Research Document: COMMUNICATION PROTOCOLS

ARJUN POGIRI

April 11, 2023

Introduction

In the field of computer networking, understanding the OSI model, Ethernet frames, and network protocol analyzers such as Wireshark is essential.MQTT (Message Queuing Telemetry Transport) ,The Publisher-Subscriber model (PUB / SUB),serial communication protocol MODBUS and RS-485. In this research document, we will explore these topics in-depth, providing a comprehensive overview of the OSI model,Ethernet Frames, Wireshark , MQTT , PUB/SUB , MODBUS and RS-485.

1 OSI Model

The OSI model is a conceptual framework that describes how data is transmitted and received over a network. It consists of seven layers, each of which serves a specific function in the process of data communication. In this section, we will discuss each layer of the OSI model in detail, along with examples.

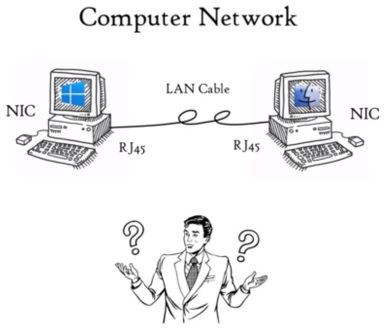

Figure 1: How it works?

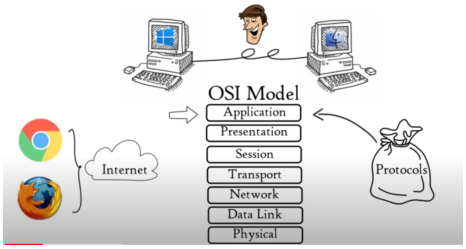

Figure 2: OSI MODEL

1.1 Physical Layer

The physical layer deals with the transmission of raw data over a physical medium such as copper cables, fiber optics, or radio waves. It provides a physical interface between the device and the network.

Examples of the physical layer:

include Ethernet, RS-232, and USB. Ethernet uses twisted pair copper cables to transmit data over a network, while RS-232 uses serial communication over a physical cable.

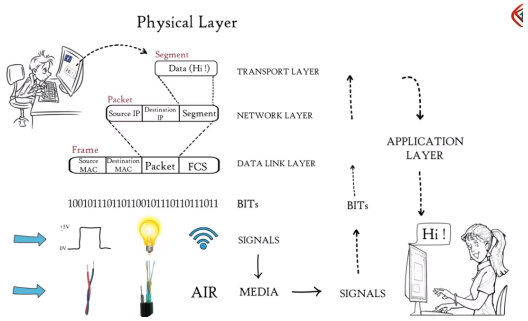

Figure 3: physical layer

1.2 Data Link Layer

The data link layer is responsible for ensuring that data is transmitted reliably and efficiently between two devices on the same physical network. It breaks data into frames, adds error-checking information, and manages access to the physical network.

Examples of the data link layer:

include MAC (Media Access Control) addresses and Ethernet switches. MAC addresses are unique identifiers assigned to network interface cards, and Ethernet switches manage the flow of data between different devices on a network.

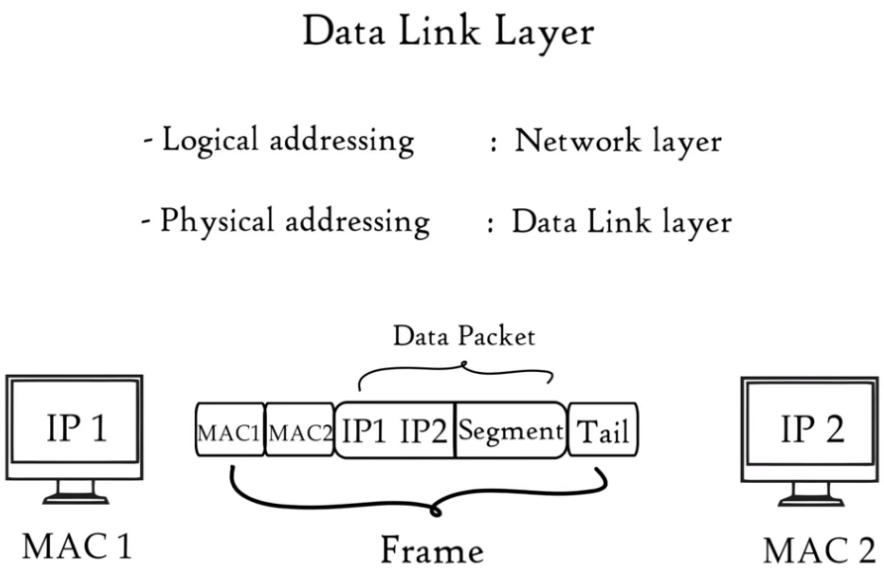

Figure 4: data link layer-1

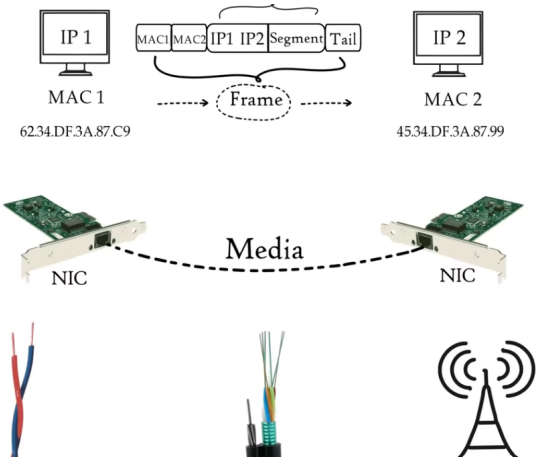

Figure 5: data link layer-2

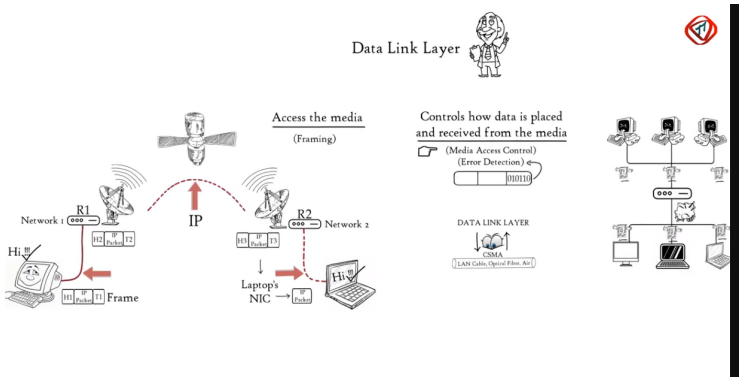

Figure 6: data link layer-3

1.3 Network Layer

The network layer is responsible for addressing, routing, and delivering data between different networks. It adds logical addresses to data packets and uses routing protocols to determine the most efficient path for data to travel. Examples of the network layer: include IP (Internet Protocol) addresses and routers. IP addresses are unique identifiers assigned to devices on a network, and routers are devices that connect multiple networks together.

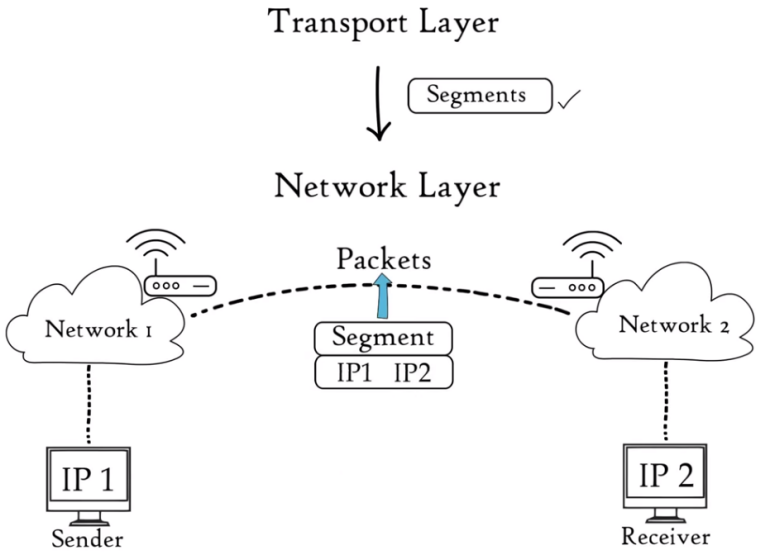

Figure 7: network layer-1

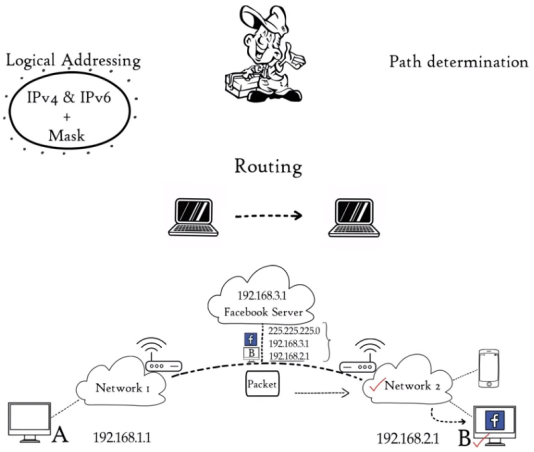

Figure 8: network layer-2

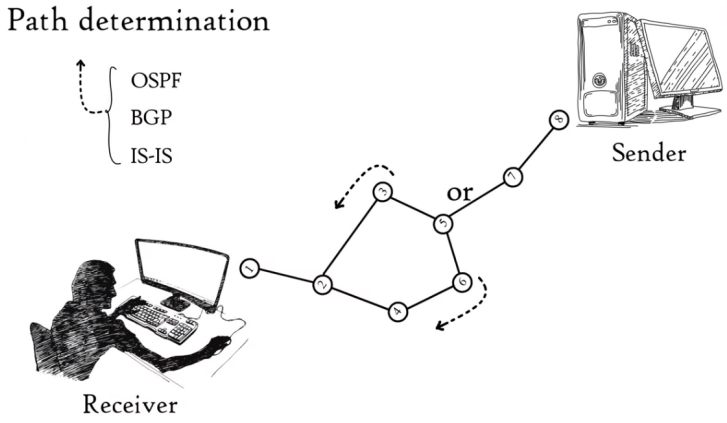

Figure 9: network layer-3

1.4 Transport Layer

The transport layer is responsible for ensuring that data is transmitted reliably and efficiently between applications on different devices. It breaks data into segments, adds error-checking information, and manages flow control. Examples of the transport layer: include TCP (Transmission Control Protocol) and UDP (User Datagram Protocol). TCP provides reliable, ordered delivery of data between applications, while UDP provides faster, but less reliable delivery.

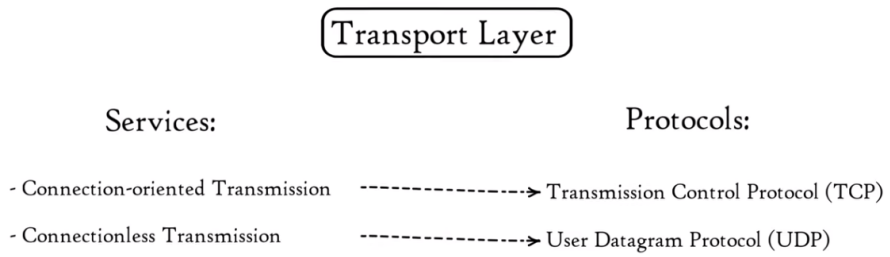

Figure 10: udp-tcp

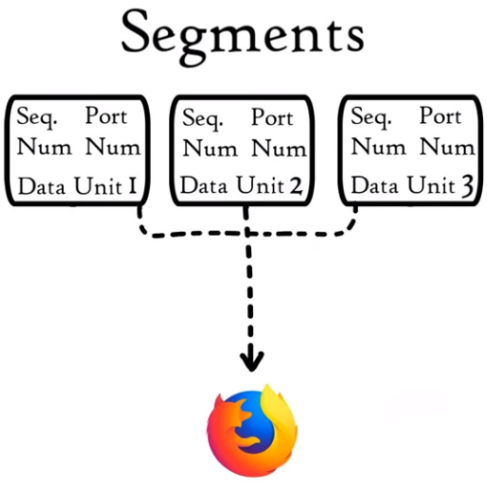

Figure 11: segments

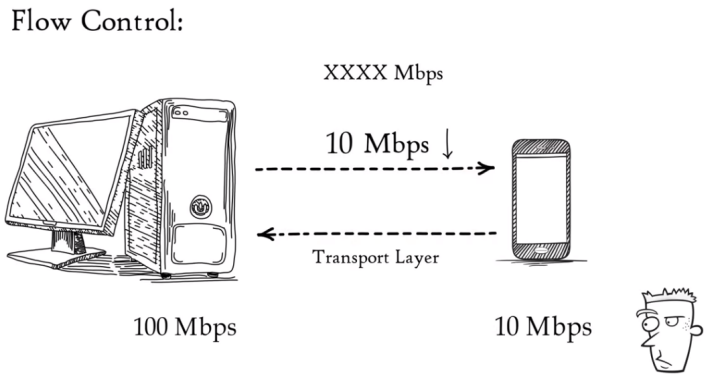

Figure 12: flowcontrol

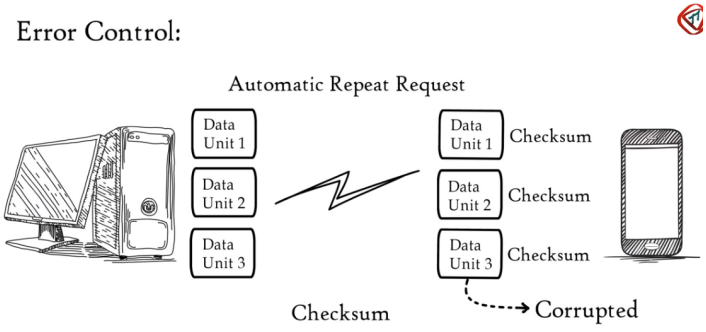

Figure 13: error control

1.5 Session Layer

The session layer is responsible for managing the communication between different applications on different devices. It establishes, maintains, and terminates sessions between applications, and manages checkpoints and recovery. Examples of the session layer: include remote procedure calls and network file systems. Remote procedure calls allow a program to execute a function on a remote device, while network file systems allow access to files on remote devices as if they were local.

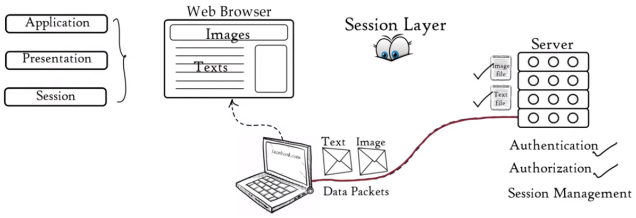

Figure 14: session layer

1.6 Presentation Layer

The presentation layer is responsible for transforming data into a format that can be understood by the application. It handles data encryption, compression, and translation between different data formats. Examples of the presentation layer: include encryption algorithms such as AES and compression algorithms such as JPEG.

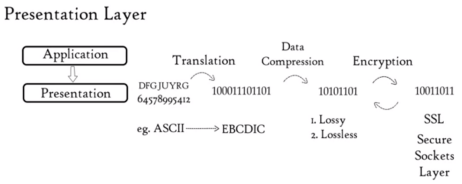

Figure 15: presentation layer

1.7 Application Layer

The application layer is responsible for providing services to end-users. It includes protocols and services such as email, file transfer, and web browsing. Examples of the application layer: include HTTP (Hypertext Transfer Protocol), SMTP (Simple Mail Transfer Protocol), and FTP (File Transfer Protocol). HTTP is used for web browsing, SMTP is used for sending email, and FTP is used for transferring files.

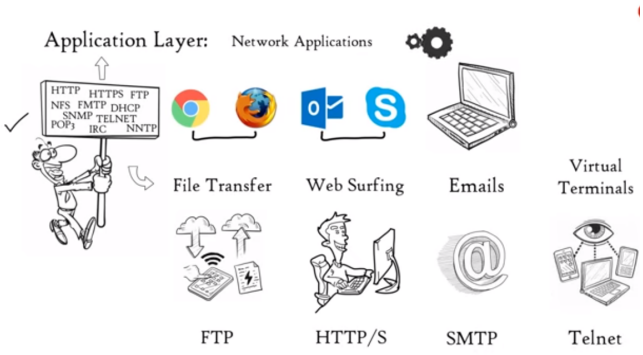

Figure 16: application layer

2 Ethernet Frames

Ethernet frames are used to transmit data on Ethernet networks, which are the most widely used local area network (LAN) technology in the world. we saw this in the datalink layer. An Ethernet frame is composed of several parts that include:

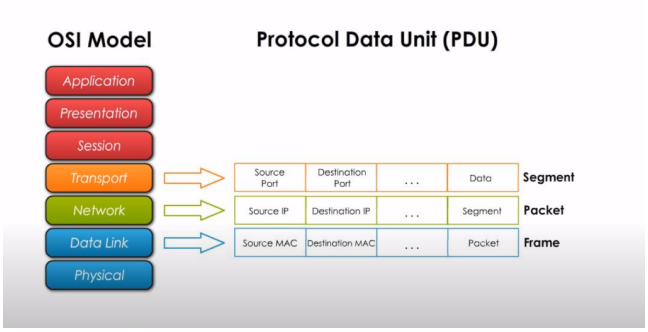

Figure 17: Ehternet

Figure 18: Ehternet Frames

2.1 Preamble :

64-bit information telling the receiving NIC(network interface card) that a frame is coming and when the formate starts. it contains alternating 0's and 1's used to synchronize the clocks of the sender and receiver.

2.2 Destination MAC address:

The destination MAC address is a unique identifier that is used to identify the intended recipient of the frame. This address is used by the network devices to decide whether or not to process the frame. If the destination MAC address matches the MAC address of the network device, then the frame is processed by that device.

2.3 Source MAC address:

The source MAC address is also a unique identifier that is used to identify the sender of the frame. This address is used by the network devices to determine where the frame originated from. This information is used to send a response back to the sender if necessary.

2.4 Type:

telling the recipient basic type of data such as ipV4 or ipV6. not higher-level data such as email or web pages.

2.5 Data:

whatever payload the frames carry such as IP packets. an IP packet delivered from the network layer. the payload limit is 1500 bytes.

2.6 Pad:

the pad is extra bits to make a frame. at least bigger than 64 bytes because any data unit smaller than 64 bytes would be considered a collision. the receiving device would simply ignore it. if is size less than 64 bytes NIC adds extra bytes.

the minimum frame size is 64 bytes

2.7 FCS:

FCS stands for frame check sequence. error checking mechanism. It is a 4-byte sequence used to verify the integrity of the data transmitted in the frame. The FCS is calculated by the sender based on the contents of the frame and is verified by the receiver.

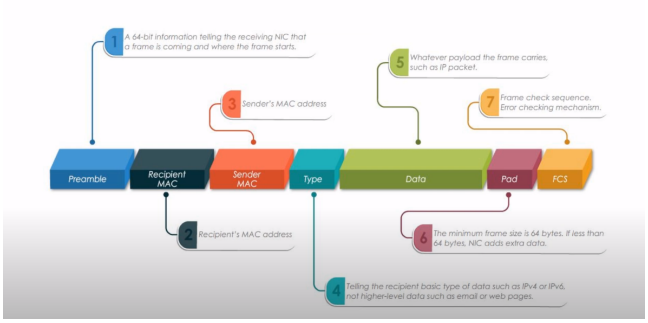

Figure 19: Frames

3 Wireshark

Wireshark is an open-source network sniffing software, which is designed to track network

Figure 20: Wireshark

packets and through the use of different filter options available in the software Wireshark is a popular open-source network protocol analyzer that allows you to inspect network traffic in real time. With Wireshark, you can capture and analyze network packets to understand how data is flowing between devices on a network. including its support for a wide range of protocols and its user-friendly interface. Wireshark supports a wide range of protocols, including TCP/IP, HTTP, DNS, and FTP, among others. It provides a user-friendly interface for visualizing and analyzing packet data, allowing you to quickly identify network issues and troubleshoot problems.

Wireshark is commonly used by network administrators, security professionals, and developers to diagnose network problems, identify security vulnerabilities, and debug network protocols.

working for :

1 . used to analyze network packets

2 . troubleshoot network issues

3 . check malicious and hacked possibilities

4 MQTT(Message Queuing Telemetry Transport)

MQTT is a lightweight messaging protocol that is designed for use in situations where network bandwidth and power consumption are limited, such as in the Internet of Things (IoT) and Machine-to-Machine (M2M) communication. MQTT components

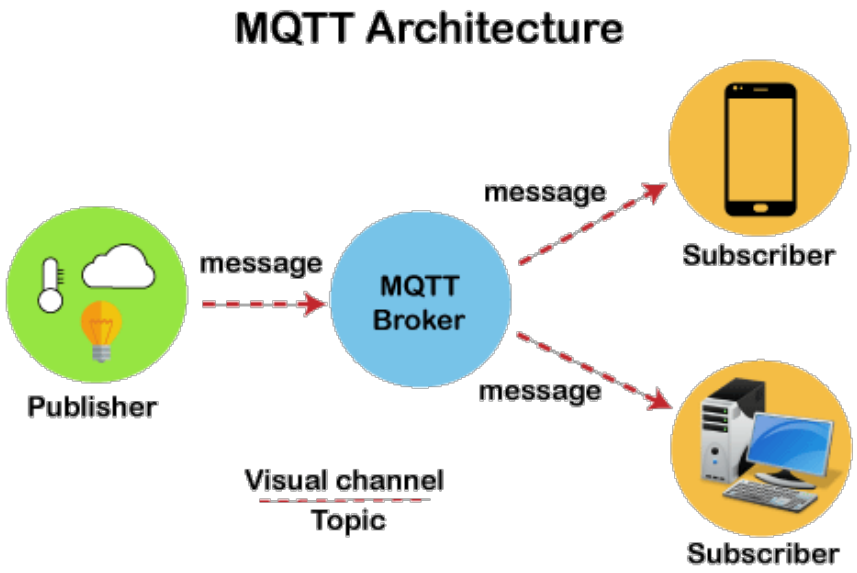

Figure 21: MQTT Architecture

4.1 MESSAGE:

the data carried by the MQTT protocol across the network when a message is transported by MQTT it contains

1. payload data: The payload data is usually in the form of a string of bytes that can be interpreted and processed by the receiving device or application. The payload data is the actual content of the message that is being transmitted and is what is ultimately processed by the receiving device or application. For example, in an IoT application, the payload data might represent sensor data such as temperature, humidity, or pressure readings, or it could be a command to control a device, such as turning on a light or activating an alarm.

2. Quality of service (QoS):

Quality of Service (QoS) refers to the level of assurance that is provided for the delivery of messages between MQTT clients (i.e., publishers and subscribers). MQTT supports three levels of QoS: 0, 1, and 2.

QoS 0: At QoS 0, the publisher sends the message to the broker once, without any confirmation of delivery. The broker then sends the message to all subscribed clients. This level of QoS provides the lowest assurance of delivery, as there is no guarantee that the message will be received by the subscriber.

QoS 1: At QoS 1, the publisher sends the message to the broker, which then acknowledges receipt of the message. The broker then sends the message to all subscribed clients and continues to try to deliver the message until it receives an acknowledgment from the subscriber. This level of QoS provides a higher assurance of delivery than QoS 0, as the message is guaranteed to be delivered at least once.

QoS 2: At QoS 2, the publisher sends the message to the broker, which then sends a confirmation to the publisher that the message has been received. The broker then sends the message to all subscribed clients and waits for an acknowledgement from the subscriber. Once the acknowledgement is received, the broker sends a confirmation back to the publisher. This level of QoS provides the highest assurance of delivery, as the message is guaranteed to be delivered exactly once.

3. Topic name: a topic name is a string that is used to identify and route messages between MQTT clients. Topics are hierarchical and are organized into a tree-like structure, with each level separated by a forward slash (/).

For example, a topic name might be "sensors/temperature/living$_r$oom", $where"sensors"isthetop-leveltopic,"temperature"isasub-topic,and"living_room"isasub-sub-topic.Inthisexample,thetopicnameindi$

4.2 Client:

clients subscribe to topics to publish and receive messages. thus subscriber and published are the special roles of a client

4.3 Server/Broker :

a program or device that acts as an intermediary between clients which publish applications messages and clients which have made subscriptions

1 . accepts network connections from clients

2 . accepts application messages published by clients

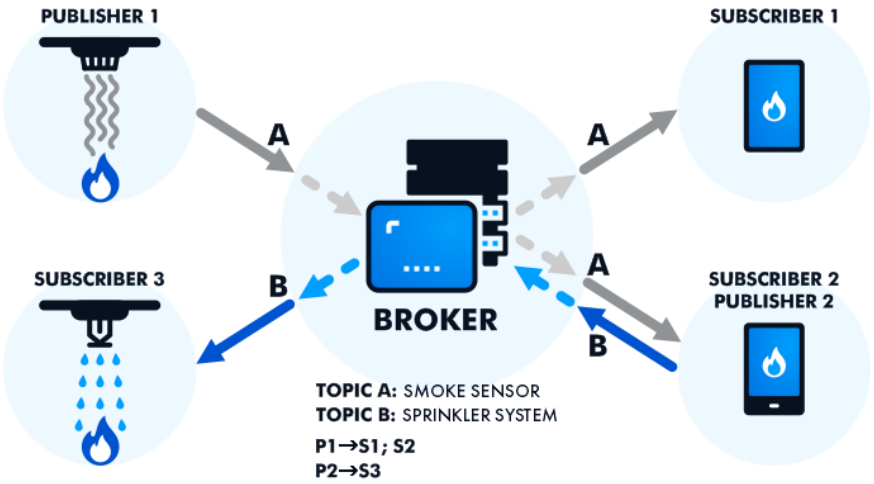

Figure 22: MQTT-workflow

3 . processes subscribe and unsubscribe requests from clients

4 . forward application messages that match client subscriptions

5 . closes the network connection from the client

4.4 Topic :

the label is attached to a message which is matched against the subscriptions known to the server

MQTT packet structure :

The MQTT protocol uses a simple packet structure for sending and receiving messages between clients and brokers.

The packet structure consists of four distinct packet types:

1 .CONNECT Packet: This packet is used to establish a connection between an MQTT client and a broker. It contains information such as the client ID, username, and password, as well as various connection options.

2 .PUBLISH Packet: This packet is used to publish a message to an MQTT broker. It contains the topic name, QoS level, message ID, and payload data.

3 .SUBSCRIBE Packet: This packet is used to subscribe to one or more topics on an MQTT broker. It contains the topic name and QoS level for each topic that the client wishes to subscribe to.

4 .UNSUBSCRIBE Packet: This packet is used to unsubscribe from one or more topics on an MQTT broker. It contains the topic names that the client wishes to unsubscribe from.

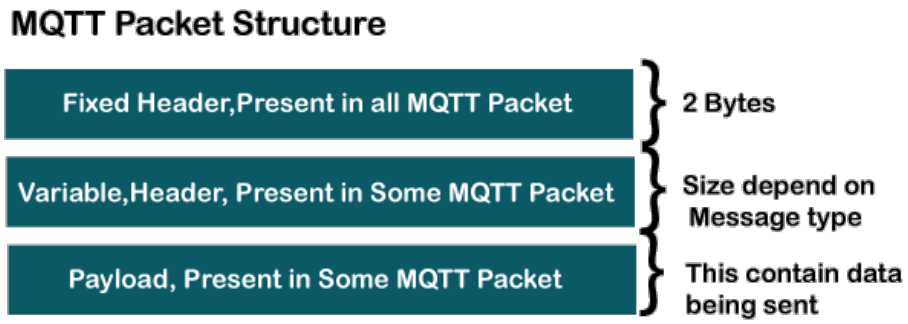

Figure 23: structure

Each packet in MQTT has a fixed header, which contains information such as the packet type, message length, and various control flags. The header is followed by a variable-length body, which contains the specific data for each packet type.

Overall, the packet structure in MQTT is designed to be simple and lightweight, making it well-suited for use in IoT applications where resources may be limited.

5 PUB / SUB

The Publisher-Subscriber model is a messaging pattern used in computer software architecture. In this model, there are two types of entities: publishers and subscribers. Publishers are responsible for generating events or messages, while subscribers receive and process these events or messages.

The basic idea behind the Publisher-Subscriber model is that publishers and subscribers are decoupled, meaning that they do not have to be aware of each other's existence. Instead, publishers simply publish events or messages to a message broker or channel, and subscribers register their interest in receiving certain types of messages from that channel. Whenever a message is published to the channel, the message broker forwards it to all subscribers that have registered for that type of message.

This model is commonly used in distributed systems and event-driven architectures, as it provides a scalable and flexible way to distribute information across multiple components without creating tight coupling between them. It is also known as the "Pub/Sub" model, and is often implemented using message queues or other messaging systems.

5.1 ADVANTAGES:

The Publisher-Subscriber (Pub/Sub) model offers several advantages over other messaging patterns, including:

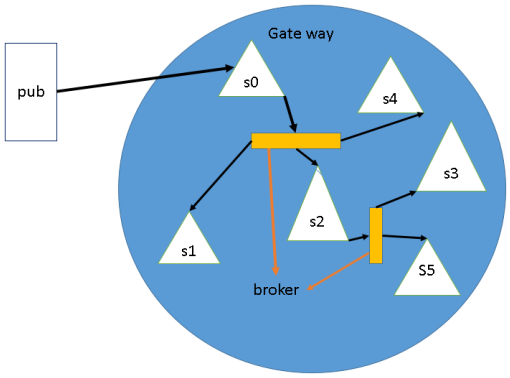

Figure 24: PUB-SUB

Scalability: Pub/Sub allows for the efficient distribution of messages to multiple subscribers, making it easy to scale a system to handle large volumes of messages.

Decoupling: The Pub/Sub model decouples publishers and subscribers, meaning that they do not have to be aware of each other's existence. This reduces the risk of tight coupling between components, which can make systems less flexible and more difficult to maintain.

Flexibility: Pub/Sub allows subscribers to receive only the messages they are interested in, which provides greater flexibility in designing a system that can handle different types of messages.

Asynchronous processing: Pub/Sub allows for asynchronous message processing, meaning that subscribers can process messages at their own pace without having to wait for the publisher to complete its work.

Reliability: Pub/Sub messaging systems often provide mechanisms for ensuring message delivery and reliability, such as message acknowledgements and guaranteed delivery.

Overall, the Pub/Sub model is a powerful and flexible way to distribute information across multiple components in a system, and it offers many advantages for building scalable, flexible, and reliable distributed systems.

5.2 DISADVANTAGES:

While the Publisher-Subscriber (Pub/Sub) model has several advantages, it also has some potential disadvantages to consider:

Complexity: Implementing a Pub/Sub system can be more complex than other messaging patterns, especially when it comes to designing the message broker and managing subscriptions.

Latency: Pub/Sub systems may introduce additional latency in message delivery, especially when compared to more direct messaging patterns such as point-to-point messaging.

Message ordering: Since Pub/Sub systems distribute messages to multiple subscribers, maintaining the order of messages can be challenging, especially when multiple subscribers are processing messages asynchronously.

Security: Pub/Sub systems may introduce additional security concerns, such as ensuring that only authorized subscribers can access certain messages or topics.

Network overhead: Pub/Sub systems may introduce additional network overhead, as messages must be transmitted to a central message broker and then distributed to multiple subscribers.

Overall, the Pub/Sub model can be an effective way to distribute information in a distributed system, but it is important to carefully consider the potential disadvantages and trade-offs before implementing this pattern.

6 MODBUS

Modbus is a widely used serial communication protocol in industrial automation systems. It defines the format of the messages that are exchanged between devices over serial communication lines.

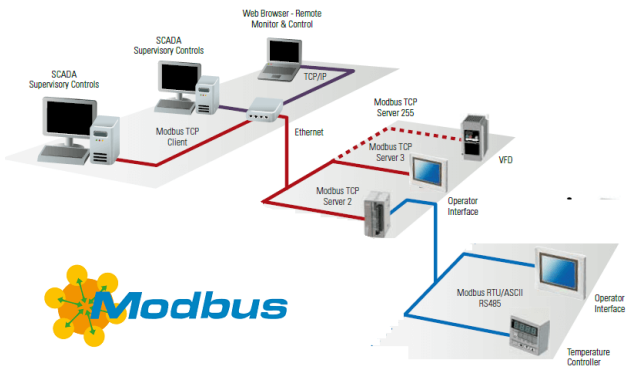

Figure 25: modbus

A Modbus message, also known as a Modbus frame, consists of several fields, including:

Address field: This field specifies the address of the device that will receive the message. It can be a broadcast address or an address of a specific device.

Function code field: This field specifies the type of action that the device should perform. Modbus defines a set of function codes, each of which corresponds to a specific action, such as reading or writing data.

Data field: This field contains the data that is being transmitted. The format and size of the data depend on the function code being used.

Error checking field: This field contains a checksum or CRC value that is used to check for errors in the message.

The frame begins with a 1-byte address field, followed by a 1-byte function code field. The data field can vary in size, depending on the function code being used. The error-checking field is typically a 2-byte checksum or CRC value.

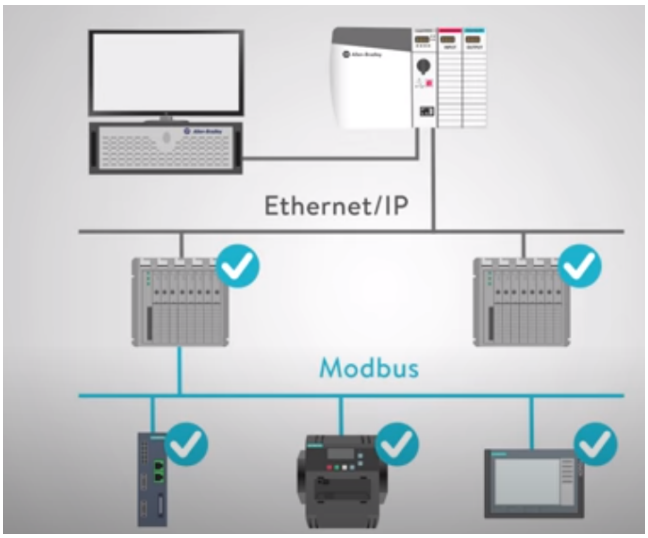

Figure 26: Modbus-1

In a Modbus transaction, the sender device sends a message to the receiver device, which then responds with a message of its own. The message exchange typically involves the following steps:

1. The sender sends a request message to the receiver, specifying the desired function code and data.

2 .The receiver device receives the message and processes the request.

3 .The receiver device sends a response message back to the sender, containing any requested data.

4 .The sender device receives the response message and processes the data as necessary.

Modbus frames are used for a variety of purposes in industrial automation systems, including reading and writing data to sensors, controlling actuators, and configuring devices. The Modbus protocol has been widely adopted in the industry due to its simplicity, flexibility, and reliability.

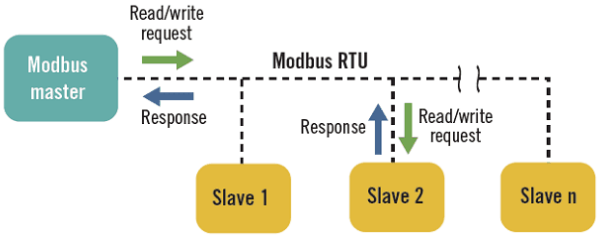

Figure 27: transaction

the modbus communication protocal is the oldest and by far most popular automation protocal in the field of process automation and SCADA (supervisory control and data acquisition)

modbus is able to function on both point to point and multidrop networks

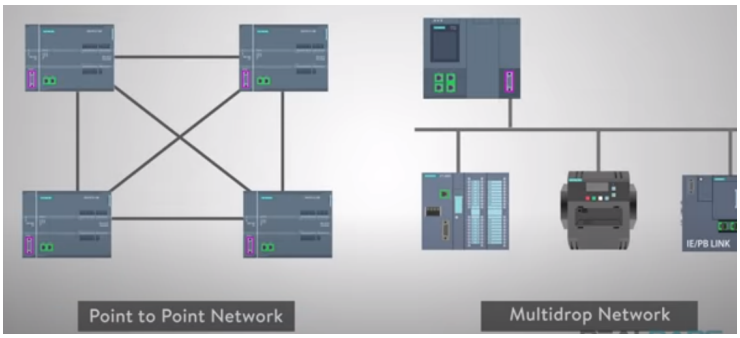

Figure 28: point to point and multipoint

7 RS-485

RS-485 is a serial communication standard used in a wide variety of industrial and commercial applications. It is a balanced, differential signaling standard that is designed to provide reliable long-distance communication over twisted pair cables. It was first introduced in 1983 by the Electronics Industries Alliance (EIA). RS-485 supports full-duplex communication over distances of up to 1,200 meters (4,000 feet) at data rates of up to 10 Mbps. It can also support half-duplex communication at distances of up to 1,200 meters (4,000 feet) at data rates of up to 2 Mbps. RS-485 uses differential signaling, which helps to reduce electromagnetic interference (EMI) and noise on the communication line.

RS-485 supports up to 32 devices on a single bus, with each device having a unique address. Devices communicate using a master-slave configuration, with the master device initiating communication and the slave devices responding to

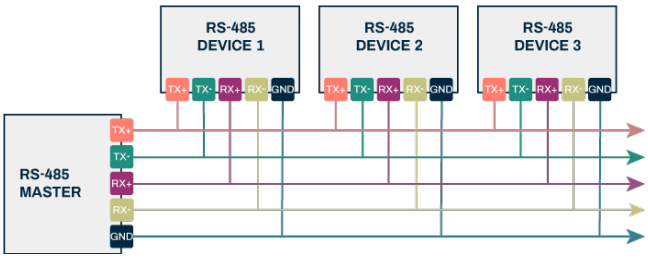

Figure 29: RS485

requests. The standard supports a variety of communication modes, including point-to-point, point-to-multipoint, and multipoint-to-multipoint.

One of the key advantages of RS-485 is its robustness and reliability in noisy industrial environments. The use of differential signaling and the ability to terminate the communication line with a resistor at each end helps to reduce EMI and noise. Additionally, the ability to support multiple devices on a single bus makes it a cost-effective solution for many applications.

RS-485 is commonly used in industrial automation, building automation, and HVAC control systems, among other applications. It is often used for communication between sensors, actuators, and other devices that need to communicate over long distances in harsh environments.

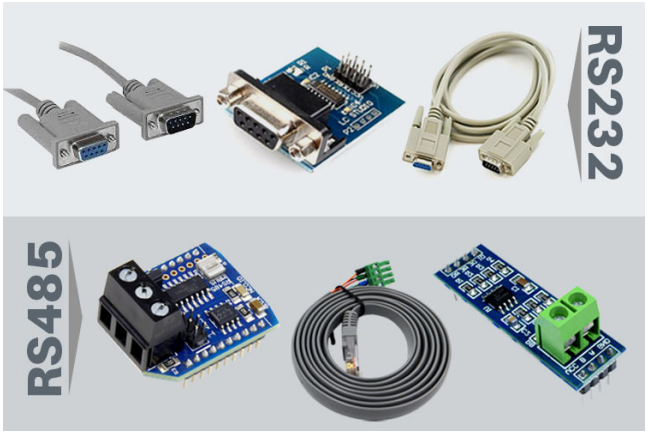

Figure 30: RS485-and-RS232

Conclusion

understanding the OSI model, Ethernet Frames, Wireshark ,MQTT , PUB /SUB , MODBUS and RS 485 are essential for anyone working in the field of Communication . By exploring these topics in depth, we have