourth observed relation which is the response speed improves with the decrease of the decay factor. This is highly expected especially after the third relation in which the overall performance is actually better for smaller decay factors. As both

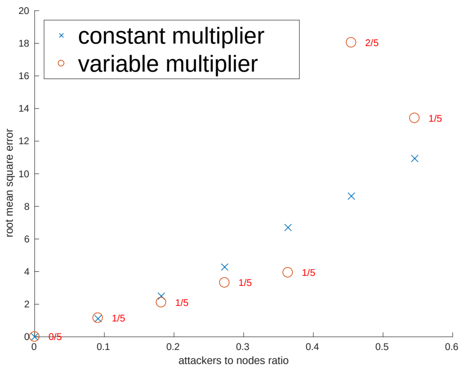


Figure 7.8: Random attack with decay factor of 0.6 in 30 hours

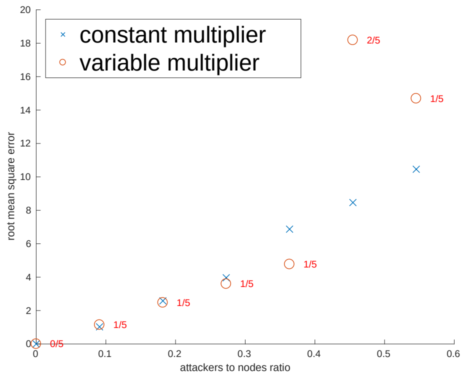
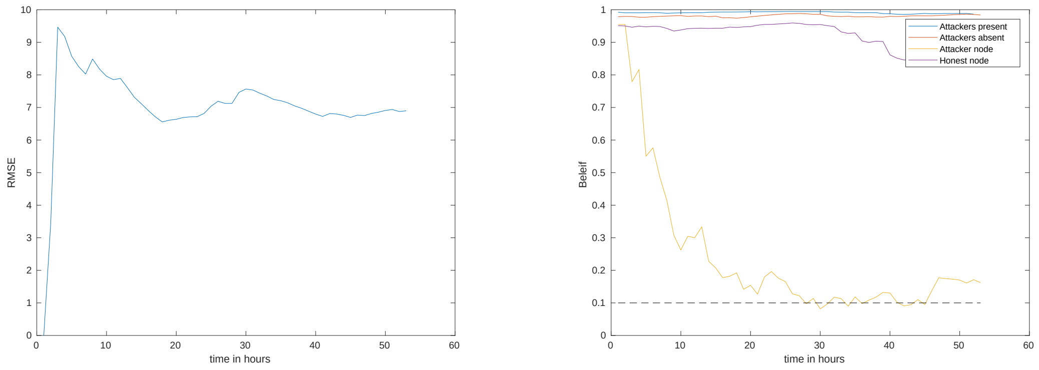


Figure 7.9: Random attack with decay factor of 0.6 in 10 hours

happen due to the fact we explained earlier that the higher the decay factor, the bigger weight old evidence has and the harder to even it out using new evidence and change the belief values.

We further elaborate the response speed and connect the opinions role, to the rest of the metrics, in judging the data through the graphs in figures 7.10 and 7.11. Noting that all of the previous figures are for the moving marker approach since it showed significantly better results. Thus, we were interested in further analyzing only the better approach for impact mitigation.

Both figures 7.10 and 7.11 have the same graphs but at different decay factors. Therefore, explaining 7.10 as an example of both would suffice. In the left subfigure 7.10a we can see how does the RMSE changes over time. While in the right subfigure 7.10b, we see the changes that happen to the final opinion about the final estimation in an environment that has attackers once and environment that attackers are absent from once. Continuing in the same subfigure, we see the changes to the trust in an honest node and attacker node in an environment where attackers are present. Noting that, these set of graphs were chosen as examples and are produced from a single run for our system.



(a) RMSE change over time

(b) Opinions change over time

Figure 7.10: Random attack with decay factor of 0.9 and attackers ratio 4/11

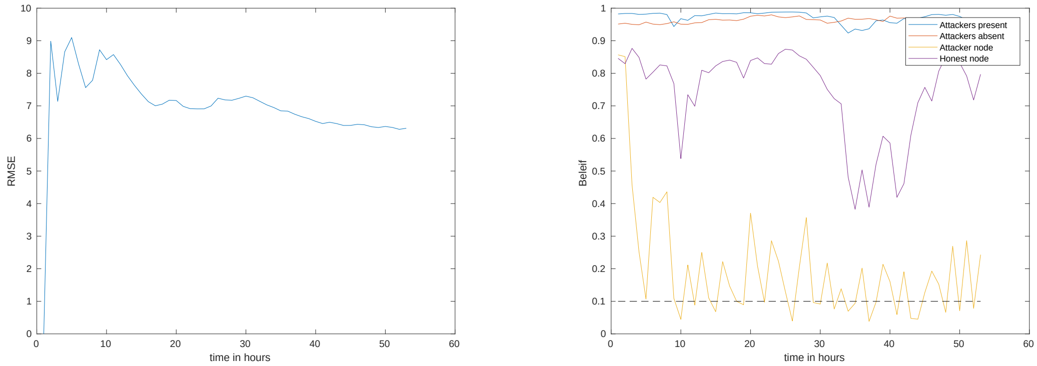
By comparing the graphs in the figures 7.10a and 7.11a, It does not really support the third relation concluded earlier. Noting that, In these figures, we could observe that for the decay factor of 0.9, the RMSE stabilized close to the 7 and for the decay factor of 0.6, the RMSE stabilized close to the 6. However, we decided that this was produced from a single run using random attacks. Which can not outweigh the result from the first set of graphs that were produced from averaging the results of 5 runs. We can also observe the fifth relation here which is the smaller the decay, the more expressive power the opinion has. And that perfectly follows the expectations we had as the decay factor decreasing makes the opinion more sensitive to change.

Next, we compare all of the graphs in the figures 7.10 and 7.11. We can easily observe the response speed by observing the slopes of the curves in figures 7.10a and 7.11a. Where in figure 7.11a, the RMSE stabilizes after 17 hours since the start of the run. While for figure 7.10a, the RMSE stabilizes after 26 hours since the start of the run. This suggests that the response speed is higher for smaller decay factors. This is also reflected in the figures 7.10b and 7.11b, where the belief in the attacker node in 7.10b falls slower than it falls in the 7.11b. All of this supports our fourth observed relation.

We can also observe from figures 7.10 and 7.11, that there are more rapid fluctuations in the belief of single nodes in figure 7.11b. This suggests the system instability which leads to the system breaking down. Therefore, these graphs support our first observed relation.

Our final observation is the dips in the final opinion and the honest node in 7.11, while both do not exist in 7.10b. This can be interpreted to support our fifth relation. However, it can be also be interpreted from another perspective that our system is too quick in its reactions as an honest node should not be distrusted and particulate matter is not a time sensitive application that we need to have such high sensitivity. However, here we go with the second interpretation because after all, we need to increase trust in the honest nodes as much as possible and decrease it in attacking nodes to get the best possible performance.

In conclusion, we observed five very important relations. The decay factor is inversely proportional to the RMSE which is the indicator of the impact of the attacks, to the response speed and to the expressiveness power of our opinions about the current events in the system. However, the decay factor is directly proportional to the stability of our system. And whenever our system becomes too unstable it becomes susceptible to breaking down completely. Also, increasing the decay factor provides better results with higher attacker ratios. And when combining the decay's factor direct proportionality relation with the expressiveness power of opinions and inverse proportionality with stability, the fact that the system becomes hyper sensitive when decreasing the decay factor to very low values is concluded.



(a) RMSE change over time

(b) Opinions change over time

Figure 7.11: Random attack with decay factor of 0.6 and attackers ratio 4/11

7.3 Injecting Marginally shifted data

In this section, we review the system's behavior when the attackers are trying to inject marginally shifted readings. We determined that adding 5, 10 or 15 to the original reading as marginally shifting the readings. We chose these values to test when the particulate matter level is already high and the attacker tries to push it over a certain threshold.

In figures 7.12 and 7.13, we can deduce two new conclusions. The first conclusion is that our system does not mitigate well the impact of such a light attack. The second relation is that the mitigation for non-aggressive attacks improves with increasing the decay factors. We can observe from the figures that the final estimation nearly moved by 10%, from what was intended by the attacker, for $1/11$ attackers' nodes to total node ratio and 50% for $5/11$ attackers' nodes to total nodes ratio. These numbers are expected numbers when we do not have a detection and mitigation system with a slight improvement for higher decay factors.

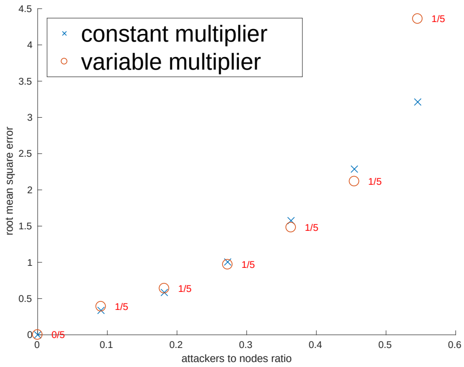


Figure 7.12: Clever attack add 5 with decay factor of 0.6

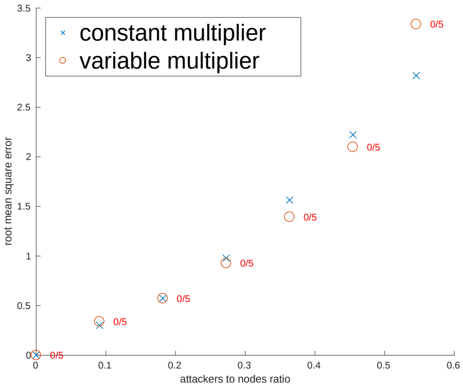


Figure 7.13: Clever attack add 5 with decay factor of 0.9

From figures 7.14 and 7.15, we can observe the system as the attacks get more aggressive. The first observation we had is that the impact is mitigated better than the previous case. This suggests that our system has better performance as the attacks get more aggressive which is a logical result. We also found that the system better performed for lower decay factors. However, we observe system instability in the case where decay factor is 0.6 as for $5/11$ attackers' nodes to total nodes ratio, the system broke 3 times out of the 5 runs we had.

While figures 7.16 and 7.17 supports the observations of the previous two sets of graphs. The observations we talk about are the better mitigation for lower decay factor as the attacks get more aggressive and the better mitigation for more aggressive attacks. However, the system is more stable than for the case of shifting by 15 than the case of shifting by 10 despite having a more aggressive attack. This suggests that the attackers were blocked faster than their ability to make the system accumulate evidence against all existing nodes.

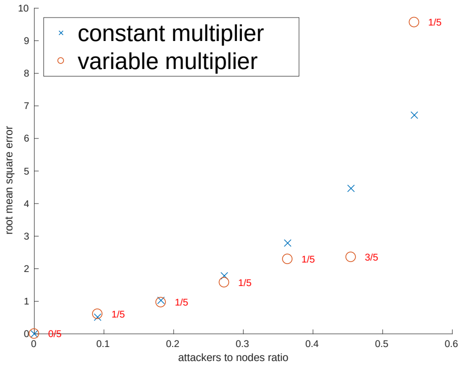


Figure 7.14: Clever attack add 10 with decay factor of 0.6

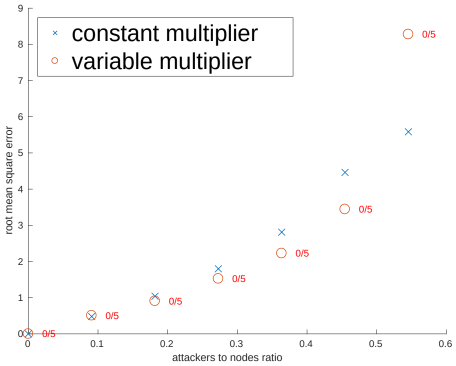


Figure 7.15: Clever attack add 10 with decay factor of 0.9

From the graphs in figure 7.18, we can interpret the reason for the previous observations. Firstly, since the shifting is only marginally the evidence against are low. This means that the evidence against can not tip the disbelief to overcome the belief quickly. This is very clear from the graph in subfigure 7.18b where clearly the trust in the attacker node falls with a gradual slope and it is never blocked. Also both graphs in the figure 7.18 show the relative few number of time steps that the attacker node is actually blocked in. This suggests that we can improve the mitigation by increasing the blocking threshold. And from the graph in the subfigure 7.18a, we can explain why the performance of higher decay factor is better. We can see that in this case sometimes adding only 5 to the reading make the new reading of the sensor approaches the value of other sensors. Thus, as clear from the graph the sensor accumulates evidence for and has some jumps to the same level of trust as the honest node.

In conclusion, We observed three relations. The mitigation of the system improves as the attacks become more aggressive. The mitigation of the system improves with the decrease of the decay factor except in extremely mild attacks owing to the fluctuations due

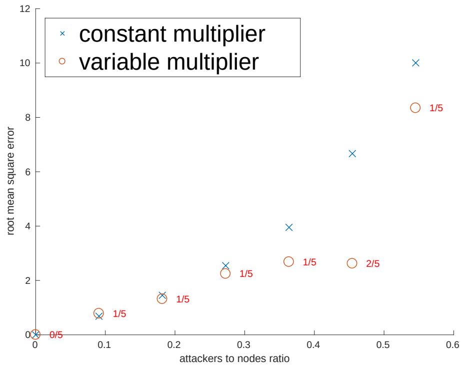


Figure 7.16: Clever attack add 15 with decay factor of 0.6

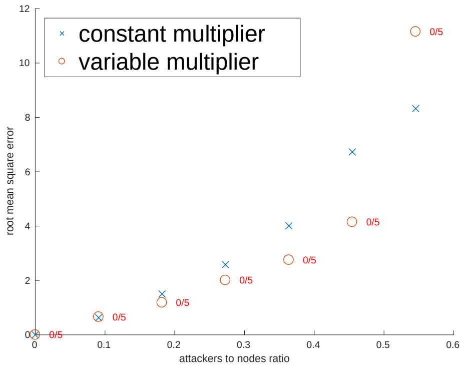
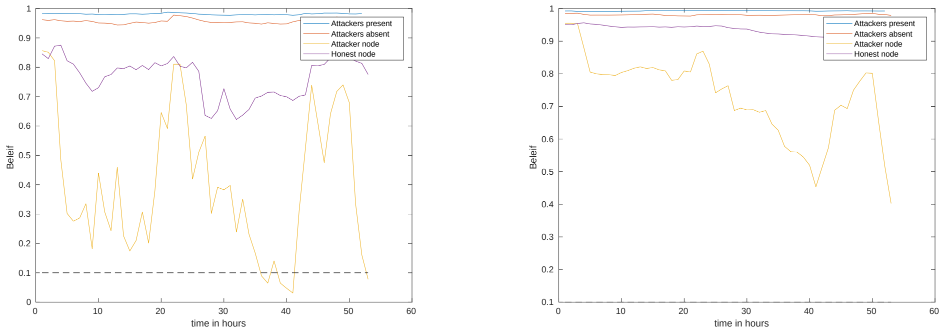


Figure 7.17: Clever attack add 15 with decay factor of 0.9

to system instability. The response speed is directly proportional to the aggressiveness of the attack.

We did not discuss the response speed extensively in this section as the results of the impact mitigation were not great. Thus, the discussion of the mitigation speed is not possible.

While clearly the system performance can be improved by increasing the blocking threshold. However, Increasing the blocking threshold require a full system analysis something to learn about other changes in this case. This is not possible in the time frame we had. Thus, we decide to leave this for future work.



(a) Decay factor 0.6

(b) Decay factor 0.9

Figure 7.18: Clever attack add 5 Opinions change over time for attackers ratio 4/11

7.4 Injecting significantly shifted data

In this section, we review the system's behavior when the attackers are trying to inject marginally shifted readings. We determined that adding 50 and 200 to the original reading as significantly shifting the readings. We chose these values as adding 50 will lead as to get readings that do not happen so often within our data set. However, they sometimes occur. While adding 200 will lead that very rarely occur.

From the figures 7.19 and 7.20, we can observe the following. Firstly, the RMSE does not exceed the 10% of the desired shift by the attacker much, for a decay factor of 0.6 and any the attackers' nodes to total nodes ratio, that does not exceed $5/11$ attackers nodes to total ratio nodes. However, the system shows instability when the attackers' ratio increase. While for a decay factor of 0.9, the RMSE keeps increasing until it approaches 20% for an attackers node to total nodes ratio of $5/11$. However, the system stays completely stable during all of the runs. All of this supports the results of the last two sections.

While by looking into the set of graphs in figures 7.21 and 7.22, we can support some of the relations observed in the last two sections. Such as the response speed is better for

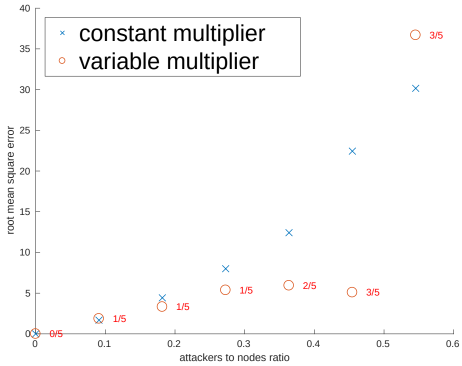


Figure 7.19: Clever attack add 50 with decay factor of 0.6

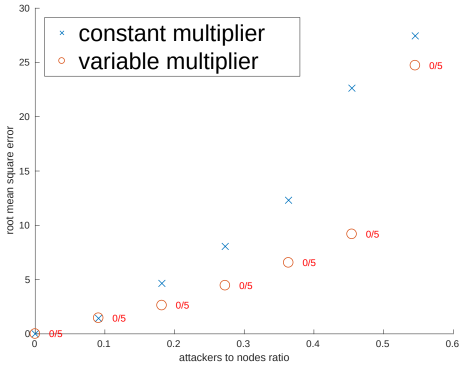
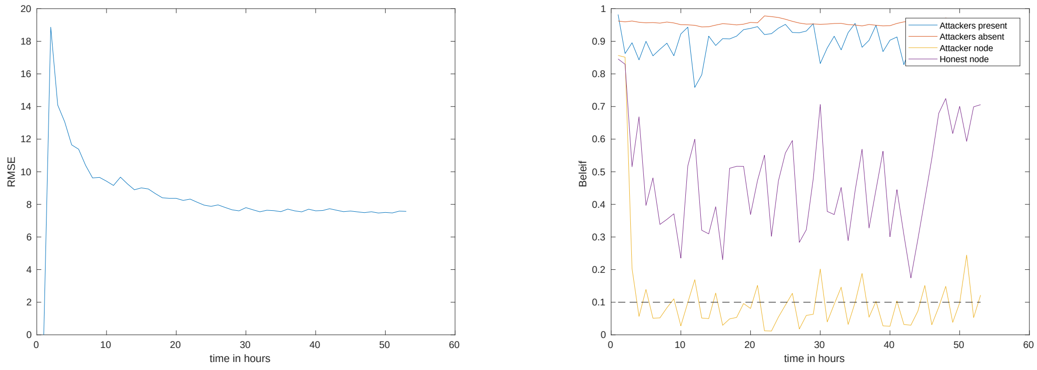


Figure 7.20: Clever attack add 50 with decay factor of 0.9

higher decay factor. This is proven by the graphs in figures 7.21a and 7.22a. Where the RMSE stabilizes in the first after 10 and after 20 hours in the second. Also, the graphs in figures 7.21b and 7.22b show also the response speed improve by blocking the attacker node faster with decreasing the decay factor and the increased stability with increasing the decay factor.

However, it is clear that by looking at the graph in figure 7.22b, you can easily deduce what is happening in the system. While looking at its corresponding graph in figure 7.21b, you will just see noise. Thus, this refutes our hypothesis that decreased decay leads to higher expressiveness power. And supporting that the interpretation that says that decreasing the decay factor to a certain limit cause the system to be hyper system sensitive more than our needs.

While clearly from the graphs in figures 7.23 and 7.24, The system is completely unstable for low decay factors in case of extremely aggressive attacks. Therefore, we will not analyze the performance in this case. However, the system mitigated the shift to only



(a) RMSE change over time

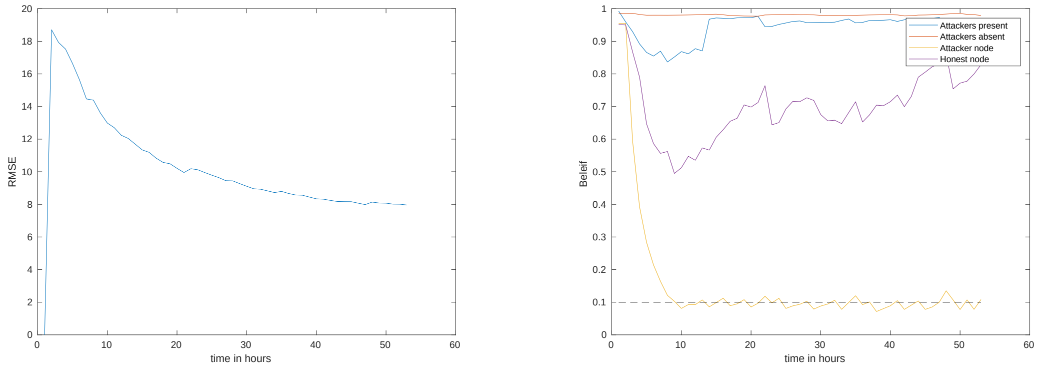
(b) Opinions change over time

Figure 7.21: Clever attack add 50 with decay factor of 0.6 and attackers ratio 4/11

10% with $5/11\%$ of the nodes are performing aggressive attacks. And with such an aggressive attack, the system showed relatively good stability compared to easier cases.

Finally from the graphs in figure 7.25, we can support our hypothesis that our system has a maximum response speed by seeing how the RMSE stabilizes after 20 hours because the honest nodes are also distrusted. Therefore the system needs time to return back to normal. However, the attacker nodes as clear from in figure 7.25b is distrusted in under 10 hours. We consider this a great result for such an organized extreme aggressive attack. Also, our claim that subjective opinion has high expressiveness power is supported in the subfigure 7.25b.

A general observation from all of the graphs that represent the impact of the attack is that for an attacker ratio of $6/11$ and a moving marker approach, the attackers clearly overtake the system and control the final reading freely. This pretty logical and expected since more than half of the network's nodes are attacking.



(a) RMSE change over time

(b) Opinions change over time

Figure 7.22: Clever attack add 50 with decay factor of 0.9 and attackers ratio 4/11

In conclusion, this section supported the following relations the decay factor is inversely proportional to the impact mitigation and the response speed. While it is directly proportional to the system stability.

However, it also refuted the relation saying that the decay factor is inversely proportional to the expressiveness power by showing that how opinions can accurately represent the system current state with a high decay factor. Thus, supporting the interpretation that was said earlier that the changes in opinion in response to mild attacks are due to the hyper sensitivity of the system. And that is not the desired behavior.

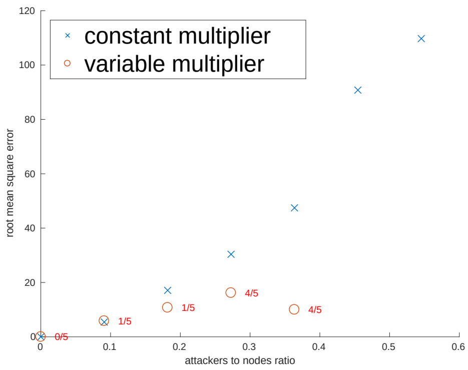


Figure 7.23: Clever attack add 200 with decay factor of 0.6

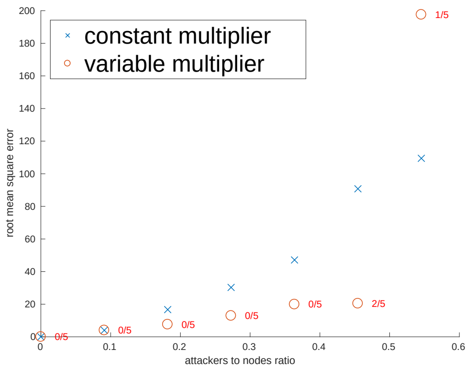
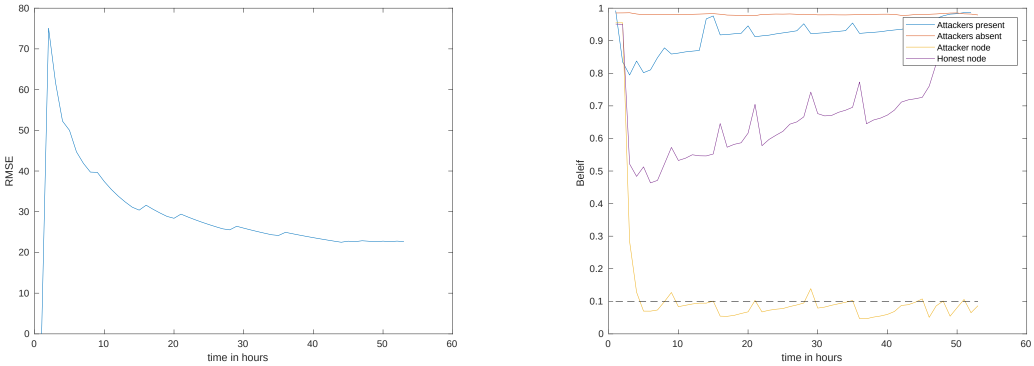


Figure 7.24: Clever attack add 200 with decay factor of 0.9

Also, this section showed great results in term of impact mitigation and mitigation speed for aggressive attacks. Suggesting that our system works perfectly fine under normal circumstances.

After seeing how the system breaks in these cases, we reached the conclusion that the system breaking is a nonlinear relation between the aggressiveness level of the attack and the way evidence are collected. However, this could not be investigated further due to the limited time frame for this work.



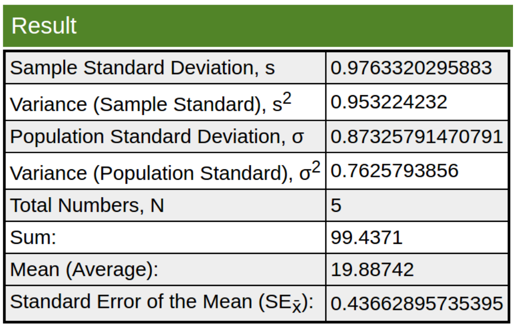
(a) RMSE change over time

(b) Opinions change over time

Figure 7.25: Clever attack add 200 with decay factor of 0.9 and attackers ratio 4/11

In all of the previous discussion, we did not state a decay factor and say this is the best. The reason for that is according to our interpretation of the results. This answer to this question changes according to the desired performance. Which is clear from our results. As for example, if we are dealing with a system that has very levels of particulate matter. We will go with a decay factor of 0.9. The best values for a specific system could be easily calculated through fine tuning and re running the test for the specific case.

Finally, we had data produced from 5 runs for our system in the previous cases the reason we did not include except the mean as an indicator of the results of the 5 runs is the fact that the standard deviation is mostly having a very small insignificant value. One example to that would be table 7.1 which contains the analysis of the differences between the 5 runs representing the clever add 200 attacks for a decay factor of 0.9 and attackers' ratio of $4/11$.



Result	
Sample Standard Deviation, s	0.9763320295883
Variance (Sample Standard), s^2	0.953224232
Population Standard Deviation, σ	0.87325791470791
Variance (Population Standard), σ^2	0.7625793856
Total Numbers, N	5
Sum:	99.4371
Mean (Average):	19.88742
Standard Error of the Mean ($SE_{\bar{x}}$):	0.43662895735395

Table 7.1: Differences of runs Clever add 200 attackers' ratio $4/11$ Decay 0.9

7.5 Periodic Injection of data

In this section, we start evaluating the system's behavior when we have a periodic injection of data. In the context of periodic injection, it is hard to define and analyze response speed. And at the same time, the opinions must fluctuate making it harder to evaluate whether fluctuations are due to system instability or just expressing the system's state. Thus, we decided to evaluate the system based impact mitigation only. Also, we evaluate the moving marker approach as it had significantly better results in the continuous attacks.

As we did before, we start this section by explaining the type of graph that we will review in all of this section. Such as the graph in figure 7.26. Each of these graphs is a specific case with a certain decay factor, added value and certain attackers' nodes to total nodes ratio. The y-axis on this graph represents the RMSE. While the x-axis represents the period length. And the legend shows the different frequencies for attacks per day tested. Finally, we need to state that if the graph showed a case where we have 0 RMSE that means that the system broke in this case.

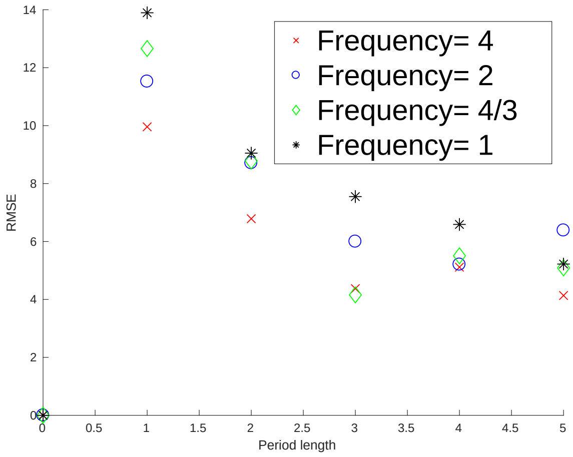


Figure 7.26: Decay factor 0.9 add 50 ratio $3/11$

From figure 7.26, we can observe our first two relations. The first is the fact the impact mitigation is directly proportional to the frequency. This pretty normal as the frequency getting larger the time between the attack and the next is smaller giving the node less time to accumulate evidence for and regain trust. Also as the frequency increase, the node approaches the continuous attack behavior. The second relation is that the impact mitigation is directly proportional to the period length. This very logical as increasing the period means the evidence against collected at each attack is larger.

And we could also note that the impact mitigation improved significantly for all periods more than one time step. This is a great result, as a period of one time step will not affect the system at all. This due to the fact that our application is not time sensitive and exceeding a threshold at one time stop will not trigger anyone to react. In conclusion for any serious attack, we have a high level of impact mitigation as the final estimation moves only by about 15% in the worst case. The same argument could also be made in most cases for a period of 2.

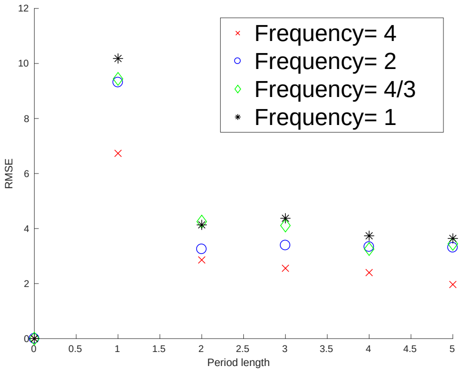


Figure 7.27: Decay factor 0.7 add 50 ratio $2/11$

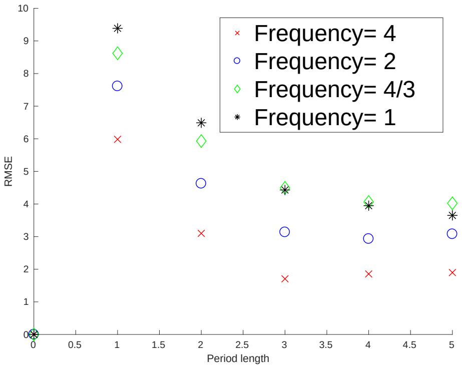


Figure 7.28: Decay factor 0.9 add 50 ratio $2/11$

From the set of graphs in figures 7.27 - 7.30, we can observe that opposite to the continuous attack the impact mitigation improves with increasing the decay factor. This is owed to the fact that retaining information with using the higher decay factor means keeping more information about previous attacks. However, we can see that as the periods get longer and the frequencies get higher this gap gets smaller until the performance at lower decay factor sometimes becomes better than at higher decay factors. This happens due to the increasing importance of reacting to the current attack quickly rather than keeping information about previous attacks.

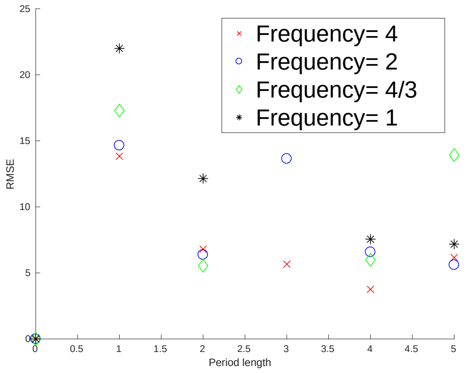


Figure 7.29: Decay factor 0.7 add 50 ratio $4/11$

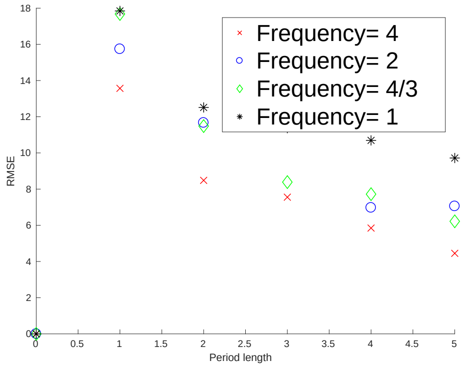


Figure 7.30: Decay factor 0.9 add 50 ratio $4/11$

From the same set of graphs, we can observe a nearly inverse linear relation between the impact mitigation and the attackers' nodes to total nodes ratio. Where at $2/11$ ratio and period of 4, the readings move with about 10%. While at $4/11$ ratio and period of 4, the readings move with about 20%. We consider a 20% change in the final readings when the ratio is $4/11$ a very decent performance.

As clear from the figures 7.31 and 7.32, we see two main observations. The first is the low decay factor starting showing some instability. The second is having a great impact mitigation as clear from the periods ranging from 2 to 4 hours where the final estimation does not move by more than 15% in the worst cases.

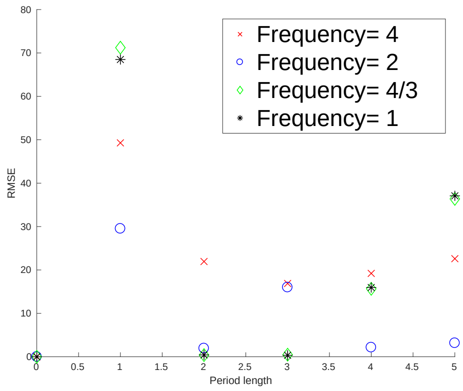


Figure 7.31: Decay factor 0.7 add 200 ratio $4/11$

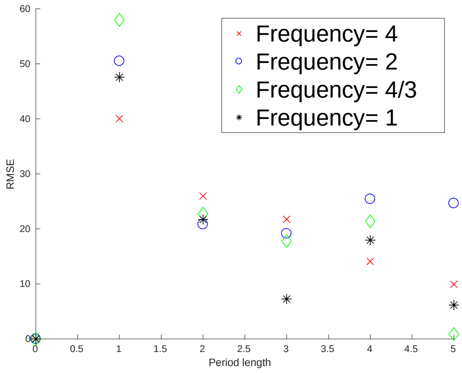


Figure 7.32: Decay factor 0.9 add 200 ratio $4/11$

In conclusion, we have five main relations. The first is the impact mitigation is directly proportional to the period length, the frequency and the aggressiveness level of the attack while being inversely proportional to the attackers' nodes to total nodes ratio. Finally, we found that for short periods and low frequencies, the impact mitigation improves with increasing the decay factor. However, the impact mitigation improves with decreasing the decay factor for long periods and high frequencies.

Also, we found that the impact mitigation is great for aggressive attacks, which were the focus of the evaluation of periodic attacks because they are the attacks that can actually affect the decision making in the system.

8 Conclusion

8.1 Results

After we have extensively analyzed the behavior of the system in the last chapter 7. We could now start discussing the meaning of the relations and observations we have found. Going back to the beginning of this work in the introduction chapter 1, where we proposed two research questions.

The first question was *”What is the robustness level of data aggregation using subjective logic?”*. The impact mitigation metric was our indicator for the robustness level. In our evaluation, the system showed a high level of robustness against random and extremely aggressive attacks and most of the periodic attacks. While it showed average performance against periodic attacks with shorter periods and lower frequencies. However, as explained earlier these attacks are not very efficient since our application is not time sensitive because we can wait for few hours to make sure the level is really a dangerous one before taking serious steps. However, the level of robustness was relatively low when it came to mild attacks. And for all cases, we had low robustness cost using the constant marker approach.

The second question was *”How efficient is subjective logic in creating a reputation system?”*. The system showed great results for forming long term trustworthiness in both points. This was clear when we kept track of the opinions about the individual attacker and honest nodes as clear from our figures in the *Evaluation* chapter 7. Those opinions showed high expressiveness to the system state. And using the operators and tools of subjective logic showed high robustness level as we have just mentioned. This means that subjective logic was able to accurately represent and update the reputations.

Now, we start discussing other applications for this system. First of all, we implemented the system in a modular fashion meaning that it could be easily modified for other applications. Such as substituting the one time step outlier detection method and putting another that produce numerical values and the system will work for another network measuring different attribute than particulate matter. Or, substituting the statistical summary module to apply different operations rather than getting an average.

However, the system is also capable of handling different applications without needing to change a thing. As this system was built only based on one property the data must follow a normal distribution. And the examples are countless for data that follows normal distribution from the temperature in an area to weights of the people.

Moreover, this system can work for the distribution of the means of many data according to the central limit theorem. Meaning, we have a wide range of applications for this system.

8.2 Future Work

After establishing the potential our system has, we now start recommending the future work that should be done to improve and better test our system.

As clear from the *Evaluation* chapter 7, the biggest problem we face is the system breaking down. It is also clear that this more likely to happen during more aggressive attacks. Thus, the first thing we recommend is implementing a threshold for normal readings and any reading exceeding it is directly excluded. This because some readings appear as sparks and have completely nothing to do with the other readings. We do not want to implement this ourselves because we prefer someone well informed in environmental engineering and studied the behavior of particulate matter to implement it. Then, the relation between the system breaking, the way evidence are collected and the aggressiveness of the attack should be investigated further.

Since we think that the system breaking down is due to the distortion in the formed distribution. We advise using a robust statistics approach beside subjective logic to ensure the distribution is as close as possible to the real data and avoid distortions in it.

Next, we predicted the improvement of impact mitigation for mild attacks if the blocking belief threshold was changed. However, this was not investigated further due to the limited time frame for this work. Thus, we recommend doing a complete system analysis while changing the blocking threshold.

After improving the system, we suggest testing it against more aggressive attacker model. Such as the alternating attackers. One alternating attacker behaves the same as a periodic one. However, the attackers coordinate with one another so that the periods do not conflict with one another. Thus, each attacker starts attacking at the end of the period of another attacker and begin at the start of the period of a third attacker. Therefore, we end up with a continuous attack without one attacker having to accumulate a lot of evidence against and each attacker has a decent period to regain trust.

We also noticed a relation between the markers and the size of the area, we are testing on. However, we did not prioritize the investigation of this relation. Thus, we see it essential as an important point to deeply understand the system that this relation must be investigated further.

After determining the perfect area size and the markers that should be placed with it, distributing the system should be looked into. This can be done by creating a cell division algorithm to virtually divide the area into cells. Some aspects to look into in this algorithm, are the inclusion of one major road per cell because in some cities such as Stuttgart streets are closed in an area when the particulate matter exceeds a certain threshold and number of sensors per cell. Also, multiple algorithms in the system to compare between different evidence for or against a particular sensor in one time step then take their average to improve the system's performance.

Finally, location attacks are to be expected if the system becomes distributed as the attacker will have the freedom to change the location of his malicious node. Thus, a mechanism to handle these changes securely should be developed.

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] R. M. Harrison and J. Yin. “Particulate matter in the atmosphere: which particle properties are important for its effects on health?” In: *Science of the total environment* 249.1 (2000), pp. 85–101.

[12] 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.

[13] 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.

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

[15] *LUFTDATEN SELBER MESSEN Archive*. `http://archive.luftdaten.info/`. 2017.

[16] 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.

[17] 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.

[18] 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.

[19] P. Schwarze, J. Øvrevik, M. Låg, M. Refsnes, P. Nafstad, R. Hetland, and E. Dybing. “Particulate matter properties and health effects: consistency of epidemiological and toxicological studies”. In: *Human & experimental toxicology* 25.10 (2006), pp. 559–579.

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

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

[22] 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.

[23] 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.

[24] 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.

[25] 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.