Getting Started with IoT Gateways

Sai Krishna Dasari

April 9, 2023

1 What are IoT Gateways?

IoT Gateways acts as a bridge, that connects the sensor networks/devices to the Cloud IoT platform.

- Gateways provide a secure, managed connection between the devices and the network.
- They collect data from multiple IoT devices and aggregate it before sending it to the Cloud or Central Network

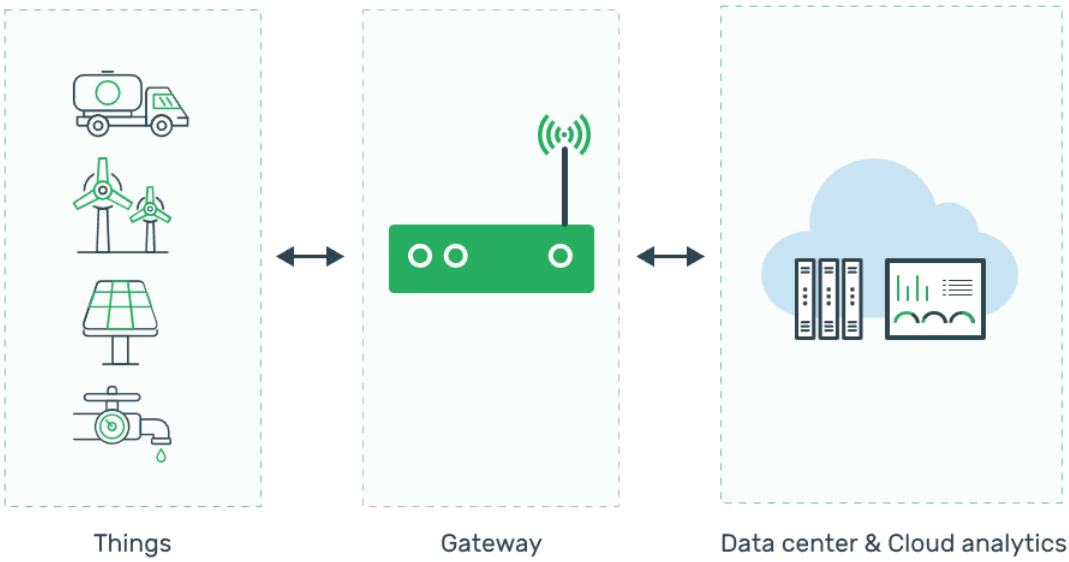

Figure 1: **IoT Gateway Block Diagram**

Why IoT Gateways are necessary?

- **Protocol translation:** IoT devices use a variety of protocols to communicate with each other and with the cloud, and these protocols may not be compatible with each other. IoT Gateways can translate between different protocols, making it possible for devices to communicate with each other and with the cloud.
- **Security:** IoT Gateways provide a secure way to transfer data between devices and the cloud. They can encrypt data and authenticate devices, protecting against data breaches and other security threats.
- **Edge computing:** IoT Gateways can perform data processing and analytics at the edge, reducing the amount of data that needs to be sent to the cloud for processing. This can help to reduce latency and bandwidth usage, and can make it possible to process data in real-time.
- **Device management:** IoT Gateways can manage the configuration and operation of IoT devices, providing a centralized way to manage a large number of devices. They can also monitor device health and diagnose issues, making it easier to troubleshoot problems.
- **Offline operation:** IoT Gateways can operate offline, allowing devices to continue to function even when they lose connectivity to the cloud. This can be important in remote or disconnected environments, where connectivity may be unreliable.

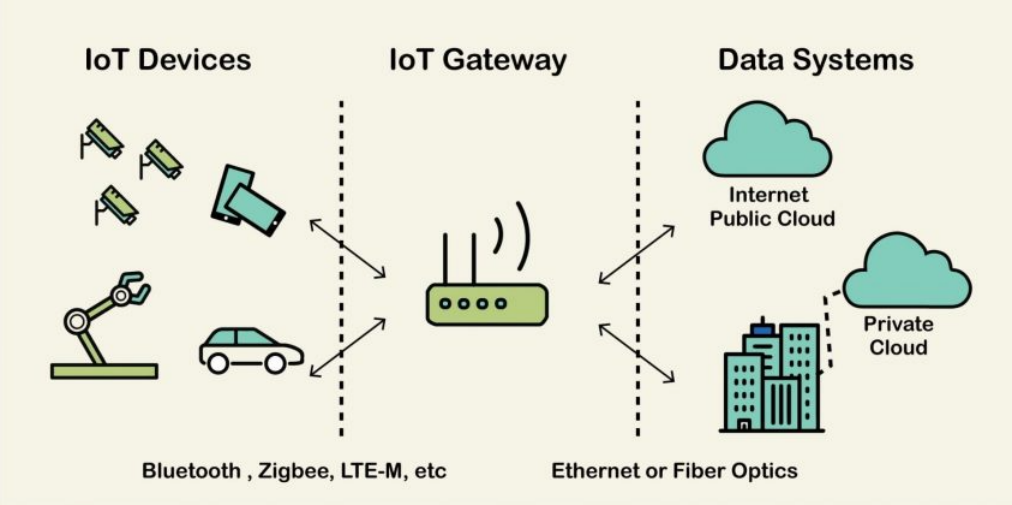

Figure 2: **IoT Gateway Block Diagram**

2 MQTT Protocol

MQTT (Message Queuing Telemetry Transport) is a lightweight messaging protocol designed for IoT (Internet of Things) devices and applications to exchange data between them. It is a publish-subscribe protocol that enables communication between devices and services.

How MQTT works ?

Let's use an example of a YouTube channel that wants to send notifications to its subscribers when a new video is uploaded.

First, the YouTube channel (publisher) would need to set up an MQTT broker (server) to manage the messaging between the publisher and subscribers. The publisher would then send a message to the broker whenever a new video is uploaded, including the video title and URL.

The subscribers would then subscribe to a particular topic on the broker, in this case, the topic for new video notifications. Whenever a new video is uploaded, the broker would then forward the message to all subscribers subscribed to that topic.

This allows subscribers to receive notifications in near real-time, even if they're using devices with limited resources or low-bandwidth networks.

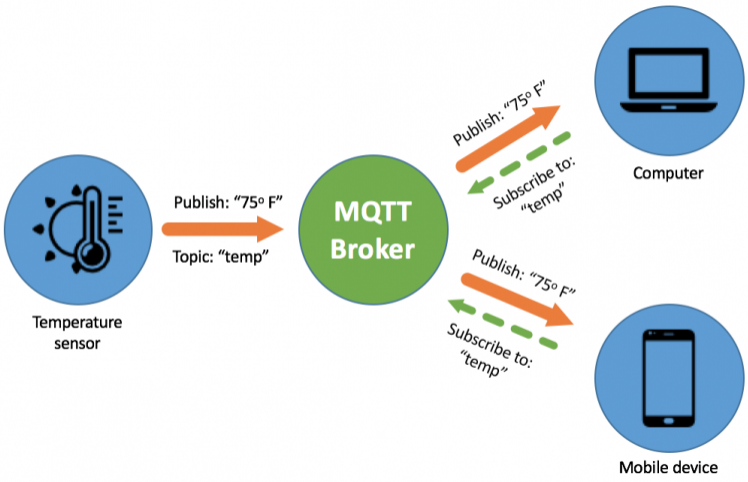

Figure 3: **MQTT Protocol Block Diagram**

In MQTT, messages are published to a broker, which then distributes them to subscribers who have subscribed to receive messages on specific topics. Topics are hierarchical and use a forward-slash to separate levels. For example, a topic could be "**sensors/temperature/room1**" where "**sensors**" is the **highest level** and "**room1**" is the **lowest level**.

Why MQTT a suitable protocol for IoT Gateways?

- Lightweight: IoT gateways often have limited processing power and memory, and MQTT's lightweight design makes it ideal for use in these resource-constrained environments.
- Low Bandwidth: IoT gateways may be connected to networks with limited bandwidth or high latency, and MQTT's efficient use of network bandwidth ensures that data can be transmitted quickly and without excessive network overhead.
- Scalability: IoT gateways often need to handle large numbers of devices and data streams, and MQTT's publish-subscribe model makes it easy to scale and manage these connections.
- Data Filtering: IoT gateways may receive a large amount of data, but only a subset of that data may be relevant for downstream processing. MQTT's topic-based subscription model allows gateways to filter incoming data and forward only the relevant data to downstream systems.
- Reliability: MQTT uses a Quality of Service (QoS) mechanism to ensure reliable message delivery, with the ability to recover from network failures, ensuring that data is transmitted consistently and accurately.
- Security: MQTT supports various security mechanisms, including TLS/SSL encryption, username/password authentication, and access control lists, which are critical for securing sensitive IoT data.

3 OSI Model

The Open Systems Interconnection (OSI) model describes seven layers that computer systems use to communicate over a network. It was the first standard model for network communications, adopted by all major computer and telecommunication companies in the early 1980s.

- **Physical layer:** This layer is responsible for the physical transmission of data over a network, including the types of cables, connectors, and hardware used.
- **Data link layer:** This layer provides error-free data transmission over a physical link. It deals with issues such as packet framing, error detection, and flow control.
- **Network layer:** This layer is responsible for routing data packets between different networks. It determines the best path for data transmission and can handle issues such as congestion and addressing.

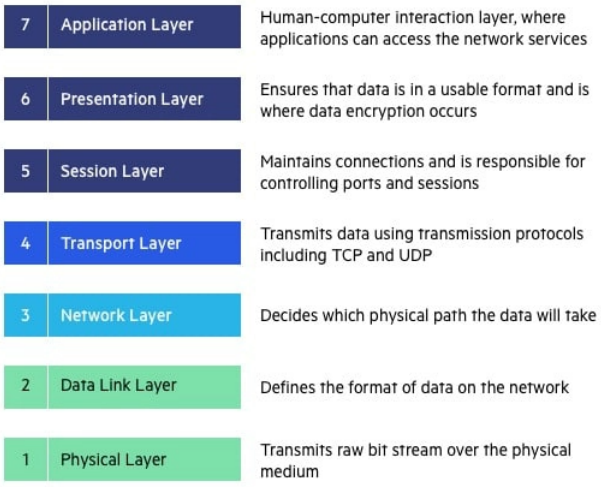

Figure 4: **OSI Model 7 Layers**

- **Transport layer:** This layer ensures the reliable transmission of data between endpoints. It provides error detection and correction, flow control, and congestion avoidance.
- **Session layer:** This layer establishes, maintains, and terminates connections between applications. It manages sessions between two communicating devices.
- **Presentation layer:** This layer provides a common language for applications to communicate with each other. It deals with issues such as data compression, encryption, and conversion.
- **Application layer:** This layer provides services and interfaces for applications to access network resources. It includes protocols for email, file transfer, and remote access.

4 ETHERNET

Ethernet is the most widely used LAN technology, which is defined under IEEE standards 802.3. The reason behind its wide usability is Ethernet is easy to understand, implement, maintain, and allows low-cost network implementation.

Ethernet operates in two layers of the OSI model, Physical Layer, and Data Link Layer.

Types of Ethernet

- Standard Ethernet (IEEE 802.3)

- Fast Ethernet (IEEE 802.3u)
- Gigabit Ethernet (IEEE 802.3ab)
- 10 Gigabit Ethernet (IEEE 802.3ae)
- 40 Gigabit Ethernet (IEEE 802.3ba)
- 100 Gigabit Ethernet (IEEE 802.3bj)
- 400 Gigabit Ethernet (IEEE 802.3bs)

Ethernet (IEEE 802.3) Frame Format:

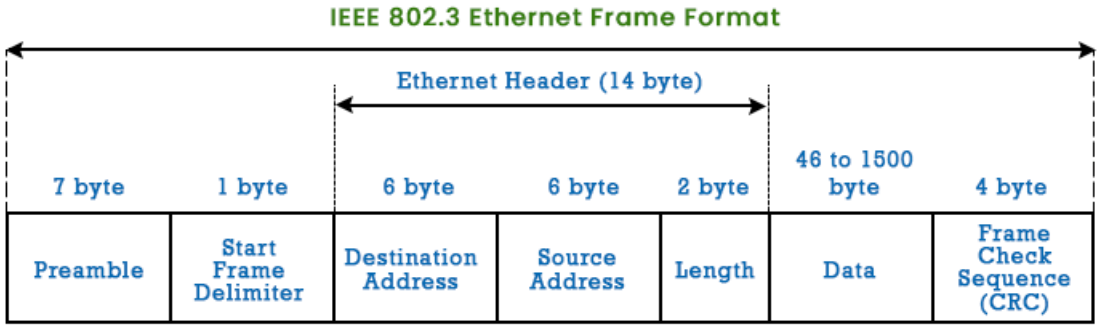

Figure 5: **Ethernet (IEEE 802.3) Frame Format**

- PREAMBLE : Ethernet frame starts with a 7-Bytes Preamble. This is a pattern of alternative 0's and 1's which indicates starting of the frame and allow sender and receiver to establish bit synchronization. Initially, PRE (Preamble) was introduced to allow for the loss of a few bits due to signal delays. But today's high-speed Ethernet doesn't need Preamble to protect the frame bits. PRE (Preamble) indicates the receiver that frame is coming and allow the receiver to lock onto the data stream before the actual frame begins.
- Start of frame delimiter (SFD) : This is a 1-Byte field that is always set to 10101011. SFD indicates that upcoming bits are starting the frame, which is the destination address. Sometimes SFD is considered part of PRE, this is the reason Preamble is described as 8 Bytes in many places. The SFD warns station or stations that this is the last chance for synchronization.
- Destination Address : This is a 6-Byte field that contains the MAC address of the machine for which data is destined. Source Address – This is a 6-Byte field that contains the MAC address of the source machine. As Source Address is always an individual address (Unicast), the least significant bit of the first byte is always 0.

- Length : Length is a 2-Byte field, which indicates the length of the entire Ethernet frame. This 16-bit field can hold a length value between 0 to 65534, but length cannot be larger than 1500 Bytes because of some own limitations of Ethernet.
- Data : This is the place where actual data is inserted, also known as Payload. Both IP header and data will be inserted here if Internet Protocol is used over Ethernet. The maximum data present may be as long as 1500 Bytes. In case data length is less than minimum length i.e. 46 bytes, then padding 0's is added to meet the minimum possible length.
- Cyclic Redundancy Check (CRC) : CRC is 4 Byte field. This field contains a 32-bits hash code of data, which is generated over the Destination Address, Source Address, Length, and Data field. If the checksum computed by destination is not the same as sent checksum value, data received is corrupted.

5 Modbus Protocol

Modbus is an industrial protocol that was developed in 1979 to make communication possible between automation devices. Originally implemented as an application-level protocol intended to transfer data over a serial layer, Modbus has expanded to include implementations over serial, TCP/IP, and the user datagram protocol (UDP).

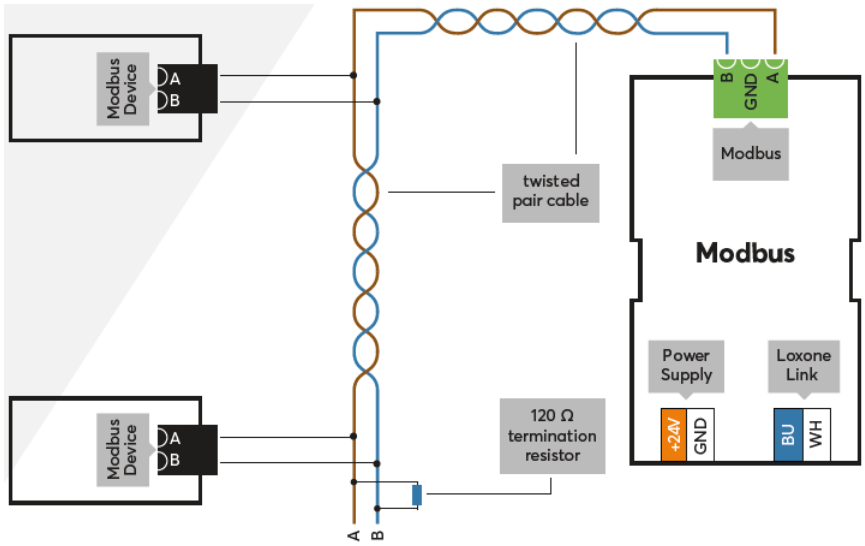

Figure 6: **Modbus Block Diagram**

Modbus Frame Format:

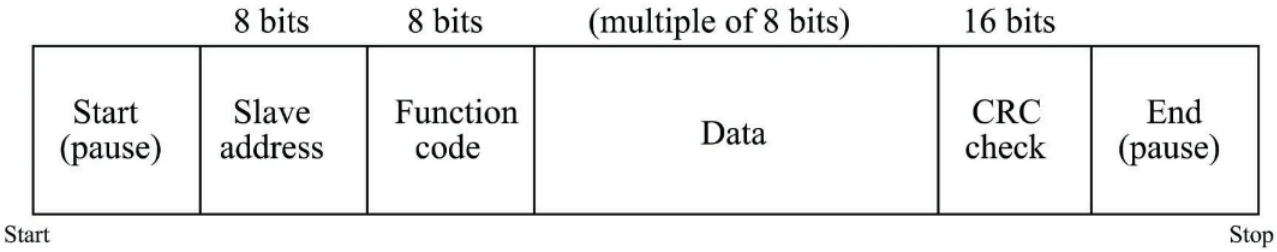

Figure 7: **Modbus RTU Frame Format**

- **Address field:** The address field is used to identify the slave device that the message is intended for. It is one byte long and can take a value between 1 and 247.
- **Function code:** The function code field indicates the type of Modbus command being sent. It is one byte long and can take a value between 1 and 127.

  Example:

  1. Read Coils (Function Code 01)
  2. Read Holding Registers (Function Code 03)
  3. Write Single Coil (Function Code 05)
  4. Write Single Register (Function Code 06)
  5. Read Device Identification (Function Code 43)
- **Data field:** The data field contains the actual data being transmitted. It can be up to 252 bytes long, depending on the function code being used.
- **CRC field:** The CRC (Cyclic Redundancy Check) field is used to ensure the integrity of the message. It is two bytes long and is calculated by performing a CRC calculation on the address, function code, and data fields.
- **Frame gap:** The frame gap is a period of silence between successive messages. It is at least 3.5 character times long.

6 RS-485 Physical Layer

RS-485 is a standard for serial communication over long distances, up to 1200 meters or more, at data rates of up to 10 Mbps. It is commonly used in industrial and building automation systems, where multiple devices need to communicate over long distances with high reliability and noise immunity.

RS-485 uses differential signaling, which means that it transmits data using two wires that carry inverted signals. This makes it less susceptible to noise and interference, as any noise that affects one wire is likely to affect the other wire in the opposite way, canceling out the noise.

RS-485 also allows multiple devices to communicate over the same bus, using a master/slave architecture. The master device initiates communication and controls the timing of the data exchange, while the slave devices respond to commands from the master.

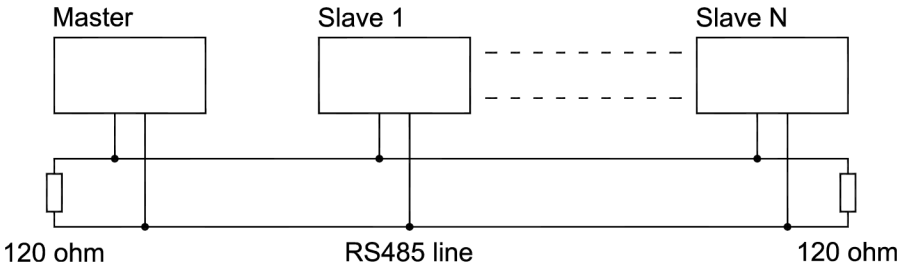

Figure 8: **RS-485 Block Diagram**

7 Wire Shark

Wireshark is a open source network packet analyzer. A network packet analyzer presents captured packet data in as much detail as possible.

You could think of a network packet analyzer as a measuring device for examining what's happening inside a network cable, just like an electrician uses a voltmeter for examining what's happening inside an electric cable (but at a higher level, of course).

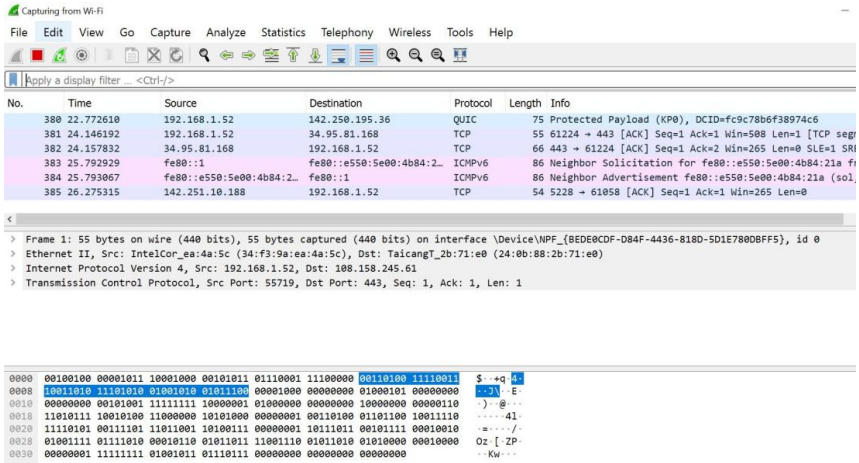

Figure 9: **Wire Shark**

References

1. https://www.techtarget.com
2. https://www.geeksforgeeks.org
3. https://www.youtube.com
4. https://www.akcp.com
5. https://www.imperva.com
6. https://www.geeksforgeeks.org
7. https://www.geeksforgeeks.org
8. https://www.se.com
9. https://www.ni.com
10. https://www.loxone.com
11. https://www.cuidevices.com