Sensor Shield: Multi-Sensor (Pulse & Ultrasonic) Cyber Attack Simulation and Mitigation for Critical Healthcare and Tactical IoT Systems

Project Submitted
to
RASHTRIYA RAKSHA UNIVERSITY
An Institution of National Importance

For the partial fulfilment for the award of the degree of

Master of Science in Cyber Security and Digital Forensics

Submitted by

UDAY H N
240031102551056

Under the Guidance of

Dr. Nitin Padariya
Assistant Professor,
School of Information Technology, Artificial Intelligence and Cyber Security,
Gandhinagar

School of Information Technology, Artificial Intelligence and Cyber Security

Rashtriya Raksha University
Lavad, Dehgam, Gandhinagar-382305, Gujarat, India
November 2025

Sensor Shield: Multi-Sensor (Pulse & Ultrasonic) Cyber Attack Simulation and Mitigation for Critical Healthcare and Tactical IoT Systems

Project Submitted
to
RASHTRIYA RAKSHA UNIVERSITY
An Institution of National Importance

For the partial fulfilment for the award of the degree of

Master of Science in Cyber Security and Digital Forensics

Submitted by

UDAY H N
240031102551056

Under the Guidance of

Dr. Nitin Padariya
Assistant Professor,
School of Information Technology, Artificial Intelligence and Cyber Security,
Gandhinagar

School of Information Technology, Artificial Intelligence and Cyber Security

Rashtriya Raksha University
Lavad, Dehgam, Gandhinagar-382305, Gujarat, India
November 2025

DECLARATION

I hereby declare that the project titled “SensorShield: Multi-Sensor (Pulse & Ultrasonic) Cyber Attack Simulation and Mitigation for Critical Healthcare and Tactical IoT Systems” “is the result of my own independent work” carried out as a requirement of the Master of Science in Cyber Security and Digital Forensics (M.Sc. CSDF) program at Rashtriya Raksha University, Gandhinagar, Gujarat.

This work has not been presented or submitted for the “award of any other qualification at this or any other institution” All external materials, digital tools, and references used in the documenting of this project have properly cited and acknowledged from my side The entire experimentation and testing were conducted within a virtualized environment, without utilizing any physical hardware components.

UDAY H N

Enrollment No: 240031102551056

Date:

CERTIFICATE

This is to certify that the project entitled "SensorShield: Multi-Sensor (Pulse & Ultrasonic) Cyber Attack Simulation and Mitigation for Critical Healthcare and Tactical IoT Systems" has been completed by

Mr. UDAY H N (Enrollment No. 240031102551056), a student of the School of Information Technology, Artificial Intelligence and Cyber Security, in as part of the academic requirements leading to the Master of Science in Cyber Security and digital forensics degree at Rashtriya Raksha University, Gandhinagar, Gujarat.

Date:

Place:

Signature and Name of Supervisor
Dr. Nitin Padariya
Assistant Professor
SITAICS

Signature and Name of School Director
Dr. Chandresh Parekha
Senior Assistant Professor
SITAICS

Seal of RRU

ACKNOWLEDGEMENT

I sincerely thank Rashtriya Raksha University and the School of Information Technology, Artificial Intelligence & Cyber Security (SITAICS) for granting me the opportunity to undertake this minor project as part of my academic journey.

I like to express my true deepest appreciation to my project mentor, Dr. Nitin Padariya, whose valuable guidance, continuous encouragement, and constructive feedback were instrumental in the successful completion of this work. Their technical expertise and suggestions truly enhanced my understanding and implementation of the project objectives.

I am really grateful to all my faculties and staff of SITAICS for their support, motivation, and true academic advice duration of this study helped me a lot

Finally, I extend my heartfelt gratitude to my family, friends for their encouragement, true support and ethical support, which highly motivated me to complete this project with dedication and extreme confidence.

UDAY H N
Enrollment No: 240031102551056

Date:

TABLE OF CONTENTS

Chapter 1 Introduction: ……...10-13

1.1 Background ……....10

1.2 Problem Statement ……10

1.3 Project Motivation ……..11

1.4 Objectives of the Project ……....12

1.5 Scope of the Project ……....12

1.6 Significance of the Project ……13

1.7 Limitations ……13

Chapter 2 Literature Review: ……13-16

2.1 IoT and Its Security Challenges ……14

2.2 MQTT Protocol Vulnerabilities ……14

2.3 Node-RED for IoT Visualization ……..15

2.4 DoS and Flooding Attacks in IoT Systems ……....15

Chapter 3 Methodology: ……..16-27

3.1 System Overview ……...16

3.2 System Architecture ……...17

3.3 Tools and Technologies Used ……...….18

3.4 System Setup & Workflow ……....19

3.5 MQTT Broker Setup ……..19

3.6 Creating Virtual Sensors ……....19

3.7 Node-RED Dashboard Setup ……20

3.8 Attack Simulation ……..23

Chapter 4 Mitigation Implementation……... 27-34

4.1 Node-RED MQTT in configuration with security ……..…28

4.2 MQTT Authentication (Username & Password) in configuration file ……..29

4.3 Access Control List (ACL) - Topic Restriction ……30

4.4 Iptables Based Rate Limiting for MQTT ……....30

Chapter 5: Discussion …….....34-36

5.1 Evaluation of Results ……..….34

5.2 Challenges Encountered ……..…35

Chapter 6: Conclusion and recommendations: …….37-38

6.1 Summary of Findings ……..…37

6.2 Future Work ……...37

Chapter 7: References: …….38

ABSTRACT

"SensorShield: Multi-Sensor (Pulse & Ultrasonic) Cyber Attack Simulation and Mitigation for Critical Healthcare and Tactical IoT Systems"

Submitted

By

UDAY H N

Supervised By

Dr. Nitin Padariya
Assistant Professor

Abstract

In this study, we start from a complete software-defined Internet of Things (IoT) environment. "This project is really about demonstrating how cyberattacks harm IoT devices and testing low-cost protection against those attacks." "I developed a virtual setup for health care and military systems via two software sensors: pulse and ultrasonic." This configuration aided me in experimenting with how attacks are performed and how security tools can prevent them (Gupta & Sharma, 2022) and "The Internet of Things continues to expand rapidly — over 15 billion connected devices were estimated by 2025" (Statista, 2025).

Experimental/Computational work done

Two virtual sensor modules were created employing Python to simulate pulse rate and distance measurements. At the heart of this data link, you have MQTT messaging protocol, and at this point it was using Mosquitto as a go between broker, to allow for information to pass between the virtual devices and the Node-RED Dashboard.

We tested the simulated network against various cyberattacks such as false data injection, topic hijacking, DoS attacks, and rogue device impersonation. User authentication, Access Control List (ACL) policies and iptables-based traffic rate limiting were set up to enhance data transmission security and verified.

Results and Discussion

During the first test, without these security layers in place, adversaries were able to manipulate or flood sensor communications. With an authentication by adding the username and the password also including ACL configurations in place with iptables integration made project possible in result I found that unauthorized connections were rejected where it is not at all allowing and rate limiting handled excessive traffic loads with direct rejection, so this only allowing or validated sensor readings appeared in the Node-RED dashboard which is only genuine, so this shows enhancing the general integrity, where it is not changing in between reliability, through out and stability of the virtual IoT network throughout.

Conclusions

The SensorShield project confirmed me that it is possible to implement a very strong robust defenses against common MQTT threats, which is very common and trending such as DoS, which is denial of service topic hijacking, and even data injection, by merely using appropriate and well implemented security protocols that are not complex which looks simple in nature. System reliability and data integrity with confidentiality were significantly increased after the intentional adaption of ACLs user authentication, and iptables based connection rate controls which is truly impressive

These measures held the core cybersecurity principles of integrity where it is not changing in between with the good accesses availability making it available all the time in consistent manner and confidentiality keeping it more confident by restricted access to sensitive data. This validates that effective security in IoT platforms is achievable with proper configuration and constant surveillance where it will be observed if it's in large critical infra without resorting to complex or high-cost commercial solutions with the possible low-cost solutions. With further growth in IoT adoption because as the tech improves the problems also increases side by side, such a practical and multi-layered security approach will defiantly continue to remain integral to achieving safety, reliability, and trust Worthey in smart systems which is really tremendous in nature

Chapter 1
Introduction

Background

An IoT is a network of various devices which are connected and in that sensors that are connected each other to share data easily which makes the work easier also make to be in vulnerable side by side which is not good because of COST having very low security implemented in it may Couse disaster if its ignored This structure promotes sector to sector automatic data exchange continuously to mention the consistent which has been proven to increase overall efficiency and user accessibility significantly (Singh, 2023). The devices in this connected structure have the capability to create and distribute and process data themselves without any interference from humans by Automation operations efficiency will be in peak and accessibility for users have been developing incredible due to this self-processing of data in major sectors such as healthcare and most importantly in defense also in smart infrastructure and making the things easier and faster

The broad security and privacy issues in IoT are well documented and require layered defenses across devices and networks (Abomhara & Køien, 2015)

Despite the many remarkable advantages of IoT expansion, it is highly vulnerable which makes attacker easy. The reason is that most IoT systems use some lightweight communication protocols such as MQTT (Gupta & Sharma 2022) this makes them so vulnerable to threats like data tampering because of low energy also topic hijacking and denial of the service attacks that result in false sensor readings which is not good so devices being controlled unauthorized, or even system downtime which makes availability down. This demands the urgent need for strong IoT security frameworks which is truly needed in current cybersecurity research.

Problem Statement

IoT is a network of several devices which are connected and sensors interconnected together for easy data sharing without the human intervention This architecture facilitates continuous and automatic exchange of data across industries, thus improving overall efficiency with good time management and accessibility to users significantly (Singh, 2023). The devices within the architecture can create also share the same and process data all by themselves with no human interference in this work

Studies show that more than 70% of IoT systems in the healthcare domain transmit sensitive

data without encryption which is highly risky in nature making them primary targets for interception and data tampering (Fernandes et al., 2017) so This autonomous processing are well capability of the created data through devices leads to massive improvements in automation and operational efficiency user accessibility across health and also some in defence and intelligent infrastructure among other areas also.

Inspire of all these great advantages the expansion of IoT is being restricted due to its increasing vulnerabilities. Only because of ignoring the security IoT systems which normally rely on lightweight communication protocols like MQTT and others which also makes it even more lightweight are particularly vulnerable to data tampering topic hijacking and DoS attacks (Gupta & Sharma, 2022). These can result in incorrect sensor readings unauthorized control of devices or even system downtime hence resulting in an urgent need for strong IoT security frameworks but at the end it always depends on user usage and human responsibility too in current cybersecurity research.

Project Motivation

My inspiration for the project came from the rapid adoption of IoT technologies in mission critical domains such as healthcare systems autonomous vehicles which is going in high speed also during my 2nd semester while in class Prakruti Parmar Assistant Professor madam thought about these problems which are arising in the real digital scenarios and loopholes in the IOT also she mentioned these are the future attacks where it is not concentrated much and also in defence infrastructure. As these environments heavily rely on real time data which is highly precious from so all everything is interlinked sensors slight misconfiguration or modification of information may lead to serious of huge disaster consequences so need to focus regarding safety and overall operations which is highly needed

Such would be the case when a medical device provides incorrect health data may lead to death if the individual or if a car wrongly evaluates distance readings by chance (Bujunuru, 2025). In light of such scenarios this project aims to evaluate security gaps in IoT through practical but few are currently in use already experimentation and to develop a virtual secure framework in the controlled environment this is capable of demonstrating attack scenarios along with explanation and understanding the defence measures against cyber attacks on the IoT ecosystem which is ongoing day by day it is improving.

Objectives of the Project

The main key goals of the project is to develop a working and simulated model of IoT communication using virtual pulse and ultrasonic sensors for full-scale testing and evaluation. It is to study and recreate various attack scenarios, such as Man in the Middle, topic hijacking, device spoofing and denial-of-service attacks, which usually target IoT systems. Further, the focus of the project is on the development and assessment of the security measures, including MQTT-based authentication of users, access control lists, and firewall configuration using iptables, which enhance the safety of the system. Visual display of sensor communication in real-time has been done on an interactive Node-RED dashboard to demonstrate various types of attacks. The project will really contribute to improving knowledge about IoT integrity of data threats that exist and makes effective ways of mitigating them in a very good way which really plays a significant role in both academic learning as well as in the development of cybersecurity research which is highly useful

Scope of the Project

This project fully focused to develop a virtual testbed for IoT that would mimic real world practical scenarios without necessarily using hardware component in this it's not used but mimicked the true real setup The setup utilizes Python based sensor simulation scripts for pulse and the distance data while the Mosquitto MQTT broker is been used for transmitting messages between devices. Which is almost similar The Node RED interface provides a graphical user interface that allows the observation of live sensor activity which is ongoing in a good manner and real time cyberattack scenarios which came very well (Mehta & Tiwari, 2018; Nekui et al., 2024)

The scope of this study is mainly to Development of two virtual sensors: Pulse & Ultrasonic. And carried out four simulated cyberattacks: false data injection, MQTT topic hijacking, with Denial of Service (DoS) and device spoofing. Which is held in a secured environment

Three defensive mechanisms will be implemented User Authentication and ACL Configuration and Network Rate Limiting (Linux Foundation, 2023; Open JS Foundation, 2024)

This setup is safe and repeatable and can easily demonstrated hence it allowed me to test IoT weaknesses without any risk to real devices are real time data which is secured in a good ethical way for analyzing both attack behavior and corresponding mitigation strategies together.

Significance of the Project

This project is highly relevant to both theoretical research and real life cyber security practice which we can clearly observe “By analyzing and demonstrating real world attack scenarios within the IoT ecosystem this project allows the learner and cybersecurity professional to identify system weaknesses together and implement defenses (Patel & Singh, 2024).” allows the learner and cybersecurity professional to more effectively identify system weaknesses and implement defensive tactics which is helpful

The results demonstrate that a set of basic defences such as rate limiting and access control and authentication with rules after the detection can effectively prevent network disruptions system crashes and data corruption which is real in nature It helps improve IoT resilience with the good reliability and also encourages the application of secure design practices in all the needed sectors within upcoming connected environments.

Limitations

The current system works only with virtualized models of sensors where I used the .py to do that and no physical hardware is involved which will be implemented in further cases The current security design relies on only basic network protection and authentication which is practical which I learnt during my academics Advanced encryption technologies such as TLS or full end to end encryption were beyond the scope and thus intentionally omitted where I dint implemented that Only MQTT protocol based attack and defense scenarios were considered while other common IoT communication protocols were not covered in this study where I felt not needed as of now in this situation

All tests were truly limited to a controlled virtual laboratory setting where it does no harm for anyone hence the results may vary once used in real IoT environments due to hardware constraints and network dynamics where I defiantly implement the same in future.

Chapter 2

Literature Review

With the acceleration of digital technologies which is truly improving every Minit the need to

protect Aswell as private information has become one of the most challenging and important topics where we need to be conceded In any industry or ecosystem which should be mandatory the increase in connected devices has drastically increased the potential attack side wise it arising surface for malicious parties which will be there always.

According to the Babar et al. (2011) IoT security can be classified across perception network and application layers each facing distinct attack vectors ranging from spoofing to denial of the service which make major.

The expansion of IoT systems in areas such as healthcare and defense comes with increasingly higher levels of device communication thus giving rise to cybersecurity risks which makes world to stay in risk condition As Kumar & Patel noted in 2021 increased exposure to multiple cyber threats creates an imbalance that makes IoT networks quite vulnerable hence protection mechanisms are really critical. This calls for strong protection mechanisms which include authentication access control and encrypted communication.

IoT and Its Security Challenges

The IoT combines multiple sensors which I mention previously control systems and smart devices linked through the internet for the faster communication to achieve an efficient environment of automation and monitoring and real time data exchange the real nature of the light communication protocols which is the main reason used in IoT therefore poses significant challenges to security and privacy were making most vulnerable

These limitations often make IoT components the most vulnerable elements within a cybersecurity framework to reiterate in the words of Singh et al. (2023)

Most IoT installations are still susceptible to unauthorized access where will get to know this by observing the previous case studies the data manipulation and network congestion because of poor settings and a lack of security measures in place Singh (2023) shows that these could bring critical systems to complete standstill.

In this respect, Kumar & Patel (2021) found that poor encryption methods and weak authentication procedures used in most IoT deployments would leave them highly vulnerable to such attacks which is most common especially via insecure MQTT configurations which attackers can manipulate to take control or inject false data within.

MQTT Protocol Vulnerabilities

MQTT works well even when network bandwidth is low where need to be focused more hence

making it a widely used protocol for small scale IoT deployments in the low budget where ignoring the security within However most research papers comment that the MQTT protocol has inherent security vulnerabilities which is there still and ongoing particularly if it is deployed using its default or out of the box configurations where need to be focused Sharma and Gupta (2021) assert

MQTT usually allows connections without authentication or encryption, making IoT systems susceptible to serious vulnerabilities, such as MITM attacks, DoS attacks, and topic hijacking incidents. Likewise, Zhou et al. (2023) presented that in the absence of authentication and authorization mechanisms and when adversaries are capable of injecting the falsified data or subscribe to sensitive communication channels, the confidentiality and integrity of the transmitted information can be severely compromised. recommended establishing secure channels of communication through the use of TLS and access control lists. Nevertheless, these enhancements lead to increased computational overhead, mostly impractical in resource-constrained IoT devices.

Accordingly, the present project focuses on the implementation of lightweight yet effective protective mechanisms, including user authentication and firewall-based rate limiting, to balance security and system efficiency in virtual IoT environments.

Node-RED for IoT Visualization

Node-RED provides a simple, intuitive visual tool for connecting, processing, and visualizing IoT sensor data in real time. (Open JS Foundation, 2024). (Brown and Morrison, 2021),

Node-RED was used in this work to visualize live data streams from the MQTT broker, including legitimate sensor readings and maliciously altered data produced in attack simulations. The various dashboards developed in this article vividly demonstrate how the IoT data integrity is affected during cyberattacks and also how security measures can restore it to normal functioning condition.

DoS and Flooding Attacks in IoT Systems

DoS attacks can slow down or completely disrupt IoT systems through the exhaustion of key resources like network bandwidth and processing power. Generally, attackers achieve it by sending an extremely high number of connection requests or continuous data packets that overwhelm the broker.

Flood-based attacks continue to dominate IoT exploitation trends, often leveraging unsecured MQTT ports (1883) to overload brokers and crash connected clients (Wang et al., 2020).
IoT networks relying on lightweight protocols like MQTT are particularly vulnerable which I mentioned earlier since repeated connection attempts can saturate the broker and block legitimate traffic rather quickly and makes it unable to processes the requested data in that prominent way
To mitigate such incidents researchers, recommend traffic throttling and anomaly-based detection in addition to rate limiting mechanisms (Kolias et al., 2017).
With this understanding this project applies a simple rate limiting approach to control the number of incoming connections which is also know as inbound traffics to prevent broker overload also to maintain stability and responsiveness in the IoT network even when exposed to possible attack scenarios where it shod be capable enough.

Chapter 3
Methodology

This chapter presents the design framework configuration process how I took the whole and implementation strategy adopted in my SensorShield project which is really admissible. The entire system was developed and tested within a controlled virtual environment in my local machine in a secured network without harming the others ensuring that different IoT cyberattack scenarios and their respective mitigation techniques could be easily understandable safely simulated without requiring real hardware but in upcoming days I will to do that also.

It covers the development of virtual sensors namely Pulse and Ultrasonic modules which is interesting the establishment of communication channels the simulation of an attack and the realization the security controls by making good use of the MQTT protocol and the Node RED platform. This structured approach allows a realistic emulation of IoT operations, enabling detection of vulnerabilities (Fernandes et al., 2017) and the assessment of protection and mechanisms under simulated attack conditions.

System Overview

The experimental setup includes two main virtual machines which is connected in the same network, shared but isolated safely

1. Attacker Machine: The machine running Kali Linux will execute all the simulated adversarial activities, such as the injection of fake data, hijacking topics, DoS flooding, and spoofing of devices.

2. Server Machine (Kali Linux running Node-RED) - This acts as a controlled environment for IoT and hosts the following components:

3. Mosquitto MQTT broker where it barrow the values and forward the same

4. Node-RED monitoring and visualization dashboard

5. Python-based virtual sensors emulating Pulse and Ultrasonic devices

All communication between the attacker and server happens over a dedicated virtual network using the following IP assignments:

• Node-RED / MQTT Broker: 192.168.194.132

• Attacker System: 192.168.194.133

System Architecture

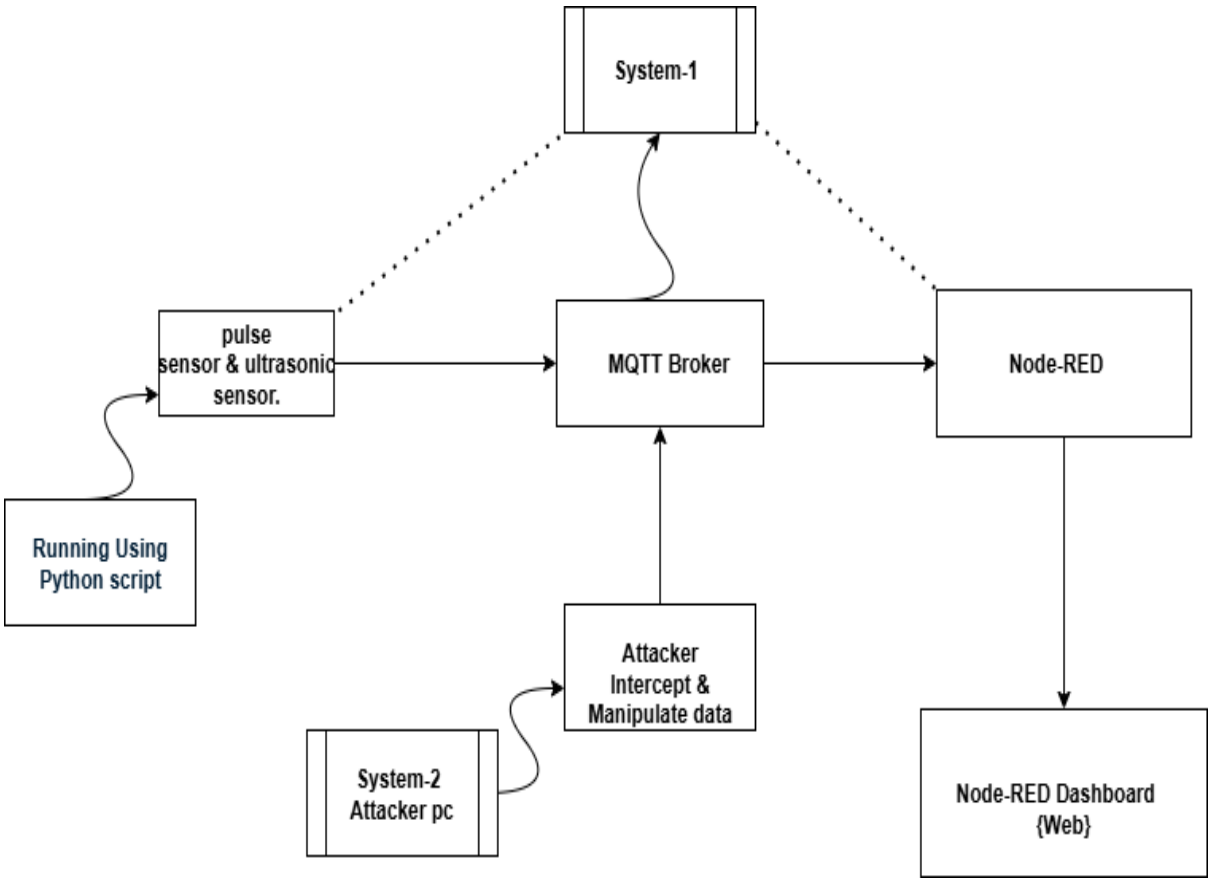

Figure 1: The *System Architecture of SensorShield*

Explanation:

1. The Pulse sensor module constantly publishes simulated heart-rate values to the MQTT topic health/pulse.
2. The Ultrasonic sensor module publishes emulated distance measurements over the MQTT topic named distance/ultrasonic.
3. The Mosquitto broker receives sensor messages and further directs them to the Node-RED dashboard for processing and visualization - Linux Foundation, 2023.
4. The Attacker machine performs various MQTT-based attack scenarios on those topics, such as false-data injection, topic hijacking, DoS, and spoofing.
5. The mitigation layer involves user authentication, an ACL, and, finally, iptables-based rate limiting to identify and block malevolent traffic, preserving data integrity.

Tools and Technologies Used

• **Kali Linux** served as the main operating environment in which both the attacker and the server systems would be simulated. It provided a secured and controlled environment to execute network-based attack scenarios along with testing the mitigation measures.

• **Mosquitto MQTT Broker:** This will provide the main communication hub for the IoT network, relaying messages between the virtual sensors and the Node-RED dashboard. Linux Foundation, 2023

• **Paho-MQTT:** Python Library are been used for publishing and subscribing to the MQTT topics; this library is used for effectively sending real-time data transmitted by simulated sensors to the broker.

• **Node-RED:** This is an open source and flow-based development tool from IBM which is most papular that has been employed to process, visualize, and monitor the live sensor readings and simulated attack traffic within the system.

• **Python 3:** Used for scripting and emulating virtual sensor behaviour, particularly generating dynamic pulse and ultrasonic readings for IoT device data.

• **Virtual Machines:** VMware provided an isolated, reproducible environment for the deployment of multiple systems without affecting the host machine, safe to experiment.

System Setup & Workflow

The following section explains the SensorShield project implementation process and operational flow. All configurations and experiments were done in a virtualized environment using Kali Linux as the primary platform and Node-RED for data visualization and monitoring. The workflow shows the entire sequence: from sensor data generation, communication via the MQTT broker, execution of an attack, to finally the application of the mitigation strategy. This approach provided a safe, controlled, and reproducible simulation of IoT-based cyberattacks with no dependence on any physical hardware component.

MQTT Broker Setup

Installed the Mosquitto MQTT broker and client tools using

apt install mosquitto mosquitto-clients

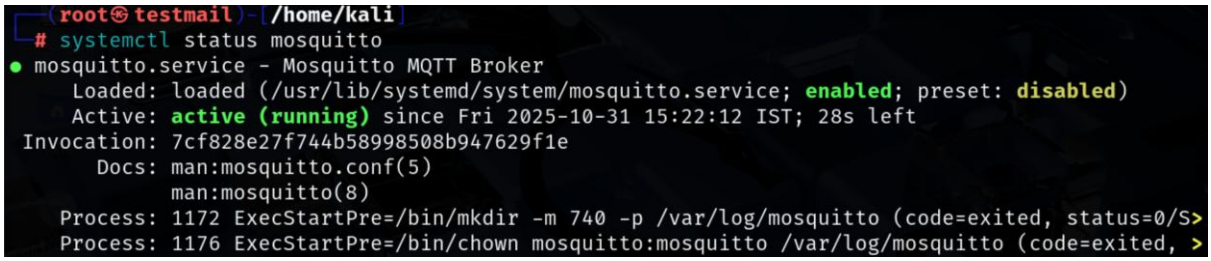

Figure 2: The Mosquitto MQTT Broker running successfully

The MQTT broker will acts as the central communicator hub between sensors (publishers) and the dashboard (subscriber).

Creating Virtual Sensors

Two virtual sensor scripts were created to simulate real IoT data

1. pulse_sensor.py generates random heart rate values between 60-100 bpm.

This makes random value creation in between given numbers that is 60 to 100 and this is adapted from to simulate the normal human pulse rate in between but it can also be changed on the basis of the requirement in real senser it actually words basis of pulse beats

Figure 3: Python script simulating pulse sensor data publishing to MQTT broker.

2. distance_sensor.py generates random distance readings between 5-150 cm.

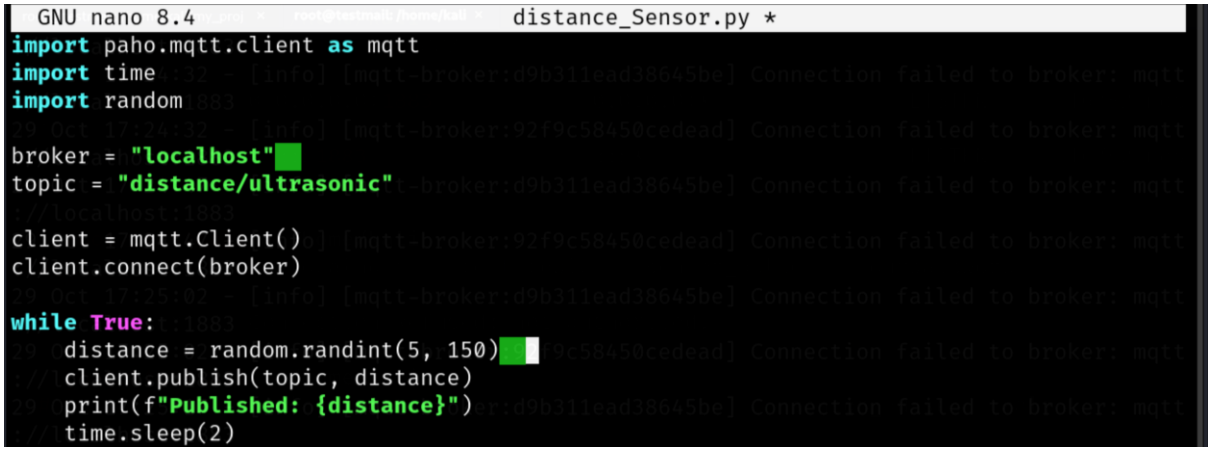

Figure 4: Python script simulating Distance sensor data publishing to MQTT broker.

Each script publishes its data to its respective MQTT topic:

- Pulse topic health/pulse
- Distance topic distance/ultrasonic

Node-RED Dashboard Setup

Opened Node-RED (http://localhost:1880) and created two flows: one for pulse sensor and one for distance sensor.

- Added MQTT input nodes to receive messages from both sensors.
- Used Function nodes to check for abnormal conditions (e.g., if pulse > 100 or distance < 10)

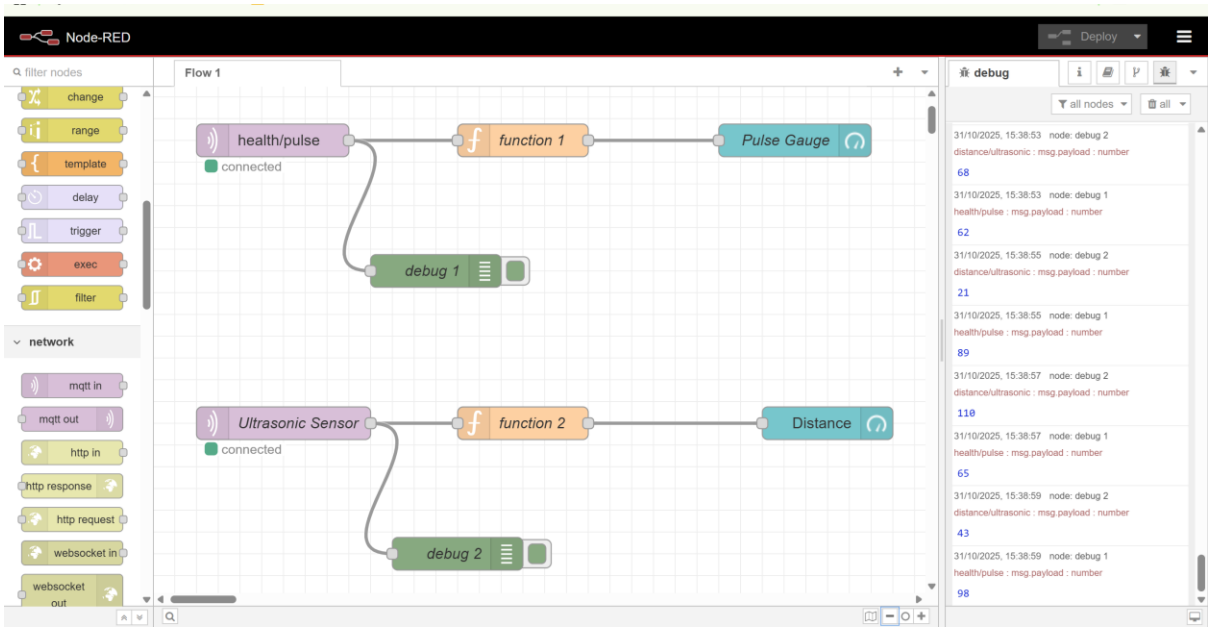

Figure 5: Node-RED flow design showing MQTT input and data processing nodes.

- Added Gauge and Chart nodes to visualize live readings on a clean dashboard.

This is the main dashboard where in a single go, we can observer both the sensers data flow and makes easier to monitor also I kept 2 sensers separately in order to mimic to the real world and makes it clear data transformation

Additionally, the internal data movement also ca be observed before the data reaches to the dashboard also it helps to observer the data folding easily in further when attacks are performed which makes doubtless

The work flow in both of the sensers is same as I mentioned earlier and makes it simpler and the .py info will be received by the sensers which are purely virtual and the function node which I have added makes the data in a structured manner and makes it suitable for dashboard where it appears the value.

In between I added the debug where it will show the data movement in between and also helps in debugging which is clearly visible in fig-5

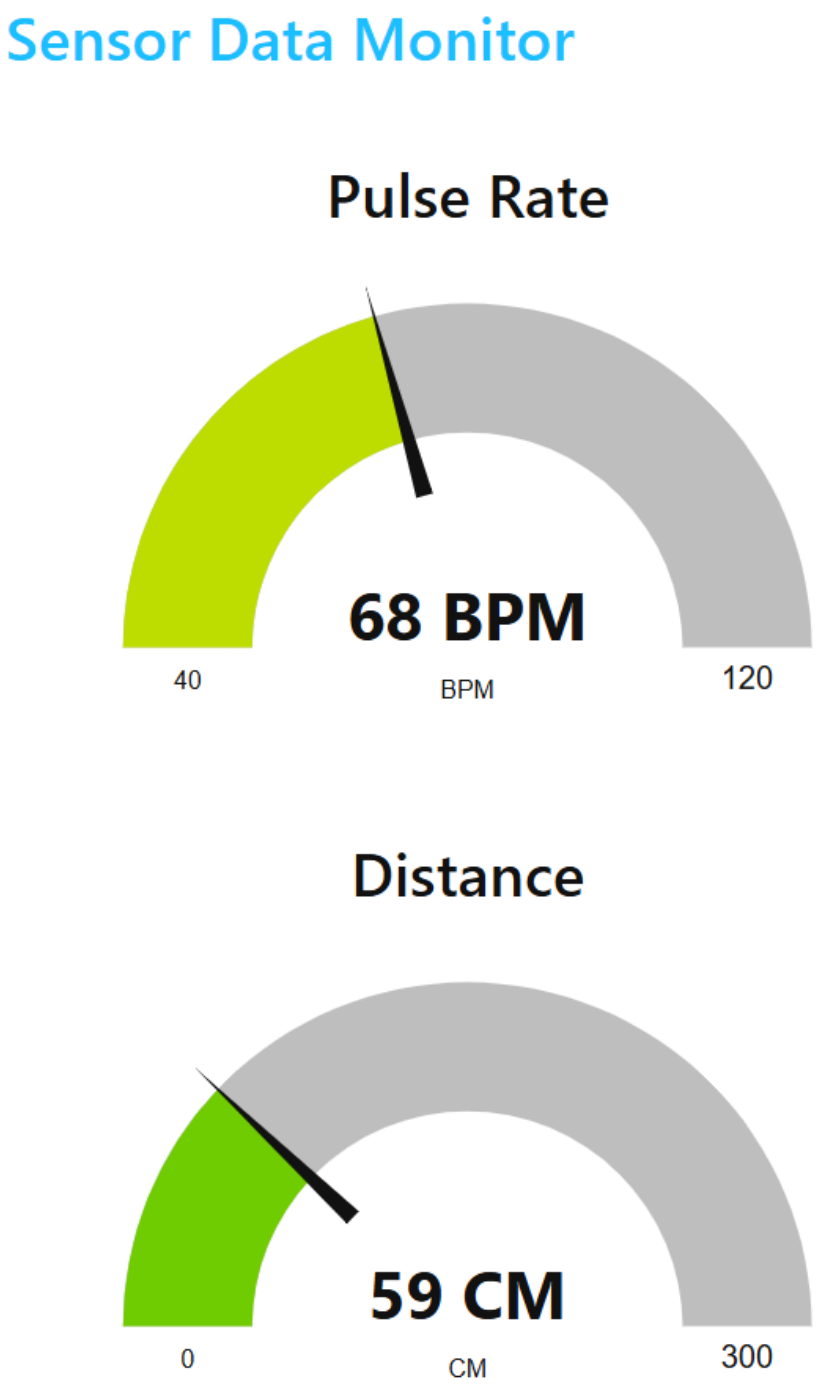

Figure 6: Node-RED dashboard displaying real-time sensor data.

The dashboard acts as the monitoring system, http://localhost:1880/ui allowing the user to observe the normal readings and notice what are all changes during simulated attacks.

Also, it varies the color if the value goes high and makes easy to notice

Attack Simulation

Different cyberattacks were performed on both topics using another Kali machine System -2

Topic Hijacking: An attacker subscribes to private topics which are present

Topic Hijacking for Pulse:

mosquitto_sub -h 192.168.194.132 -t health/pulse

Figure 7: showing Topic Hijacking for Pulse

Topic Hijacking for Ultrasonic:

mosquitto_sub -h 192.168.194.132 -t distance/ultrasonic

Figure 8: showing Topic Hijacking for Ultrasonic

Fake Data Injection (MITM) – **Injecting** or sending false sensor values.

Fake Data Injection (MITM) Injecting for pulse

- mosquitto_pub -h 192.168.194.132 -t health/pulse -m 2
- mosquitto_pub -h 192.168.194.132 -t health/pulse -m 6
- mosquitto_pub -h 192.168.194.132 -t health/pulse -m 8
- mosquitto_pub -h 192.168.194.132 -t health/pulse -m 17

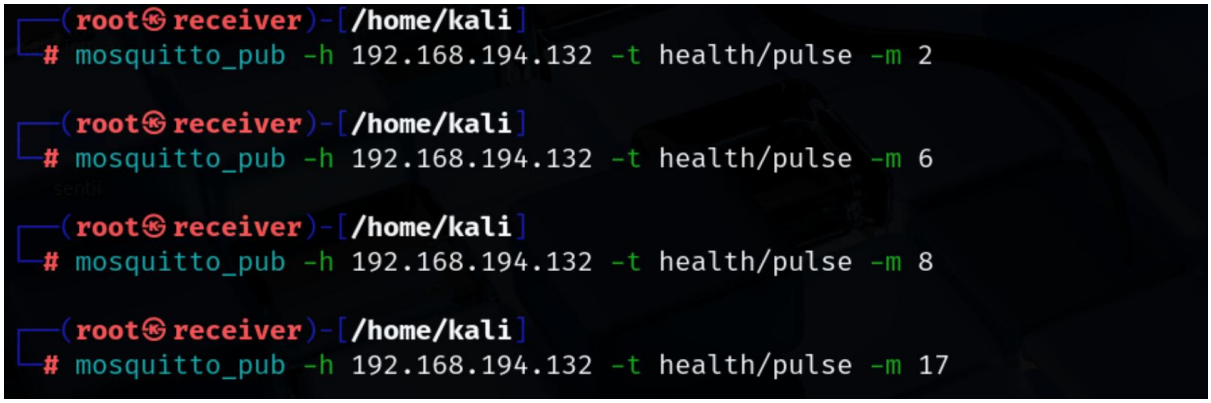

Figure 9: Attacker terminal where Data Injection that is (MITM) for pulse and its publish commands.

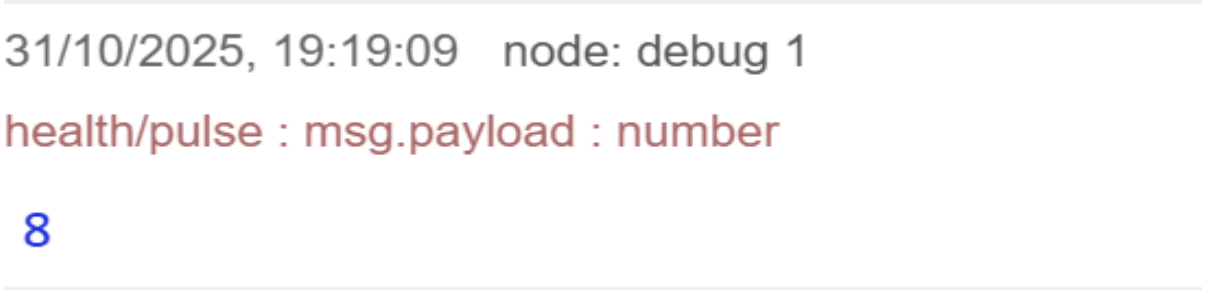

Figure 10: showing Fake Data Injected value (MITM) Injected for pulse

Fake Data Injection (MITM) Injecting for Ultrasonic:

- mosquitto_pub -h 192.168.194.132 -t distance/ultrasonic -m 8904
- mosquitto_pub -h 192.168.194.132 -t distance/ultrasonic -m 204
- mosquitto_pub -h 192.168.194.132 -t distance/ultrasonic -m 25
- mosquitto_pub -h 192.168.194.132 -t distance/ultrasonic -m 299999

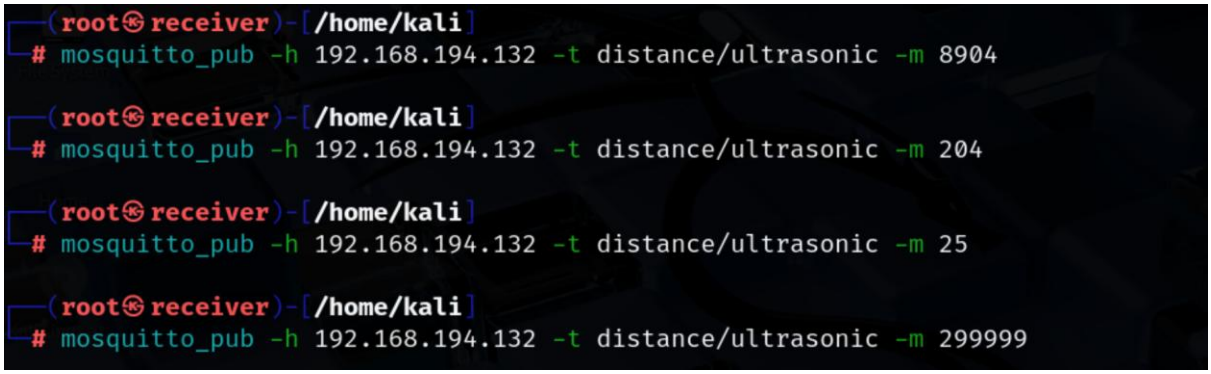

Figure 11: Attacker terminal: Data Injection (MITM) for Ultrasonic and its Published commands.

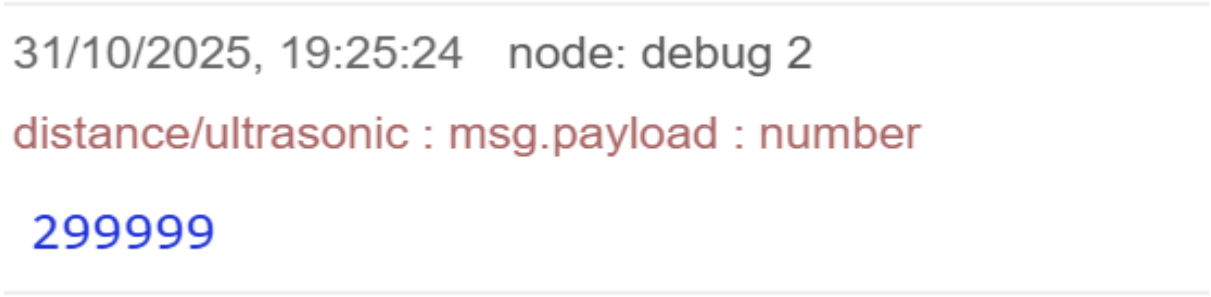

Figure 12: showing Fake Data Injected value (MITM) Injected for Ultrasonic

DoS (Denial of Service) – Flooding MQTT topics with random data to disrupt the system.

DoS (Denial of Service) on pulse

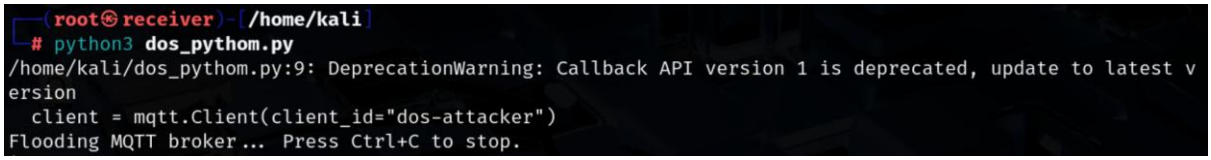

Figure 13: Attacker terminal showing DoS (Denial of Service) on pulse

For both the sensers there are separate python code for the respective topic in which they are running and ip address and in the above fig it is showing the attack it performing in that the value will go vey down and makes it unavailable for the user and sensed the massive requests but it's in secured network and performed in a controlled environment

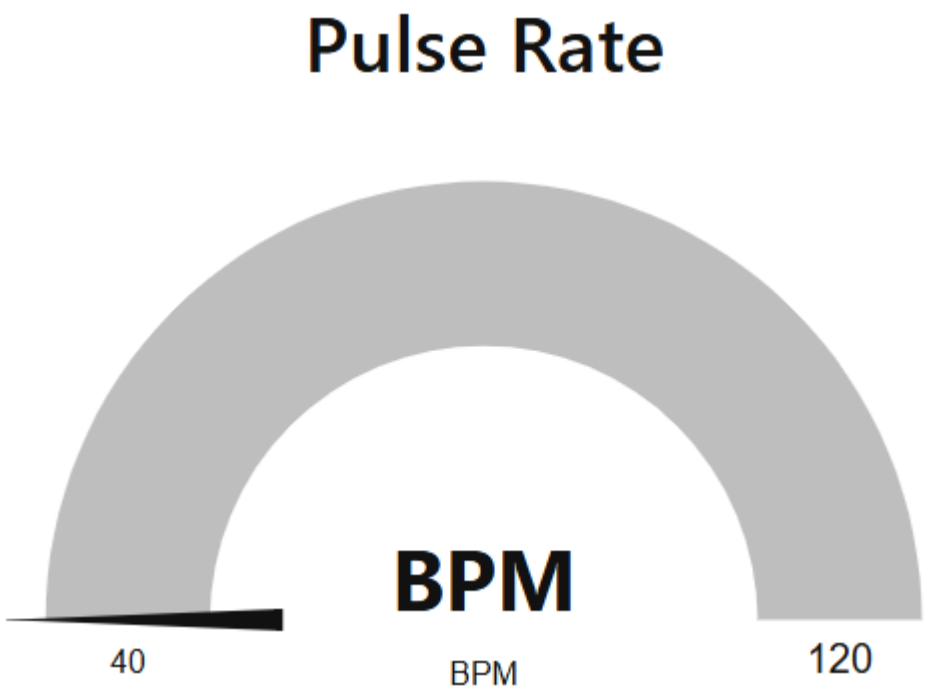

Figure 14: showing dashboard reaction during DoS (Denial of Service) on pulse

DoS (Denial of Service) on Ultrasonic

python3 distance_dos.py

```
┌──(root㉿receiver)-[/home/kali]
└─# python3 distance_dos.py
/home/kali/distance_dos.py:9: DeprecationWarning: Callback API version 1 is deprecated, update to latest
 version
  client = mqtt.Client(client_id="dos-attacker")
Flooding MQTT broker... Press Ctrl+C to stop.
```

Figure 15: Attacker terminal showing DoS (Denial of Service) on Ultrasonic

In this it is clearly targeted the ultrasonic sensers which is mainly of calculating the distance when the python script was run in the controlled environment and it became unavailable and making it more flooded and the systems goes down

But this is done in the controlled entrainment and making the attack visible and noticeable and this helps in makes mitigation phase in order to know the processes which occurred

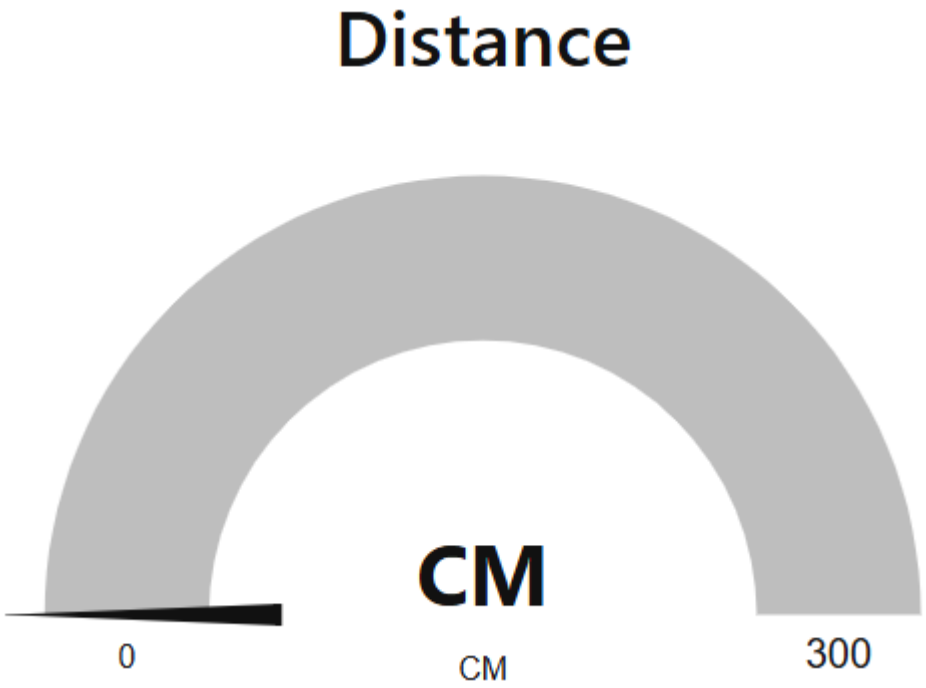

Figure 16: showing dashboard reaction during DoS (Denial of Service) on Ultrasonic

Chapter 4
Mitigation Implementation

Overview

A set of protective settings were implemented in the MQTT broker and Node-RED to secure the integrity then confidentiality of all the sensor data transmissions through. These configurations were designed to safeguard the system from data manipulation, topic hijacking, and denial of the service (DoS) attack.

The implemented security controls emphasize practicality and simplicity making them suitable for both virtual laboratory simulations and small-scale real world IoT deployments when it also comes to the hardware.

Following the successful simulation of all attack scenarios where I simulated above authentication mechanisms were enforced within the Mosquitto broker to restrict unauthorized access to the environment and ensure that only verified clients like genuine could publish or can subscribe to sensitive MQTT topics which is running in the secured environment.

Node-RED MQTT in configuration with security

During this phase user authentication parameters were set up for the “MQTT broker node in Node RED” to enhance the security of communication where it is available In this section the Security settings involve creating a user account named “dashboard” where it can be observed in the above figure and to control and monitor access to the broker. Where it will be in between This configuration will further ensure that only authenticated clients possessing valid credentials can connect other are not allowed thereby preventing unauthorized devices from publishing or subscribing to the MQTT topics which makes it secure in the 1st layer
Besides authentication ACLs were then set up to specify permissions at the topic level even if topic leaks but not the credentials.

This step was important in restricting each users publish and subscription permissions so that the pulse and ultrasonic sensors would interact with their respective topics only. This kind of granularity in control contributes to data isolation which keeps separate and make it more systematic and minimizes the possibility of data leakage or cross topic interference which might be possible

In addition, the MQTT nodes were configured to use credential-based connections and to verify certificates within the Node-RED dashboard. The interface configuration on the dashboard displayed only verified sensor inputs to ensure that no untrusted or injected data appeared in the visualization. Collectively, these configurations of authentication, ACLs, and TLS encryption set up a multilayer security framework for the MQTT-based Node-RED environment. This ensures not only CIA (data confidentiality, integrity, availability) but also how a secure message exchange protocol can be practically implemented in IoT-based virtual simulations.

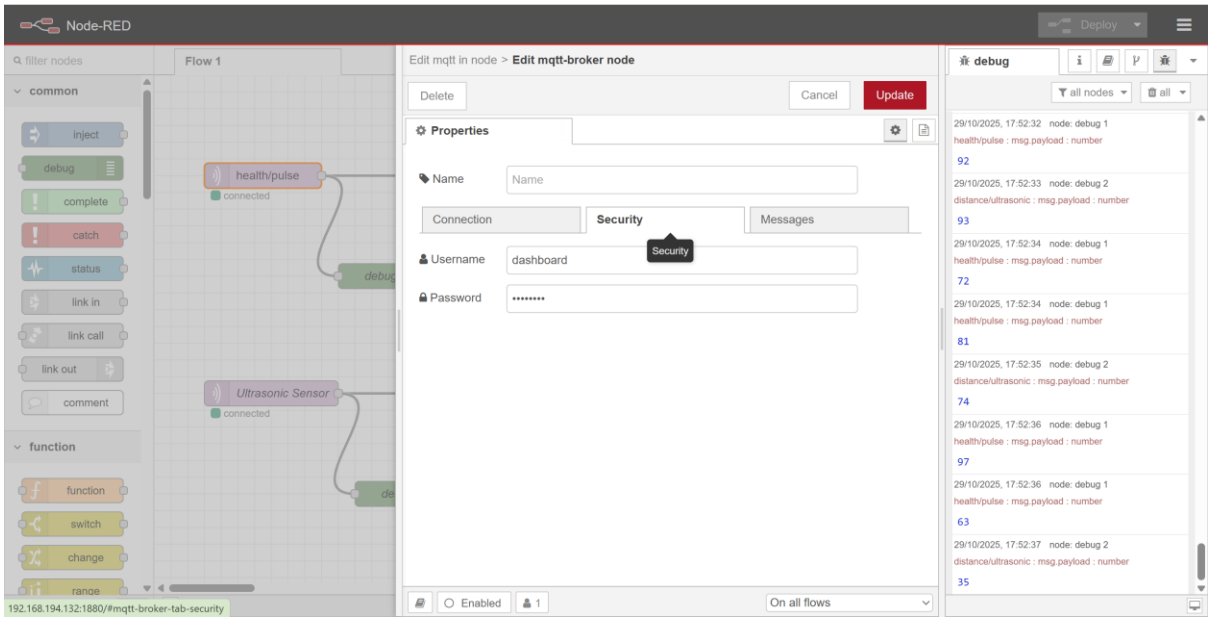

Figure 17: MQTT Broker Security Configuration in Node-RED

This figure shows security tab of the MQTT broker node in locally hosted one which is Node-RED, where the username (dashboard) and password are configured to ensure authenticated access to an MQTT broker.

MQTT Authentication (Username & Password) in configuration file

Disabled anonymous access and created user accounts for each component
Which is pulse senser distance senser and the dashboard.

Files / Paths
/etc/mosquitto/mosquitto.conf — main broker config

```
 GNU nano 8.4                          /etc/mosquitto/mosquitto.conf *
persistence true
persistence_location /var/lib/mosquitto/

log_dest file /var/log/mosquitto/mosquitto.log

include_dir /etc/mosquitto/conf.d

listener 1883
allow_anonymous false
password_file /etc/mosquitto/passwd
acl_file /etc/mosquitto/acl
#allow_anonymous true
```

Figure 18: showing mosquitto configuration file

/etc/mosquitto/passwd — mosquitto password file

Figure 19: showing passwd file

Access Control List (ACL) - Topic Restriction

Created an ACL file that restricts which user can publish or read each topic.

File / Path
/etc/mosquitto/acl

Figure 20: showing Access Control List (ACL) file

Even if user authentication credentials are breached, the Access Control List (ACL) still guarantees granular security by enforcing specific access restrictions for every client account within the Mosquitto broker. For instance, sensor accounts are permitted to publish only to their designated topics, while the dashboard user is limited to subscribing and viewing sensor data without the ability to inject new messages.
The broker automatically blocks and rejects any attempt by an attacker to execute operations outside the permissible access rights thereby protecting the system's security and data integrity.

Iptables Based Rate Limiting for MQTT

To prevent **message flood or DoS attacks** on the MQTT broker by limiting how many new connections a client can make within a specific time.
Using **iptables** (Linux built-in firewall tool)

In this step, I used **iptables** to protect the MQTT broker from excessive connection attempts. The purpose of this rule is to prevent attackers from initiating excessive MQTT connection attempts, particularly those that are redundant or deceitful.

It does **not** replace authentication or ACLs — it just slows down or blocks simple floods (DoS) from one IP during a demo or in a small lab.

The Rules are saved in this fallowing file

/home/kali/mqtt_rules.txt

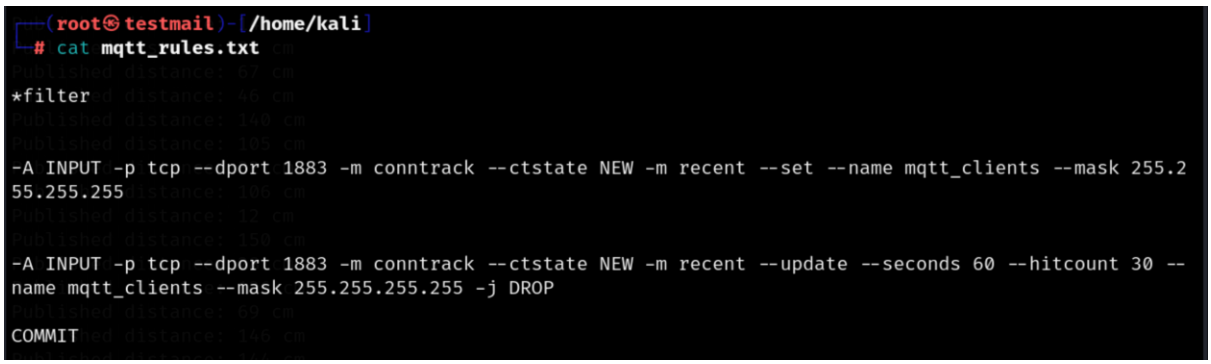

Figure 21: it's clearly showing the iptables Rule File which is (mqtt_rules.txt) Implemented for MQTT Flood Protection

Rule 1: Track new MQTT connection attempts

"-A INPUT -p tcp -m tcp --dport 1883 -m conntrack --ctstate NEW -m recent --set --name mqtt_clients --mask 255.255.255.255"

Rule 2: Drop if same IP sends more than 30 requests in 60 seconds

"-A INPUT -p tcp -m tcp --dport 1883 m conntrack -ctstate NEW -m recent -update -seconds 60 --hitcount 30 --rttl --name mqtt_clients --mask 255.255.255.255 -j DROP"

A **rate-limiting rule is configured within the iptables firewall** specifically to **prevent excessive connection attempts originating from a single source**. The policy temporarily blocks any IP address that initiates more than 30 new connection requests within a 60-second interval.

This approach helps mitigate denial-of-service (DoS) and flooding attacks by automatically dropping suspicious traffic, thereby ensuring that legitimate sensor communications remain stable and uninterrupted.

therefore, if an attacker sending too much of packets where it gets blocked automatically.

Activation Command To **re-add (enable)**

sudo iptables-restore < /home/kali/mqtt_rules.txt

To verify

sudo iptables -L INPUT -v --line-numbers

```
┌──(root㉿testmail)-[/home/kali]
└─# sudo iptables -L INPUT -v --line-numbers
Chain INPUT (policy ACCEPT 978 packets, 39597 bytes)
num   pkts bytes target     prot opt in     out     source               destination
1        0     0            tcp  --  any    any     anywhere             anywhere             tcp dpt:18
83 ctstate NEW recent: SET name: mqtt_clients side: source mask: 255.255.255.255
2        0     0 DROP       tcp  --  any    any     anywhere             anywhere             tcp dpt:18
83 ctstate NEW recent: UPDATE seconds: 60 hit_count: 30 name: DEFAULT side: source mask: 255.255.255.255
```

Figure 22: Confirms iptables Rule File (for port 1883) it's active

To remove (disable) my MQTT rules temporarily I'm using

sudo iptables -D INPUT 1

sudo iptables -D INPUT 1

Testing Basic Rate Limiting (Stop Message Floods) (iptables) on Pulse

python3 dos_pythom.py

```
┌──(root㉿receiver)-[/home/kali]
└─# python3 dos_pythom.py
/home/kali/dos_pythom.py:9: DeprecationWarning: Callback API version 1 is deprecated, update to
latest version
  client = mqtt.Client(client_id="dos-attacker")
Flooding MQTT broker ... Press Ctrl+C to stop.
```

Figure 23: Attacker terminal showing DoS (Denial of Service) on pulse

sudo iptables -L INPUT -v --line-numbers

In this testing phase I noticed that there is not folding happening in this where the mitigation truly worked in this case and which is highly effective

That I we can observe in the dashboard and where the system is running smoothly and also archiving the availably and integrity

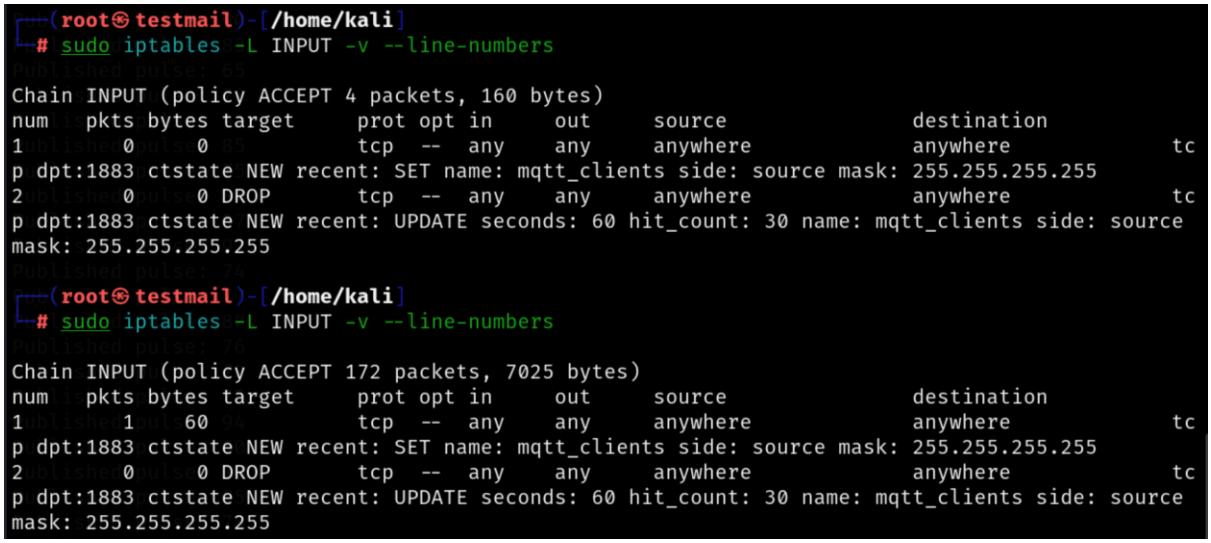

Figure 24: screenshot showing packet counts (point to DROP rule) Protection.

Testing Basic Rate Limiting (Stop Message Floods) (iptables) on Ultrasonic

python3 distance_dos.py

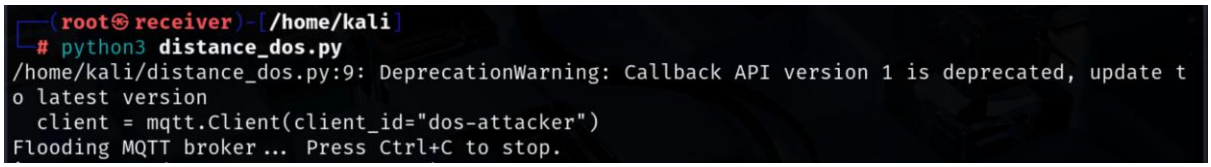

Figure 25: Attacker terminal showing DoS (Denial of Service) on Ultrasonic

sudo iptables -L INPUT -v --line-numbers

Figure 26: screenshot showing packet counts (point to DROP rule) Protection.

Broker mosquitto log evidence

tail -n 40 /var/log/mosquitto/mosquitto.log

Figure 27: showing (192.168.194.133) being rejected with not authorized.

During testing, the attacker system operating at IP address 192.168.194.133 attempted to establish a connection with the MQTT broker on two separate occasions. In both cases, the broker denied access and returned the message “not authorized”, confirming that the authentication and ACL configurations were functioning as intended.

This validation step effectively prevented unauthorized clients from publishing or subscribing to MQTT topics or to my environment and its works perfectly which is secured now thereby ensuring that only legitimate sensor nodes could transmit or receive data within the network in a secured way and ethically

CHAPTER 5
DISCUSSION
EVALUATION OF RESULTS

- After adding authentication and other security settings, all the attack scripts were run again in order to check how the system working
- The MQTT broker successfully and finally blocked unauthorized users from publishing or subscribing to any of the topics which are running.
- The **Node RED dashboard** showed only **real data** coming from the real trusted Pulse and Ultrasonic sensors which is impressive and satisfied
- These tests proved that the **security settings worked properly** and kept the system **safe and stable** during different types of attacks even dint faced the issue of system down or performance

Through the implementation of virtual sensors within a fully simulated network with a high

robust and reliable IoT environment could be established and Attack scenarios like data manipulation spoofing of devices and Data integrity is one of the key pillars of IoT trustworthiness which is impressive alongside confidentiality and also availability mentioned (Shin & Kim, 2022). DoS were clearly presented and makes full down but then mitigated in a good manner “Applying authentication and rate limiting rules which blocked unauthorized connection attempts” (Singh, 2023).

CHALLENGES ENCOUNTERED

While the SensorShield project had highly successful outcomes, there were several obstacles encountered during its design with implementation and testing stages. A presentation of these challenges provides valuable insight into the practical challenges presented when an IoT-based cyber-security framework deployment is simulated in a real, current world environment.

Ensuring Reliable Data Flow and Topic Separation in MQTT Communication

One of the main challenges was to maintain clear isolation between MQTT topics when multiple virtual sensors were publishing data concurrently. Improper configuration of topics could lead to a scenario in IoT ecosystems where one compromised device might access or influence another sensor's data stream. During the initial experiments, both pulse and ultrasonic sensors transmitted simultaneously, and a misconfigured topic structure resulted in partial data overlap. Implementing robust topic hierarchy, access control lists, and user-specific authentication made sure that each virtual sensor communicated strictly within its own assigned topic, therefore also preserving data integrity with privacy.

Balancing between the System Performance During an Attack Simulations

Maintaining operational stability while simulating cyberattacks, in particular during Denial-of-Service tests, was another key challenge. The repeated execution of attack scripts occasionally caused temporary slowdowns in the Mosquitto broker and Node-RED dashboard, mainly due to the excessive traffic generated. By fine-tuning broker parameters, configuring iptables-based rate limiting, and monitoring CPU and memory usage, stable performance could be achieved while realistically portraying DoS effects. This balance needed to be found in order to properly

simulate realistic attack behaviour versus system resilience for practical cybersecurity defense.

Integrating Multi-Layer Security Without Breaking Functionality

There were many challenges to this integration: the implementation of multi-layered security with authentication, ACL policies, and firewall rules. A single configuration needed precision, and any small syntax error, missing file path, or mis ordered parameters within mosquitto.conf would break broker operations or disconnect valid clients. Many iterations were needed to combine the settings correctly so that the system remained secure yet continued to allow legitimate data flow to the Node-RED interface.

Ensuring Network Stability Between Virtual Machines

Since it was all going on in a virtual environment, stable connectivity between the two Kali Linux machines was important. Intermittent network glitches caused time-outs and packet drops between the attacker system and the MQTT broker. These were overcome by thoroughly debugging the virtual network adapters, checking the IP addresses of the two machines' configurations (192.168.194.132 and 192.168.194.133), and conducting preliminary testing of the network before running the simulation to ensure their stability.

Verifying Security Logs and Monitoring in Real Time

Continuous monitoring of system logs was vital to establishing the effectiveness of the security measures applied. This consisted of reviewing the broker log file at /var/log/mosquitto/mosquitto.log to confirm both successful and blocked connection attempts. It ensures real-time log analysis for threat detection and incident response in IoT security environments, providing practical experience with operational monitoring.

Altogether, these challenges reinforced the necessity of proper configuration, real-time monitoring, and a layered security design in safeguarding the IoT systems against cyber threats.

CHAPTER 6
CONCLUSION AND RECOMMENDATIONS

SUMMARY OF FINDINGS

This project has shown both the benefits of IoT and how security problems can cause data issues when protection is missing. I tested different attacks such as DoS, data injection, and spoofing in a small virtual lab. After adding login checks, topic control, and firewall rules, only real sensor data appeared on the dashboard. The results have proven that even simple steps such as passwords and rate limits can prevent many IoT attacks. Keeping the security from the very beginning helps build trust and makes the IoT systems more reliable.

FUTURE WORK

Although this project successfully demonstrated the simulation of IoT-based cyberattacks and corresponding defence mechanisms in a virtual environment, there remain several opportunities for enhancement and expansion in Furter research's

1. The next steps for integration with physical hardware could include deployment of real sensors like Pulse and Ultrasonic modules along with the ESP32 microcontroller. These are to check communication reliability and system behaviour in practical hardware environments using the same MQTT framework.
2. TLS/SSL Encryption Implementation: The introduction of end-to-end encryption to the MQTT data transmission would enhance the data confidentiality and protect it from eavesdropping during device-to-device message exchanges.

3. AI-Driven Intrusion Detection: Integrating machine learning-based anomaly detection models could let the system automatically identify irregular data patterns and respond to a possible intrusion in real time.

4. Cloud-Based Deployment: Hosting the MQTT infrastructure on cloud platforms such as AWS IoT Core or HiveMQ Cloud could help analyze scalability, latency, and a performance in the largescale IoT real environments.

5. Defence Against Advanced Attacks: Future work can extend mitigation layers to safeguard

against sophisticated threats such as replay attacks, data exfiltration, and IoT-targeting ransomware.

These developments can potentially enable SensorShield to transition from a virtual demonstration model into a fully deployable IoT defence framework applicable to real-world smart and critical infrastructure systems.

CHAPTER 7

REFERENCES

1. Abomhara, M., & Køien, G. M. (2015). *Security and privacy in the Internet of Things: Current status and open issues.* Computers & Electrical Engineering, 52, 1–16.
2. Statista. (2025). *Number of connected IoT devices worldwide from 2019 to 2030.* https://www.statista.com/statistics/1101442/iot-number-of-connected-devices-worldwide/
3. Babar, S., Mahalle, P., Stango, A., Prasad, N., & Prasad, R. (2011). *Proposed security model and threat taxonomy for the Internet of Things (IoT).* 6th International Conference on Network Security and Applications (CNSA), 420–429.
4. Gupta, A., & Sharma, P. (2022). *Analysis of MQTT protocols and vulnerabilities in IoT systems.* International Journal of Computer Applications, 180(2), 12–18.
5. Hunkeler, U., Truong, H. L., & Stanford-Clark, A. (2008). *MQTT-S — A publish/subscribe protocol for Wireless Sensor Networks.* IEEE Conference on Communication Systems Software and Middleware.
6. Open JS Foundation. (2024). *Node-RED documentation.* https://nodered.org/docs
7. Linux Foundation. (2023). *Mosquitto MQTT Broker Manual (v2.0.21).* https://mosquitto.org
8. Fernandes, E., Jung, J., & Prakash, A. (2017). *Security analysis of emerging smart home applications.* IEEE Symposium on Security and Privacy, 636–654.
9. Kolias, C., Kambourakis, G., Stavrou, A., & Voas, J. (2017). *DDoS in the IoT: Mirai and other botnets.* Computer, 50(7), 80–84.
10. Wang, Y., Xu, D., Zhang, C., & Zhou, M. (2020). *IoT DDoS attack detection and mitigation using machine learning.* IEEE Access, 8, 45198–45211.
11. Shin, D., & Kim, J. (2022). *Enhancing data integrity in IoT systems through decentralized trust mechanisms.* Sensors, 22(19), 7294.
12. Rahman, S., et al. (2022). *Enhancing MQTT security using TLS and ACLs.* International Journal of Network Security, 24(2), 85–92.