

Lab 2: Capture and Analyze IoT Device Traffic

Lab Scenario

As a professional ethical hacker or pen tester, you must have sound knowledge to capture and analyze the traffic between IoT devices. Using various tools and techniques, you can capture the valuable data flowing between the IoT devices, analyze it to obtain information on the communication protocol used by the IoT devices, and acquire sensitive information such as credentials, device identification numbers, etc.

Lab Objectives

- Capture and analyze IoT traffic using Wireshark

Overview of IoT and OT Traffic

Many IoT devices such as security cameras host websites for controlling or configuring cameras from remote locations. These websites mostly implement the insecure HTTP protocol instead of the secure HTTPS protocol and are, hence, vulnerable to various attacks. If the cameras use the default factory credentials, an attacker can easily intercept all the traffic flowing between the camera and web applications and further gain access to the camera itself. Attackers can use tools such as Wireshark to intercept such traffic and decrypt the Wi-Fi keys of the target network.

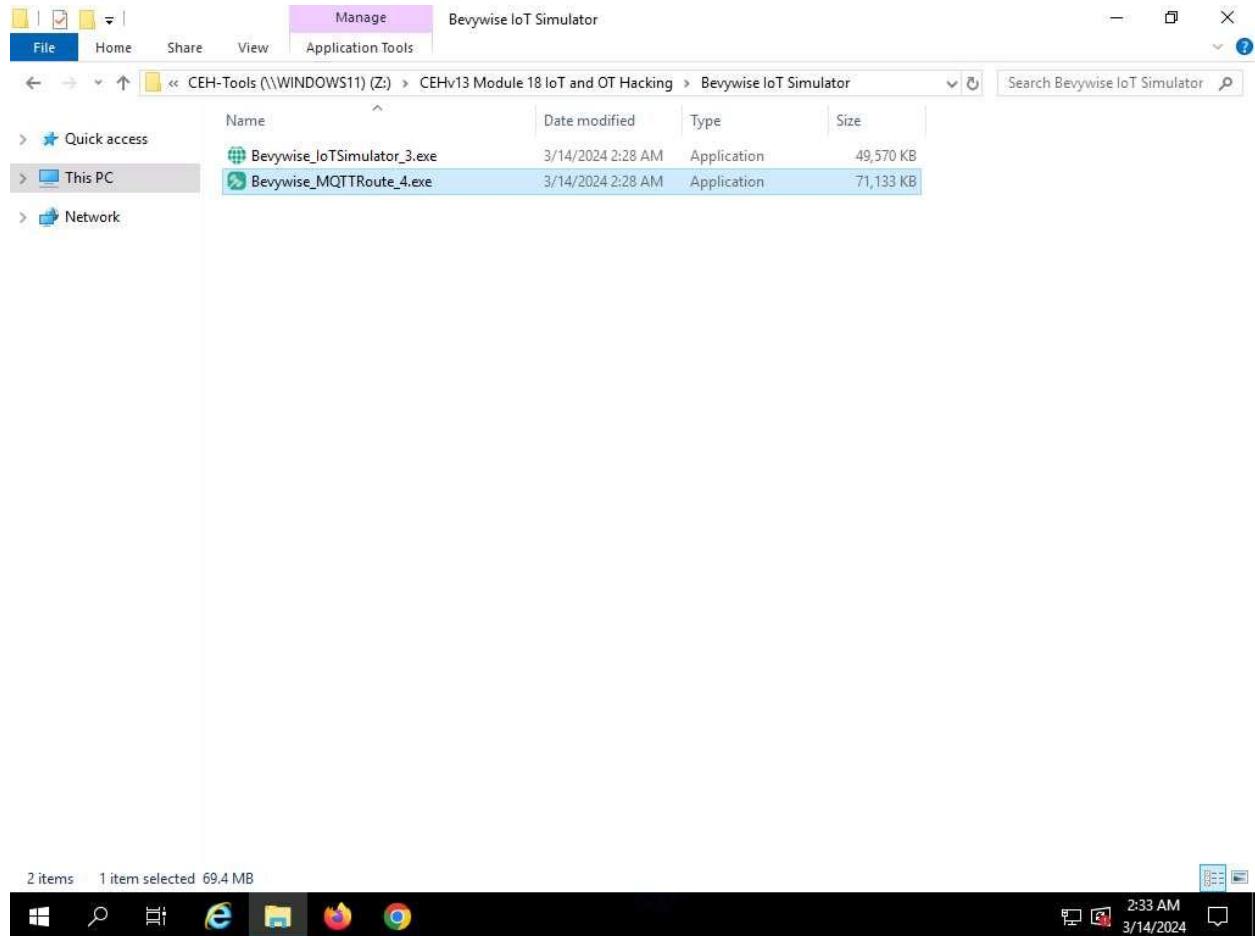
Task 1: Capture and Analyze IoT Traffic using Wireshark

Wireshark is a free and open-source packet analyzer. It facilitates network troubleshooting, analysis, software and communications protocol development, and education. It is used to identify the target OS and sniff/capture the response generated from the target machine to the machine from which a request originates.

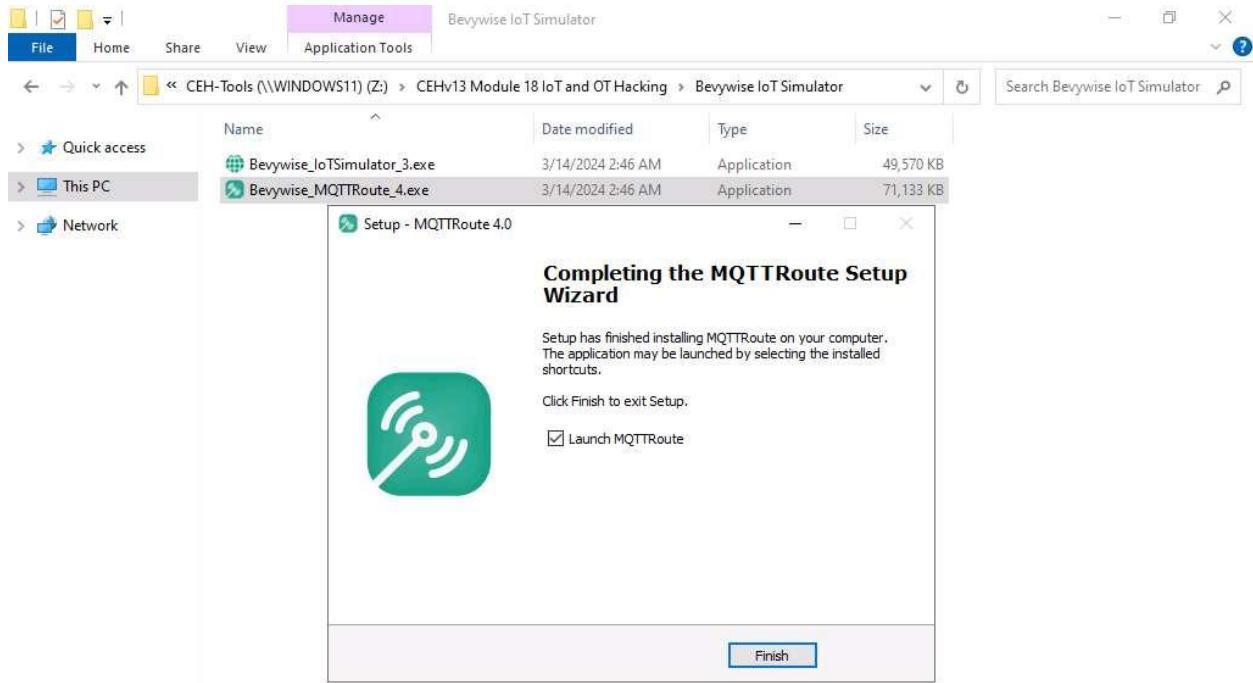
MQTT is a lightweight messaging protocol that uses a publish/subscribe communication pattern. Since the protocol is meant for devices with a low-bandwidth, it is considered ideal for machine-to-machine (M2M) communication or IoT applications. We can create virtual IoT devices over the virtual network using the Bevywise IoT simulator on the client side and communicate these devices to the server using the MQTT Broker web interface. This interface collects data and displays the status and messages of connected devices over the network.

Here, we use Wireshark to capture and analyze traffic between IoT devices.

1. To install the **MQTT Broker** on the **Windows Server 2019**, click [Windows Server 2019](#) to launch **Windows Server 2019** machine. Click [Ctrl+Alt+Delete](#) and login with **Administrator/Pa\$\$w0rd**.
2. Navigate to **Z:\CEHv13 Module 18 IoT and OT Hacking\Bevywise IoT Simulator** folder and double-click on the **Bevywise_MQTTRoute_4.exe** file.



3. If **Open File - Security Warning** popup appears, click **Run**.
4. The **Setup - MQTTRoute 4.0** window opens. Select **I accept the agreement** and click on **Next**. Follow the wizard driven steps to install the tool.
5. After the installation completes, click on **Finish**. Ensure that **Launch MQTTRoute** is checked.



6. The MQTTRoute will execute and the command prompt will appear. You can see the **TCP** port using **1883**.

```
[MQTTROUTE]14:03:2024 02:51:49 - Bevywise MQTTRoute 4.0 - build 1122-001
[MQTTROUTE]14:03:2024 02:51:50 - Bevywise HTTP UI WebSocket - Listening on port 8081 for clients..
This software is licensed to 'Bevywise Networks'. This Software is for the sole use of the Licensee and the Licensee cannot rent, lease, sell, assign, or transfer rights to the Software. This software is protected by copyright laws.

Licensed Clients:10
Licensed for any MAC address

[MQTTROUTE]14:03:2024 02:51:50 - You are using - MQTTRoute - DEVELOPER version
[MQTTROUTE]14:03:2024 02:51:50 - You can connect upto 10 clients
[MQTTROUTE]14:03:2024 02:51:50 - TCP Port - 1883           WebSocket Port - 10443
[MQTTROUTE]14:03:2024 02:51:50 - View your connected devices via your browser at - http://localhost:8080 or http://<machine-ip>:8080
[MQTTROUTE]14:03:2024 02:51:50 - Authentication Disabled
[MQTTROUTE]14:03:2024 02:51:50 - Remote Authentication Disabled
```

7. We have installed MQTT Broker successfully and leave the Bevywise MQTT **running**.

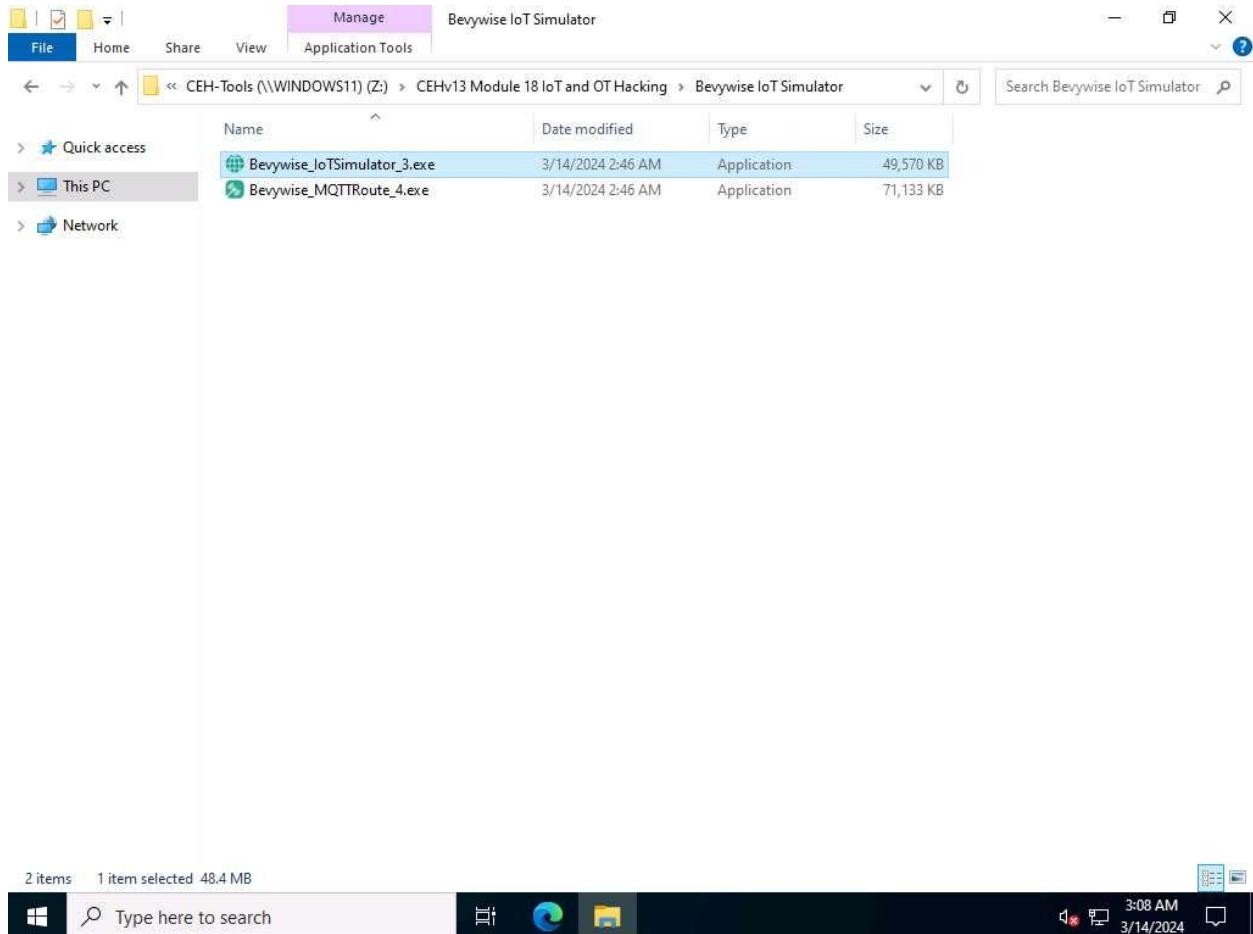
8. To create IoT devices, we must install the **IoT simulator** on the client machine.

9. Click [Windows Server 2022](#) to switch to **Windows Server 2022** machine.

Click [Ctrl+Alt+Delete](#) and login with **Administrator/Pa\$\$w0rd**.

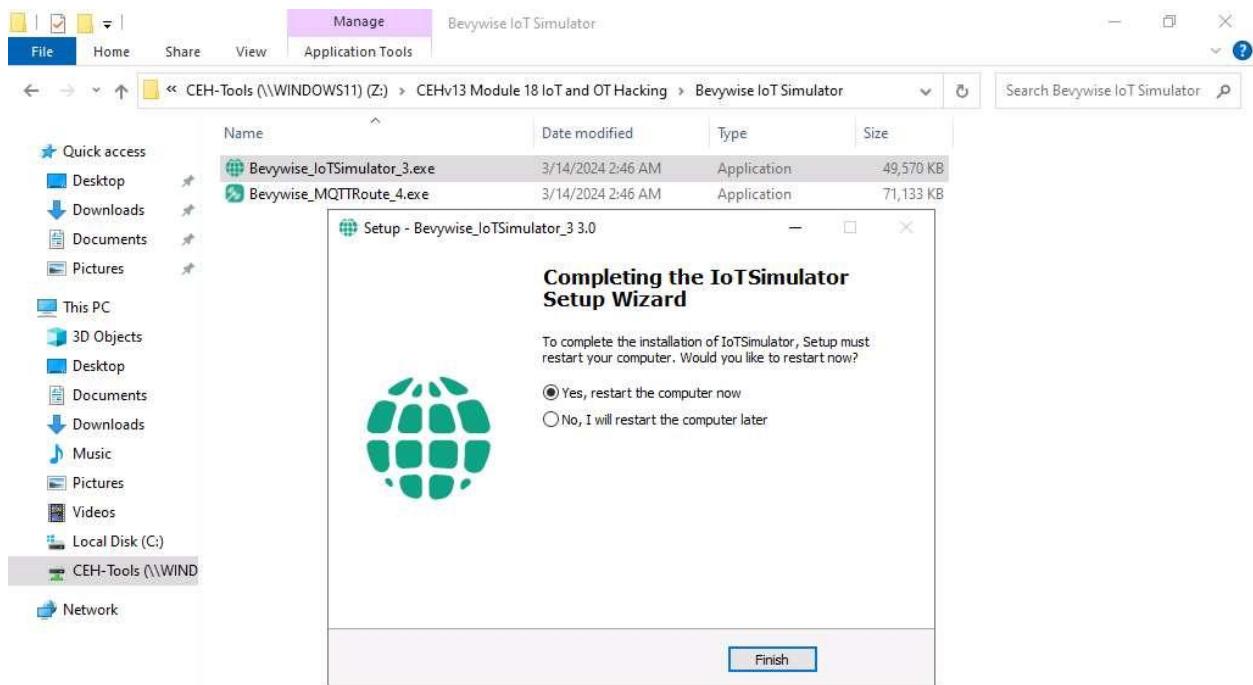
If the network screen appears, click **Yes**.

10. Navigate to **Z:\CEHv13 Module 18 IoT and OT Hacking\Bevywise IoT Simulator** folder and double-click on the **Bevywise_IoTSimulator_3.exe** file.

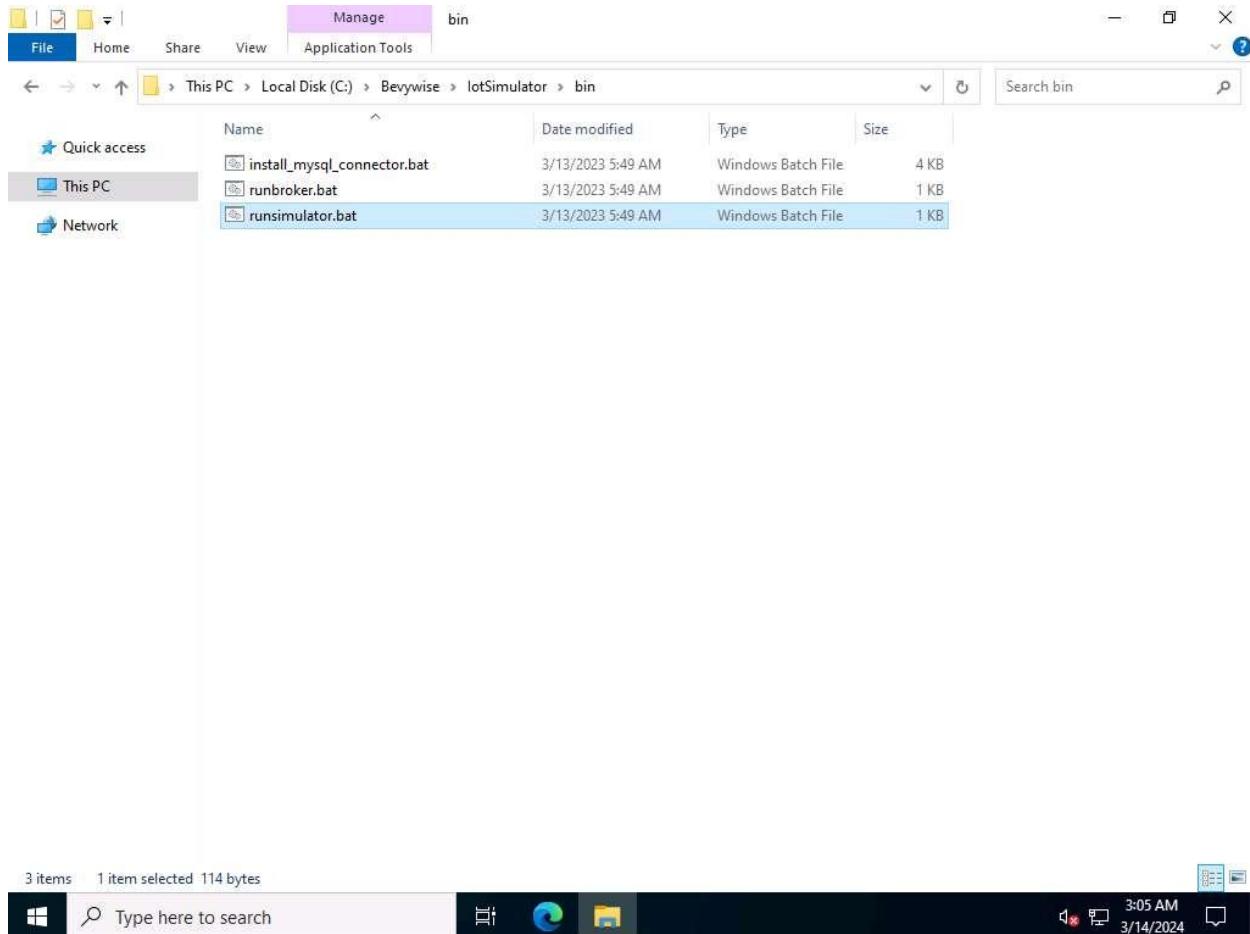


11. If **Open File - Security Warning** popup appears, click **Run..**
12. The **Setup-IoTSimulator_3 3.0** setup wizard opens. Select **I accept the agreement** and follow the wizard driven steps.
13. To complete the installation, select **Yes, restart the computer now** and click on **Finish** to complete the installation.

If restart computer option does not appear, then continue from **Step#16**.



14. After restarting, Bevywise IoT Simulator is installed successfully. To launch the **IoT simulator**, navigate to the **C:\Bevywise\IoTSimulator\bin** directory and double-click on the **runsimulator.bat** file.



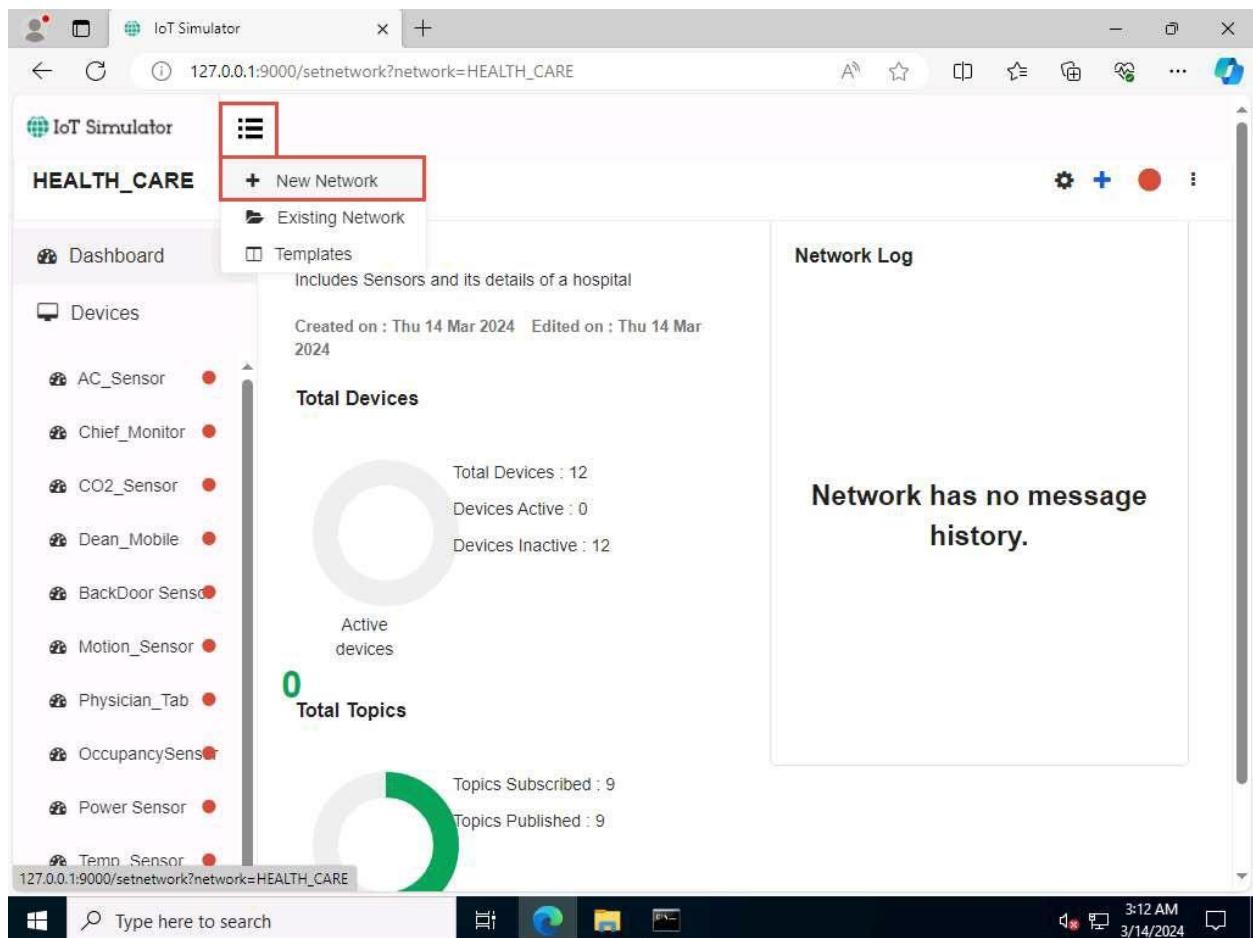
15. Upon double-clicking the **runsimulator.bat** file opens in the command prompt. If **How do you want to open this?** pop-up appears, select **Microsoft Edge** browser and click **OK** to open the URL http://127.0.0.1:9000/setnetwork?network=HEALTH_CARE.

If the URL directly opens in Microsoft Edge browser, then continue.

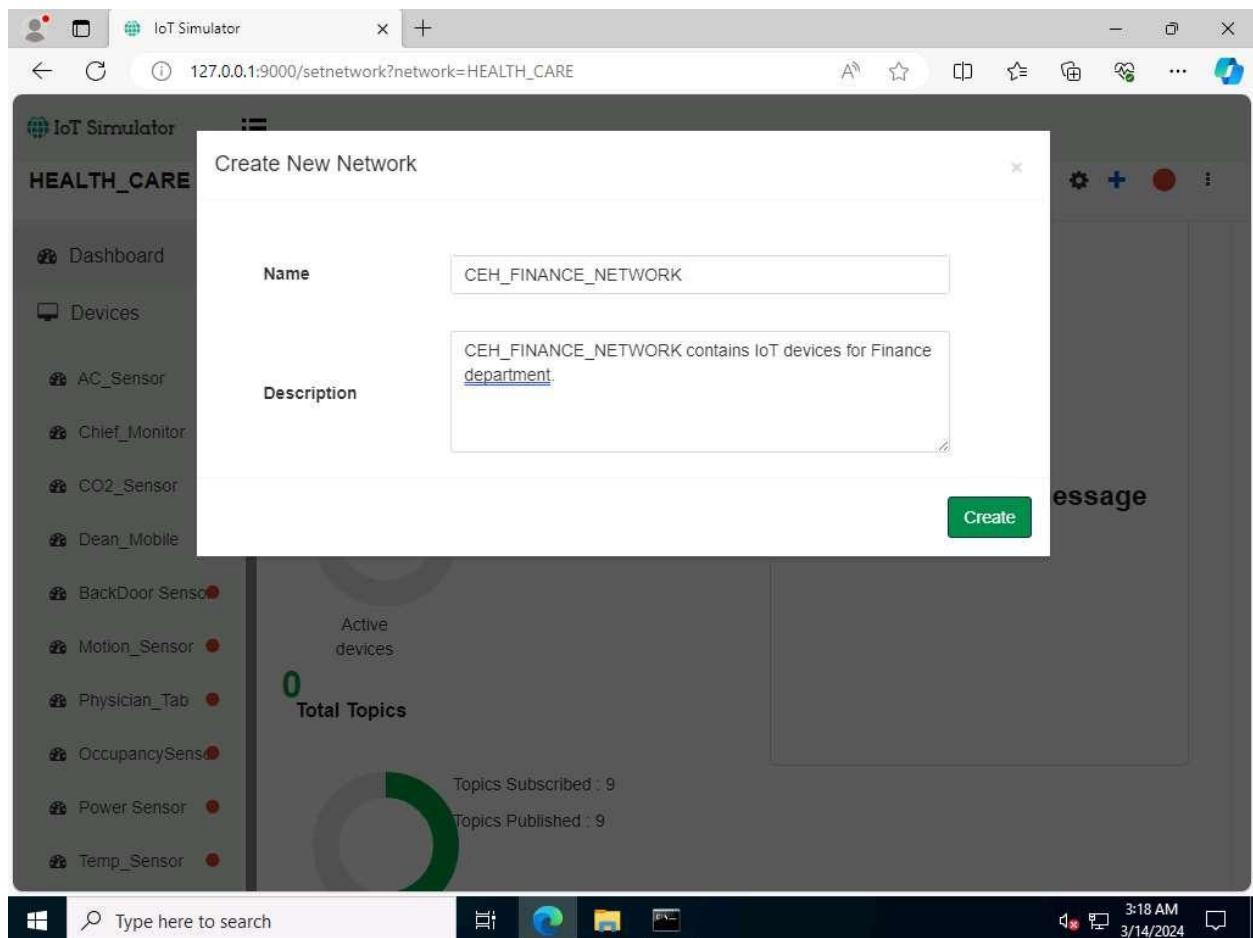
16. The web interface of the IoT Simulator opens in Edge browser. In the IoT Simulator, you can view the default network named **HEALTH_CARE** and several devices.

The screenshot shows a web browser window titled "IoT Simulator" with the URL "127.0.0.1:9000/setnetwork?network=HEALTH_CARE". The main content area displays a "HEALTH_CARE" network dashboard. On the left, a sidebar lists "Devices" and their status: AC_Sensor (red dot), Chief_Monitor (red dot), CO2_Sensor (red dot), Dean_Mobile (red dot), BackDoor Sensor (red dot), Motion_Sensor (red dot), Physician_Tab (red dot), OccupancySensor (red dot), Power Sensor (red dot), and Temp_Sensor (red dot). The main panel has two sections: "Description" (including a brief description of sensors and details of a hospital, creation date, and edit date) and "Network Log" (which states "Network has no message history"). Below these are two circular progress indicators: one for "Total Devices" (12 total, 0 active, 12 inactive) and one for "Total Topics" (0 topics subscribed, 9 topics published).

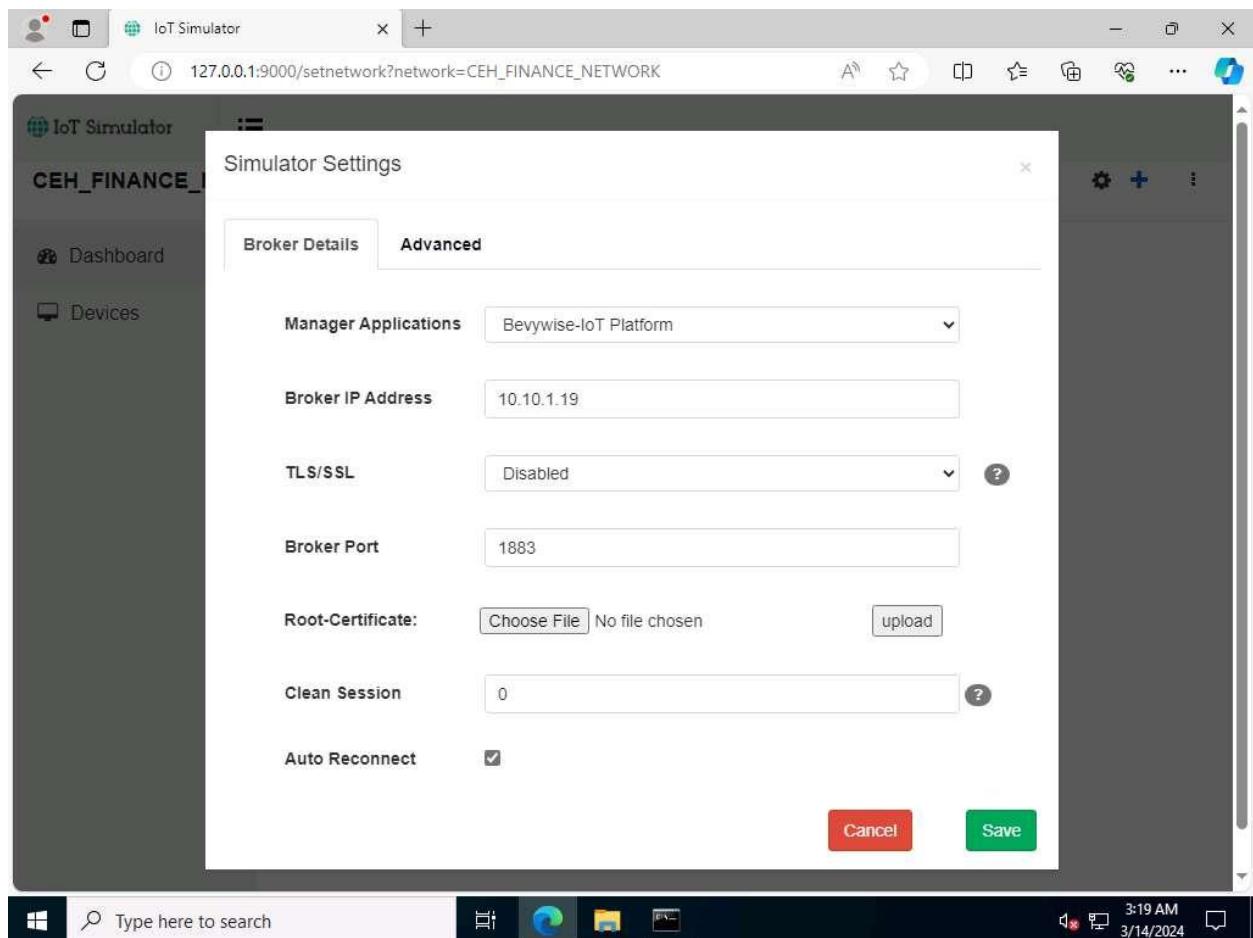
17. Next, we will create a **virtual IoT network** and **virtual IoT devices**. Click on the **menu** icon and select the **+New Network** option.



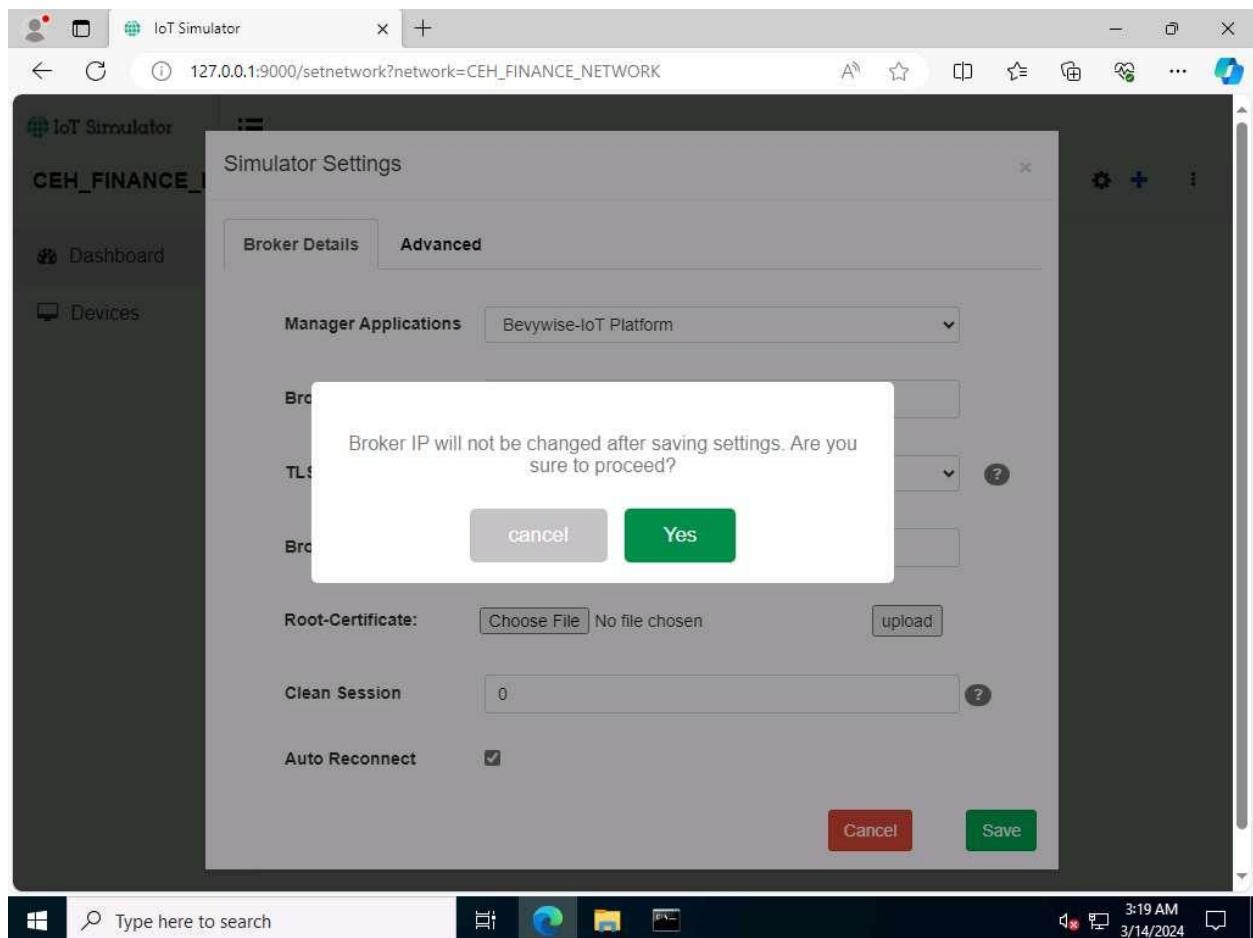
18. The **Create New Network** popup appears. Type any name (here, **CEH_FINANCE_NETWORK**) and description. Click on **Create**.



19. In the next screen, we will setup the **Simulator Settings**. Set the **Broker IP Address** as **10.10.1.19** (the IP address of the **Windows Server 2019**). Since we have installed the Broker on the web server, the created network will interact with the server using MQTT Broker. Do not change default settings and click on **Save**.

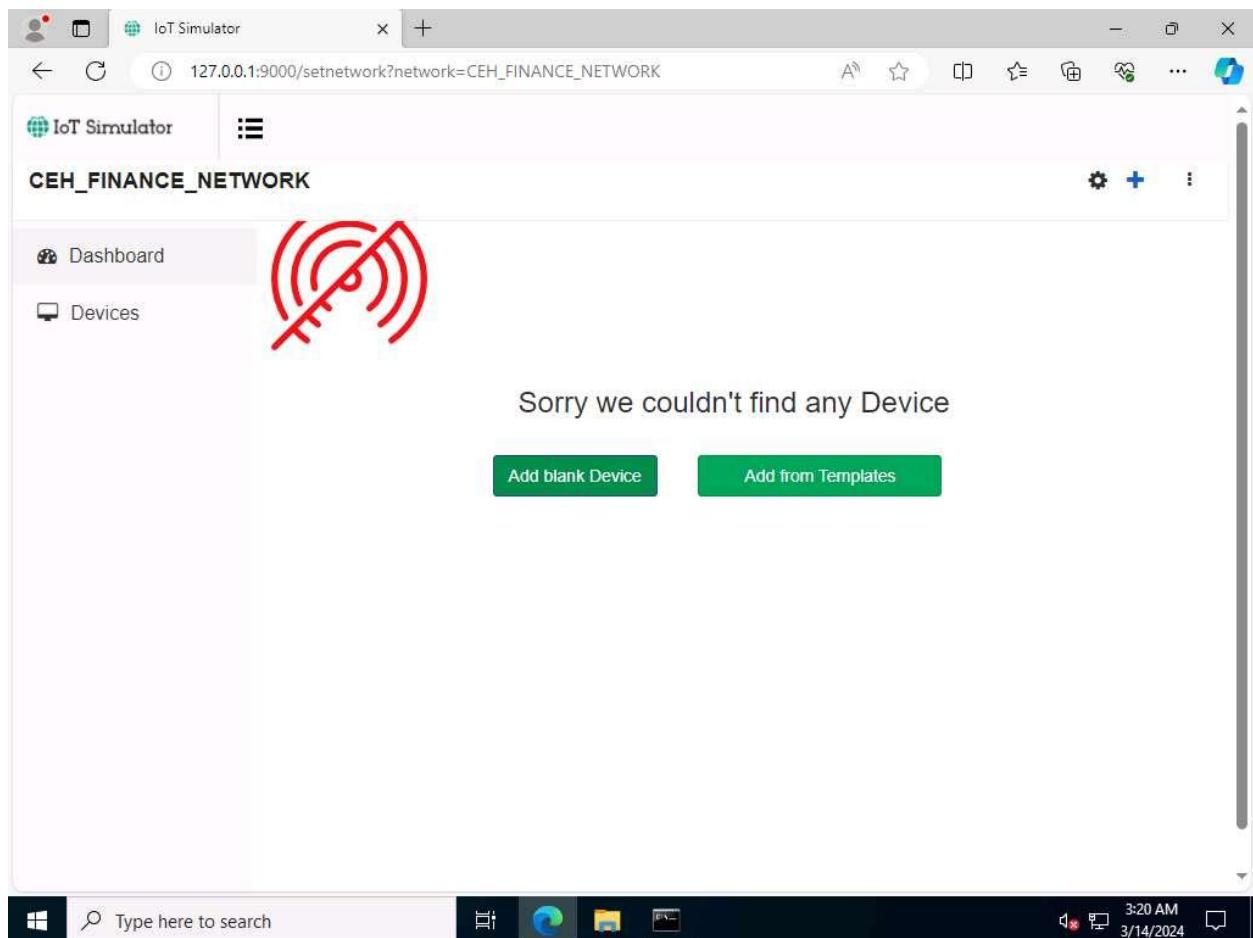


20. To proceed with the network creation, click on **Yes**.

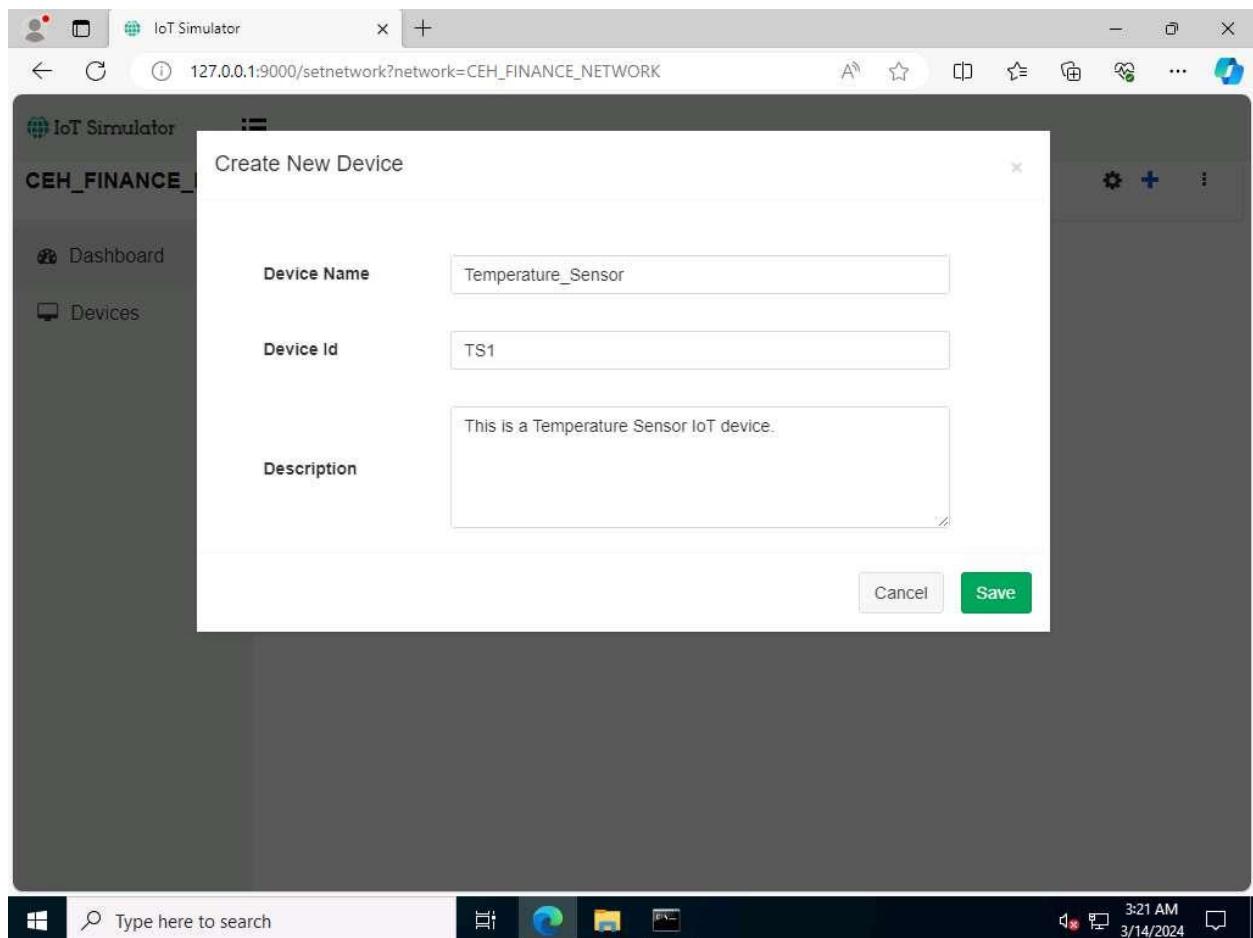


If **Configuration Saved** pop-up appears. Click on **OK** to continue. This step completes the creation of the virtual IoT network.

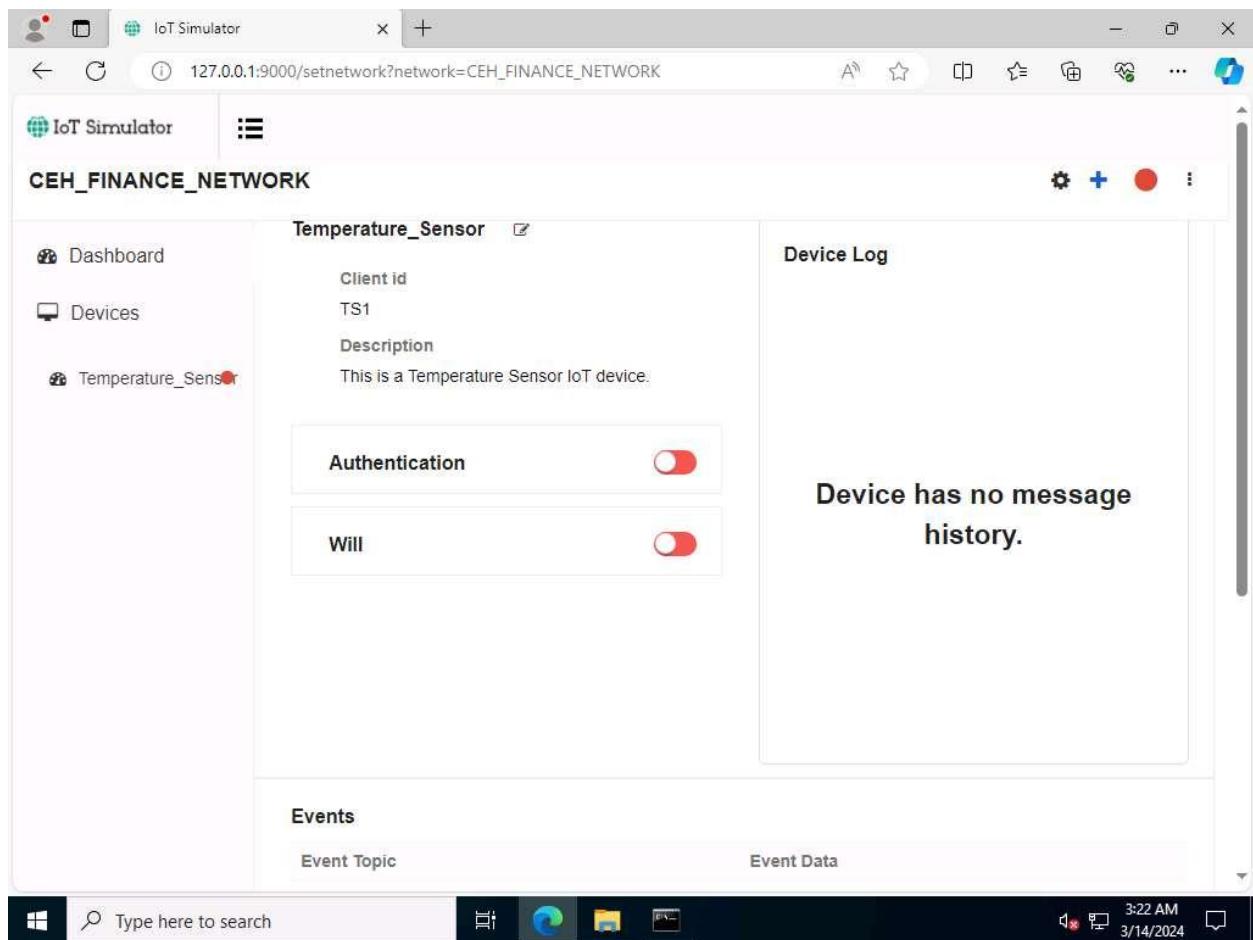
21. To add IoT devices to the created network, click on the **Add blank Device** button.



22. The **Create New Device** popup opens. Type the device name (here, we use **Temperature_Sensor**), enter Device Id (here, we use **TS1**), provide a **Description** and click on **Save**.



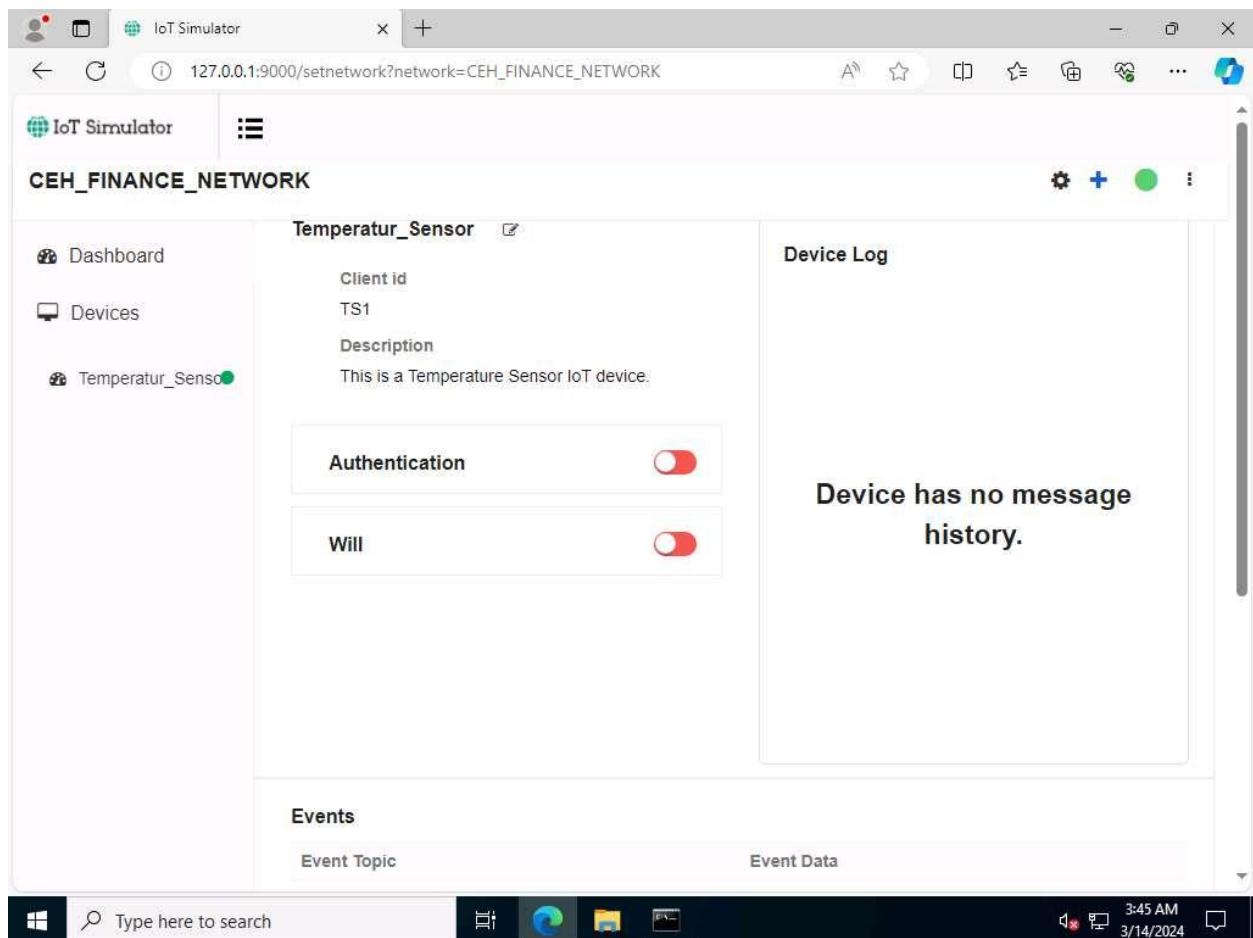
23. The device will be added to the **CEH_FINANCE_NETWORK**.



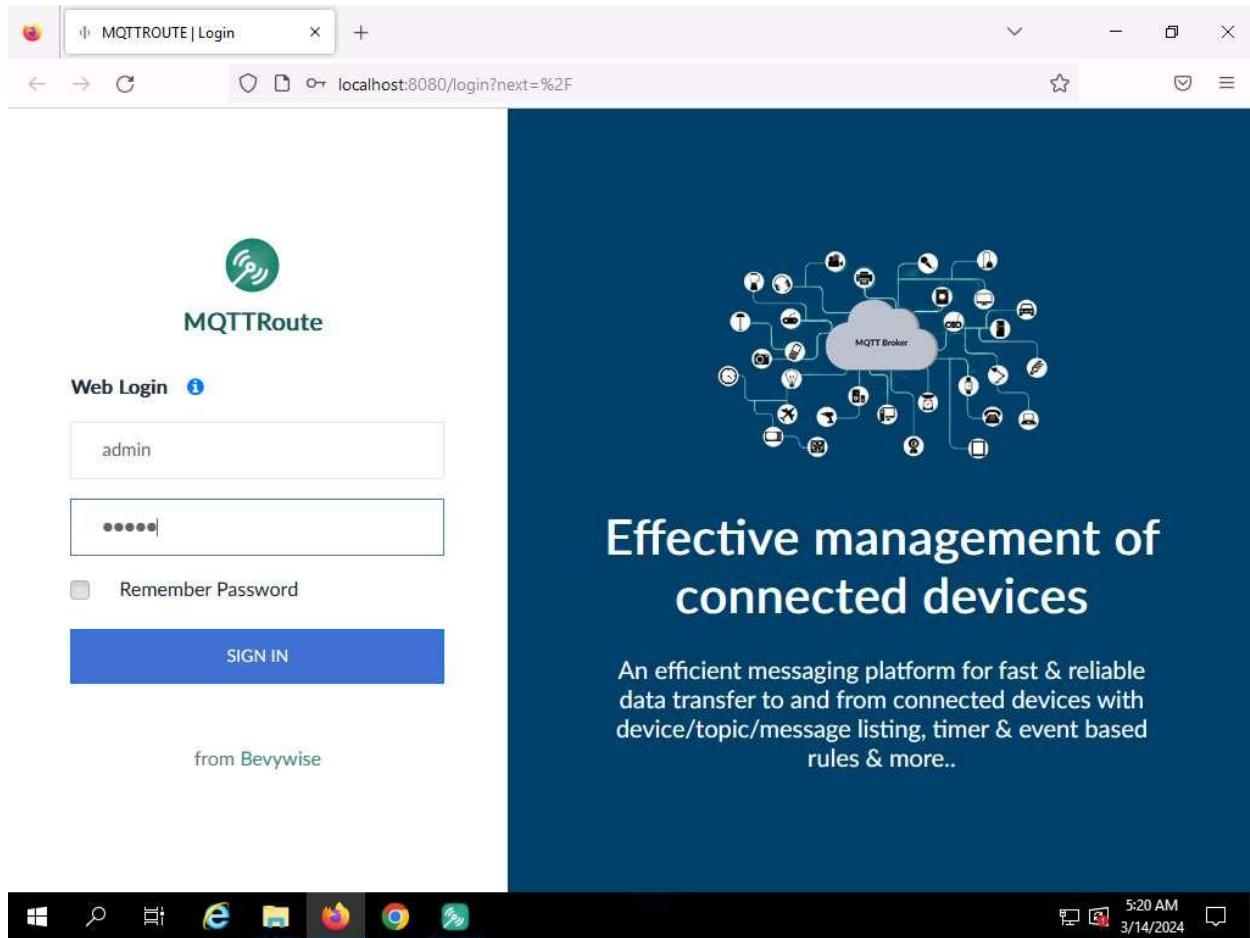
24. To connect the Network and the added devices to the server or Broker, click on the **Start Network** red color circular icon in right corner.

The screenshot shows a web browser window titled "IoT Simulator" with the URL "127.0.0.1:9000/setnetwork?network=CEH_FINANCE_NETWORK". The main content area displays a device configuration for a "Temperature_Sensor". On the left sidebar, there are links for "Dashboard", "Devices", and "Temperature_Sensor". The main panel has a title "Temperature_Sensor" with a close button. It contains fields for "Client id" (set to "TS1") and "Description" (set to "This is a Temperature Sensor IoT device"). Below these are two toggle switches: "Authentication" (on) and "Will" (on). To the right, a "Device Log" section displays the message "Device has no message history." At the bottom of the page is a Windows taskbar with a search bar, pinned icons for File Explorer, Edge, and File History, and a system tray showing the date and time.

25. When a connection is established between the network and the added devices and the web server or the MQTT Broker, the red button turns into green.



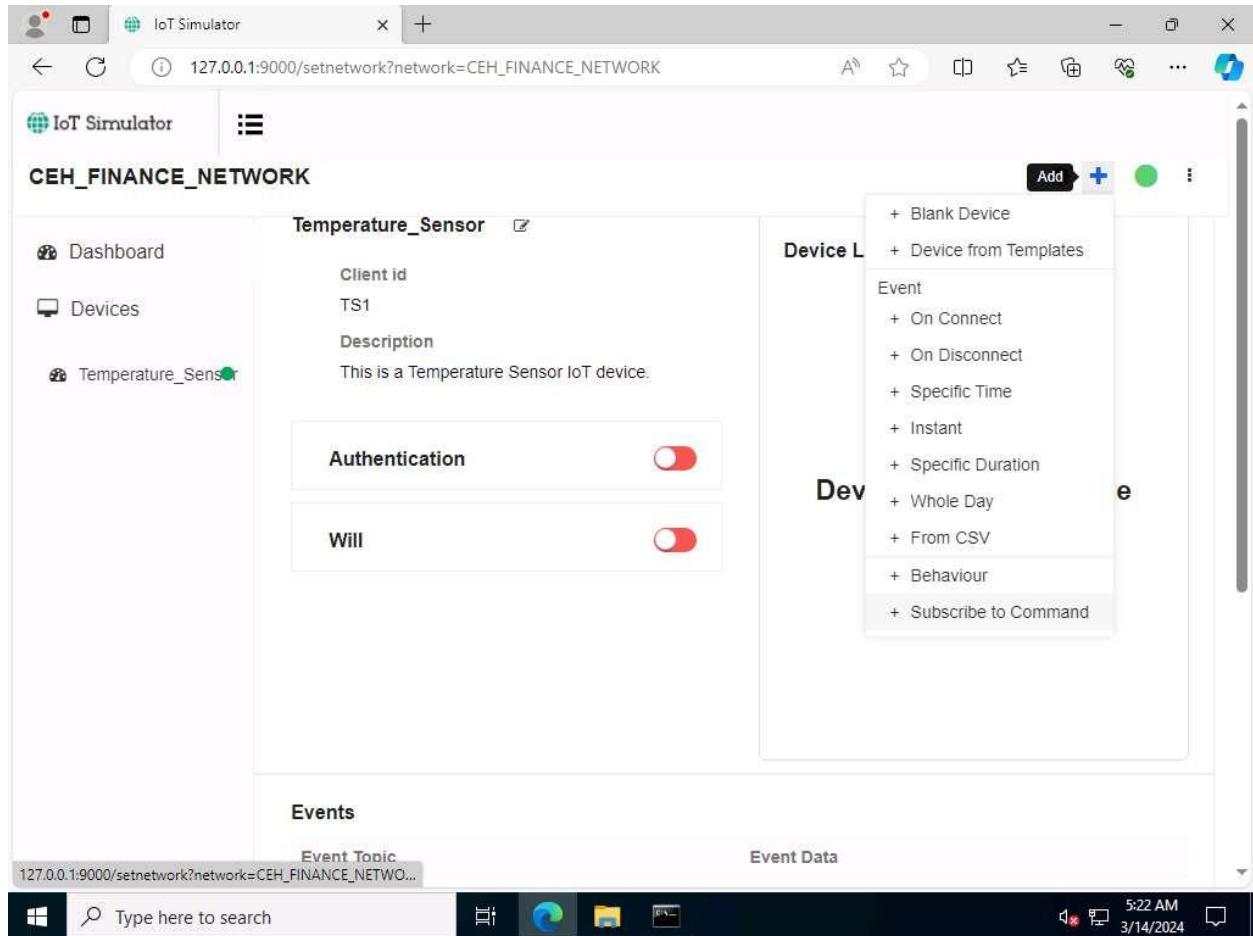
26. Next, switch to the [Windows Server 2019](#) machine. Open a web browser, and go to <http://localhost:8080> and login using **admin/admin** (here, we are using **Firefox** Browser).



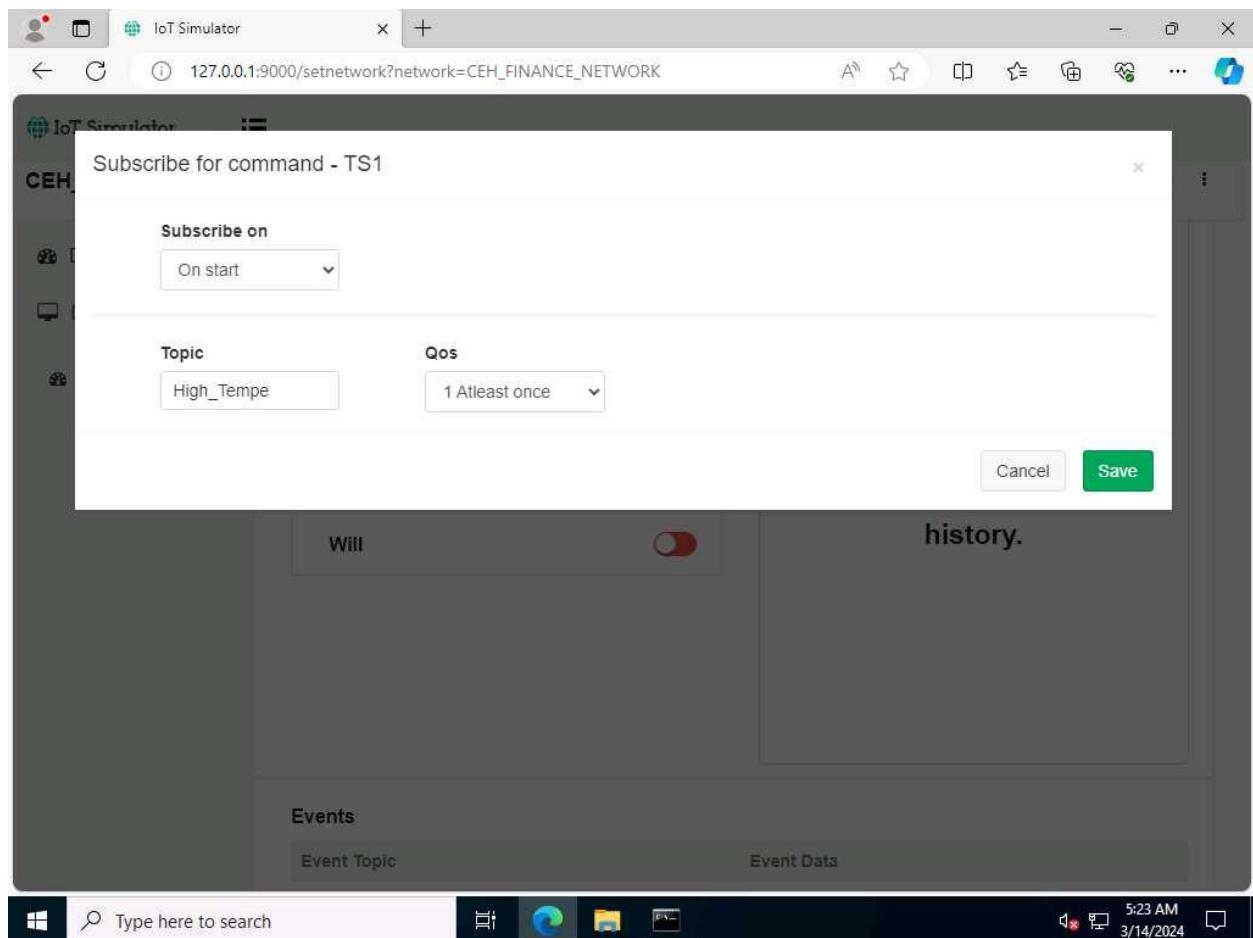
27. Since the Broker was **left running**, you can see a connection request from machine **10.10.1.22** for the device **TS1** under **Recent Connections** section.

The screenshot shows the MQTTRoute web application running on localhost:8080. The top navigation bar includes a refresh icon, a search bar with the text 'MQTTRoute', and a plus sign icon. The main dashboard features four summary cards: 'Active Devices' (1), 'Total Devices' (1), 'Events' (0), and 'Commands' (0). Below these are two sections: 'Recent Events' and 'Recent Device Log', both of which display 'No Data Found'. A third section, 'Recent Connections', shows one entry: 'TS1' connected from '10.10.1.22' at 'Today 05:15:32'. This connection entry is highlighted with a red border. To the right is a 'Recent Disconnections' section, which also shows 'No Data Found'. The bottom of the screen displays a Windows taskbar with icons for File Explorer, Task View, Edge, Firefox, Google Chrome, and FileZilla, along with system status icons and the date/time '5:21 AM 3/14/2024'.

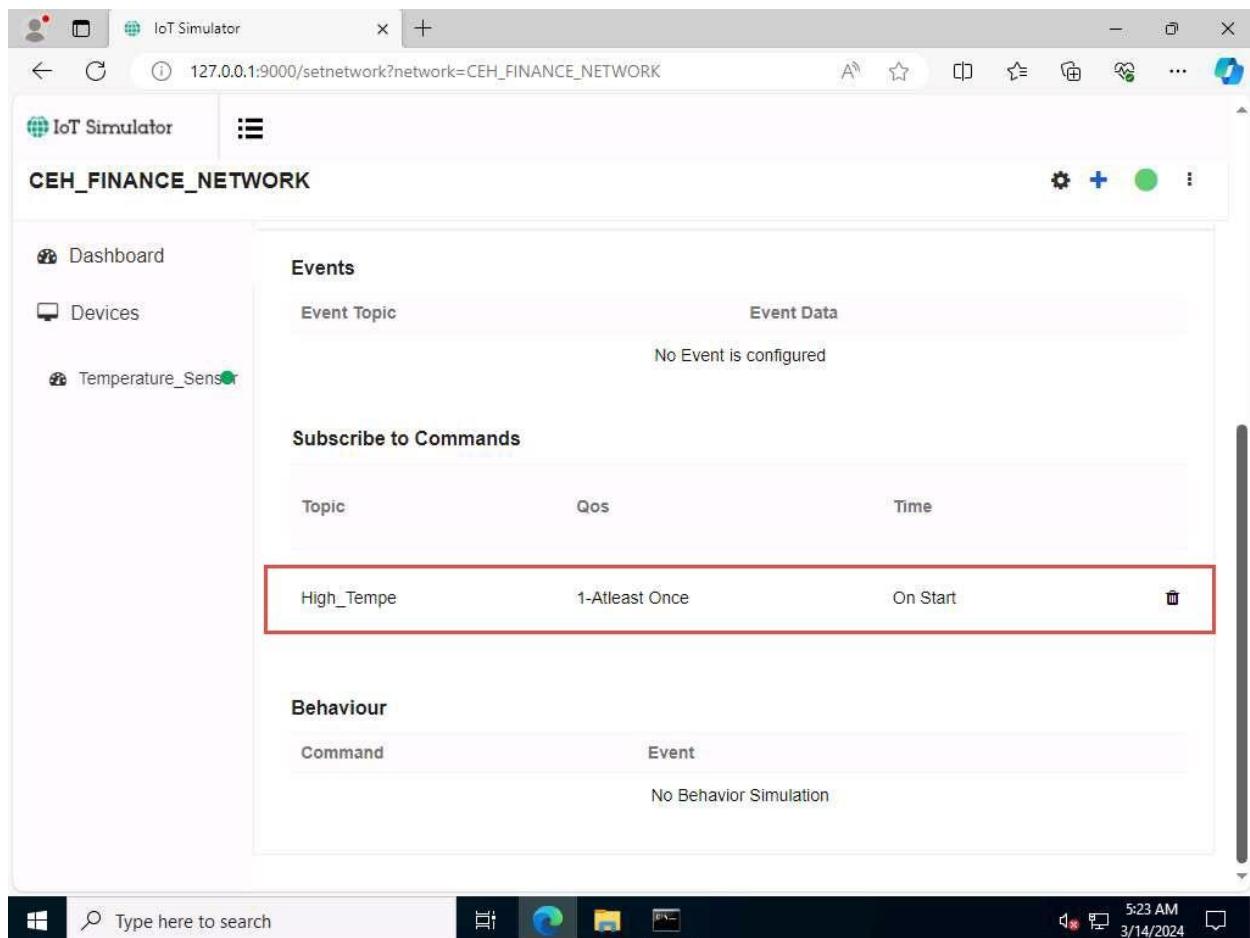
28. Switch back to [Windows Server 2022](#) machine.
29. Next, we will create the **Subscribe command** for the device Temperature_Sensor.
30. Click on the **Plus icon in the top right corner** and select the **Subscribe to Command** option.



31. The **Subscribe for command - TS1** popup opens. Select **On start** under the **Subscribe on tab**, type **High_Tempe** under the **Topic tab**, and select **1 Atleast once** below the **Qos** option. Click on **Save**.



32. Scroll down the page, you can see the **Topic** added under the **Subscribe to Commands** section.



33. Next, we will capture the traffic between the **virtual IoT network** and the **MQTT Broker** to monitor the secure communication.
34. Minimise the Edge browser. Click **Type here to search** field on the **Desktop**, search for **wireshark** in the search bar and select **Wireshark** from the results.
35. The Wireshark Application window appears, select the **Ethernet** as interface.

Make sure you have selected interface which has **10.10.1.22** as the IP address.

If Software update popup appears click on **Skip this version**.



Welcome to Wireshark

Capture

...using this filter: Enter a capture filter ...

All interfaces shown ▾

- Ethernet
- Adapter for loopback traffic capture
- Local Area Connection* 9
- Local Area Connection* 8
- Local Area Connection* 7
- Event Tracing for Windows (ETW) reader

Learn

[User's Guide](#) · [Wiki](#) · [Questions and Answers](#) · [Mailing Lists](#) · [SharkFest](#) · [Wireshark Discord](#) · [Donate](#)

You are running Wireshark 4.2.3 (v4.2.3-0-ga15d7331476c). You receive automatic updates.



36. Click on the **Start Wireshark** icon to start the capturing packets, leave the Wireshark running.
37. Leave the IoT simulator running and switch to the [Windows Server 2019](#) machine.
38. Navigate to **Devices** menu and click on connected device i.e.**TS1**.

The screenshot shows a web browser window titled "MQTTRoute" with the URL "localhost:8080/#page-single-device". The navigation bar includes links for "Beyywise", "Dashboard", "Devices" (which is highlighted with a red border), "Topics", "Rules", "Device Log", and a gear icon. Below the navigation bar is a table with columns: Device Name, Device Id, Status, Will Topic, Will Qos, Will Message, and Time. A single row is visible for device "TS1", with "Active" status and a timestamp of "Today 22:02:28".

Below the table is a section titled "Send Command" with tabs for "Events", "Commands", "Subscribe Topics", and "Send Command" (which is active). It contains two input fields: "Topic" containing "High_Tempe" and "Message" containing "Alert for High Temperature". A "Submit" button is located at the bottom right of this section.

39. Now, we will send the command to **TS1** using the **High_Tempe** topic.

40. In **Send Command** section, select **Topic** as **High_Tempe**, type **Alert for High Temperature** in **Message** field and click on the **Submit** button.

The screenshot shows a web browser window titled "MQTTRoute" with the URL "localhost:8080/#page-single-device". The main header includes the Bevywise logo, Dashboard, Devices, Topics, Rules, Device Log, and a gear icon. Below this is a table with columns: Device Name, Device Id, Status, Will Topic, Will Qos, Will Message, and Time. A single row is present for "TS1" with Device Id "TS1", Status "Active", Will Topic "High_Tempe", Will Message "Alert for High Temperature.", and Time "Today 22:02:28". Below the table is a navigation bar with tabs: Events, Commands, Subscribe Topics, and Send Command (which is selected). Under the "Send Command" tab, there are two input fields: "Topic" containing "High_Tempe" and "Message" containing "Alert for High Temperature.". A blue "Submit" button is located to the right of the message field. The taskbar at the bottom of the screen shows various application icons.

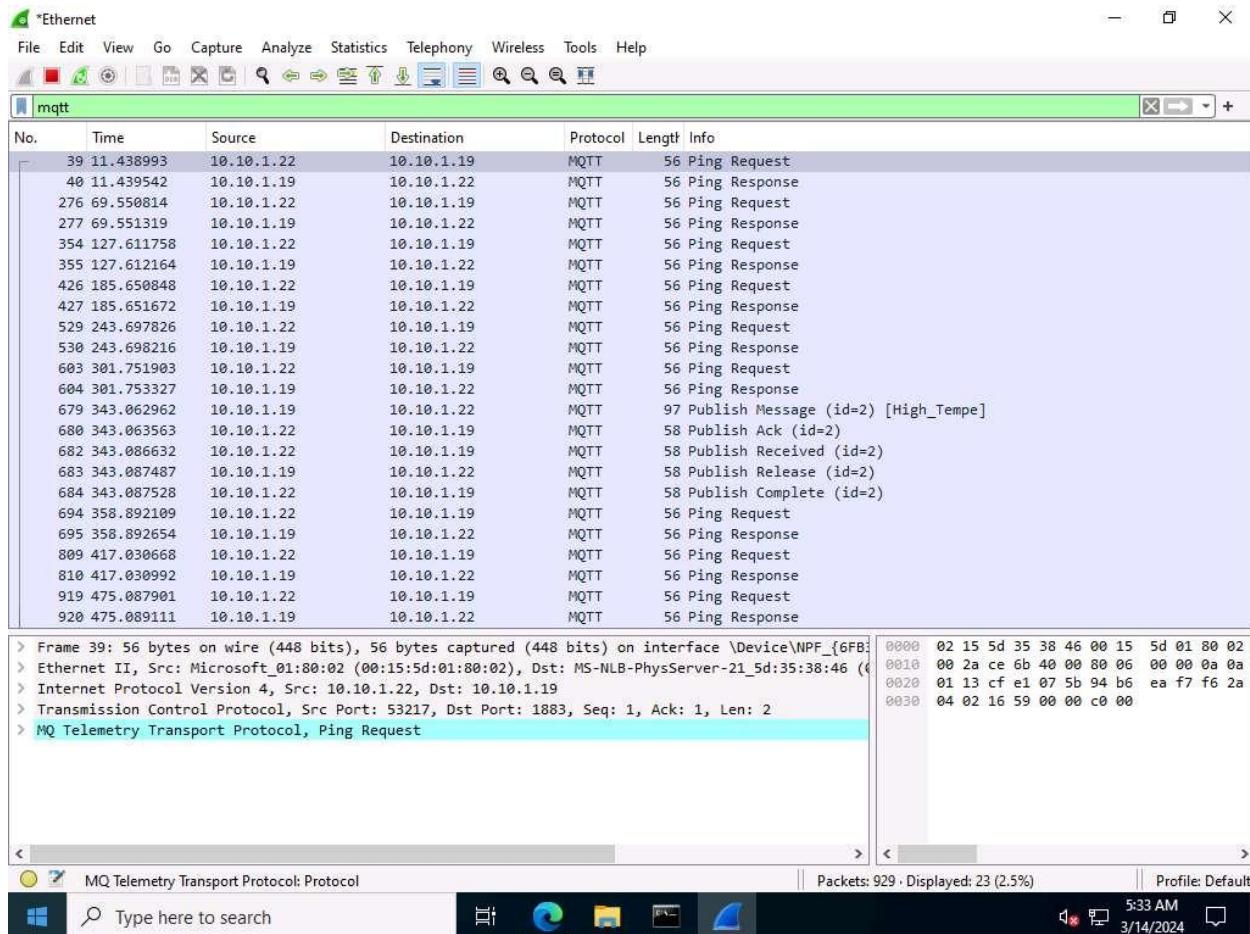
41. **Message sent to TS1** appears under **Message** box which indicates that the message was successfully sent to TS1.

The screenshot shows a web browser window titled "MQTTRoute" at the URL "localhost:8080/#page-single-device". The Bevywise interface has a dark header with tabs for Dashboard, Devices, Topics, Rules, Device Log, and Settings. Below the header is a table with columns: Device Name, Device Id, Status, Will Topic, Will Qos, Will Message, and Time. One row is shown for "TS1" with "TS1" in the Device Id column and "Active" in the Status column. The Time column shows "Today 22:02:28". Below the table is a navigation bar with tabs: Events, Commands, Subscribe Topics, and Send Command. The "Send Command" tab is active. Under "Topic", the value "High_Tempe" is entered. Under "Message", there is a large text input field. A red box highlights the message "Message send to TS1" in a feedback box, and a blue "Submit" button is visible.

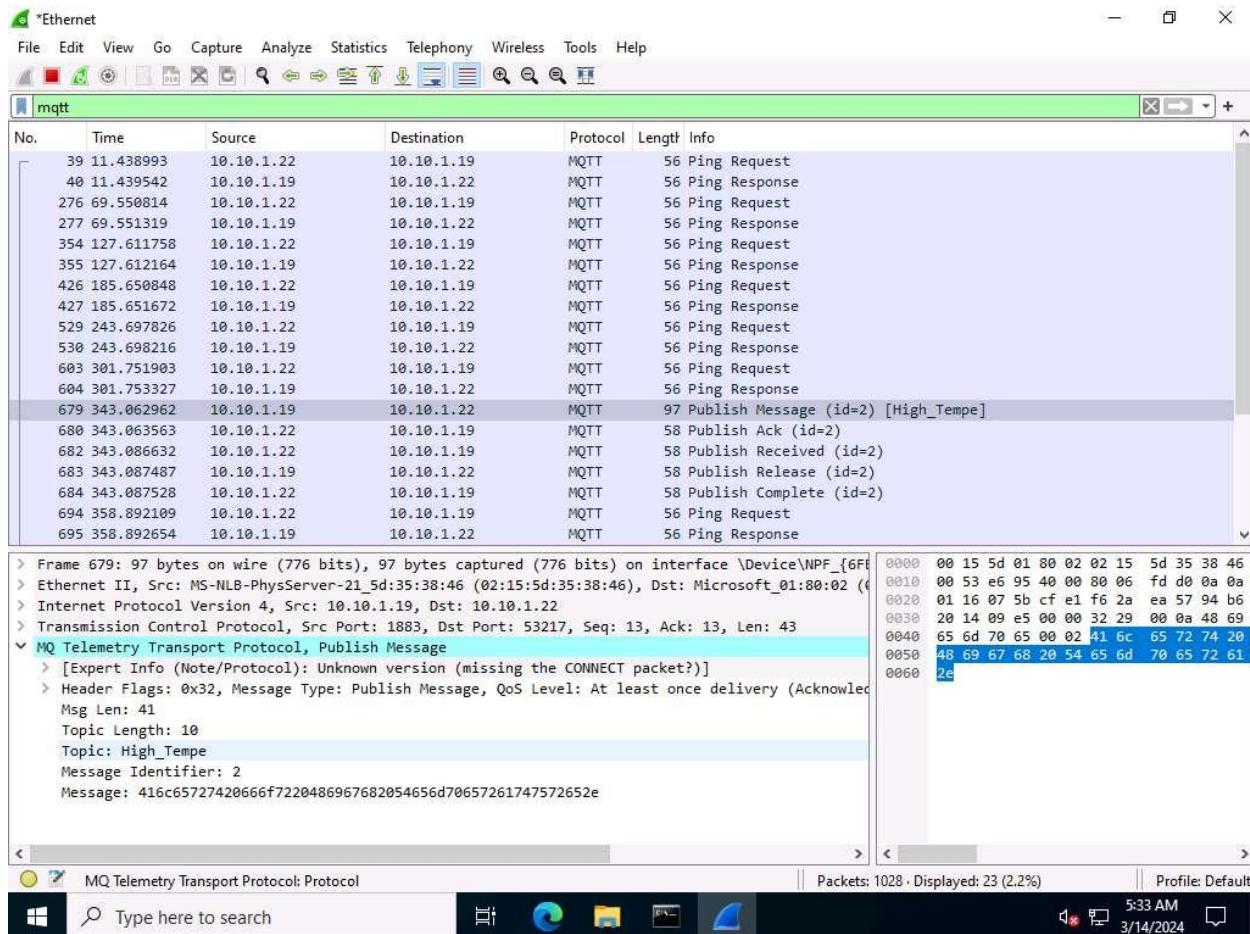
42. The message has been sent to the device using this topic.
43. Next, switch to [Windows Server 2022](#) machine.
44. We have left the IoT simulator running in the web browser. To see the alert message, maximise the Edge browser and expand the arrow under the connected **Temperature_Sensor, Device Log** section.
45. You can see the alert message "**Alert for High Temperature**"

The screenshot shows the IoT Simulator application running in a browser window. The URL is 127.0.0.1:9000/setnetwork?network=CEH_FINANCE_NETWORK. The main content area displays a device configuration for a 'Temperature_Sensor' named 'TS1'. The device has a description: 'This is a Temperature Sensor IoT device.' Under the 'Authentication' section, both 'Authentication' and 'Will' toggles are turned on. To the right, a 'Device Log' panel shows a command received by the device: 'Command received by Device TS1' at 'Thu Mar 14 05:30:45 2024'. The log entry includes a 'Topic' field with 'High_Tempe' and a 'Message' field with '"Alert for High Temperature."'. At the bottom of the screen, a Windows taskbar is visible with icons for File Explorer, Edge, Task View, and others.

46. To verify the communication, we have executed **Wireshark** application, switch to the Wireshark traffic capturing window.
47. Type **mqtt** under the **filter** field and press **Enter**. To display only the MQTT protocol packets.



48. Select any **Publish Message** packet from the **Packet List** pane. In the **Packet Details** pane at the middle of the window, expand the **Transmission Control Protocol**, **MQ Telemetry Transport Protocol**, and **Header Flags** nodes.
49. Under the **MQ Telemetry Transport Protocol** nodes, you can observe details such as **Msg Len**, **Topic Length**, **Topic**, and **Message**.
50. Publish Message can be used to obtain the message sent by the MQTT client to the broker.

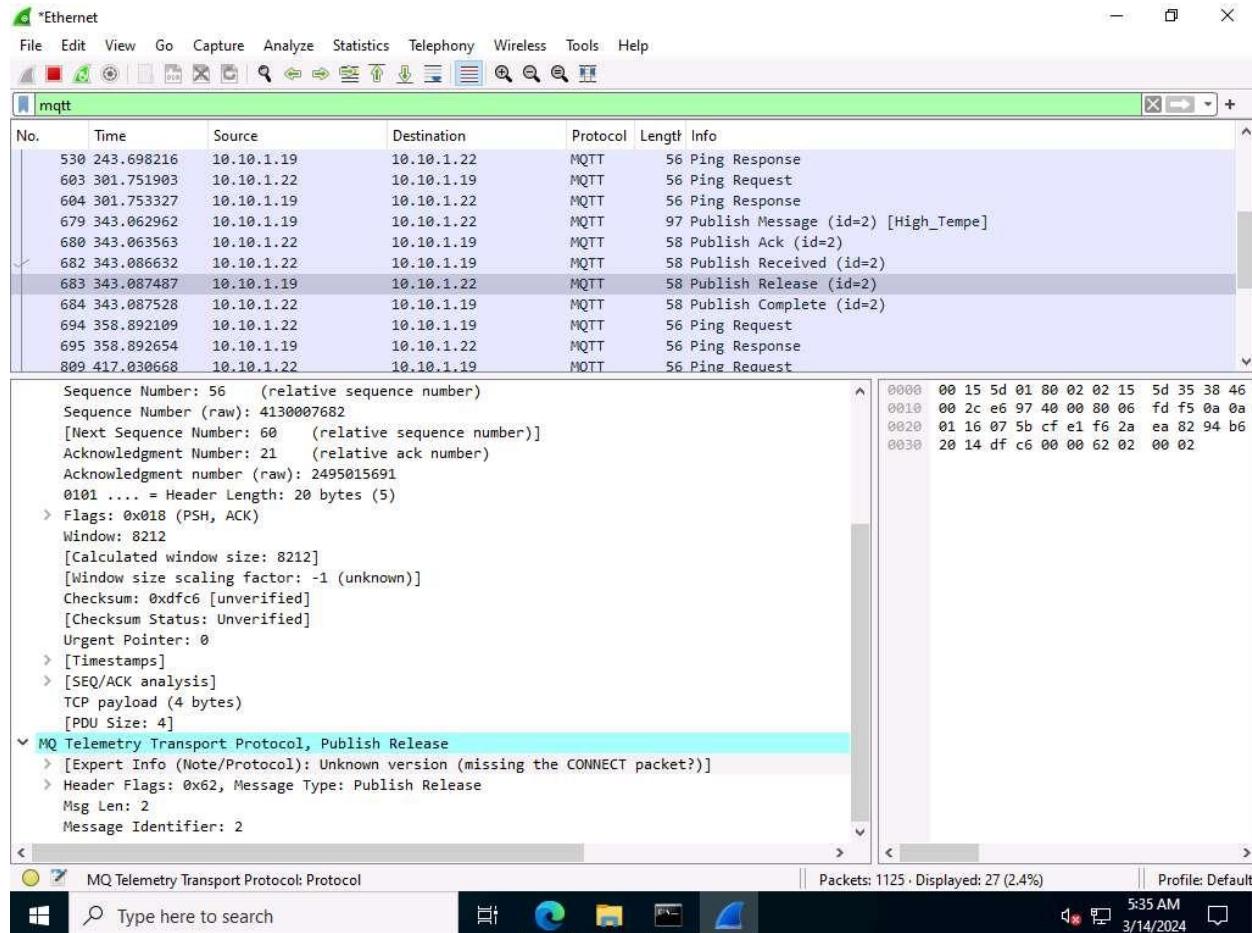


Note: After establishing a successful connection with the MQTT broker, the MQTT client can publish messages. The headers in the Publish Message packet are given below:

- Header Flags: Contains information regarding the MQTT control packet type.
- DUP flag: If the DUP flag is 0, it indicates the first attempt at sending this PUBLISH packet; if the flag is 1, it indicates a possible re-attempt at sending the message.
- QoS: Determines the assurance level of a message.
- Retain Flag: If the retain flag is set to 1, the server must store the message and its QoS, so it can cater to future subscriptions matching the topic.
- Topic Name: Contains a UTF-8 string that can also include forward slashes when it needs to be hierarchically structured.
- Message: Contains the actual data to be transmitted.
- Payload: Contains the message that is being published.

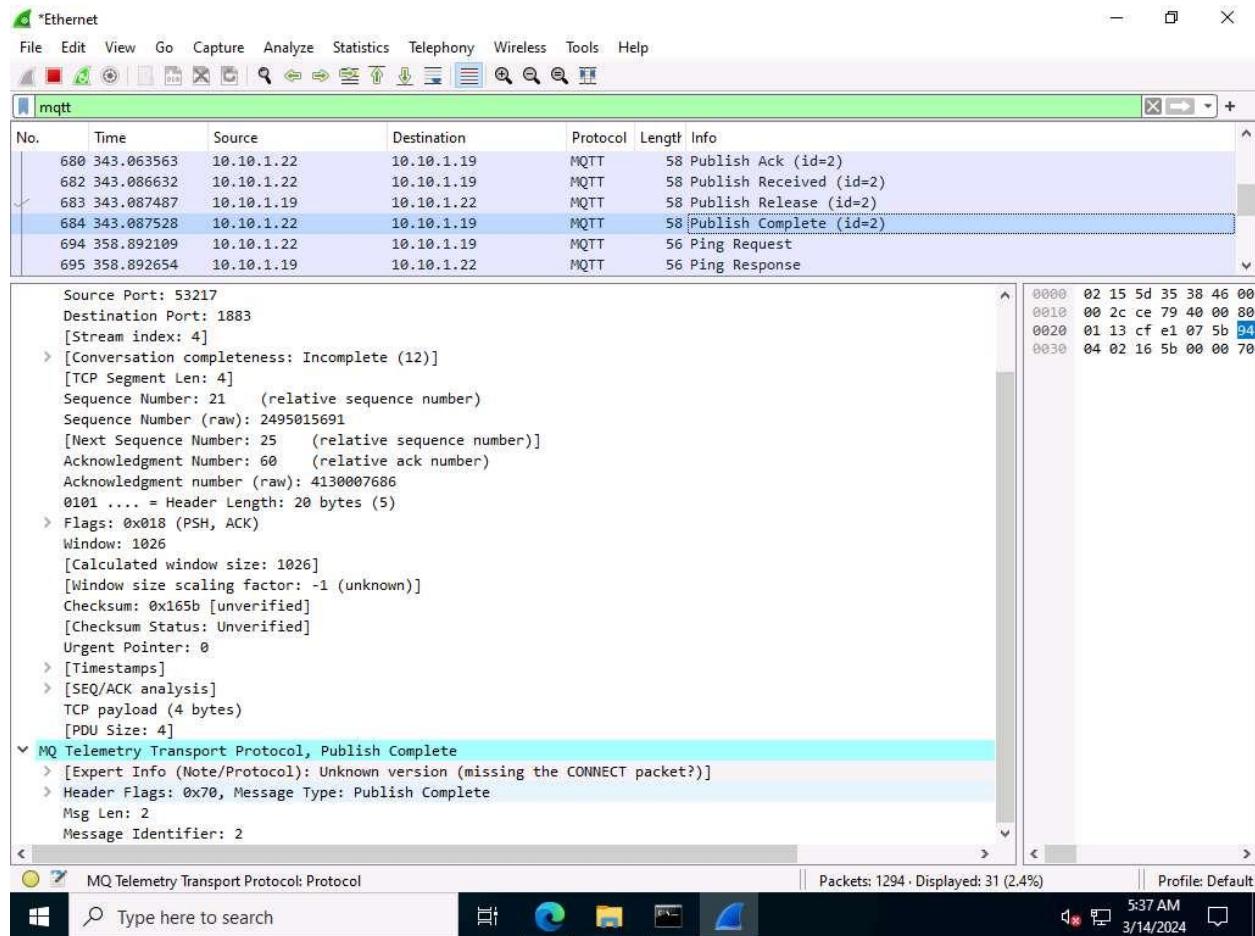
51. Select any **Publish Release** packet from the **Packet List** pane. In the **Packet Details** pane at the middle of the window, expand the **Transmission Control Protocol**, **MQ Telemetry Transport Protocol**, and **Header Flags** nodes.

52. Under the **MQ Telemetry Transport Protocol** nodes, you can observe details such as **Msg Len**, **Message Type**, **Message Identifier**.



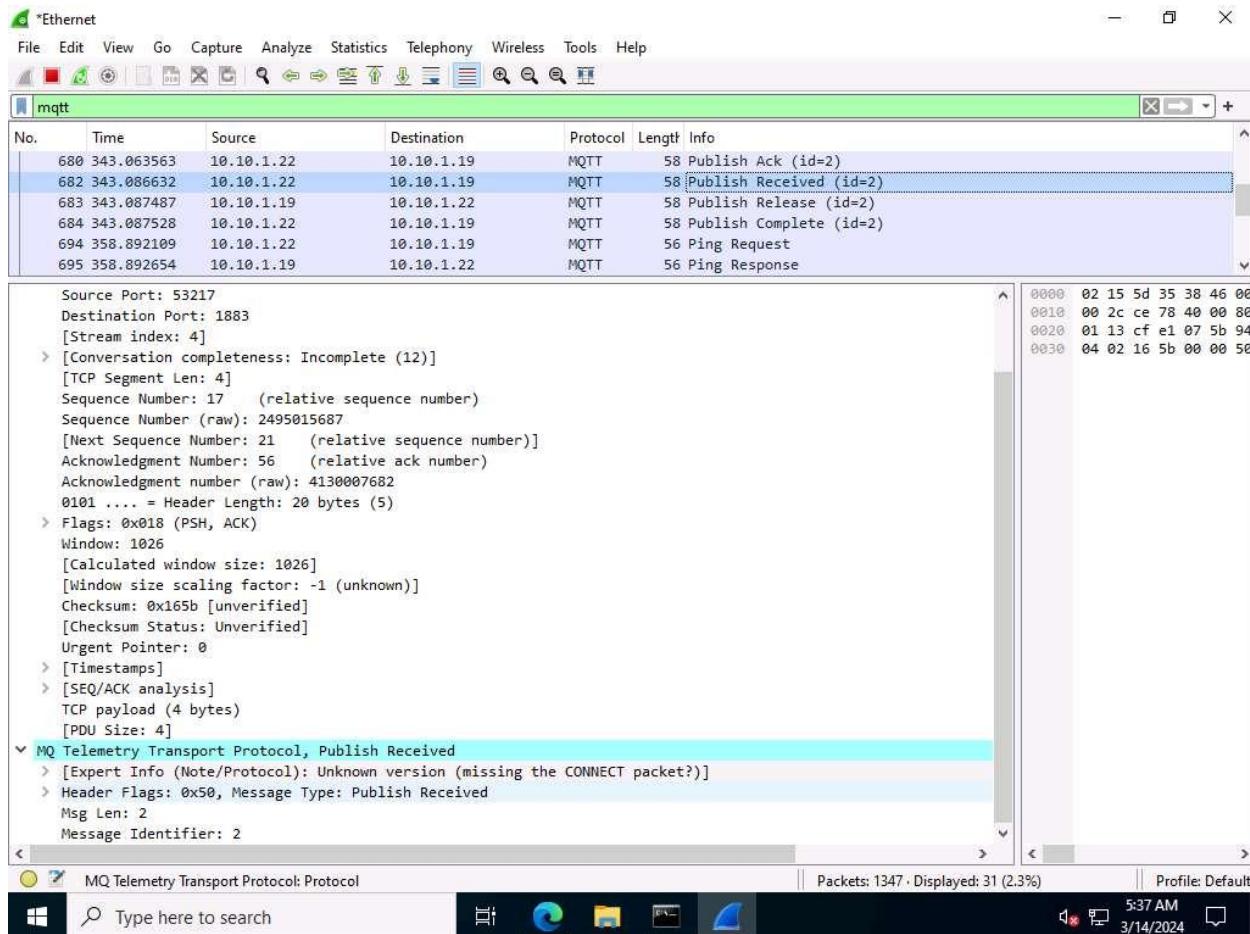
Note: A Publish Release (PUBREL) packet is the response to a Publish Received (PUBREC) packet.

53. Now, scroll down, look for the **Publish Complete** packet from the **Packet List** pane, and click on it. In the **Packet Details** pane at the middle of the window, expand the **Transmission Control Protocol**, **MQ Telemetry Transport Protocol**, and **Header Flags** nodes.
54. Under the **MQ Telemetry Transport Protocol** nodes, you can observe details such as **Msg Len** and **Message Identifier**.



Note: The Publish Complete (PUBCOMP) packet is the response to a Publish Release (PUBREL) packet.

55. Now, scroll down, look for the **Publish Received** packet from the **Packet List** pane, and click on it. In the **Packet Details** pane at the middle of the window, expand the **Transmission Control Protocol**, **MQ Telemetry Transport Protocol**, and **Header Flags** nodes.
56. Under the **MQ Telemetry Transport Protocol** nodes, you can observe details such as **Message Type**, **Msg Len** and **Message Identifier**.



57. Similarly you can select **Ping Request**, **Ping Response** and **Publish Ack** packets and observe the details.
58. This concludes the demonstration of capturing and analyzing MQTT protocol packets. Here, we analyzed different processes involved in the communication between an MQTT client and an MQTT broker using Wireshark. Understanding these metrics as well as the workflow can help you in quickly identifying the MQTT-related issues.
59. Close all open windows and document all the acquired information.

Question 18.2.1.1

Use Wireshark and Bevywise MQTT Route and Bevywise IoT Simulator to capture and analyze traffic between IoT devices. What is the default TCP port used by Bevywise MQTT Route to establish connection with Bevywise IoT Simulator?

Question 18.2.1.2

Use Wireshark and Bevywise MQTT Route and Bevywise IoT Simulator to capture and analyze traffic between IoT devices. What is the default WebSocket port used by Bevywise MQTT IoT Simulator to establish connection with Bevywise MQTT Route?