ISBN: 978-628-96613-1-6. ISSN: 2414-6390. Digital Object Identifier: https://dx.doi.org/10.18687/LACCEI2025.1.1.1789

SECURITY ANALYSIS OF AN IOT COMMUNICATION SYSTEM WITH A VIRTUAL MACHINE IN AWS

Stephanee Carolina Villagra Bautista, Ing. En Telecomunicaciones y Electronica [1,a]; Alberto Max Carrasco Bardales, Msc. En Sistemas Mecatrónicos [2,b]; Juan Miguel Mejía Saavedra, Ing. En Mecatrónica [1,c]; Juan Carlos Dubón Portillo, Mg. en Finanzas [1,d]

[1]Facultad de Ingeniería, Universidad Tecnológica Centroamericana (UNITEC), San Pedro Sula, Honduras, [2]School of Advanced Technology, Algonquin College, Ottawa, Canada

[a]*stephaneevillagra@unitec.edu,*[b]*carrasa@algonquincollege.com,* [c]*juan.mejia@unitec.edu.hn,* [d] *juan.dubon@unitec.edu.hn*

Abstract*– This research explored the issue of security when communicating with IoT devices using the MQTT protocol in conjunction with cloud computing. IoT devices pose a greater risk than electronic devices because they do not have software updates, so if a vulnerability is found, the device will remain that way, in addition to allowing access to your network and local data. Despite the risks that IoT devices can pose, the growth in their use shows the importance of streamlining processes and activities through their widespread use in projects in different areas, such as home automation and industry. The use of communication protocols such as MQTT is used when talking about IoT, since with its addition a secure channel is used for the passage of information from the IoT device to a broker that allows the flow of data to the receiver. Cybersecurity protects us from attacks from the Internet by protecting the systems we use, therefore performing penetration tests with specific penetration tools help to perform an analysis that revealed the weaknesses of the IoT communication system and then proceed to reduce vulnerabilities with services in AWS, configurations in the intermediary broker MQTT and rules to protect passwords.

Keywords--* Cloud computing, cybersecurity, IoT, MQTT, Pentesting.

I. INTRODUCTION

Communication is an important factor for the human being, since many years ago a way was sought to obtain a response from the recipient without delay and with clarity. Currently, communications are made in real time according to the communication protocol used. These protocols that allow the flow of data from transmitter to receiver are made through the internet, so that this data goes through immense data traffic that can be potentially exploited by threat actors through different types of denial-of-service attacks, brute force attacks and data integrity attacks.

The security of information passing through the internet is always an important problem, as security is an important factor in using this technology. If the internet connection is not safe, it would cause us to opt for another type of technology that offers us higher security, and in this way, we waste the potential of that technology for its low levels of security. Using a system that integrates different tools to provide real-time communication that is dynamic and adding cloud services that already have implemented security policies is intended to obtain a system that allows stable and secure communication. But even if we have this hypothesis about the system, the way to check the security of the system is through penetration tests that indicate the weaknesses of the system, seeing these weaknesses from a threat agent's perspective.

This paper is organized as follows. Section II presents key concepts about AWS, IoT, virtual machine, MQTT, and communication system security. Section III focuses on the methodology and explains the software to be used to perform penetration tests. Section IV shows the results obtained through the steps indicated in the methodology. Section V provides conclusions and recommendations on how best practices can be used to protect an IoT communication system built from scratch against cyber-attacks.

II. CONTEXT

Knowing the precedents of cyber security in IoT devices will help us to better understand the importance of security regardless of the significant advances that have been made in recent years and how to achieve a risk reduction using a method of real-time communication that provides security to devices. The theoretical foundation of the components that are used in the telemetric monitoring system which consists of the ESP, AWS with EC2 services for the creation of instances that provide a logical space to install virtual machines. Similarly, this chapter covers the communication process of the MQTT protocol and the connection with a cloud server as well as supporting the existing complications to protect IoT devices.

Since the 1990s, the use of IoT has grown. The IoT has attracted many attackers trying to exploit the IoT. With the evolution of the IoT, the result has been the improvement of different software to attack different devices connected to the Internet. Over the years, many threats have been discovered against them, but many continue to prevail with great strength to infiltrate and infect systems.

One of the many lethal threats to IoT devices was discovered in 2015, known as KTN-RM, or Remaiten, this threat specifically targets IoT devices. KTN-RM is a combination of Tsunami and Gafgyt, both of which are Linux malware [2].

Another relevant threat is Mirai, which exploits vulnerabilities in IoT devices to spread and infect other IoT

devices. The vulnerabilities are mostly in the management part and by exploiting them, cybercriminals can gain control through remote access. The infected devices in a DDoS attack become bots, where they become part of a botnet army with many bots. Mirai has been one of the most prolific IoT malware families for years and was responsible for the largest DDoS attack in history in 2016. In 2021, 76% of the attacks blocked by Zscaler on IoT devices were from the Mirai family [3].

In 2023 there was an 18% increase in IoT device traffic compared to previous assessments in 2021. But while usage has increased, so have the challenges. The ThreatLabz team found approximately 300,000 blocked attacks from known IoT threat actors representing a 400% increase in IoT malware attacks compared to the previous year. Mirai and Gafgyt bots had the highest percentage share of IoT malware attacks [4].

A. IoT

The Internet of Things (IoT) is widely used today because it allows devices to be connected over the Internet, made possible by their ability to collect and share data. These devices are usually equipped with microcontrollers such as Arduino, sensors, actuators and internet connectivity [5].

In this way, users can remotely monitor various devices that are part of the IoT, such as lights, fans, heating/cooling systems, and make decisions based on the information provided by the sensors [6].

Through the Internet of Things, various advances have been achieved for industry and people through the implementation of smart cities. A smart city is an urban area that collects data from various devices to effectively manage urban resources. The smart city IoT infrastructure connects numerous devices to an Internet-protocol based low-power wireless network, shares massive amounts of data, and facilitates the development of new service [7].

Another application of the Internet of Things is in the commercial sector, in a way that allows us to simplify processes. In the residential, commercial and industrial sectors, IoT is being used to better manage energy consumption, either through the use of a reliable monitoring, control and data acquisition system (SCADA) for home monitoring and control systems [6].

But the use of IoT devices presents vulnerabilities in their software, and this produces threats to infrastructures and users' privacy. For many years, multiple research and efforts have been made to bring to market devices with a design that implements security.

However, these efforts are still far from being translated into real deployments, since devices such as microcontrollers do not have patch updates to enact and maintain device security [8].

One of the solutions for better security in the IoT area is with access control, this is a fundamental security mechanism to protect the IoT ecosystem, which includes cloud computing and edge computing services along with smart devices with leading cloud and IoT service providers such as Amazon Web Services (AWS), Google Cloud Platform (GCP) and Azure [9].

The paradigm of cybersecurity in the industrial environment, given by the combination of these two technologies, forces to provide security solutions for cities and intelligent machines to operate together and to ensure a better protection against all kinds of attacks from cyber attackers. The complementary methodologies are based on the collaboration of teams responsible for assessing the security of IT and OT networks. These teams have the sole purpose of assessing the risks that may occur on the Internet in order to proceed to apply the necessary security mitigations [10].

Currently, the use of IoT in factories is of utmost importance to have the provision of advanced technologies, so the use of artificial intelligence and machine learning that can help detect and prevent cyber threats. Since the use of the two mentioned technologies can help prevent, monitor and identify suspicious behavior of a production system or network to prevent attacks [10].

The use of different technologies for the use of IoT in sectors such as smart homes can be very expensive and with a complex structure if it is necessary to implement different security filters as in the industrial one. Therefore, the implementation of tools such as ESP8266 help us to have a simple connection to the Internet of different types of sensors and actuators.

B. ESP

The most commonly used microcontrollers in IoT projects are ESP's. The ESP8266 is a low-power microcontroller with a powerful microprocessor. This device allows different modes of operation to connect sensors or actuators in a network and process the data in this network, which can reach the Internet or remain in a private network.

The modules ESP32 is a powerful module that has various applications such as monitoring and control of various sensors at the same time, everything necessary to develop irrigation systems or air flow systems as well as the creation of security systems in homes. It is very functional to control electrical devices such as lights in the rooms of a house as well as design models of trackers, all possible with the different integrated sensors it has.

Using ESP from mobile applications can be easy if the device is in configuration state, activate the access point mode where it creates its own network with SSID and password, for this you need to run the TCP/IP server responsible for identifying incoming connections. The mobile application then obtains the IP and port number of the QR code to establish the connection. The QR code data is in JSON format. This allows for easy interpretation and storage of nested and complex configurations [11].

The most commonly used connection to send data to the Internet is based solely on activating the station mode, which connects to an existing Wi-Fi network where it obtains its IP address and then forwards the data through a secure transmission channel to the services designed to manage traffic, such as EC2 AWS, or directly to a web page.

C. AWS

Amazon Web Service (AWS) is one of the most used cloud platforms today and according to the Stackoverflow report. AWS has a large number of services that can be used at different levels of security; for this reason, this article mainly talks about those that are applicable to microservices to increase security, with a focus on encryption and protection of data, files and databases [12].

In [12] noted that the Amazon Web Services has a level of security high enough to call it impenetrable, this complies with the expected security standards for the proposed case study where it was expected not to be able to access information stored under the security level provided by the Amazon Web Service key management service.

In addition, AWS complies with the three main features of the information: integrity, availability and confidentiality. The availability of data is provided by the APIs that are hosted in the ECS, the APIs are responsible for acting as an interface to provide data to those who have access to them, as well as allowing the integrity of them, since, being inside an EKS, the APIs cannot be altered in order to modify the information; finally, the confidentiality of the data, is treated by means of the data encryption services using the keys provided by KMS [12].

MQTT uses brokers to manage topics and list the subscribers to each topic. Brokers forward any new message posted to a specific topic to all subscribers of that topic. For example, if DHT11 posts to home/living_room/light, any other device subscribed to that topic will receive the message [9].

With the design proposed with the MQTT protocol the previous steps must be implemented because they are the essential topology to have access to the Internet through the module and in the same way the basic security policies such as the creation of a password to prevent connections with unauthorized devices to the network. The different applications of this system can be smart homes and, in the industry, since with the addition of Wi-Fi modules you can have a scalability in the field of integration of different low power sensors and with the creation of secure networks in the ESP Data can be transported using the MQTT protocol with your kernel installed in a cloud using the Linux operating system with your UBUNTU distribution.

Each broker can be installed in different ways, as many have a graphical interface or only a command line for their installation. The latter can be installed on virtual machines, and to implement a secure space, you can use the EC2 AWS service.

The Amazon Elastic Compute Cloud (Amazon EC2) EC2 service provides scalable, on-demand computing capacity in the Amazon Web Services (AWS) cloud. Using Amazon EC2 reduces hardware costs so you can develop and deploy applications faster. With EC2, you can launch as many virtual servers as you need, configure security and networking, and manage storage [13].

D. Virtual Machine

Data centers face a big problem about power consumption for cloud computing, which has not long been adequately addressed by data center developers. One way to reduce power consumption is to add virtual machines (VMs) to the cloud computing environment [14].

Virtual machine consolidation remains the most deployed strategy to manage both performance and energy consumption. Most of existing energy efficiency techniques save energy against the cost on performance degradation [15].

Using multiple virtual machines in the Amazon cloud it is found that even in the presence of two editorials on the same topic, similar results, but with a difference between minimum and maximum delays, this is due to the timing accuracy in single machine testing versus cloud computing with millisecond accuracy and in the virtual machine a timing accuracy down to microseconds [16].

Using the Linux operating system kernel and UBUNTU distribution provides a working environment for web browsing, multimedia playback, and document creation as if you were on a physical machine. With UBUNTU Server, you get secure and high-quality implementations to connect to a wide range of services or applications, databases, email services, etc. All this easily. With technological advances, UBUNTU is a choice that is widely used for having a cloud structure, since it is possible to be in private or public cloud environments.

Therefore, using virtual machines in the Amazon Web Services cloud consolidates its efficiency by giving the system the ability to dynamically register the brokers needed by customers [16].

In addition, virtual machines can help simplify tasks in a simple and efficient way, as recovery time and revisions can be easily reduced using the virtual machine concept [17].

E. MQTT

MQTT is designed to reliably transmit messages without the problem of low bandwidth and high network latency. It is currently widely used for communication with IoT devices. Both Amazon and Facebook Messenger for mobile devices use the MQTT protocol [18].

The concepts of "Publisher", "Subscriber", "Topic" and "Payload" are used for MQTT communication. Each of these concepts means how the transmitter and receiver of information passing through the MQTT protocol will be identified. The importance of this protocol lies in your Broker or also called server since that is an agent who is responsible for distributing messages (Payload) from the publisher to users interested in a particular topic to receive data from the publisher that are called subscribers.

When the client wants to send information to the broker, the client must establish a connection with the broker, where the client requests the broker to connect to it with the CONNECT message, then the broker sends the acknowledgement of this connection request to the requesting client. And the information that MQTT delivers to the subscriber is known as an application message.

Create a publisher and different subscribers, create different publishers and a subscriber or different publishers and subscribers, but on the subject, it produces that the speed of information transfer is less than the serial connection, as it depends on the speed of Wi-Fi subscribers, since using MQTT to send many data can be observed that if there is data loss or delay while publishing messages from two subscribers at the same time, as shown in Fig. 1 [5].

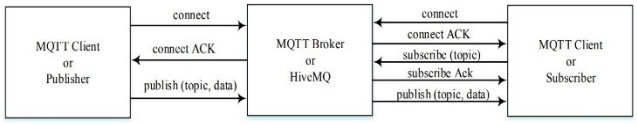

Fig. 1 System operation with MQTT [5]

When adding subscribers either globally or locally to the publisher, the MQTT protocol introduces delays, this has been demonstrated by tests where the publishing and receiving times of the subscriber were affected by the same protocol [16].

Since the broker does not have the capacity to scale well compared to other systems, it delays the process of information transfer to the subscriber, besides the fact that it has no data processing, so event processing is not considered in many MQTT brokers.

Although the MQTT protocol has disadvantages, it has many advantages when using it, as it has the ability to transfer data faster than the HTTP protocol, achieving data transfer six times faster than HTTP. Therefore, the use of MQTT is an option for real-time data acquisition hardware application based on the Internet of Things [19].

The ESP8266, MQTT and EC2 connection process consists of first successfully connecting the ESP8266 to the Wi-Fi network, then the ESP module will attempt to connect to the EC2 cloud server. AWS will check the root CA, device, private key, MQTT host, topic and name of things. If these are OK, it will allow the MQTT client to connect [20].

Currently, there are many brokers used in MQTT, these are responsible for sending and receiving data from the publisher, so this is paramount for proper communication of IoT devices. Some of the brokers used are Adafruit, HiveMQ and Mosquitto, but the one used in this research is Mosquitto, it is free and can be downloaded on virtual machines. These different brokers MQTT has three types of evaluation in the quality of the service so the quality of service that can provide you the broker will define the type of communication that will have the system [21].

The quality of service has three levels consisting of QoS 0, QoS 1, QoS 2, where QoS 0 defines that the message will be sent once and in case of failure may lead to non-delivery of the transmitted information. QoS 1 guarantees the delivery of information and in case of failure may lead to duplicate messages being sent. QoS 2 guarantees that the message is received by the topic subscriber but only once.

The Mosquitto broker is a single-threaded application. It uses an infinite loop, called the main_loop, in which Mosquitto manages the TCP sockets of the connected clients and keeps track of which topic each subscriber is connected to, thus establishing communication between publisher and subscriber, with the Mosquitto broker acting as an intermediary [22].

The main_loop works by repeating the same operation for each loop, where each loop performs a request to the system 'poll()', which inserts the newly connected subscriber located in the subscription list, also inserts the message in the message queue with the topic [22].

It then compares the topic of each message in the message queue with the topic in the subscriber list, and if the two topics match, the message is copied to the message buffer, and the message pointer of the matching subscriber in the subscriber list is linked to the copied message in the message buffer and sent to each subscriber in the subscriber list in order of subscription [22].

Fig. 2 shows the average message processing rate and latency of the brokers using 100% of the CPU. This reflects that Mosquitto outperforms other brokers by having message processing in its different QoS [23].

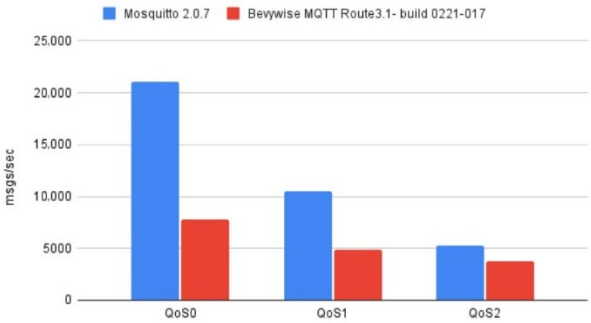

Fig. 2 Projected message rate (messages/sec) of non-scalable brokers with 100% compute CPU usage in the cloud evaluation environment [23]

The MQTT protocol with its broker represents a low complexity for clients (publishers and subscribers). However, its default configuration relies on a single broker, which is a potential bottleneck. As IoT systems can grow in size, it may be necessary to implement security to protect complex architectures [24]. For non-critical IoT applications, vertically scaling a single broker can help the system meet increasing customer demand. However, when more than one broker is required for the communication system, clustering or federating multiple brokers is often the way to achieve scalability. [24].

MQTT broker or a subscriber could be a performance bottleneck. Fig. illustrates a typical architecture of collecting data by MQTT. A massive amount of data from publishers concentrates on the broker, and thus it might not accept all published data. Even though the broker can process all the data, the subscriber might not catch up with the amount of data from the broker [25].

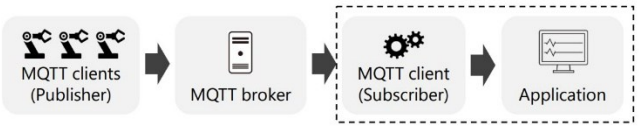

Fig. 3 IoT data collection using MQTT [25]

The use of a vertically scalar broker means that it runs directly on a physical machine, which can strengthen the broker, but even if there is some degree of fault tolerance, a failure on the physical machine can wreak havoc on the entire system [24].

IoT and cloud computing are becoming more and more fashionable and vital in the computing world. In the advancement of computing, the MQTT protocol is of much importance as it enables us to use IoT which provides connected things we have never seen before [26].

By using a virtual machine installed on a cloud server, the overhead of data migration between brokers is expected to be low, making data migration negligible compared to the overhead of MQTT publishing [16].

With the ability to provision/de-provision granular virtual resources provided by cloud computing, it empowers MQTT brokers by enabling the elasticity features that help brokers manage a wide variety of integrated IoT data every day [26].

F. Cybersecurity

Internet of Things (IoT) devices often lack cybersecurity capabilities for their customers or organizations, making it impossible to mitigate their risks from targeted attacks. While manufacturers can help their customers by providing and facilitating cybersecurity information, these devices are driven more by how well they work and sensor integration than by advancements to protect them.

Cybersecurity risks for IoT devices can be viewed from two high-level risk mitigation perspectives. The first is to protect the cybersecurity of the device itself, by preventing the misuse of the system that ultimately causes negative impact to the customer or attacks other organizations. And the second is to protect the confidentiality, integrity, and/or availability of data (including personal information) that is collected, stored, processed, or transmitted to or from the IoT device [27].

Security in IoT devices is a mandatory requirement for the development of technology and the advancement of IoT infrastructures. Having a cybersecurity certification on these devices, in addition to warranties and maintaining that certification throughout the life of the device, promotes consumer confidence in IoT devices [8].

But with the disruption of IoT devices and applications, attackers are exploiting weak authentication and access control mechanisms to gain unauthorized device access, infect a system, and cause damage [9].

Since they are extremely weak without security measures, attackers can connect to the security camera and once it is connected to the Internet, they can obtain the video signals through the open RTSP ports, as well as test the common URLs of these cameras and manage to broadcast video signals on them [28].

Research shows that 10.58% of vulnerable devices have critical vulnerabilities, 40.35% have high vulnerabilities, while 67.96% of devices have one or more low vulnerabilities. This may be reflected in the fact that many devices need to implement additional services to maintain security in order to reduce risk [29].

One of the ways to protect systems is to identify the expected customers of the IoT device from the earliest design stage. By doing this, it is possible to determine the cybersecurity capabilities that should be implemented in IoT devices. All of these capabilities should be tailored to the needs of the customers, such as: For enterprise customers, the devices may need to integrate with their log management servers, but not for a residential customer [27].

Penetration tests are performed to protect communication systems, as they help to identify the risks that may occur when an attacker gains access to the organization's computer system and networks. Performing these tests allows you to create a plan to mitigate and close security breaches before an actual attack on the system occurs. Conducting a penetration test helps organizations reduce financial and information losses that would have caused loss of customer confidence due to security breaches in addition to helping shape important aspects of the information security strategy by identifying vulnerabilities quickly and accurately [30].

Penetration testing requires a lot of time, effort and knowledge, depending on the complexity of the organization. Therefore, penetration testing helps improve the knowledge and skills of the people involved in the process. It is considered a quality assurance tool that benefits both business and operations. [30].

With the use of technologies such as network monitoring and firewall, network access control technology, distributed intrusion detection technology, contact-after-repair technology partitioning, lightweight encryption/decryption algorithm, and anti-malware and antivirus software technology, IoT devices can mitigate cyberattacks, but at a high monetary value [31].

One of the most important factors indicating increased vulnerability is social engineering. It is a term used to describe an attack that relies entirely in human error. It gathers valuable and sensitive information through the use of psychological manipulation to trick legitimate users. This kind of attack is very dangerous since users' mistakes are less predictable. PEN testing helps the organization evaluate their staff adherence to the organizations policies and procedures. It also helps improving the security training provided for the employees [30,32].

In this research a simple monitoring system is developed with an ESP8266 that collects the data of a humidity sensor such as the DHT11 (publisher). Sensor data is sent through the MQTT communication protocol that has a broker installed in AWS with EC2, as shown in Fig. 4, that facilitates the creation of UBUNTU virtual machines. The broker used is Mosquitto

which after being installed in the virtual machine and will allow the data of the publisher to be processed and delivered to the subscribers. With this structure it will be possible to analyze the weaknesses of the system and in the same way it will be possible to make security configurations for the mitigation of attacks to the system.

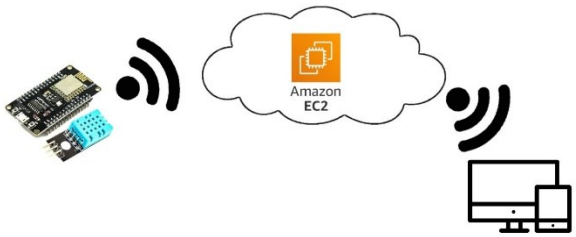

Fig. 4 Integrative image

Knowing the advantages offered by cloud services as well as virtual machines and protocols for communication raises a system with lower latency since communication is in real time adding that communication passes through the AWS cloud that involves security in the installed instances. Uses penetration tests is mandatory to have a source of security validation for any infrastructure that connects to the Internet.

III. RELATED WORK

Several methodologies have been proposed for intrusion detection in IoT environments. For instance, Aggarwal and Srivastava (2016) implemented a machine learning-based intrusion detection system on MQTT networks [33]. Another approach by Zare and Iqbal (2020) used SCADA frameworks with MQTT to monitor traffic anomalies [34]. Compared to these, our work applies a penetration testing framework directly on an AWS-hosted MQTT system, offering a practical evaluation of real vulnerabilities.

Unlike detection-oriented solutions that rely on anomaly detection or machine learning, our approach proactively tests for vulnerabilities through ethical hacking techniques. This provides a hands-on analysis, although it may not detect passive or evolving threats as efficiently as automated systems.

III. METHODOLOGY

This project proposes a systematic approach since the performance of penetration tests always follow a sequence of steps, in addition to the action of these tests make use of specific tools and methods to understand the structure of the system, these tools depending on which area of the system you want to analyze use the same steps for their operation, as shown in Fig. 5.

A. Tools for penetration testing

1. Wireshark

This software is used in the telemetry system to analyze all encrypted packets passing through the network. In order to know how the data passes through the system and thus visualize the reliability of the data.

2. Nmap

It is an open source command line tool executable in the virtual machine that allows us to know the open ports of the telemetry system through a scan of IP addresses that also serves to detect applications that are installed in the network communication.

3. Hping3

This VM-executable tool in AWS performs the analysis and assembly of TCP/IP packets flowing into the communication system. With this software the denial of service attack is performed as it allows the sending of packets for system saturation.

4. Aircrack-ng

This software allows you to obtain passwords from WiFi networks using a packet analyzer. This tool allows you to know the security of the WiFi network in which the ESP8266 is connected for sending data to the broker that is installed in the virtual machine.

B. Study methodology

Penetration testing is carried out with software that allows the identification of vulnerabilities in a network or any technological system. All these tools mentioned above must fulfill three phases for their correct function.

1. This first phase is test preparation, in which the information of the system in which the test is applied is collected and documented, in order to know what type of tool will be used for its respective analysis.

2. The second phase is test implementation, which is where the selected penetration tool is already used, which consists of three fundamental steps for efficient use, which are:

2.1 Information gathering

When implementing the tool, it is necessary to scan and identify all the logical and physical information of the system and to have a collection of information about the vulnerability to want to analyze using the penetration tool.

2.2 Vulnerability analysis

This step indicates the finding of vulnerability in the system and with this discovery helps understanding how vulnerability originated. This also promotes education by learning to know about detecting, eliminating and avoiding vulnerabilities.

2.3 Vulnerability exploits

In this step the system is infected with an attack from its vulnerability with the help of exploits that are mechanisms to take advantage of the vulnerability found. Exploit defines implementing actions to harm the system either by denying the service by sending unauthorized messages saturating the system or getting the information going through the system.

3. The third phase “test analysis” consists of mitigating the vulnerability found, investigating countermeasures and risk strategies in the face of exploited vulnerability. This phase must be implied if the likelihood of exploiting weakness is accepted, avoided, reduced or transferred. Moreover, this phase involves

the documentation of the results obtained in the previous phases.

C. *Validation Methodology*

The system penetration tests evaluate the results and implement security improvements in terms of authentication and security advice. After these mitigation actions, the same system tests are used again to determine if the mitigation measures were one of the four ways to manage a risk: reduce, avoid, transfer or accept it.

IV. RESULTS

The communication system was easy to implement together with the connection of the DHT11 to the ESP8266 as well as the programming made for the sending of data by port 1883 intended for MQTT communications, as shown in Fig. 6.

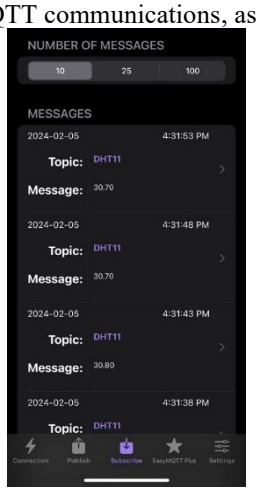

Fig. 5 Data reception

A. *Test Preparation*

The scope to run the tests is to check access to the system without credentials through open communication ports and access WiFi networks and DoS attacks.

To perform the test, as shown in Fig. 7, four tools were downloaded to analyze, scan and perform DoS attacks as well as brute force attacks on WiFi networks.

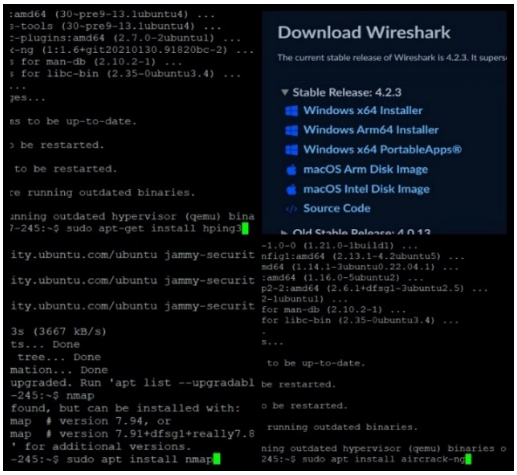

Fig. 6 Tool installation

The result using Nmap, as shown in Fig. 8, when analyzing the VM network was the initial information was from port 21 FTP without version information and port 22 SSH indicating the use of Ubuntu Linux 2.0.

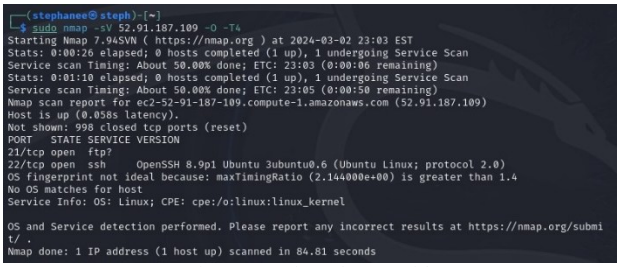

Fig. 7 Scanning of ports with Nmap

An NMAP command, as shown in Fig. 9, designed for vulnerability analysis resulted in a CVE-2011-1002 vulnerability and more information on other open ports such as the 1883 dedicated to MQTT communications.

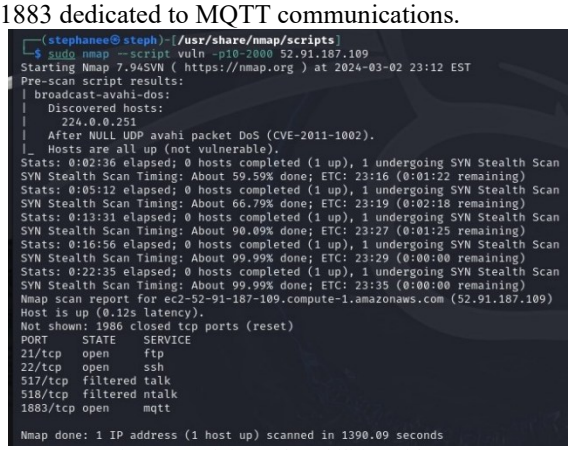

Fig. 9 Examining vulnerabilities with Nmap

When performing Nmap brute force attacks to obtain credentials on VM, credentials were not obtained to access them as shown in Fig. 10.

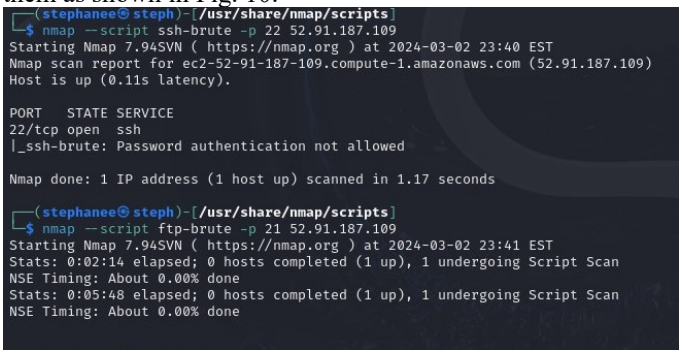

Fig. 10 Vulnerability exploitation in open ports

The denial of service attack performed on the VM IP address using Ping flood with HPING3 failed to affect the system and instead weakened the attacker machine's internet connection, as shown in Fig. 11.

Fig. 11 Error al ingresar a Putty ante DoS

When the vulnerability is not exploited, the following test consists of a password theft with Aircrack-ng to the LAN network where the ESP8266 located in a WiFi network is connected to perform the audit, as shown in Fig. 12.

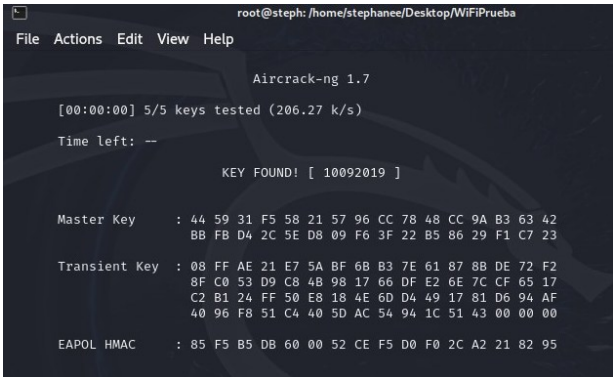

Fig. 12 Cracking of WiFi network

With the private IP address obtained from access to the WiFi network, as shown in Fig. 13, the vulnerability analysis was performed where DoS vulnerability was exploited using the Hping3 tool.

Fig. 13 Exploitation of vulnerability in ESP8266

With the Wireshark tool it was possible to visualize the sending of packets that saturated the ESP8266 that consisted of more than two hundred ninety-two thousand one hundred and eighty-one packets in less than 13 seconds, as shown in Fig. 14.

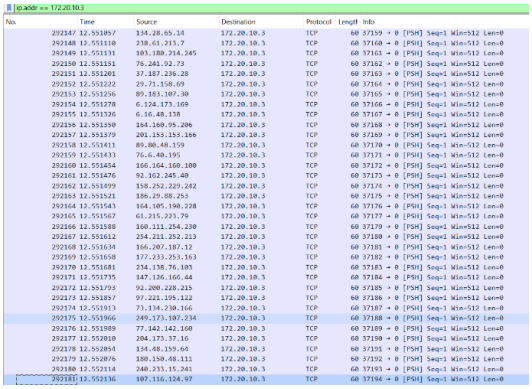

Fig. 14 Scanning of incoming ESP8266 traffic

With the attack carried out, the sending of data from the DHT11 sensor to the VM in AWS is disabled, so the communication system was successfully exploited.

B. Test Analysis

To reduce password thefts to WiFi networks, network security was configured by placing stronger passwords and changing them periodically every 8 or more days. As another security measure were limited connections to any device users are created with passwords from Broker Mosquitto and linked to MQTT communication.

To limit attacks to other ports, ports were restricted to only the necessary ones such as port 1883 and port 80 by configuring VM exit rules in AWS, as shown in Fig. 15.

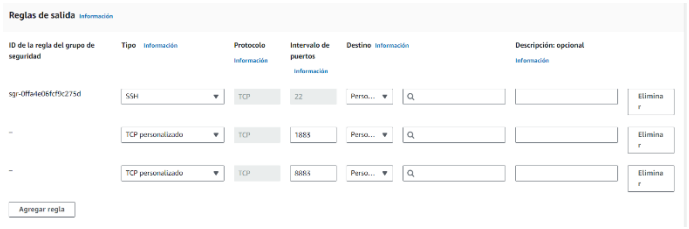

Fig. 15 Port restriction in the instance

Using IAM AWS allows access to VM resources where it allows the integration of different services to the EC2 instance, as shown in Fig. 16.

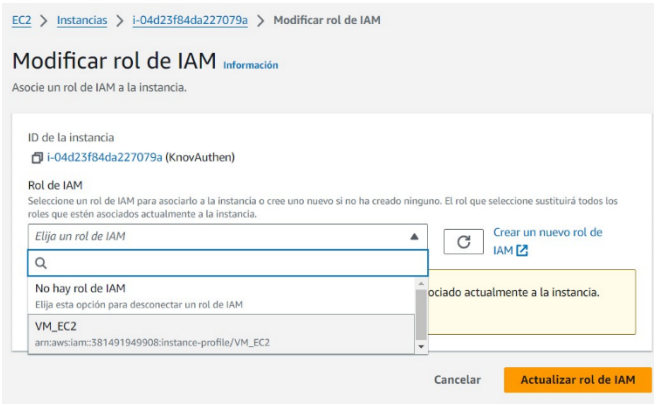

Fig. 16 Role addiction

To improve VM logins and mitigate an attack on port 22, the AWS Systems Manager (SSM) service was used to start controlled sessions, as shown in Fig. 17.

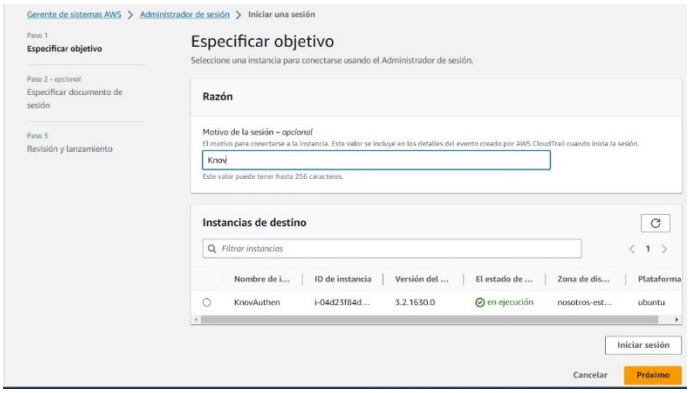

Fig. 17 Secure login with SSM

To keep a monitoring of system activity the CloudWatch tool was included and thus the performance of the VM when processing MQTT system messages was known. It also helped us to collect information about the installed instance such as the collection of CPU metrics, Memory, Disk and many more metrics for activation, as shown in Fig. 18.

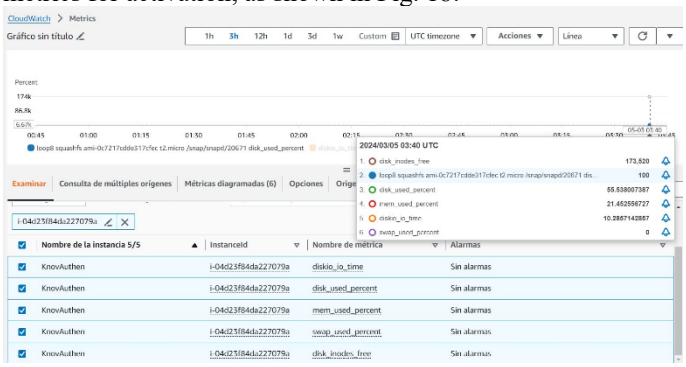

Fig. 18 CloudWatch metrics

VI. LIMITATIONS

One limitation of our approach is its dependency on the knowledge and experience of the Pentester. Moreover, the scope of our analysis is limited to the MQTT protocol on AWS infrastructure and may not generalize to other IoT platforms or protocols. The testing environment does not fully replicate real-world deployment complexities such as device heterogeneity, large-scale data traffic, and long-term operation scenarios.

V. SUMMARY AND CONCLUSIONS

In this research, the weaknesses of the system were identified, along with their respective actions to mitigate the vulnerabilities of the system. Performing these actions in the communication systems is a necessity for the use of these applications to reduce the risks that communication systems may have when passing their information to the Internet. Security in any communication device is vital to protect the information of individuals and organizations, since important information passes through the devices and can give rise to different attacks such as data theft, falsification of information or denial of services. The communication system implemented promised a high level of security by applying cloud services and the MQTT protocol according to the information gathered. Although the design of the system was simple, the ability to communicate is in real time by having an intermediary in the communication such as the Mosquitto Broker. The use of these factors together makes an effective communication every five seconds and in security analysis to the VM located in EC2 AWS it was verified that the DoS attacks to the VM in AWS were not satisfactory, since AWS has security in its connections through port 22 (SSH) to only achieve access with their respective private key file PuTTY. The MQTT traffic and the use of the VM command line in AWS could not be avoided due to the amount of packets sent from the Kali machine is not affected at all, but affected the machine where the attack was carried out. This was because AWS has a built-in service called AWS Shield that protects applications running on AWS from DDoS attacks, plus the bandwidth of the victim is greater than that of the attacker. But with the private IP address of the ESP8266 obtained from the Wi-Fi network access, vulnerability analysis was performed where the DoS vulnerability was exploited. Another communication vulnerability was access to all users from the Mosquitto Broker which was resolved by applying authentication to the Broker. In addition to maintaining console access with secure credentials, but continuing to implement improvements to logins, a controlled and monitored login is achieved that allows only primary usage to the customer linked to their AWS account. One way to mitigate vulnerabilities in Wi-Fi networks is to change passwords to passwords of up to 15 digits and characters. As well as reducing areas of brute force attacks to open ports that are of no use to the IoT communication system.

REFERENCES

[1] C. Aggarwal y K. Srivastava, «Securing IOT devices using SDN and edge computing», en 2016 2nd International Conference on Next Generation Computing Technologies (NGCT), oct. 2016, pp. 877-882. doi: 10.1109/NGCT.2016.7877534.

[2] ThreatLabZ Report, «The State of Encrypted Attacks | 2022 ThreatLabZ Report | Zscaler». Accedido: 28 de enero de 2024. [En línea]. Disponible en: https://info.zscaler.com/resources-industry-reports-the-state-of-encrypted-attacks-2022

[3] Zscaler ThreatLabz, «Zscaler ThreatLabz 2023 Enterprise IoT and OT Threat Report | Zscaler». Accedido: 27 de enero de 2024. [En línea]. Disponible en: https://info.zscaler.com/resources-industry-reports-threatlabz-2023-enterprise-ioT-ot-threat-report?_gl=1*1e1npk*_ga*NDM3MTMzNjE2LjE3MDYzOTQyNzI.*_ga_10SPJ4YJL9*MTcwNjM5NDI3MS4xLjAuMTcwNjM5NDI4Mi40OS4wLjA.&_ga=2.60490221.136761948.1706394272-437133616.1706394272

[4] M. Kashyap, V. Sharma, y N. Gupta, «Taking MQTT and NodeMcu to IOT: Communication in Internet of Things», Procedia Computer Science, vol. 132, pp. 1611-1618, ene. 2018, doi: 10.1016/j.procs.2018.05.126.

[5] A. Zare y M. T. Iqbal, «Low-Cost ESP32, Raspberry Pi, Node-Red, and MQTT Protocol Based SCADA System», en 2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), sep. 2020, pp. 1-5. doi: 10.1109/IEMTRONICS51293.2020.9216412.

[6] D. Z. Fawwaz, S.-H. Chung, C.-W. Ahn, y W.-S. Kim, «Optimal Distributed MQTT Broker and Services Placement for SDN-Edge Based Smart City Architecture», Sensors, vol. 22, n.o 9, Art. n.o 9, ene. 2022, doi: 10.3390/s22093431.

[7] A. Khurshid, R. Alsaaidi, M. Aslam, y S. Raza, «EU Cybersecurity Act and IoT Certification: Landscape, Perspective and a Proposed Template Scheme», IEEE Access, vol. 10, pp. 129932-129948, 2022, doi: 10.1109/ACCESS.2022.3225973.

[8] Bhatt Smriti, P. KThanh Kim, Maanak Gupta, James Benson, Jaehong Park, y Ravi Sandhu, «Attribute-Based Access Control for AWS Internet of Things and Secure Industries of the Future». Accedido: 24 de diciembre de 2023. [En línea]. Disponible en: https://ieeexplore.ieee.org/abstract/document/9502070

[9] A.-M. Berindei, C. Ilie, y B. Florentina, «The Cyber Security Paradigm in Industry 4.0», International Journal of Mechatronics and Applied Mechanics, n.o 13, pp. 226-229, 2023.

[10] I. Berizka, R. Romanyshyn, y O. Savitskyy, «ПРОГРАМНО-КОНФІГУРОВАНЕ IOT-РІШЕННЯ НА ОСНОВІ МІКРОКОНТРОЛЕРА ESP32», Електроніка та інформаційні технології/ Electronics and information technologies, n.o 19, Art. n.o 19, dic. 2022, doi: 10.30970/eli.19.7.

[11] B. C. C. Sánchez, E. a sitio externo E. enlace se abrirá en una ventana nueva, C. Olarte, y E. a sitio externo E. enlace se abrirá en una ventana nueva, «Análisis de seguridad entre microservicios con Amazon Web Service», Revista Logos, Ciencia & Tecnologia, vol. 14, n.o 2, pp. 42-52, 2022, doi: 10.22335/rlct.v14i2.1546.

[12] Amazon Web Services, Inc, «What is Amazon EC2? - Amazon Elastic Compute Cloud». Accedido: 30 de enero de 2024. [En línea]. Disponible en: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html

[13] M. Ghobaei-Arani, M. Shamsi, y A. A. Rahmanian, «An efficient approach for improving virtual machine placement in cloud computing environment», Journal of Experimental & Theoretical Artificial Intelligence, vol. 29, n.o 6, pp. 1149-1171, nov. 2017, doi: 10.1080/0952813X.2017.1310308.

[14] Y. Saadi y S. El Kafhali, «Energy-efficient strategy for virtual machine consolidation in cloud environment», Soft Comput, vol. 24, n.o 19, pp. 14845-14859, oct. 2020, doi: 10.1007/s00500-020-04839-2.

[15] F. Azzedin y T. Alhazmi, «Secure Data Distribution Architecture in IoT Using MQTT», Applied Sciences, vol. 13, n.o 4, Art. n.o 4, ene. 2023, doi: 10.3390/app13042515.

[16] S. Kumar y K. Kumar, «IRSC: Integrated automated Review mining System using Virtual Machines in Cloud environment», en 2018 Conference on Information and Communication Technology (CICT), oct. 2018, pp. 1-6. doi: 10.1109/INFOCOMTECH.2018.8722387.

[17] H. C. Hwang, J. Park, y J. G. Shon, «Design and Implementation of a Reliable Message Transmission System Based on MQTT Protocol in IoT», Wireless Pers Commun, vol. 91, n.o 4, pp. 1765-1777, dic. 2016, doi: 10.1007/s11277-016-3398-2.

[18] R. A. Atmoko, R. Riantini, y M. K. Hasin, «IoT real time data acquisition using MQTT protocol», J. Phys.: Conf. Ser., vol. 853, n.o 1, p. 012003, may 2017, doi: 10.1088/1742-6596/853/1/012003.

[19] S. Chakraborty y P. S. Aithal, «Let Us Create a Physical IoT Device Using AWS and ESP Module», International Journal of Management, Technology and Social Sciences (IJMTS), vol. 8, n.o 1, Art. n.o 1, mar. 2023, doi: 10.47992/IJMTS.2581.6012.0265.

[20] N. Imtiaz Jaya y Md. F. Hossain, «A Prototype Air Flow Control System for Home Automation Using MQTT Over Websocket in AWS IoT Core», en 2018 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), oct. 2018, pp. 111-1116. doi: 10.1109/CyberC.2018.00032.

[21] K. Hwang, J. M. Lee, I. H. Jung, y D.-H. Lee, «Modification of Mosquitto Broker for Delivery of Urgent MQTT Message», en 2019 IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE), oct. 2019, pp. 166-167. doi: 10.1109/ECICE47484.2019.8942800.

[22] B. Mishra, B. Mishra, y A. Kertesz, «Stress-Testing MQTT Brokers: A Comparative Analysis of Performance Measurements», Energies, vol. 14, n.o 18, Art. n.o 18, ene. 2021, doi: 10.3390/en14185817.

[23] M. A. Spohn, «On MQTT Scalability in the Internet of Things: Issues, Solutions, and Future Directions». Accedido: 19 de diciembre de 2023. [En línea]. Disponible en: https://ojs.wiserpub.com/index.php/JEEE/article/view/1687

[24] R. Banno y T. Yoshizawa, «A Scalable IoT Data Collection Method by Shared-Subscription with Distributed MQTT Brokers», en Mobile Networks and Management, C. T. Calafate, X. Chen, y Y. Wu, Eds., en Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering. Cham: Springer International Publishing, 2022, pp. 218-226. doi: 10.1007/978-3-030-94763-7_17.

[25] M. L. Pham, T. T. Nguyen, y M. D. Tran, «A Benchmarking Tool for Elastic MQTT Brokers in IoT Applications», International Journal of Information and Communication Sciences, vol. 4, n.o 4, Art. n.o 4, dic. 2019.

[26] M. Fagan, K. N. Megas, K. Scarfone, y M. Smith, «Foundational cybersecurity activities for IoT device manufacturers», National Institute of Standards and Technology, Gaithersburg, MD, NIST IR 8259, may 2020. doi: 10.6028/NIST.IR.8259.

[27] Y. Seralathan et al., «IoT security vulnerability: A case study of a Web camera», en 2018 20th International Conference on Advanced Communication Technology (ICACT), feb. 2018, pp. 172-177. doi: 10.23919/ICACT.2018.8323686.

[28] R. Williams, E. McMahon, S. Samtani, M. Patton, y H. Chen, «Identifying vulnerabilities of consumer Internet of Things (IoT) devices: A scalable approach», en 2017 IEEE International Conference on Intelligence and Security Informatics (ISI), Beijing, China: IEEE, jul. 2017, pp. 179-181. doi: 10.1109/ISI.2017.8004904.

[29] H. M. Z. A. Shebli y B. D. Beheshti, «A study on penetration testing process and tools», en 2018 IEEE Long Island Systems, Applications and Technology Conference (LISAT), may 2018, pp. 1-7. doi: 10.1109/LISAT.2018.8378035.

[30] Z. Yang, Y. Yue, Y. Yang, Y. Peng, X. Wang, y W. Liu, «Study and application on the architecture and key technologies for IOT», en 2011 International Conference on Multimedia Technology, jul. 2011, pp. 747-751. doi: 10.1109/ICMT.2011.6002149.

[31] M. O. Reddy, «MQTT-SN GATEWAY INTEGRATION WITH AWS CLOUD MQTT SERVER AND IOT APPLICATION».

[32] C. Aggarwal and K. Srivastava, "Securing IOT devices using SDN and edge computing," 2016 2nd Int. Conf. on Next Generation Computing Technologies (NGCT), 2016, pp. 877–882. doi: 10.1109/NGCT.2016.7877534.

[33] A. Zare and M. T. Iqbal, "Low-Cost ESP32, Raspberry Pi, Node-Red, and MQTT Protocol Based SCADA System," 2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), 2020, pp. 1–5. doi: 10.1109/IEMTRONICS51293.2020.9216412.