

Module 18: IoT and OT Hacking

Lab 1: Perform Footprinting using Various Footprinting Techniques

Lab Scenario

As a professional ethical hacker or pen tester, your first step is to gather maximum information about the target IoT and OT devices by performing footprinting through search engines, advanced Google hacking, Whois lookup, etc.

The first step in IoT and OT device hacking is to extract information such as IP address, protocols used (MQTT, ModBus, ZigBee, BLE, 5G, IPv6LoWPAN, etc.), open ports, device type, geolocation of the device, manufacturing number, and manufacturer of the device.

Lab Objectives

- Gather information using online footprinting tools

Overview of Footprinting Techniques

Footprinting techniques are used to collect basic information about the target IoT and OT platforms to exploit them. Information collected through footprinting techniques includes IP address, hostname, ISP, device location, banner of the target IoT device, FCC ID information, certification granted to the device, etc.

Task 1: Gather Information using Online Footprinting Tools

The information regarding the target IoT and OT devices can be acquired using various online sources such as Whois domain lookup, advanced Google hacking, and Shodan search engine. The gathered information can be used to scan the devices for vulnerabilities and further exploit them to launch attacks.

In this task, we will focus on performing footprinting on the MQTT protocol, which is a machine-to-machine (M2M)/"Internet of Things" connectivity protocol. It is useful for connections with remote locations where a small code footprint is required and/or network bandwidth is at a premium.

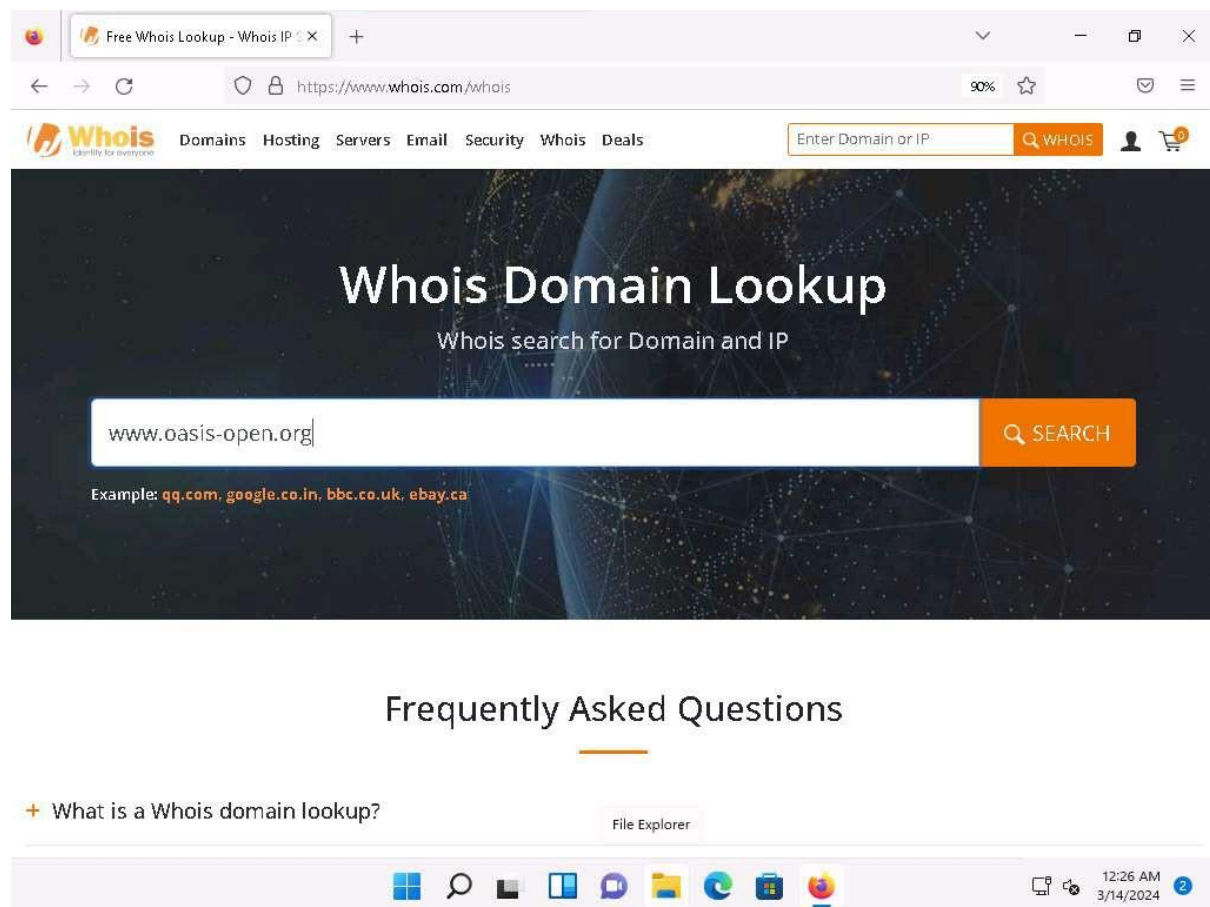
You can also select a protocol or device of your choice to perform footprinting on it.

1. By default **Windows 11** machine selected, click Ctrl+Alt+Delete. Login with **Admin/Pa\$\$w0rd**.

Networks screen appears, click **Yes** to allow your PC to be discoverable by other PCs and devices on the network.

2. Launch any web browser, go to <https://www.whois.com/whois> (here, we are using **Mozilla Firefox**).
3. The **Whois Domain Lookup** page appears; type **www.oasis-open.org** in the search field and click **SEARCH**.

Oasis is an organization that has published the MQTT v5.0 standard, which represents a significant leap in the refinement and capability of the messaging protocol that already powers IoT.



4. The result appears, displaying the following information, as shown in the screenshots: Domain Information, Registrant Contact, and Raw Whois Data.

This information is about the organization that has developed the MQTT protocol, and it might help keep track of the modifications and version changes of the target protocol.

Whois oasis-open.org

https://www.whois.com/whois/oasis-open.org

Whois
Liberty for everyone

Domains Hosting Servers Email Security Whois Deals

Enter Domain or IP

WHOIS

On Sale!

.LIFE

.LIFE @ \$2.48 \$35.00

Introducing

WORDPRESS
HOSTING

\$5.48 /mo

VIEW MORE

Raw Whois Data

Domain Name: oasis-open.org
Registry Domain ID: 2bc33180c6aa48c180bb9e4f887737bd-LROR
Registrar WHOIS Server: http://whois.directnic.com
Registrar URL: http://www.directnic.com
Updated Date: 2024-01-23T07:30:05Z
Creation Date: 1998-03-04T05:00:00Z
Registry Expiry Date: 2025-03-03T05:00:00Z
Registrar: DNC Holdings, Inc.
Registrar IANA ID: 291
Registrar Abuse Contact Email: abuse@directnic.com
Registrar Abuse Contact Phone: +1.8778569598
Domain Status: clientDeleteProhibited https://icann.org/epp#clientDeleteProhibited
Domain Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited
Domain Status: clientUpdateProhibited https://icann.org/epp#clientUpdateProhibited
Registry Registrant ID: REDACTED FOR PRIVACY
Registrant Name: REDACTED FOR PRIVACY
Registrant Organization: OASIS Open
Registrant Street: REDACTED FOR PRIVACY
Registrant City: REDACTED FOR PRIVACY
Registrant State/Province: MA
Registrant Postal Code: REDACTED FOR PRIVACY
Registrant Country: US
Registrant Phone: REDACTED FOR PRIVACY
Registrant Phone Ext: REDACTED FOR PRIVACY
Registrant Fax: REDACTED FOR PRIVACY
Registrant Fax Ext: REDACTED FOR PRIVACY
Registrant Email: Please query the RDDS service of the Registrar of Record identified in this ou

12:31 AM
3/14/2024

Whois lookup reveals available information on a hostname, IP address, or domain.

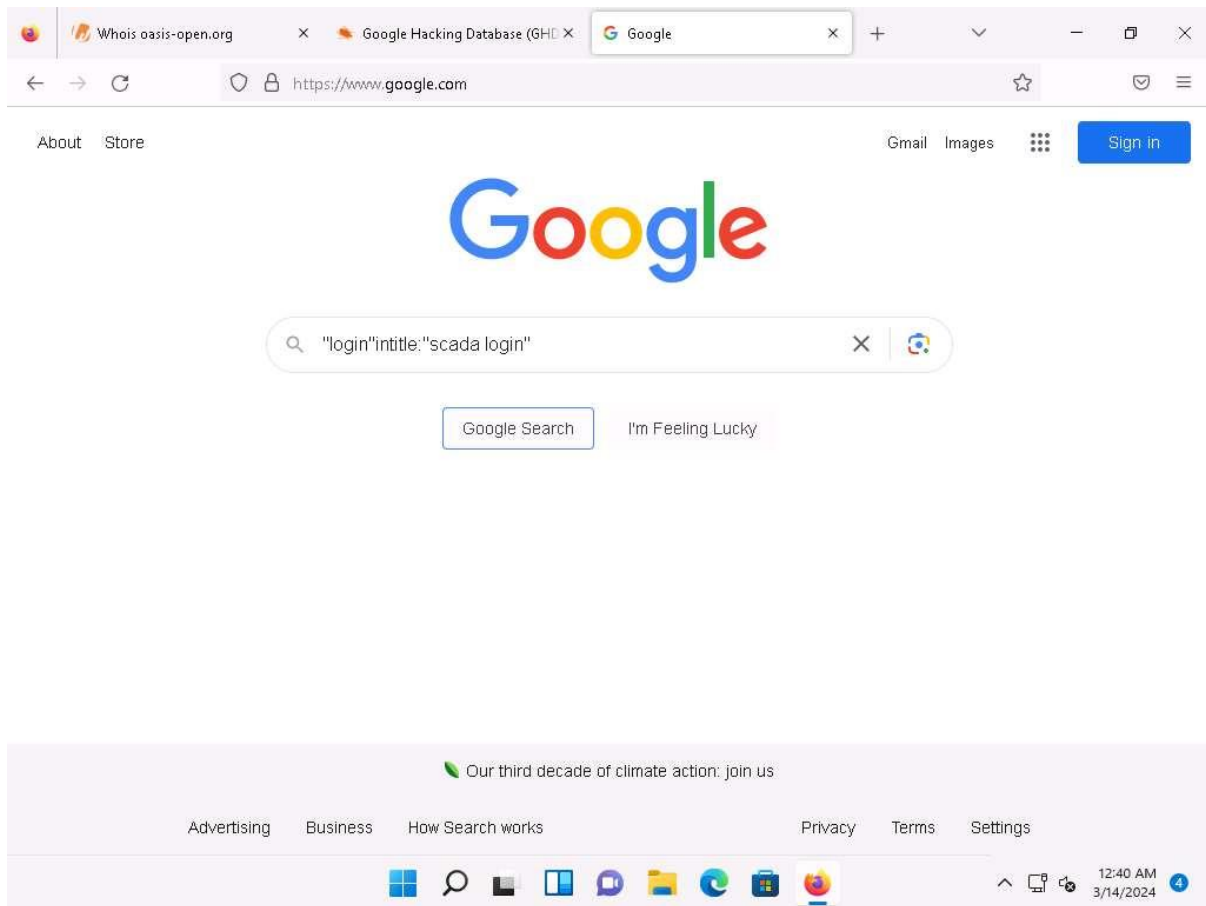
- Now, open a new tab, and go to **<https://www.exploit-db.com/google-hacking-database>**.
- The **Google Hacking Database** page appears; type **SCADA** in the **Quick Search** field and press **Enter**.
- The result appears, which displays the Google dork related to SCADA, as shown in the screenshot.

The screenshot shows the Exploit Database website with a search for 'SCADA'. The results table is highlighted with a red border and contains the following data:

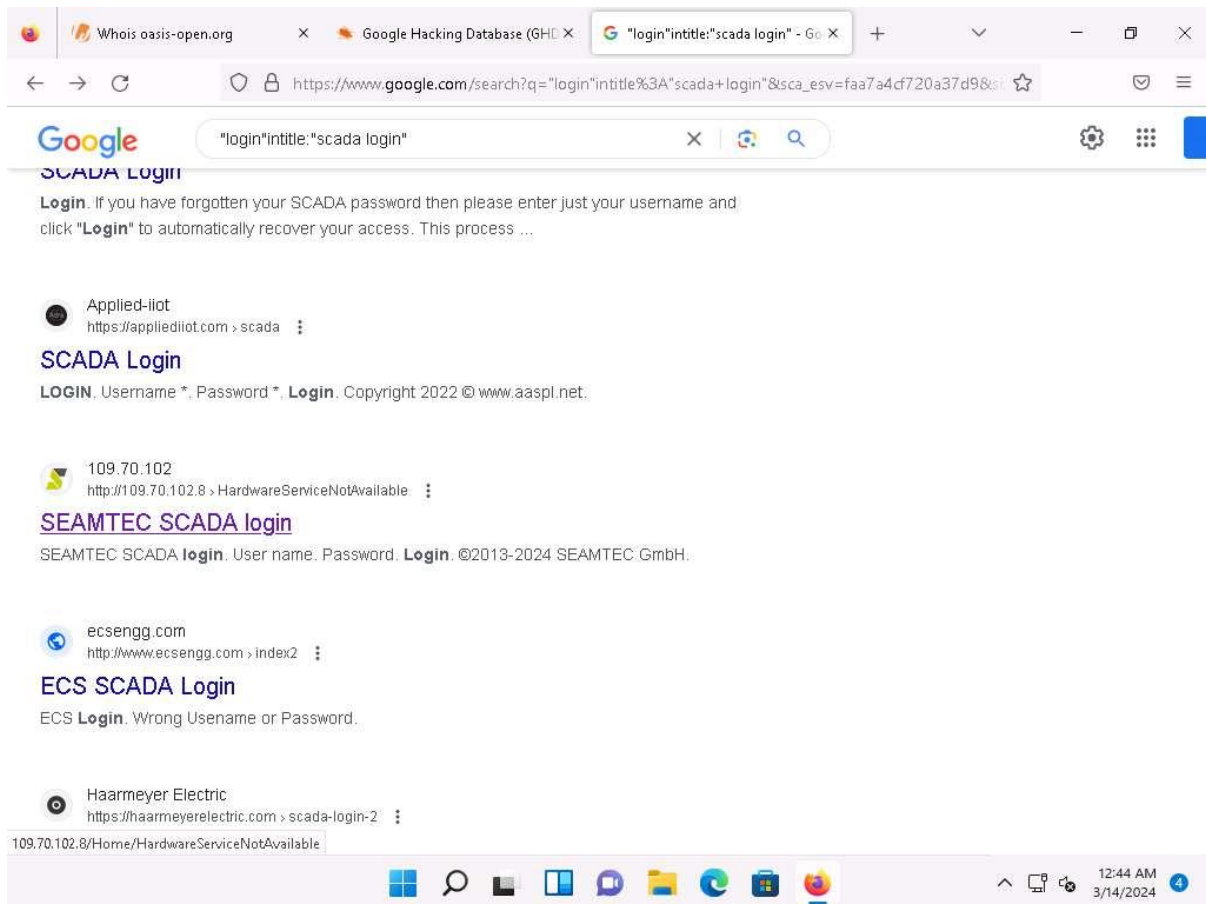
| Date Added | Dork | Category | Author |
|------------|---|--------------------------------|-------------------------|
| 2023-04-06 | inurl:"/scada-vis" | Files Containing Juicy Info | Parsa Rezaie Khiabanloo |
| 2021-10-04 | intitle:"index of SCADA" | Sensitive Directories | Romell Marin Cordoba |
| 2021-09-20 | intitle:inurl:"SCADA login" | Pages Containing Login Portals | Cyber Shelby |
| 2021-09-16 | intitle:"CirCarLife Scada" inurl:/html/index.html | Various Online Devices | Alexandros Pappas |
| 2020-05-28 | "login" intitle:"*scada login" | Pages Containing Login Portals | Alexandros Pappas |
| 2019-04-22 | intitle:"index of" scada | Sensitive Directories | Aman Bhardwaj |
| 2018-04-06 | "login" intitle:"scada login" | Pages Containing Login Portals | Bruno Schmid |

Showing 1 to 7 of 7 entries (filtered from 7,915 total entries)

8. Now, we will use the dorks obtained in the previous step to query results in Google.
9. Open a new tab and go to <https://www.google.com>. In the search field, enter **"login" intitle:"scada login"**.



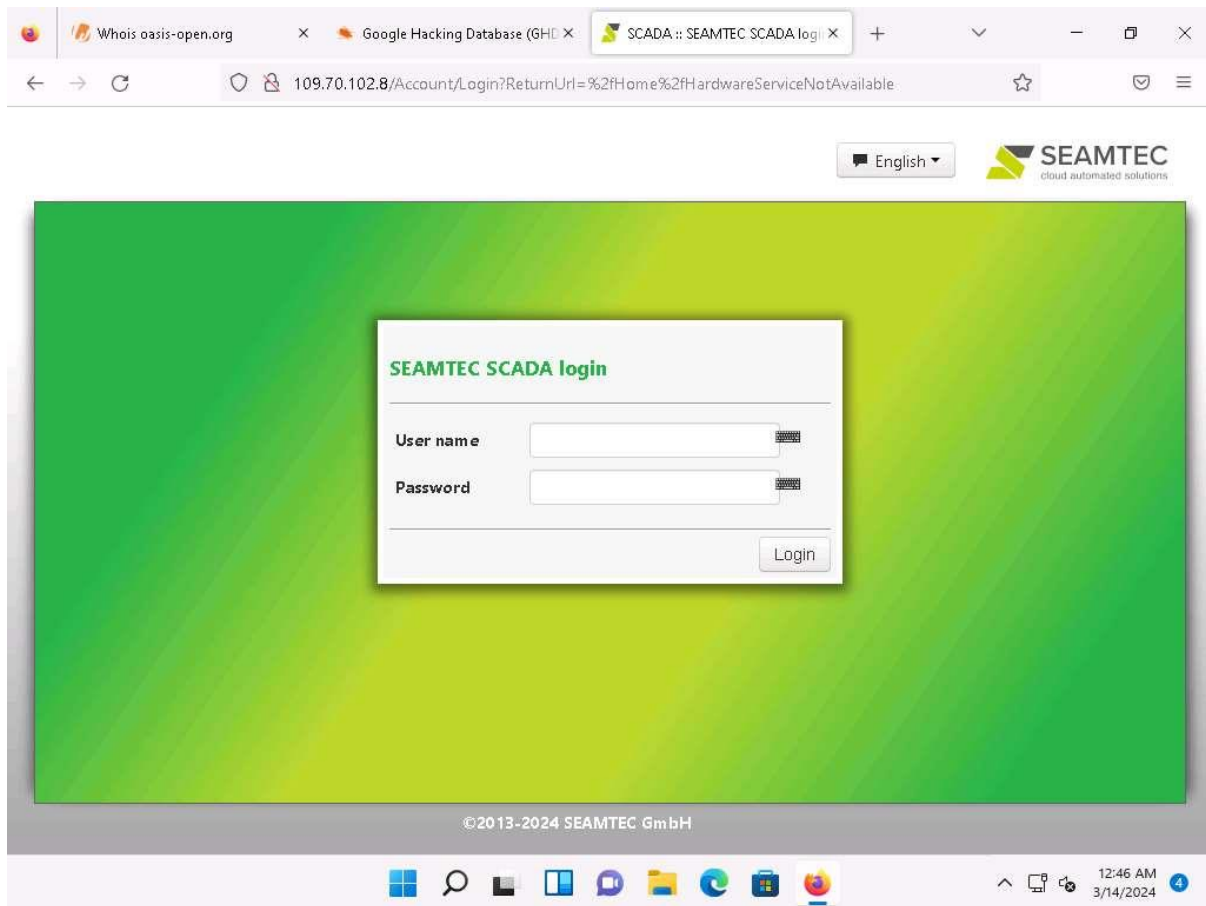
10. The search result appears; click any link (here, **SEAMTEC SCADA login**).



Advanced Google hacking refers to the art of creating complex search engine queries by employing advanced Google operators to extract sensitive or hidden information about a target company from the Google search results.

11. The **SEAMTEC SCADA login** page appears, as shown in the screenshot.

In the login form, you can brute-force the credentials to gain access to the target SCADA system.

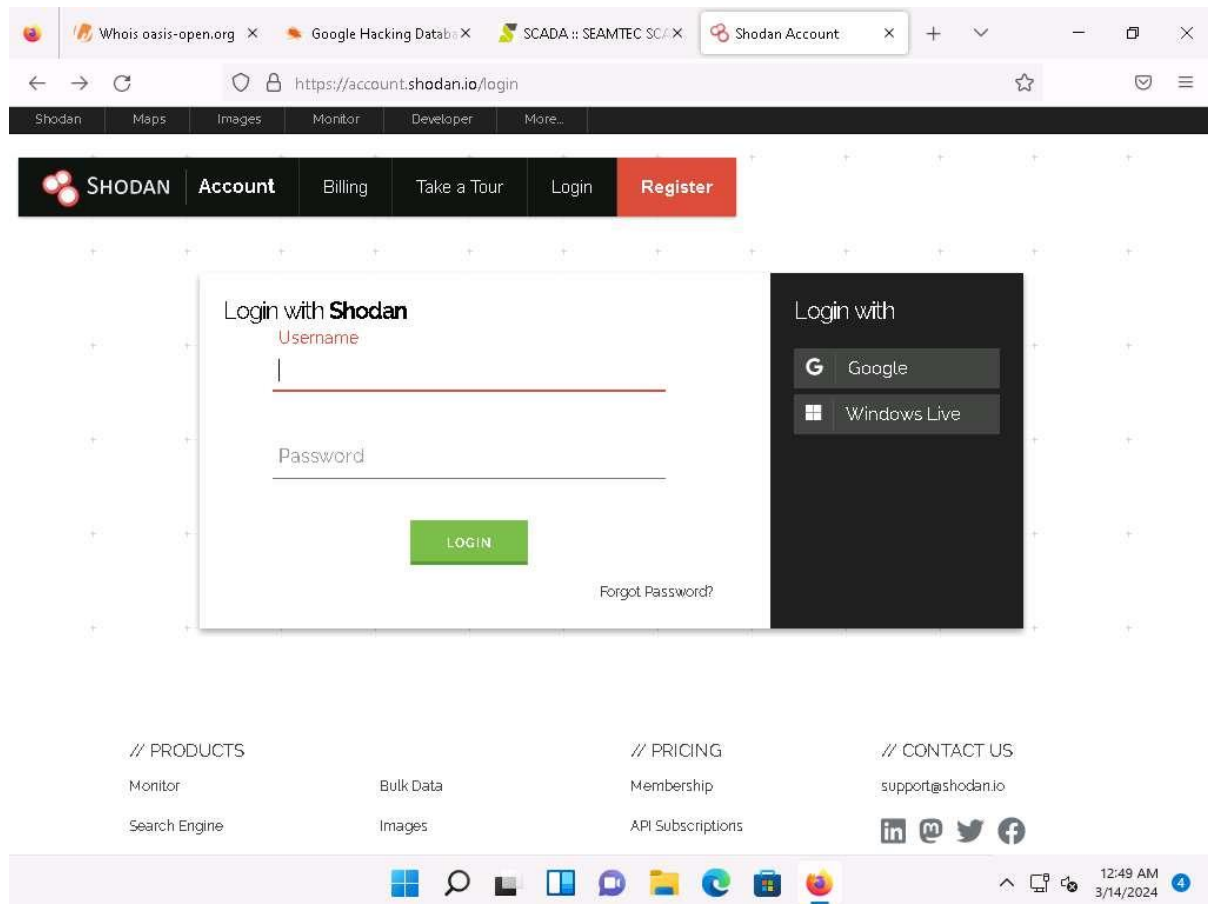


12. Similarly, you can use advanced search operators such as **intitle:"index of" scada** to search sensitive SCADA directories that are exposed on sites.

13. Now, in the browser window, open a new tab and go to **https://account.shodan.io/login**.

14. The **Login with Shodan** page appears; enter your username and password in the **Username** and **Password** fields, respectively; and click **Login**.

If you do not have an existing account, then go to the **Register** option to register yourself .

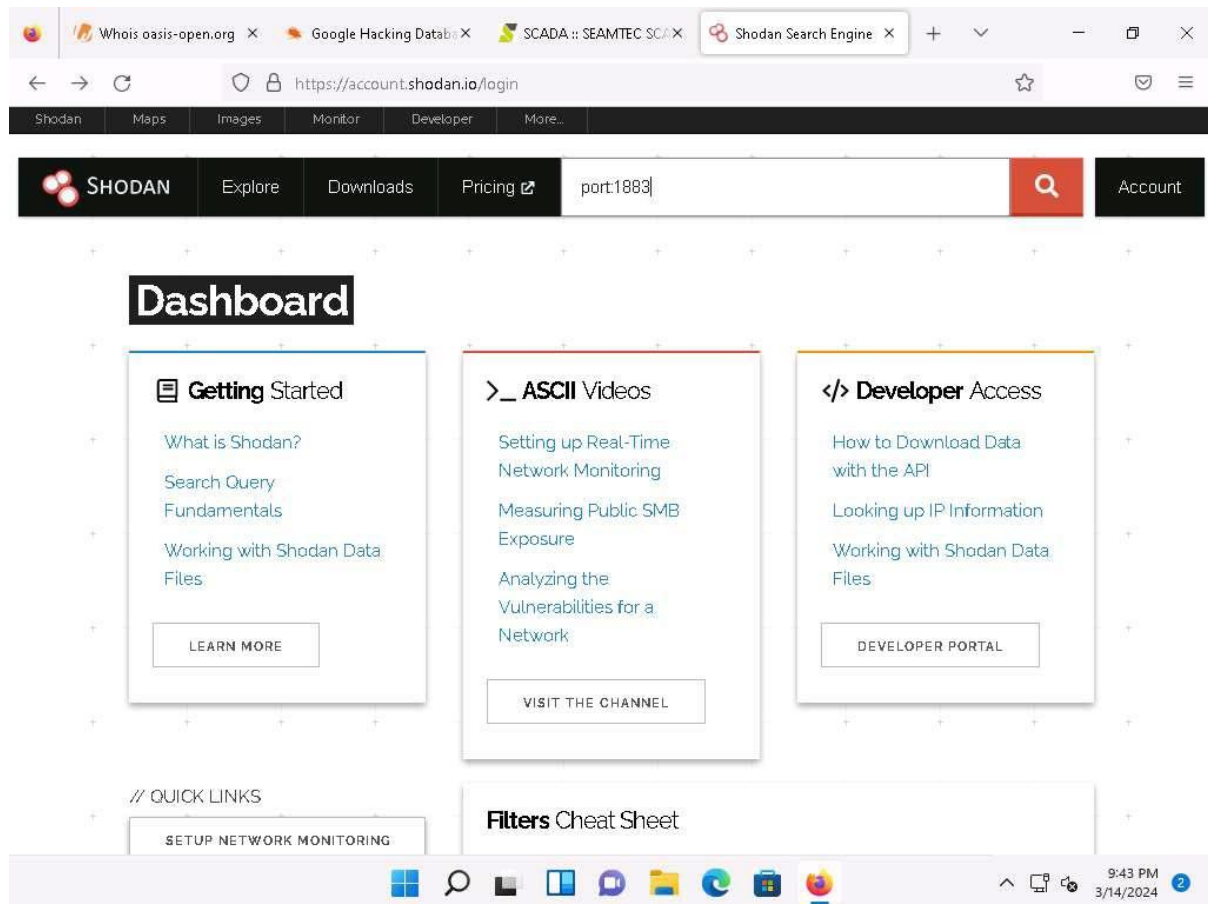


15. The **Account Overview** page appears, which displays the account-related information. Click on **Shodan** on top-left corner of the window to go to the main page of **Shodan**.

If the **Would you like Firefox to save this login for shodan.io?** notification appears, click **Don't Save**.

16. The **Shodan** main page appears; type **port:1883** in the address bar and press **Enter**.

Port 1883 is the default MQTT port; 1883 is defined by IANA as MQTT over TCP.



17. The result appears, displaying the list of IP addresses having port 1883 enabled.

18. Click on any IP address to view its detailed information.

Whois oasis-open.org x Google Hacking Database x SCADA :: SEAMTEC SCADA x port1883 - Shodan Search x


https://account.shodan.io/login

TOTAL RESULTS
1,018,738

View Report Browse Images View on Map

Access Granted: Want to get more out of your existing Shodan account? Check out [everything you have access to.](#)

TOP COUNTRIES



| Country | Count |
|--------------------|---------|
| United States | 418,011 |
| Korea, Republic of | 363,848 |
| China | 105,214 |
| Japan | 18,113 |
| Germany | 13,585 |

[More...](#)

TOP ORGANIZATIONS

| Organization | Count |
|-----------------------|---------|
| SK Broadband Co ... | 357,215 |
| Google LLC | 355,579 |
| Alhyn Computing C... | 40,166 |
| Flyio, Inc. | 22,508 |
| Huawei Public Clou... | 10,559 |

[More...](#)

TOP PRODUCTS

34.49.29.35

35.29.49.34.b.googleusercontent.com

Google LLC

United States, Kansas City

cloud

No data returned

2024-03-15T04:43:20.440441

130.211.8.229

229.8.211.130.b.googleusercontent.com

Google LLC

United States, Kansas City

cloud

No data returned

2024-03-15T04:43:13.001874

213.188.219.148

Flyio, Inc.

United States, Chicago

No data returned

2024-03-15T04:43:00.028015

39.125.233.41

SK Broadband Co Ltd

South Korea, Republic of, Yeosu

MQTT Connection Code: 0

Topics:

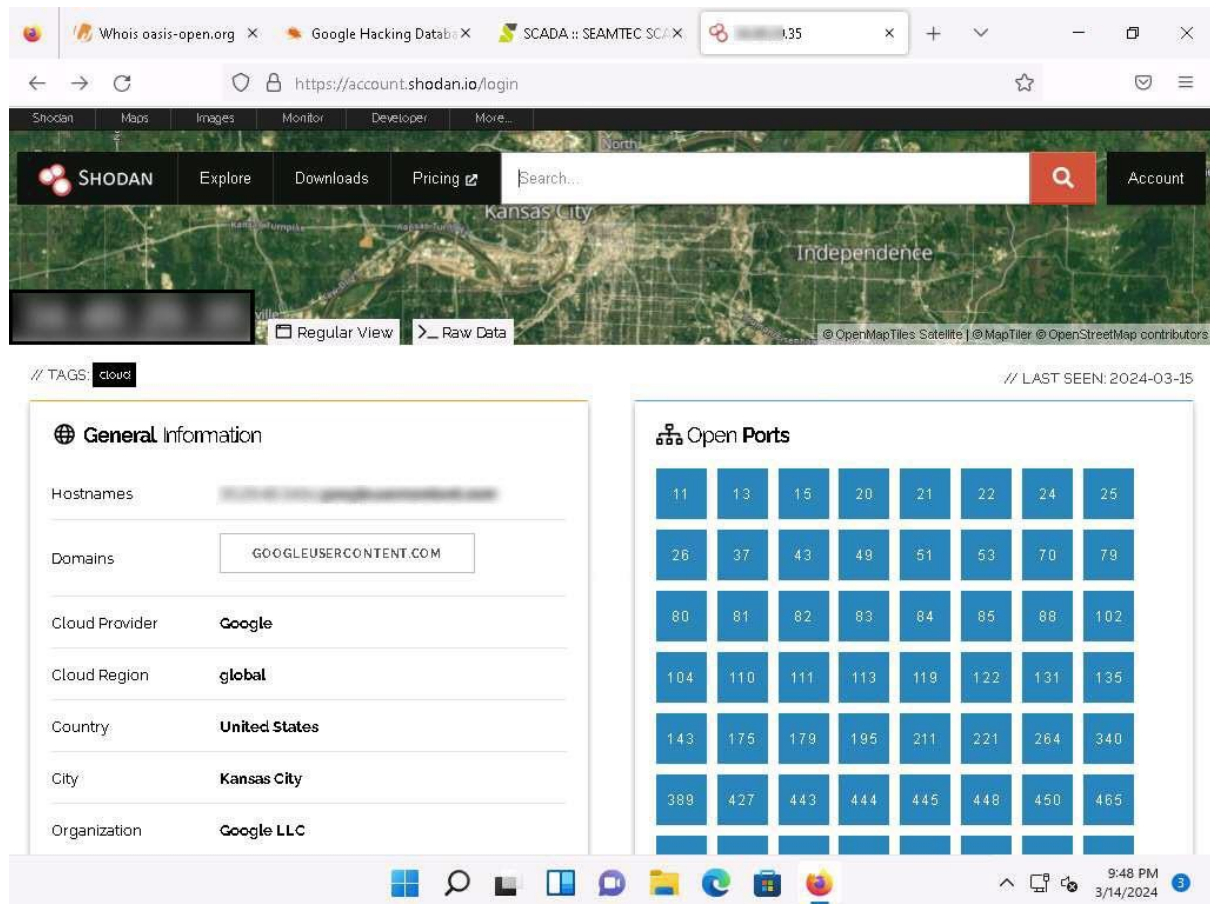
2024-03-15T04:42:48.171868

58.236.75.116

2024-03-15T04:42:46.756183

9:44 PM 3/14/2024

19. Detailed results for the selected IP address appears, displaying information regarding **Ports, Services, Hostnames, ASN**, etc. as shown in the screenshot.



20. Similarly, you can gather additional information on a target device using the following Shodan filters:

- **Search for Modbus-enabled ICS/SCADA systems:**

port:502

- **Search for SCADA systems using PLC name:**

"Schneider Electric"

- **Search for SCADA systems using geolocation:**

SCADA Country:"US"

21. Using Shodan, you can obtain the details of SCADA systems that are used in water treatment plants, nuclear power plants, HVAC systems, electrical transmission systems, home heating systems, etc.

22. This concludes the demonstration of gathering information on a target device using various techniques such as Whois lookup, advanced Google hacking, and Shodan search engine.

23. Close all open windows and document all the acquired information.

Question 18.1.1.1

Use the Shodan search engine to collect the IP addresses with MQTT enabled. Perform a search using the MQTT port number. Which port number will you enter in the search field to obtain the desired result?

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.

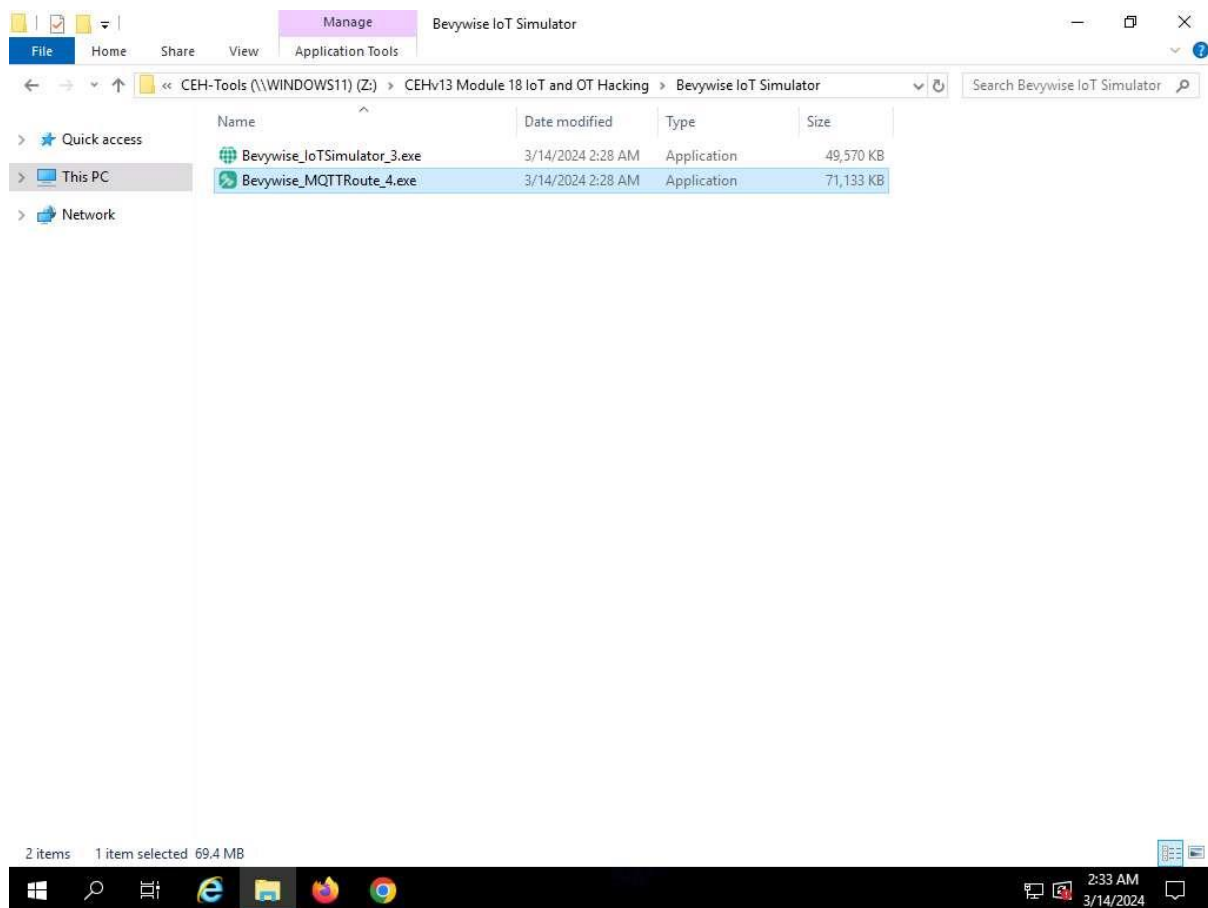
Task 1: Capture and Analyze IoT Traffic using Wireshark

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.

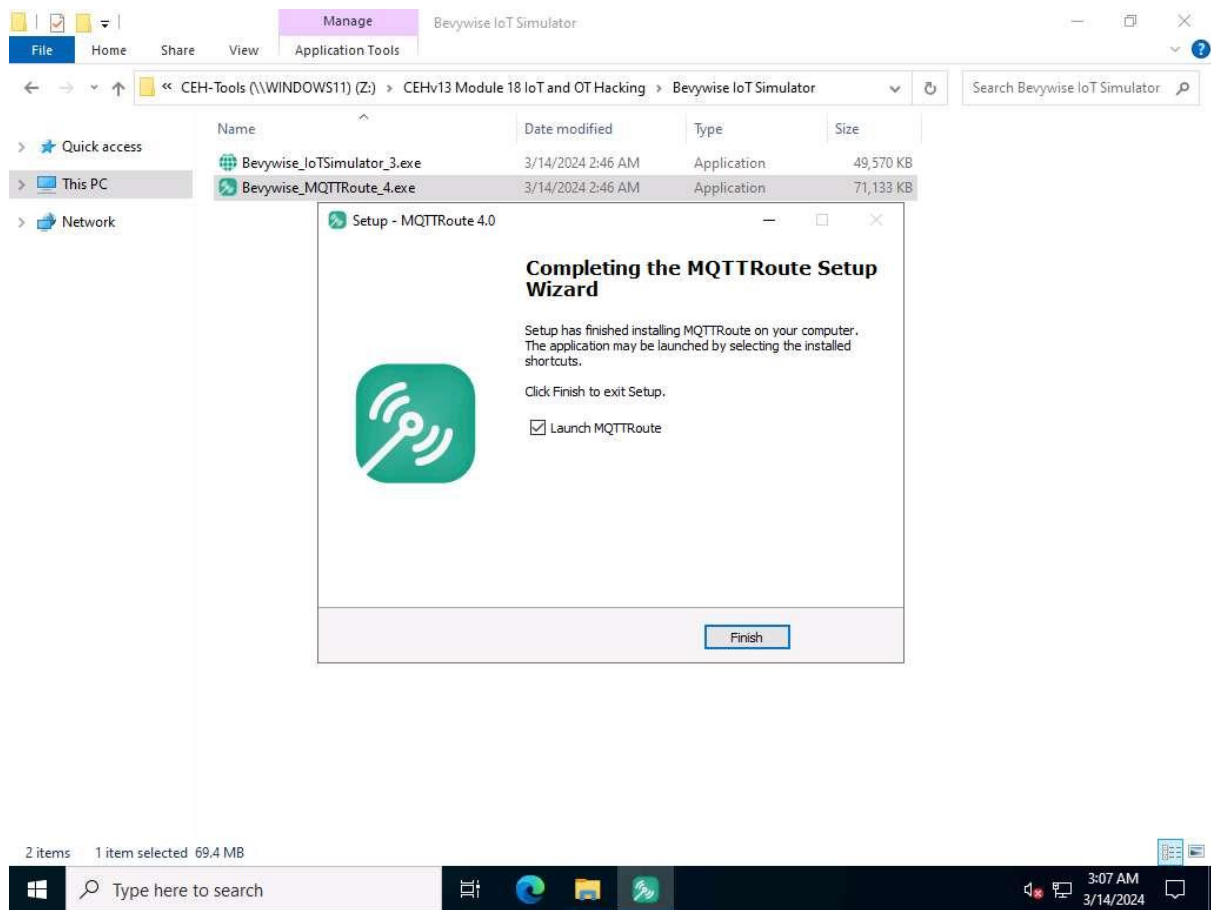
MQTT is a lightweight messaging protocol that uses a publish/subscribe communication pattern. Since the protocol is meant for devices with a low-bandwidth, it is considered ideal for machine-to-machine (M2M) communication or IoT applications. We can create virtual IoT devices over the virtual network 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.

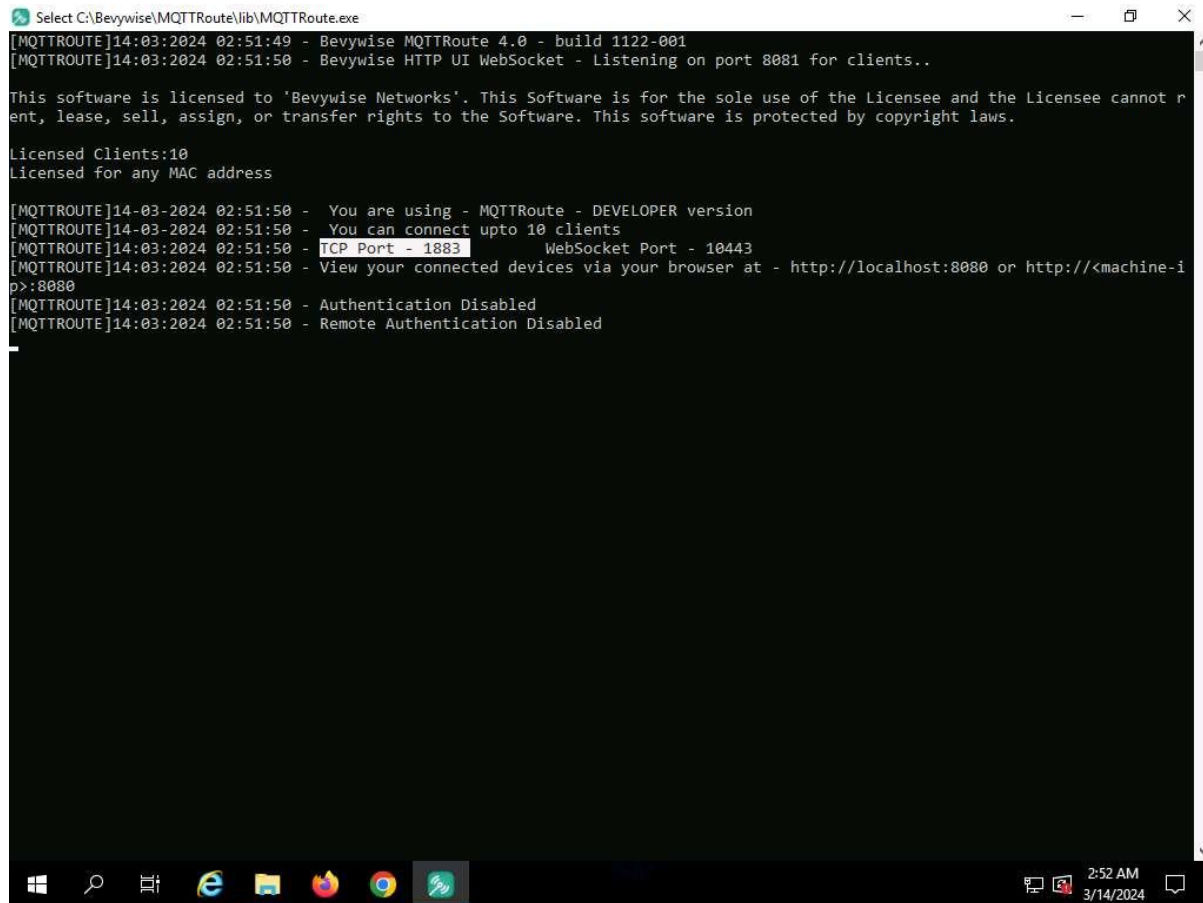
1. To install the **MQTT Broker** on the **Windows Server 2019**, click Windows Server 2019 to launch **Windows Server 2019** machine.
Click Ctrl+Alt+Delete and login with **Administrator/Pa\$\$w0rd**.
2. Navigate to **Z:\CEHv13 Module 18 IoT and OT Hacking\Bevywise IoT Simulator** folder and double-click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



```
Select C:\Bevywise\MQTTRoute\lib\MQTTRoute.exe
[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..

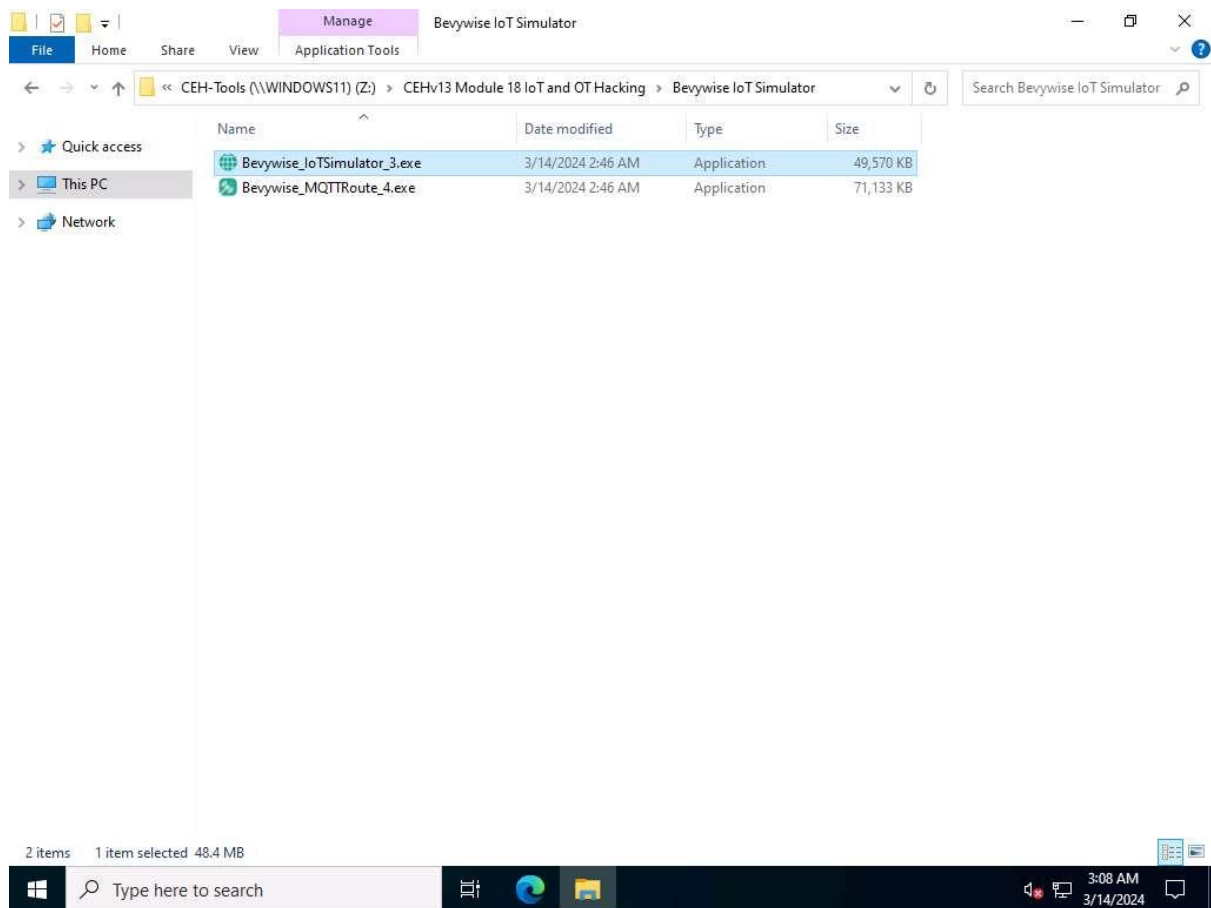
This software is licensed to 'Bevywise Networks'. This Software is for the sole use of the Licensee and the Licensee cannot r
ent, lease, sell, assign, or transfer rights to the Software. This software is protected by copyright laws.

Licensed Clients:10
Licensed for any MAC address

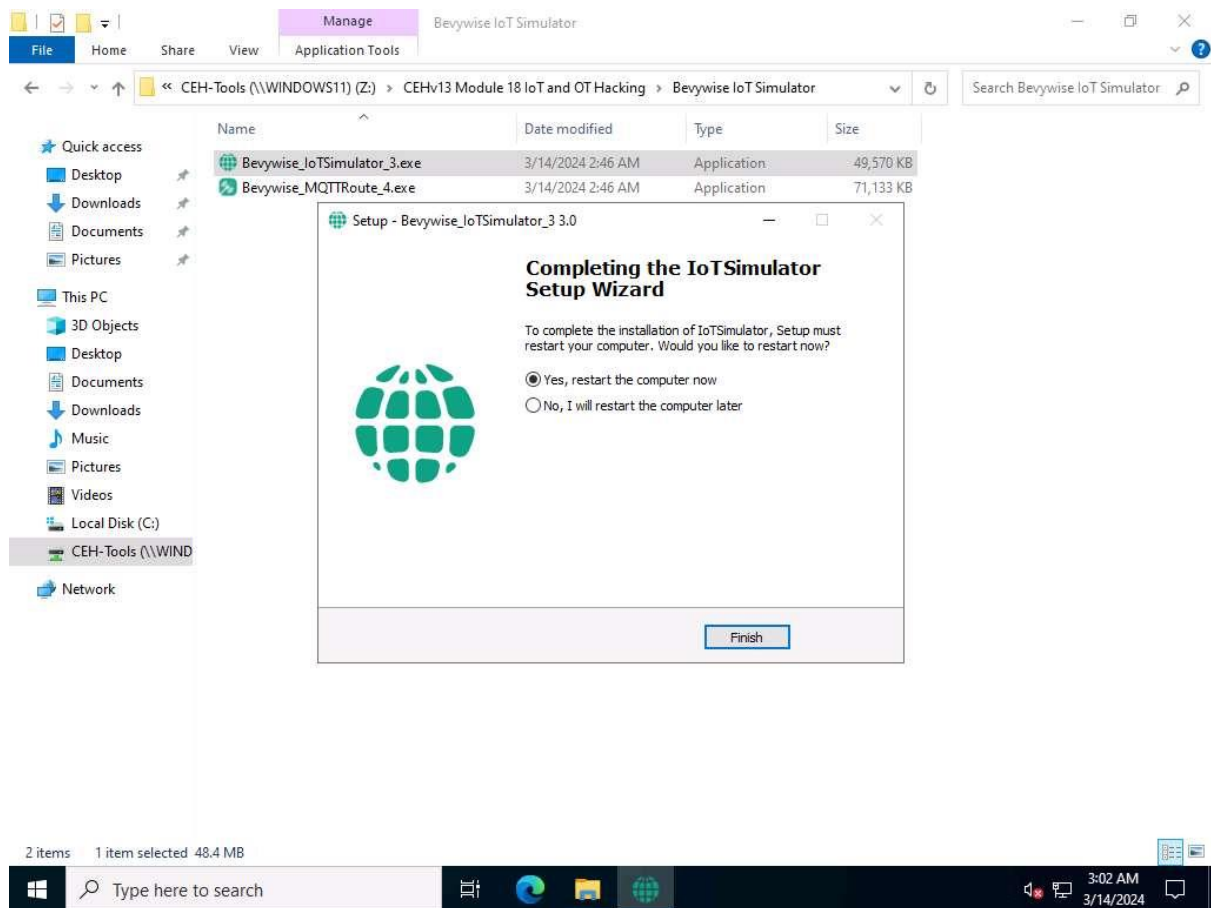
[MQTTRoute]14-03-2024 02:51:50 - You are using - MQTTRoute - DEVELOPER version
[MQTTRoute]14-03-2024 02:51:50 - You can connect upto 10 clients
[MQTTRoute]14:03:2024 02:51:50 - TCP Port - 1883 Websocket Port - 10443
[MQTTRoute]14:03:2024 02:51:50 - View your connected devices via your browser at - http://localhost:8080 or http://<machine-i
p>: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**.

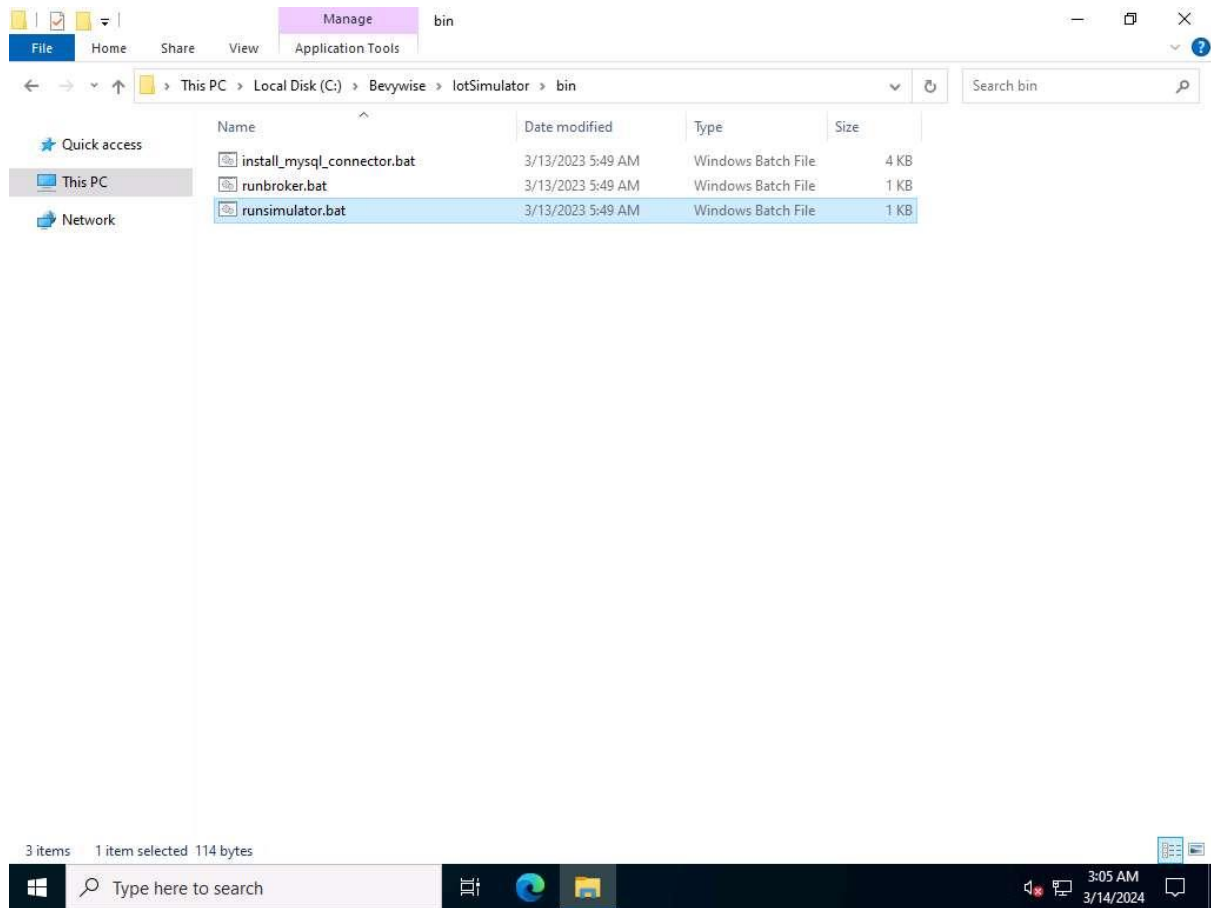
If the network screen appears, click **Yes**.
10. Navigate to **Z:\CEHv13 Module 18 IoT and OT Hacking\Bevywise IoT Simulator** folder and double-click on the **Bevywise_IoTSimulator_3.exe** file.



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**.



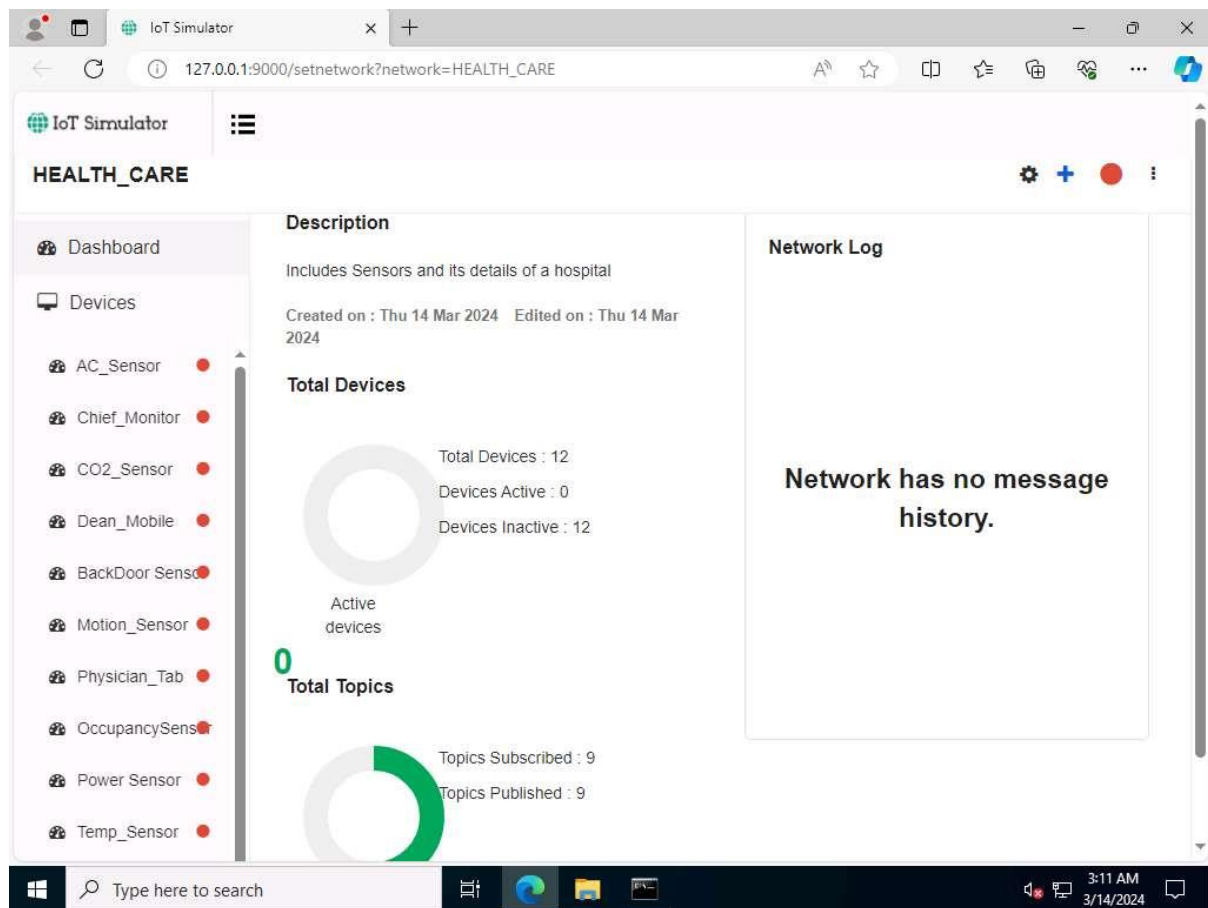
14. After restarting, Bevywise IoT Simulator is installed successfully. To launch the **IoT simulator**, navigate to the **C:\Bevywise\IoT Simulator\bin** directory and double-click on the **runsimulator.bat** file.



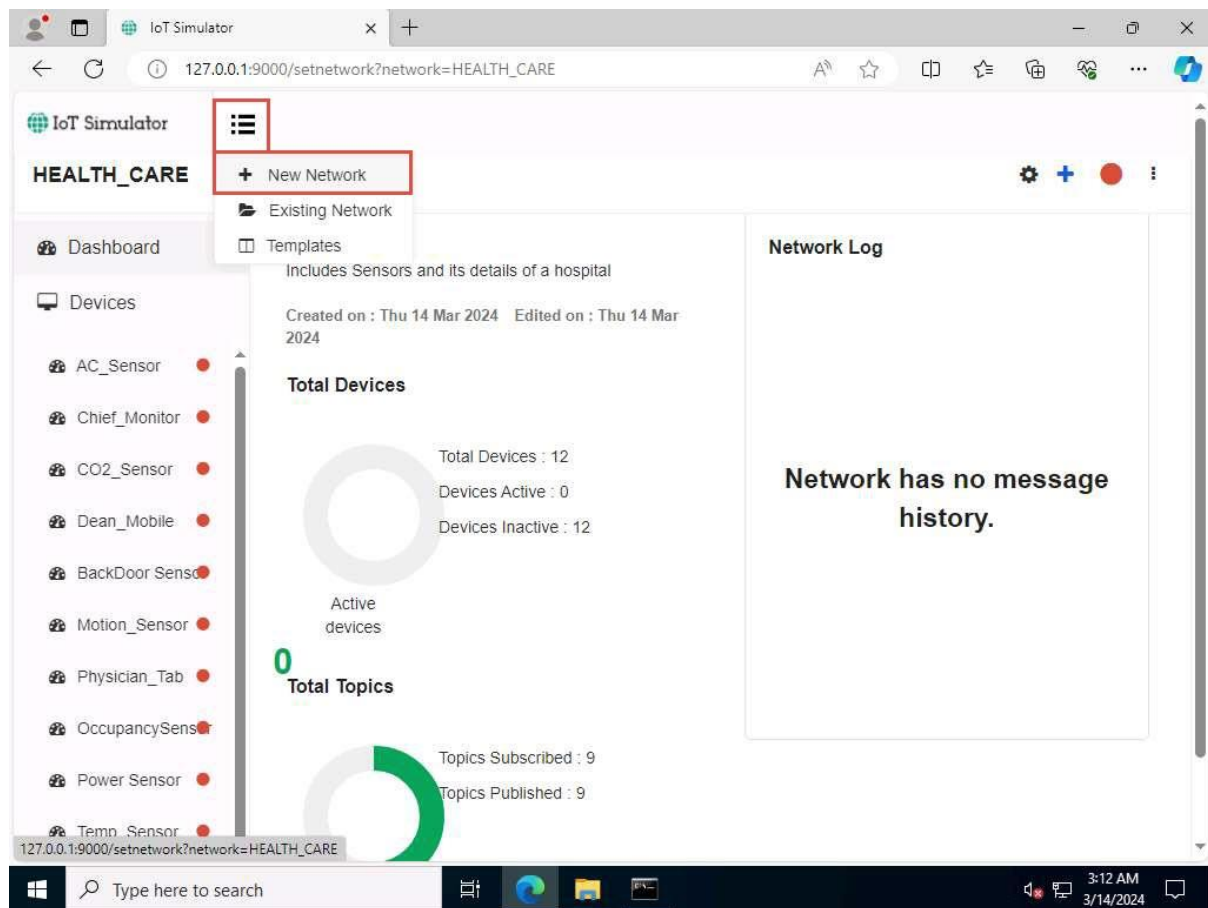
15. Upon double-clicking the **runsimulator.bat** file opens in the command prompt. If **How do you want to open this?** pop-up appears, select **Microsoft Edge** browser and click **OK** to open the URL **`http://127.0.0.1:9000/setnetwork?network=HEALTH_CARE`**.

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

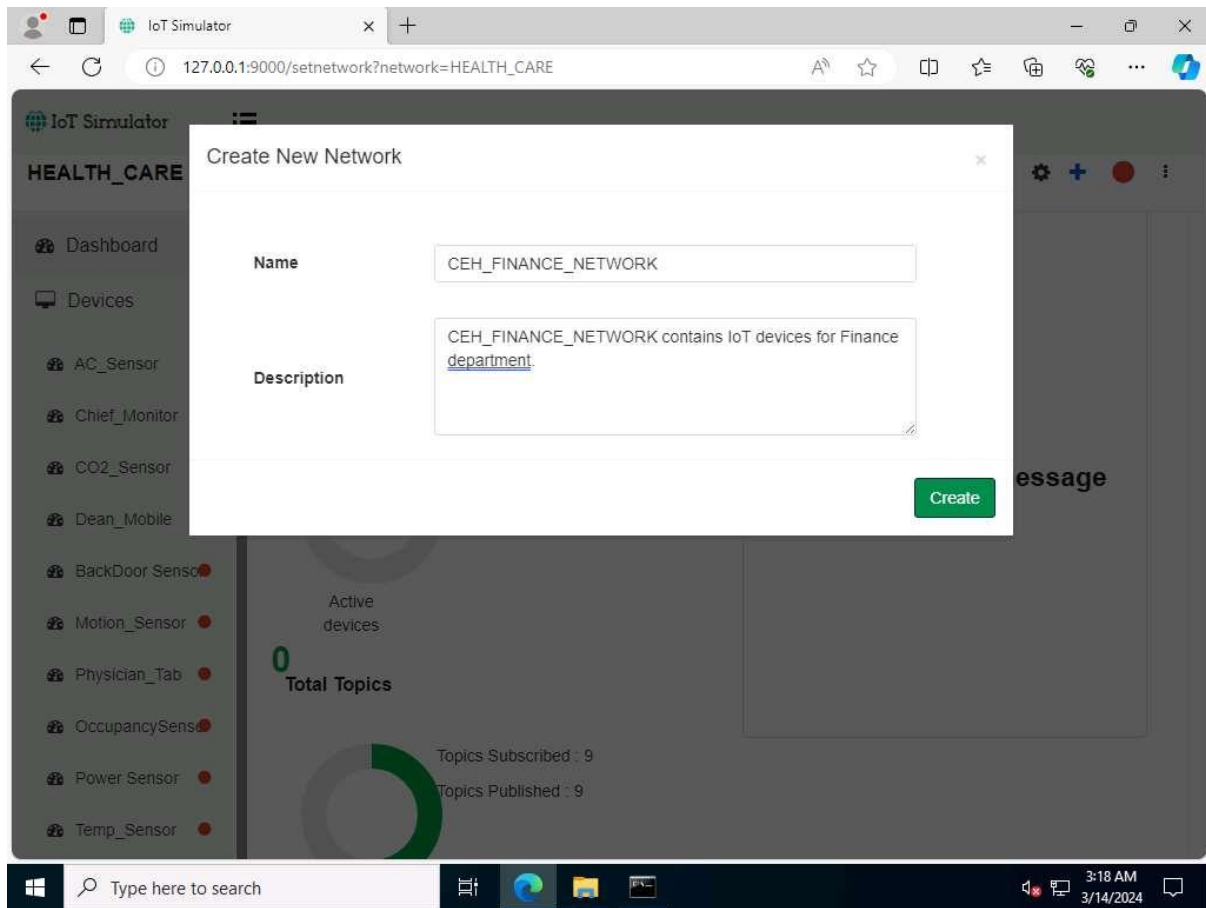
16. The web interface of the IoT Simulator opens in Edge browser. In the IoT Simulator, you can view the default network named **HEALTH_CARE** and several devices.



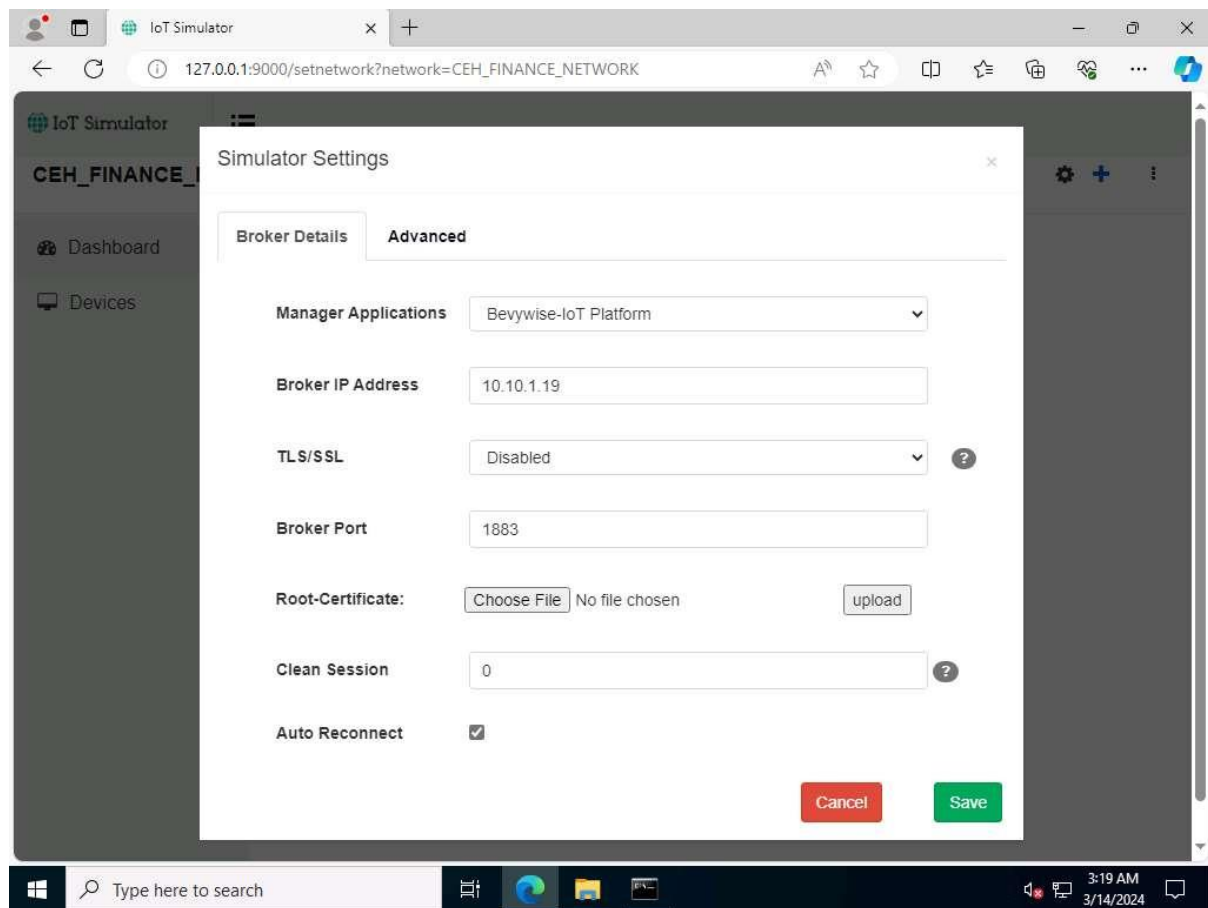
17. Next, we will create a **virtual IoT network** and **virtual IoT devices**. Click on the **menu** icon and select the **+New Network** option.



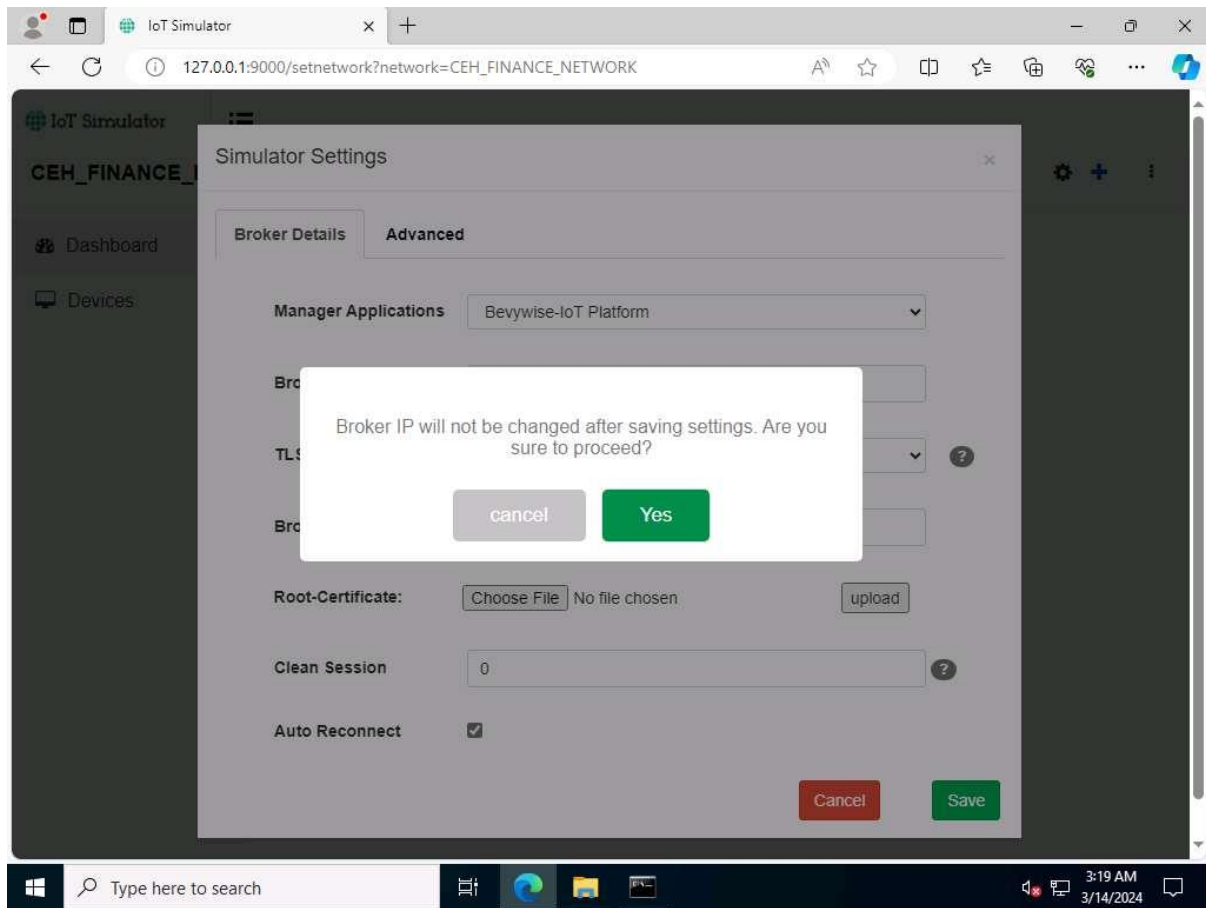
18. The **Create New Network** popup appears. Type any name (here, **CEH_FINANCE_NETWORK**) and description. Click on **Create**.



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**.

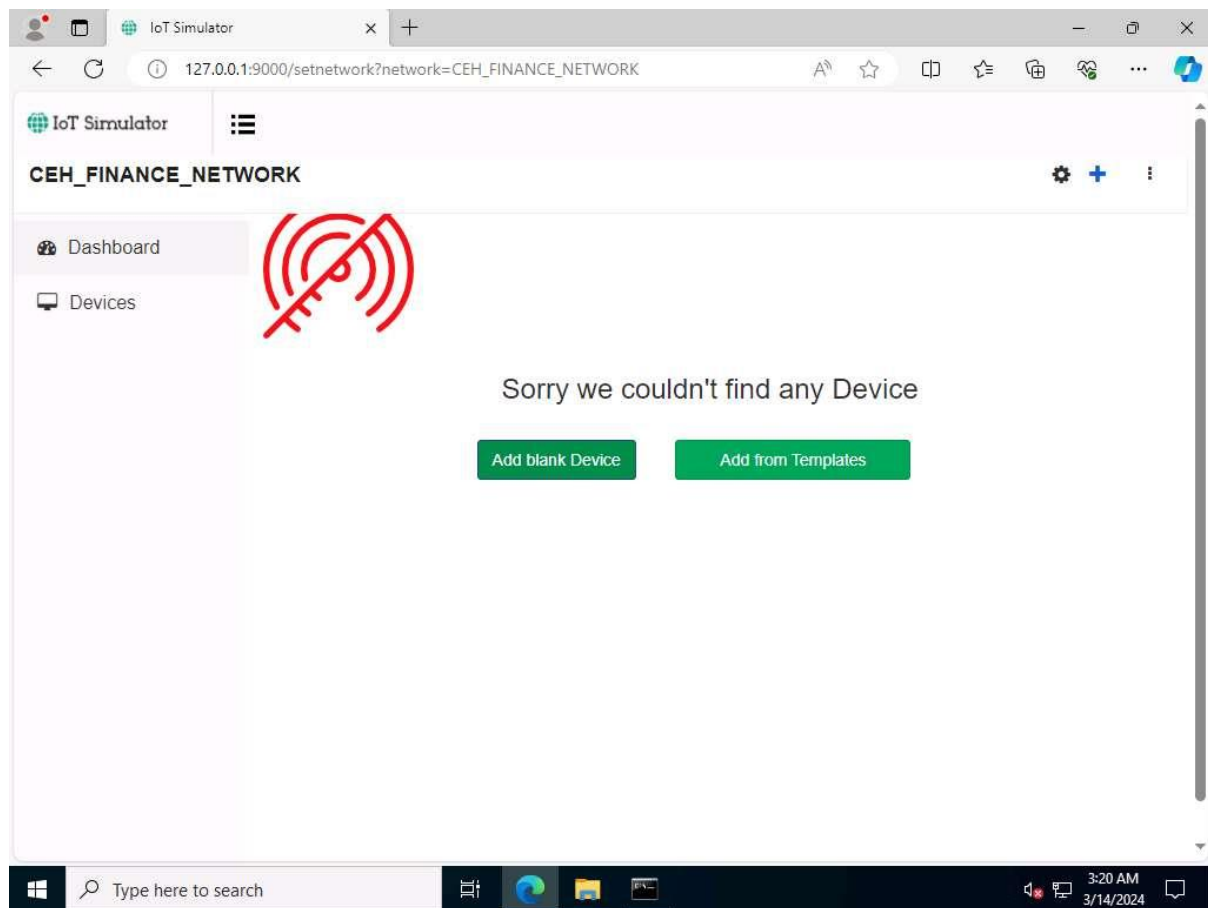


20. To proceed with the network creation, click on **Yes**.

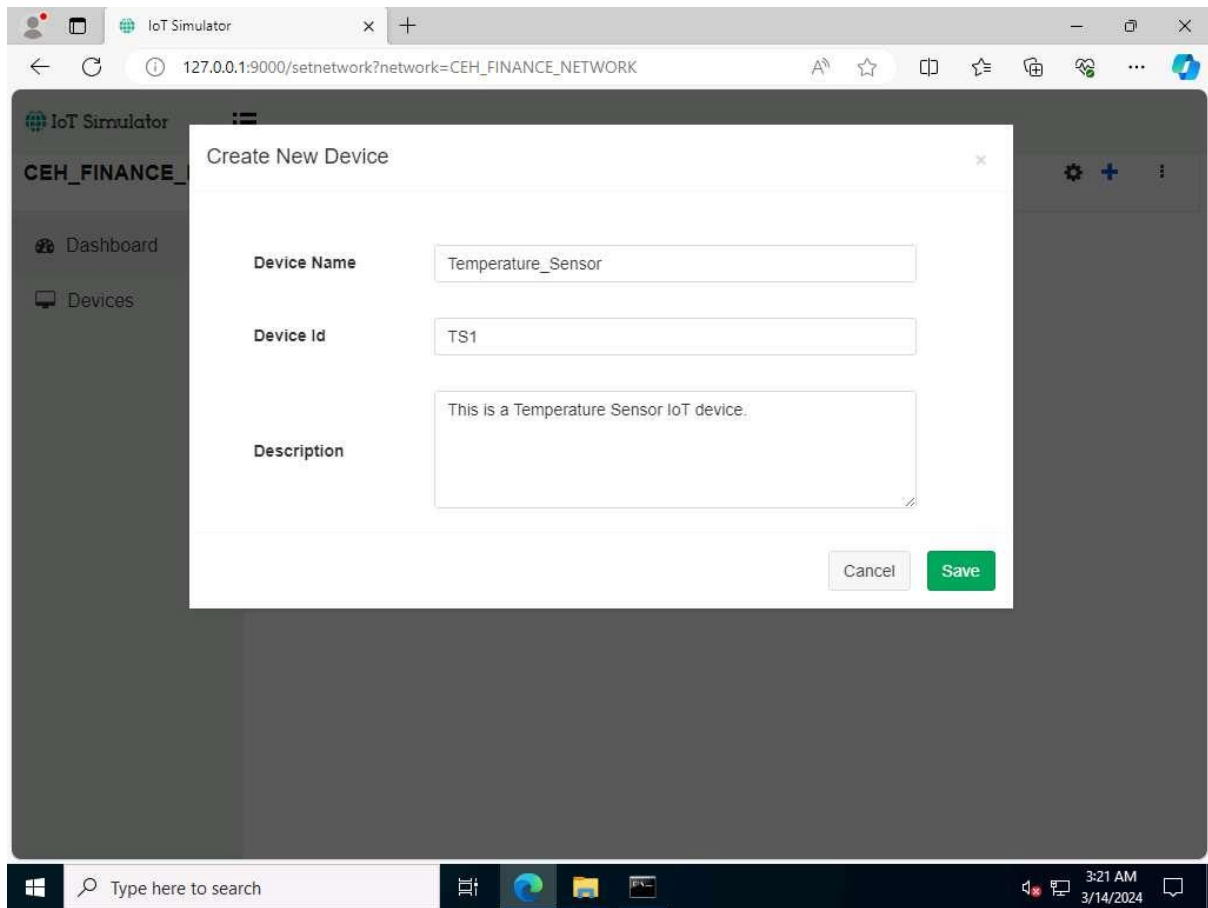


If **Configuration Saved** pop-up appears. Click on **OK** to continue. This step completes the creation of the virtual IoT network.

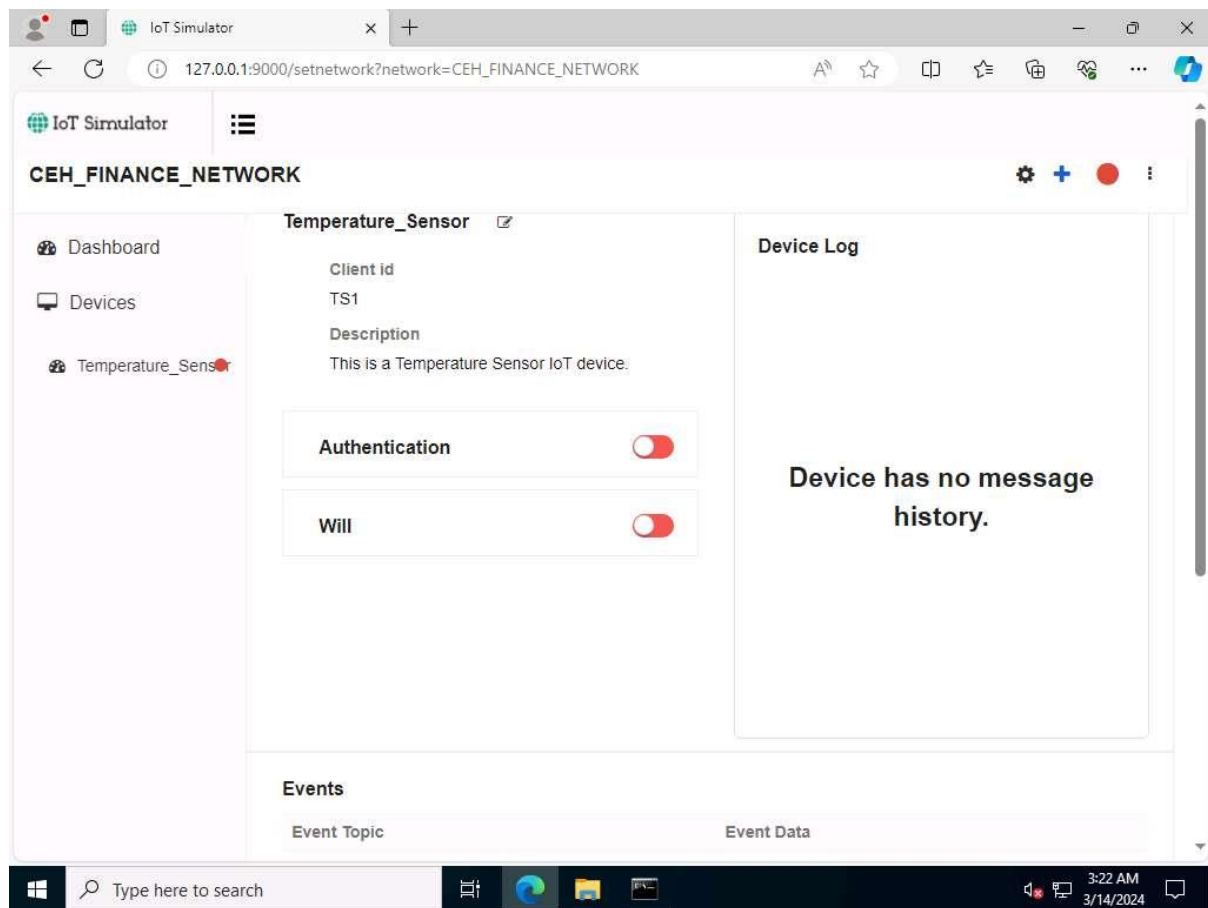
21. To add IoT devices to the created network, click on the **Add blank Device** button.



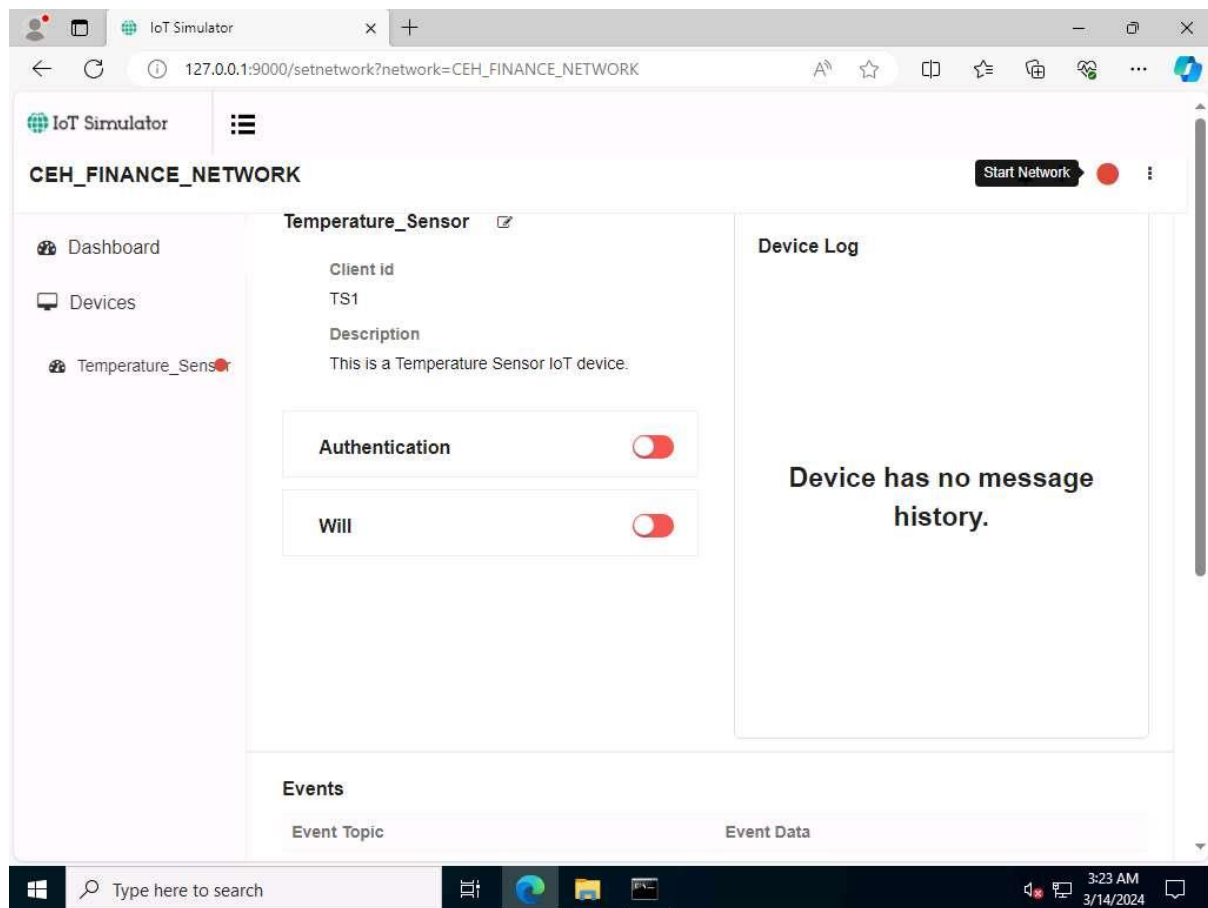
22. The **Create New Device** popup opens. Type the device name (here, we use **Temperature_Sensor**), enter Device Id (here, we use **TS1**), provide a **Description** and click on **Save**.



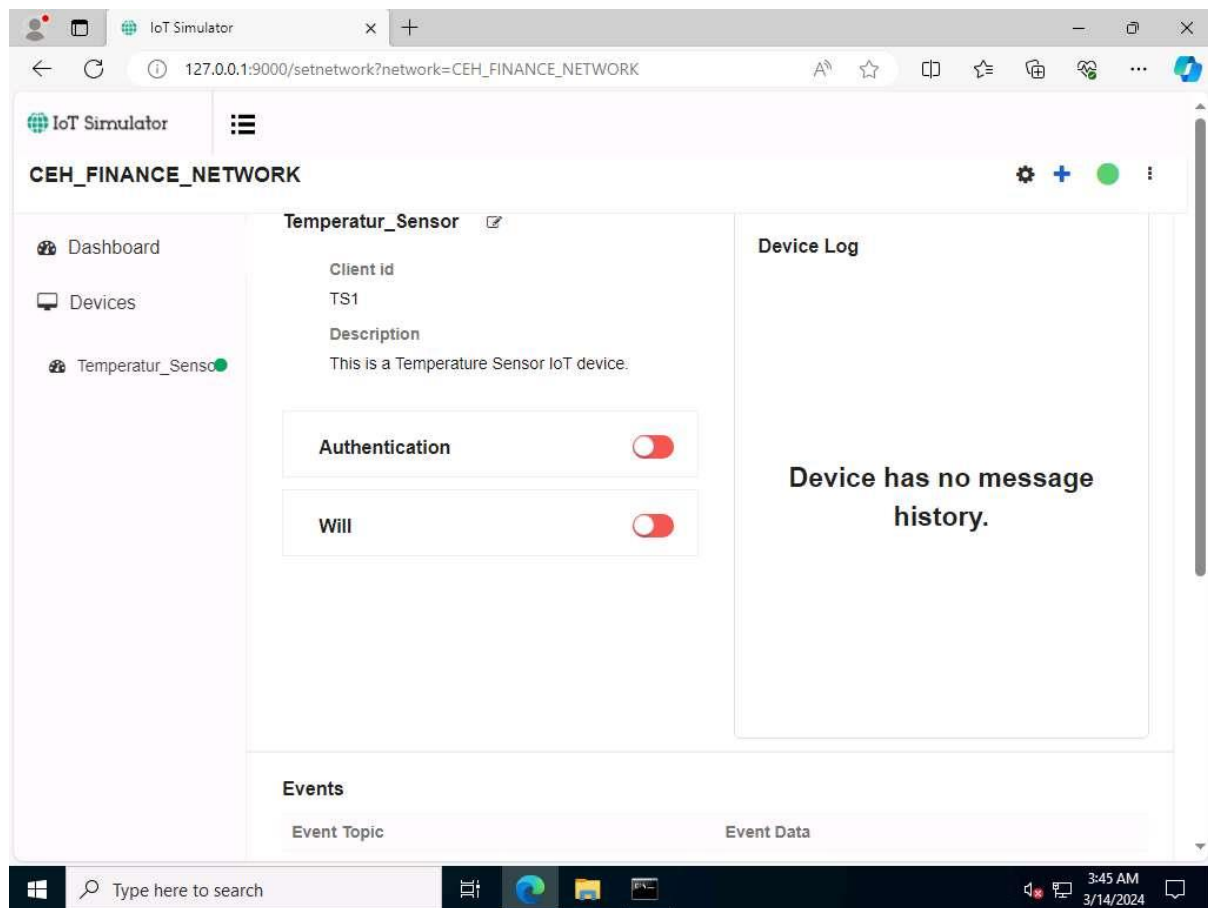
23. The device will be added to the **CEH_FINANCE_NETWORK**.



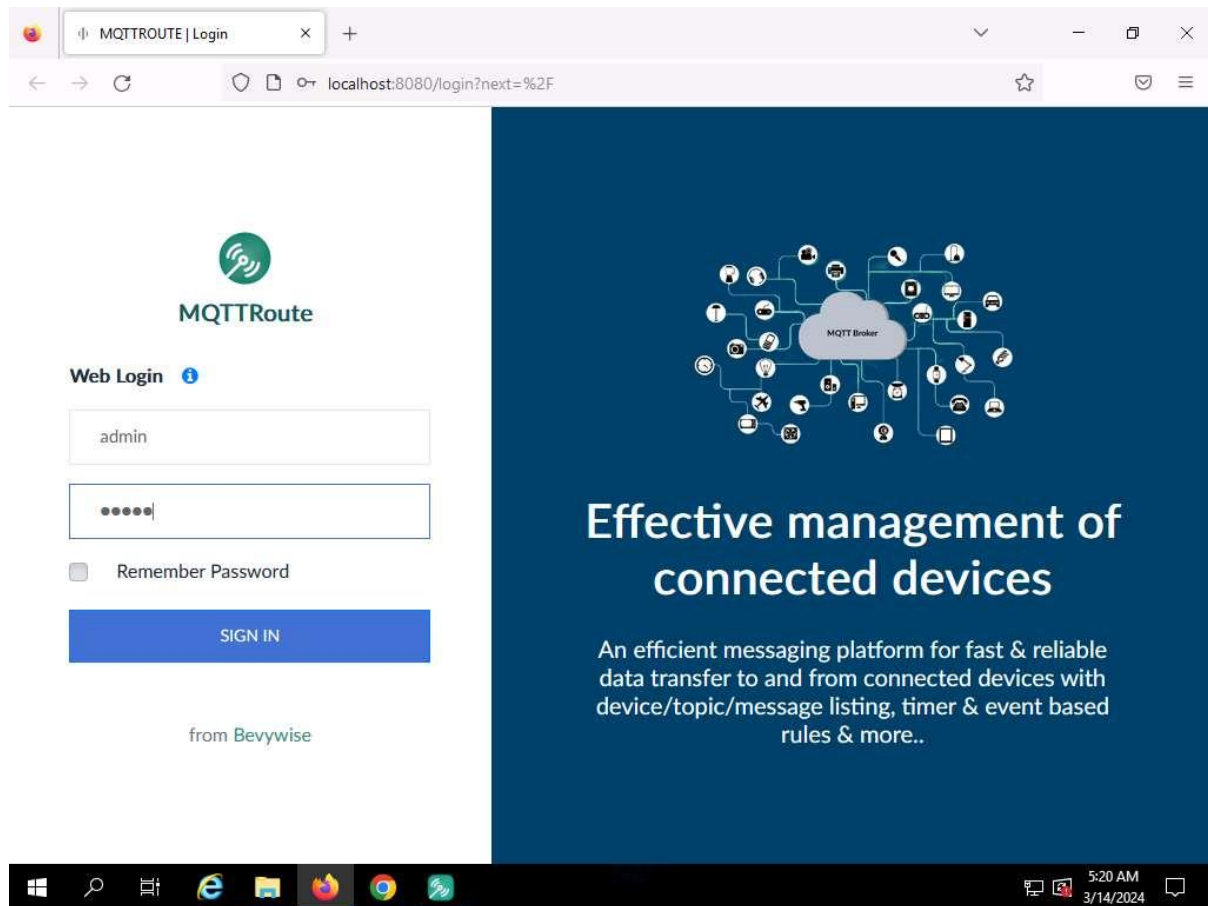
24. To connect the Network and the added devices to the server or Broker, click on the **Start Network** red color circular icon in right corner.



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.

MQTTRoute

localhost:8080

Active Devices
1

Total Devices
1

Events
0

Commands
0

Recent Events

| Device Id | Topic | Message | Time |
|---------------|-------|---------|------|
| No Data Found | | | |

Recent Device Log

| Device Id | IP | Status | Time |
|---------------|----|--------|------|
| No Data Found | | | |

Recent Connections

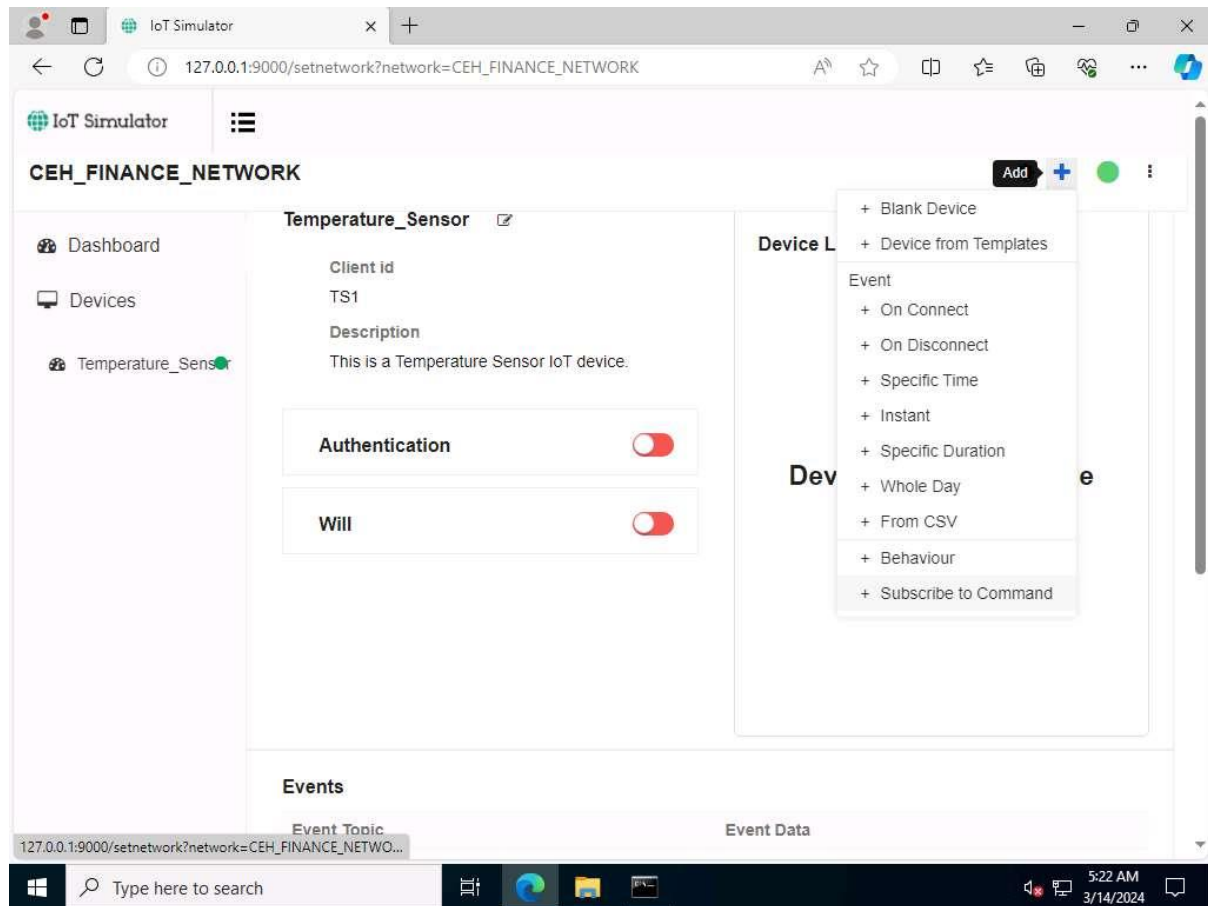
| Device Id | IP | Time |
|-----------|------------|----------------|
| TS1 | 10.10.1.22 | Today 05:15:32 |

Recent Disconnections

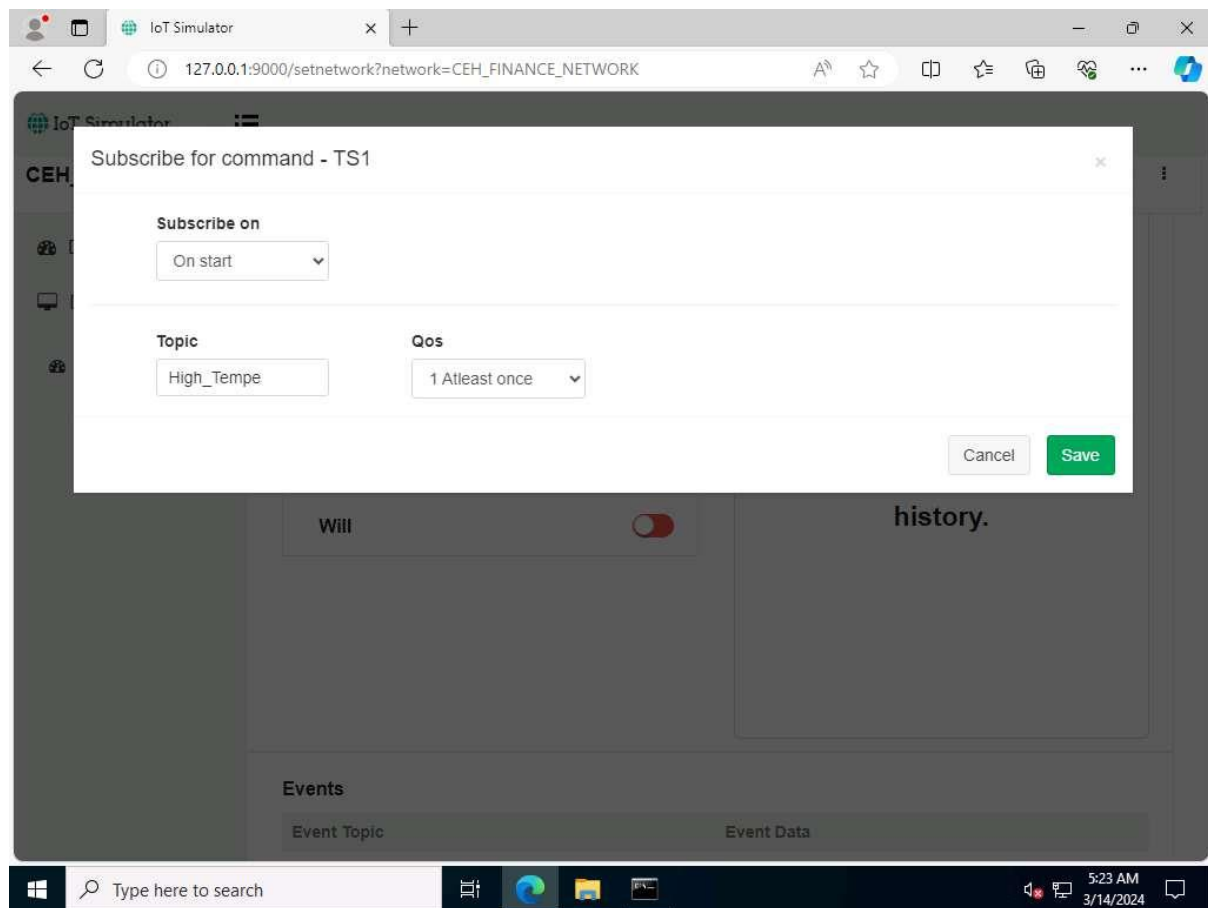
| Device Id | IP | Time |
|---------------|----|------|
| No Data Found | | |

Windows taskbar at the bottom shows the time as 5:21 AM on 3/14/2024.

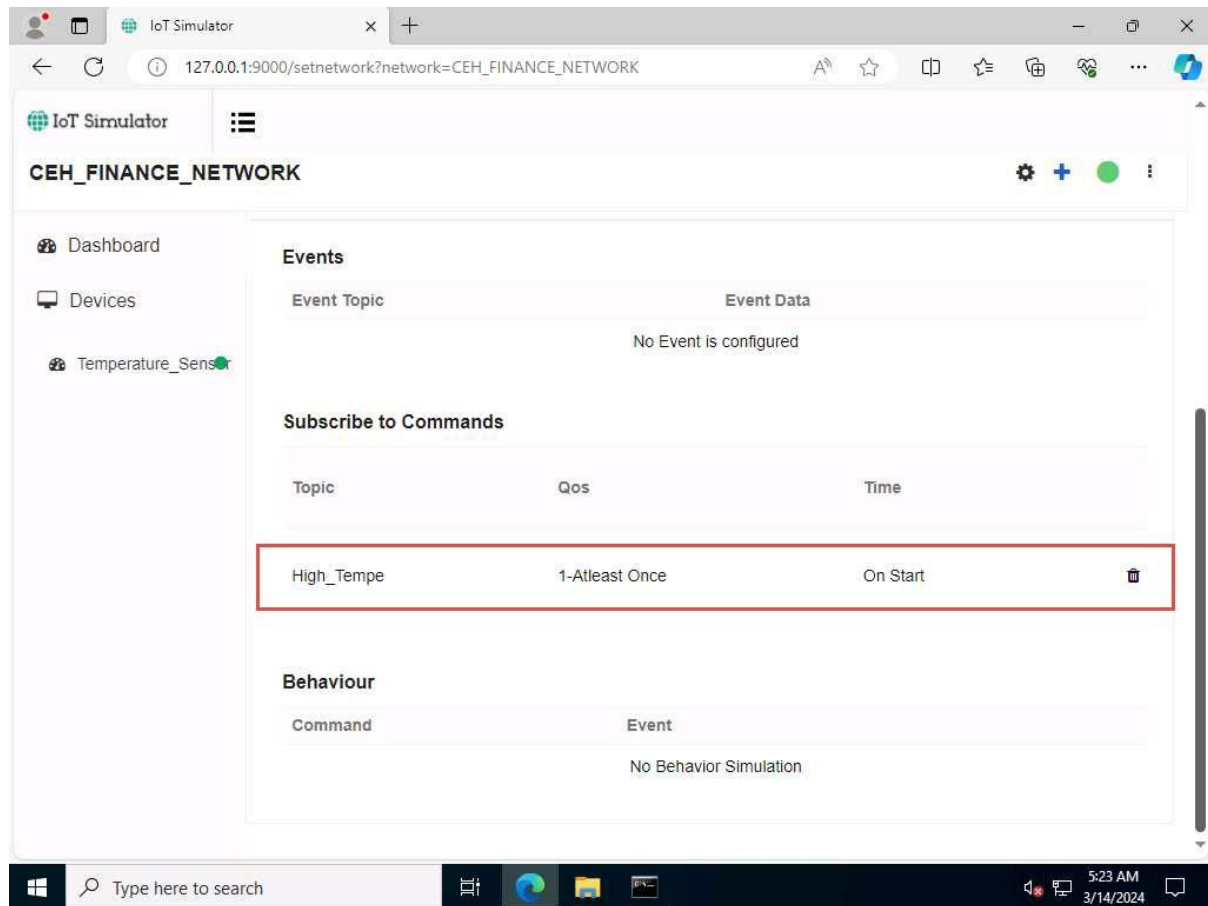
28. Switch back to Windows Server 2022 machine.
29. Next, we will create the **Subscribe command** for the device Temperature_Sensor.
30. Click on the **Plus** icon in **the top right corner** and select the **Subscribe to Command** option.



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.

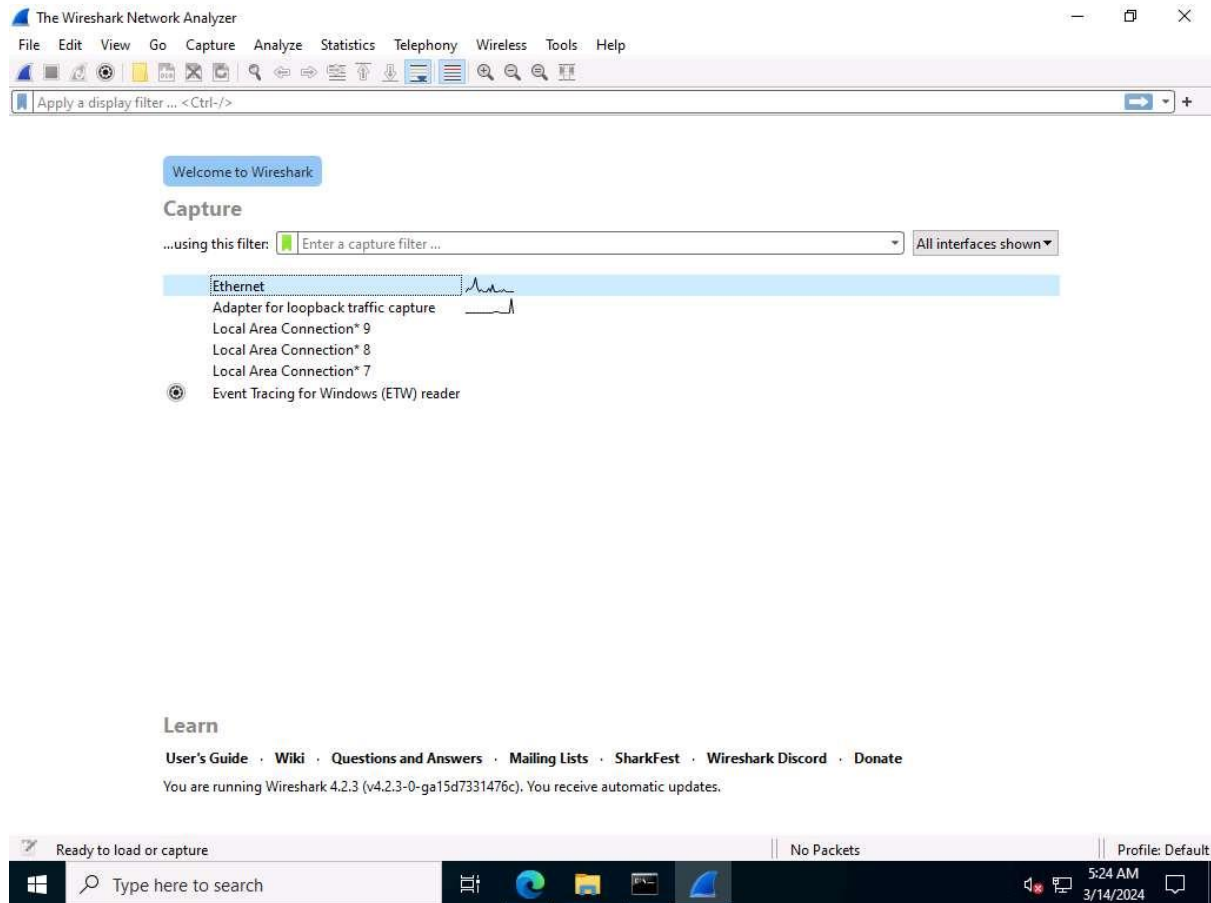


33. Next, we will capture the traffic between the **virtual IoT network and the MQTT Broker** to monitor the secure communication.

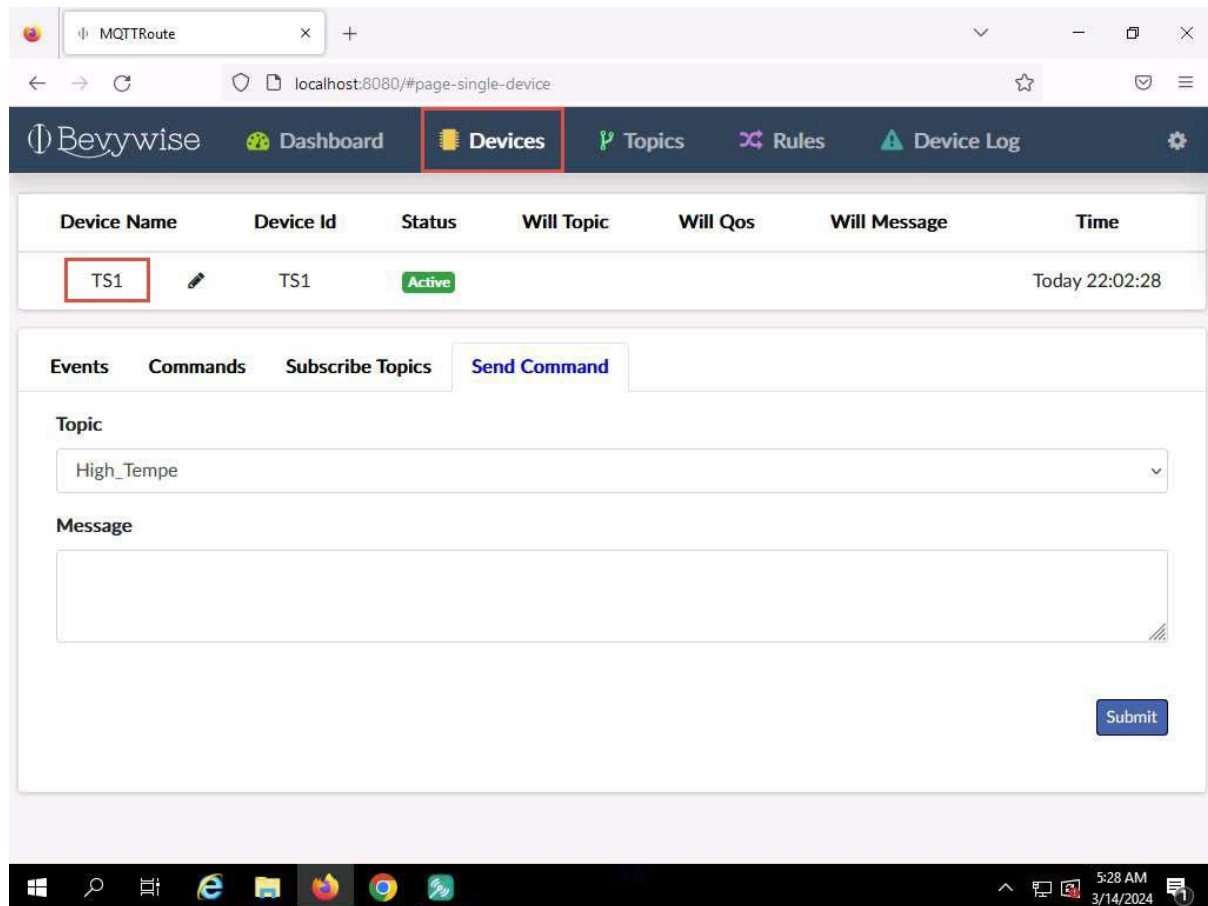
34. Minimise the Edge browser. Click **Type here to search** field on the **Desktop**, search for **wireshark** in the search bar and select **Wireshark** from the results.

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**.



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**.



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.

MQTTRoute

localhost:8080/#page-single-device

Beywise Dashboard Devices Topics Rules Device Log

| Device Name | Device Id | Status | Will Topic | Will Qos | Will Message | Time |
|-------------|-----------|--------|------------|----------|--------------|----------------|
| TS1 | TS1 | Active | | | | Today 22:02:28 |

Events Commands Subscribe Topics Send Command

Topic

High_Tempe

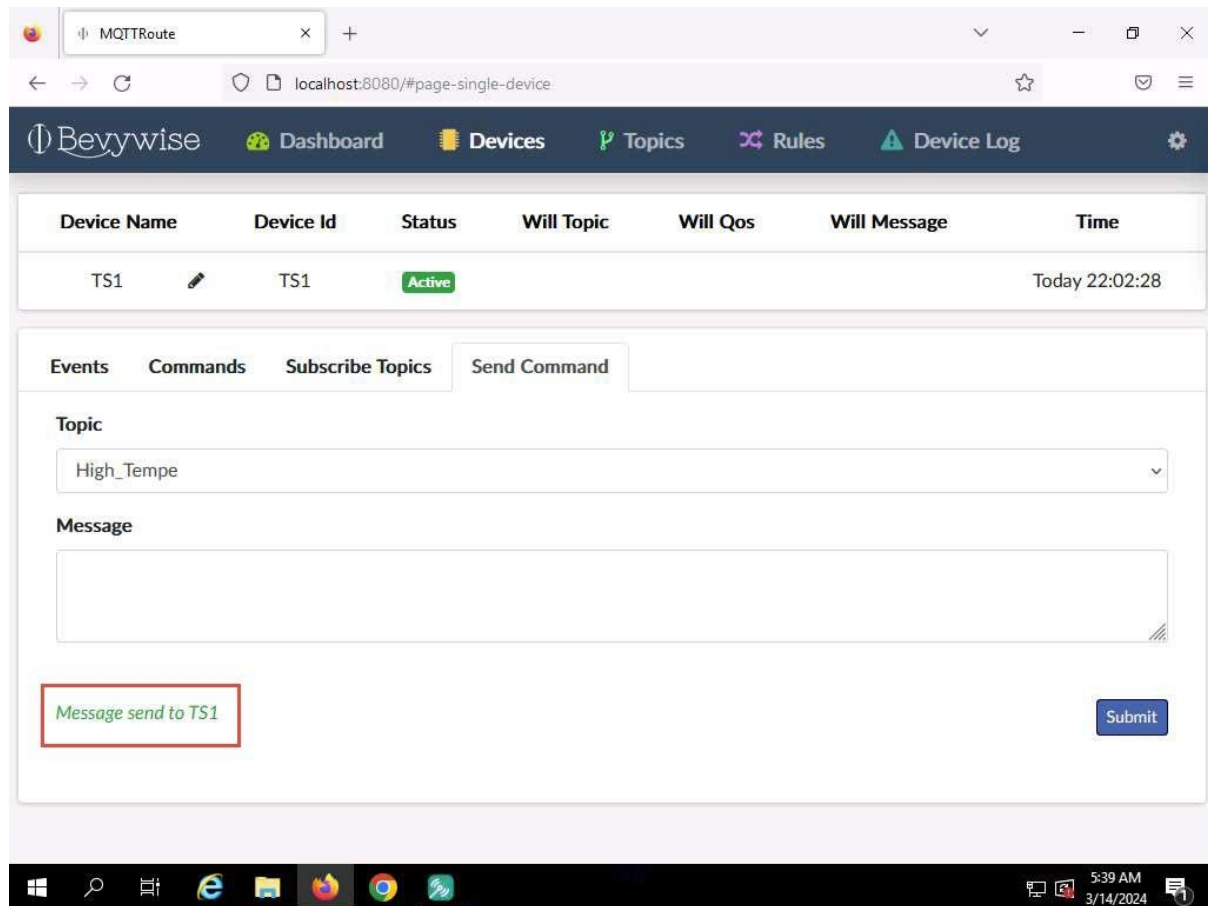
Message

Alert for High Temperature.

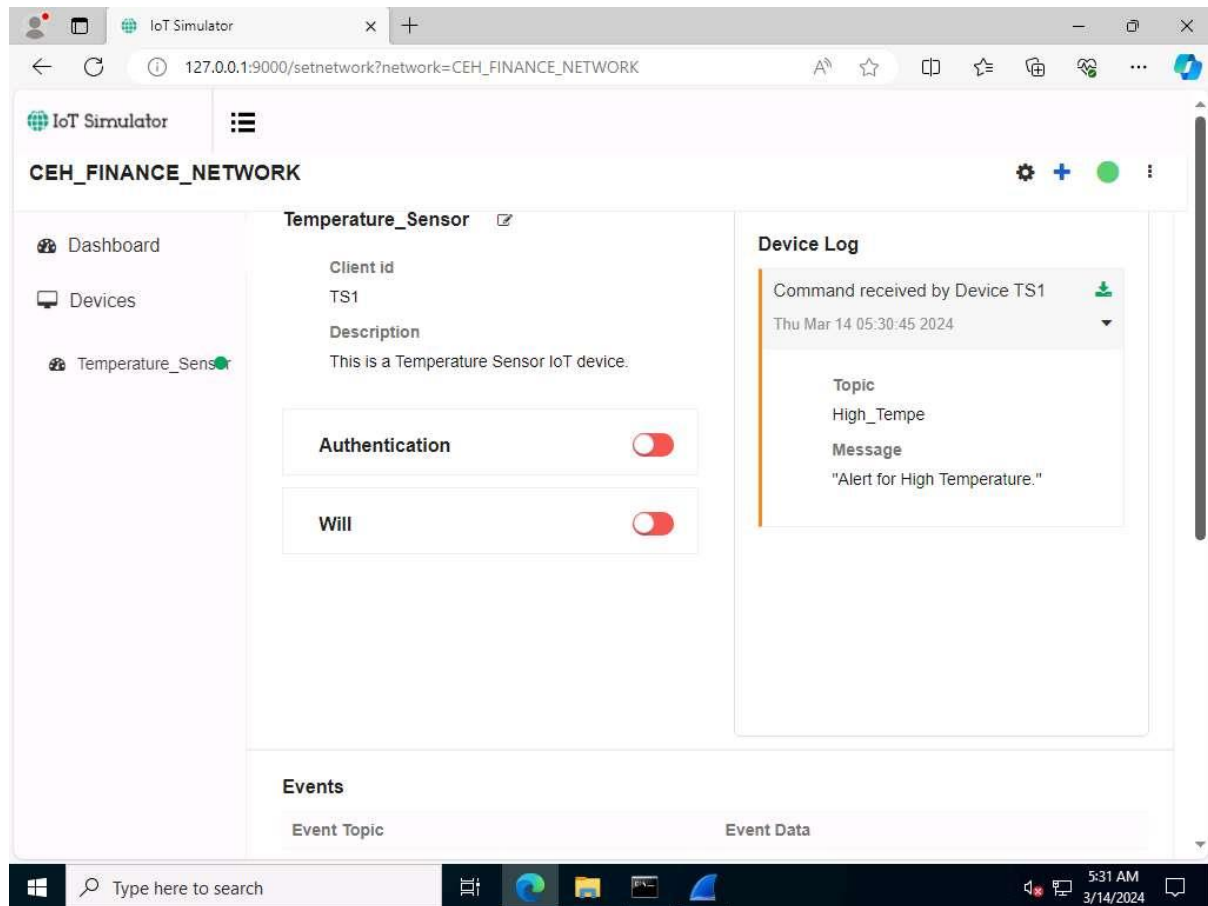
Submit

5:30 AM 3/14/2024

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



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**"



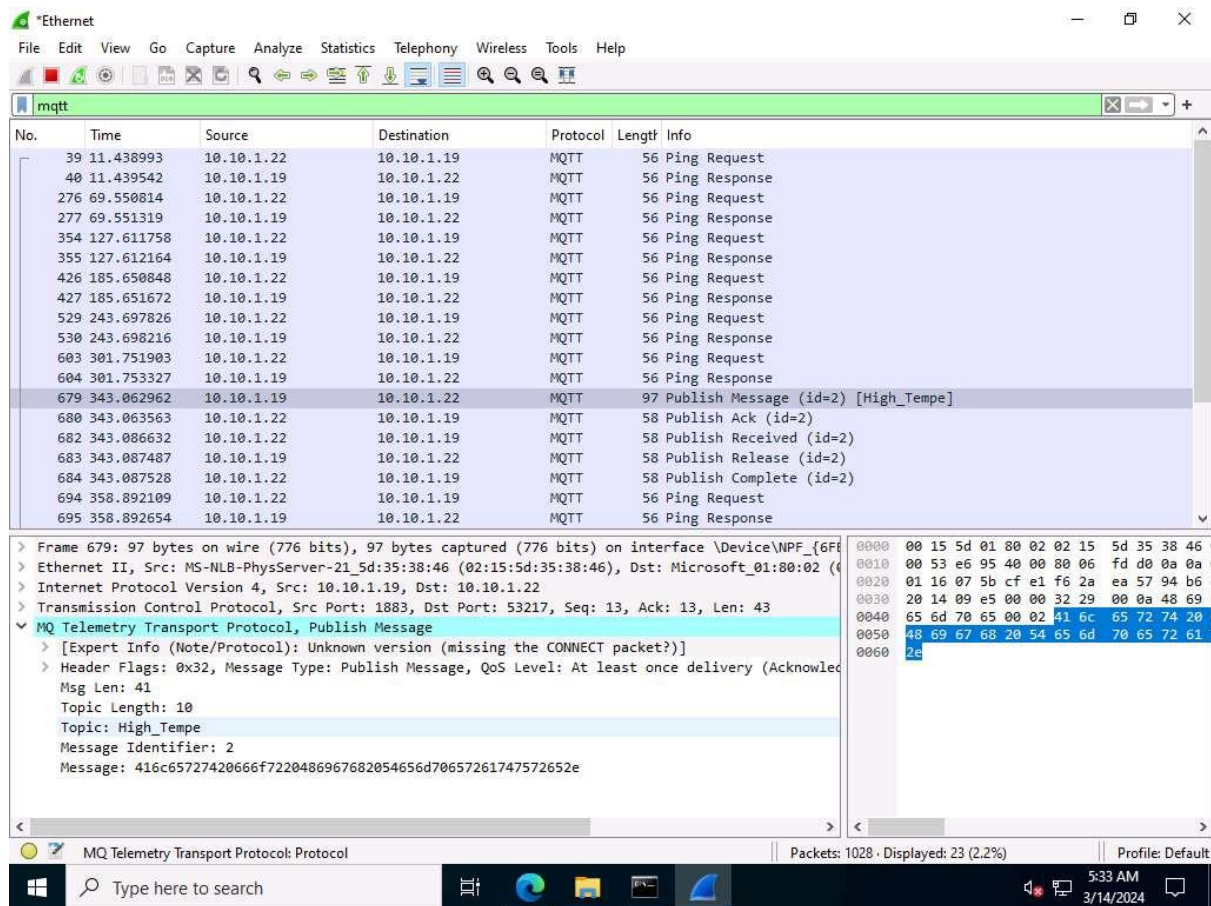
46. To verify the communication, we have executed **Wireshark** application, switch to the Wireshark traffic capturing window.
47. Type **mqtt** under the **filter** field and press **Enter**. To display only the MQTT protocol packets.

The screenshot displays a Wireshark capture of network traffic on the 'Ethernet' interface. The filter is set to 'mqtt'. The packet list shows a series of MQTT messages between 10.10.1.22 and 10.10.1.19. The packet details pane for packet 39 (MQTT Ping Request) is expanded, showing the following structure:

- Frame 39: 56 bytes on wire (448 bits), 56 bytes captured (448 bits) on interface \Device\NPF_{6FB...}
- Ethernet II, Src: Microsoft_01:80:02 (00:15:5d:01:80:02), Dst: MS-NLB-PhysServer-21_5d:35:38:46 (00:15:5d:01:80:02)
- Internet Protocol Version 4, Src: 10.10.1.22, Dst: 10.10.1.19
- Transmission Control Protocol, Src Port: 53217, Dst Port: 1883, Seq: 1, Ack: 1, Len: 2
- MQ Telemetry Transport Protocol, Ping Request

The packet bytes pane shows the raw data for the selected packet, including the MQTT header and the ping request payload.

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.

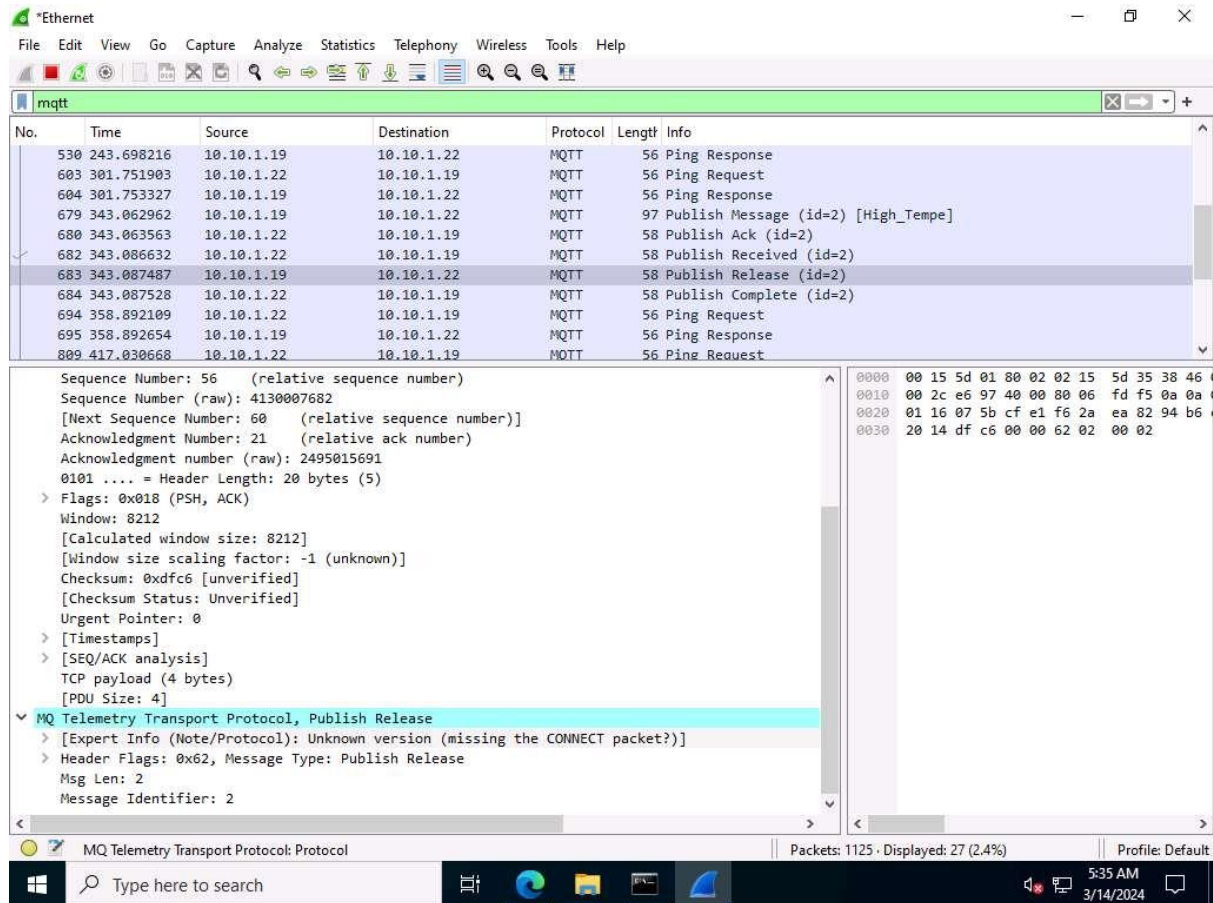


Note: After establishing a successful connection with the MQTT broker, the MQTT client can publish messages. The headers in the Publish Message packet are given below:

- 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.

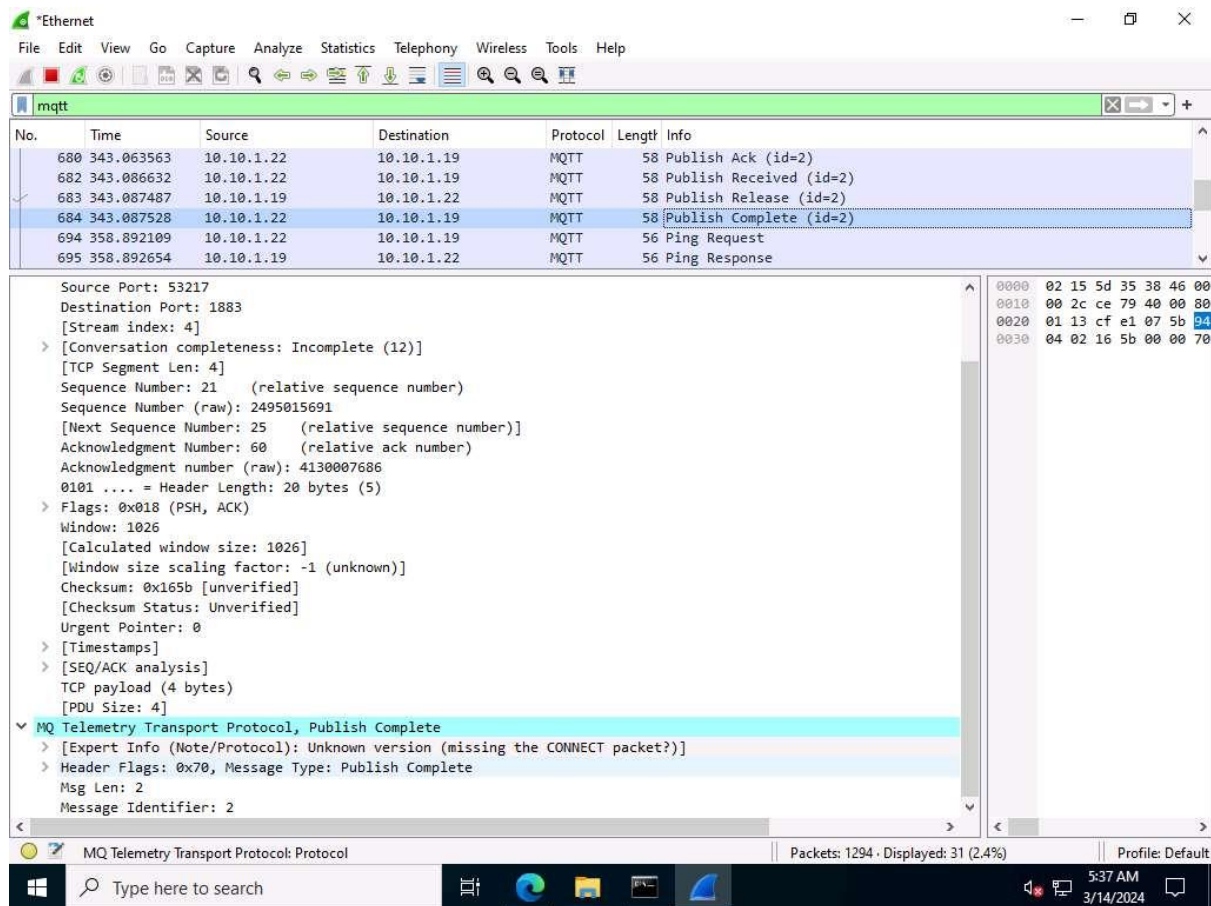
51. Select any **Publish Release** packet from the **Packet List** pane. In the **Packet Details** pane at the middle of the window, expand the **Transmission Control Protocol**, **MQ Telemetry Transport Protocol**, and **Header Flags** nodes.

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



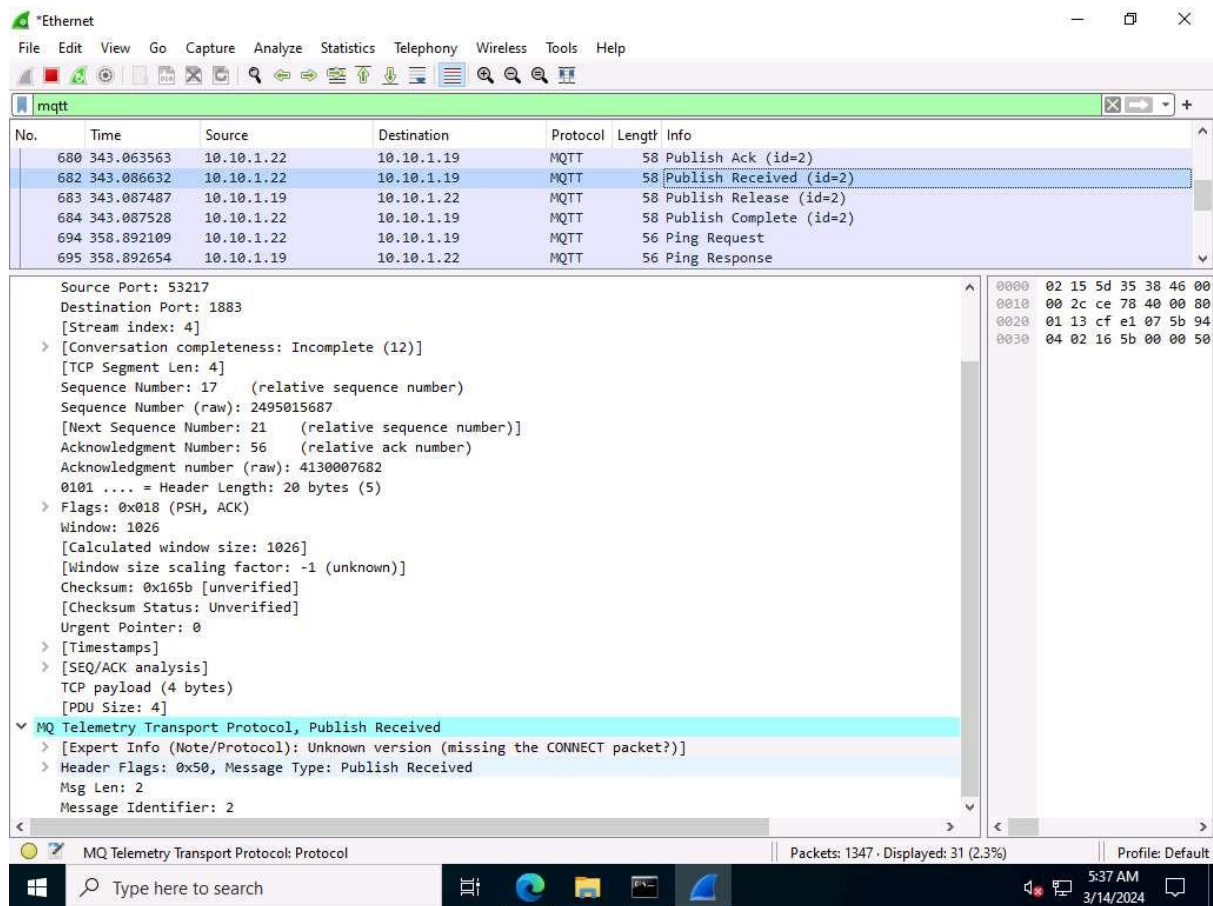
Note: A Publish Release (PUBREL) packet is the response to a Publish Received (PUBREC) packet.

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?

Lab 3: Perform IoT Attacks

Lab Scenario

As an ethical hacker or penetration tester, you must have sound knowledge in implementing various techniques to exploit vulnerabilities and launch attacks on target IoT devices or networks.

Potential vulnerabilities in the IoT system can result in major problems for organizations. Most IoT devices come with security issues such as the absence of a proper authentication mechanism or the use of default credentials, absence of a lock-out mechanism, absence of a strong encryption scheme, absence of proper key management systems, and improper physical security.

Lab Objectives

- Perform replay attack on CAN protocol

Overview of IoT Attacks

Owing to the significant growth of the paradigm of the IoT, an increasing number of devices are entering our lives every day. From the automation of homes to healthcare applications, the IoT is everywhere. However, despite the ability of IoT devices to make our lives easier and more comfortable, we cannot underestimate the risk of cyber-attacks. IoT devices lack basic security, thus making them prone to various types of cyber-attacks.

Task 1: Perform Replay Attack on CAN Protocol

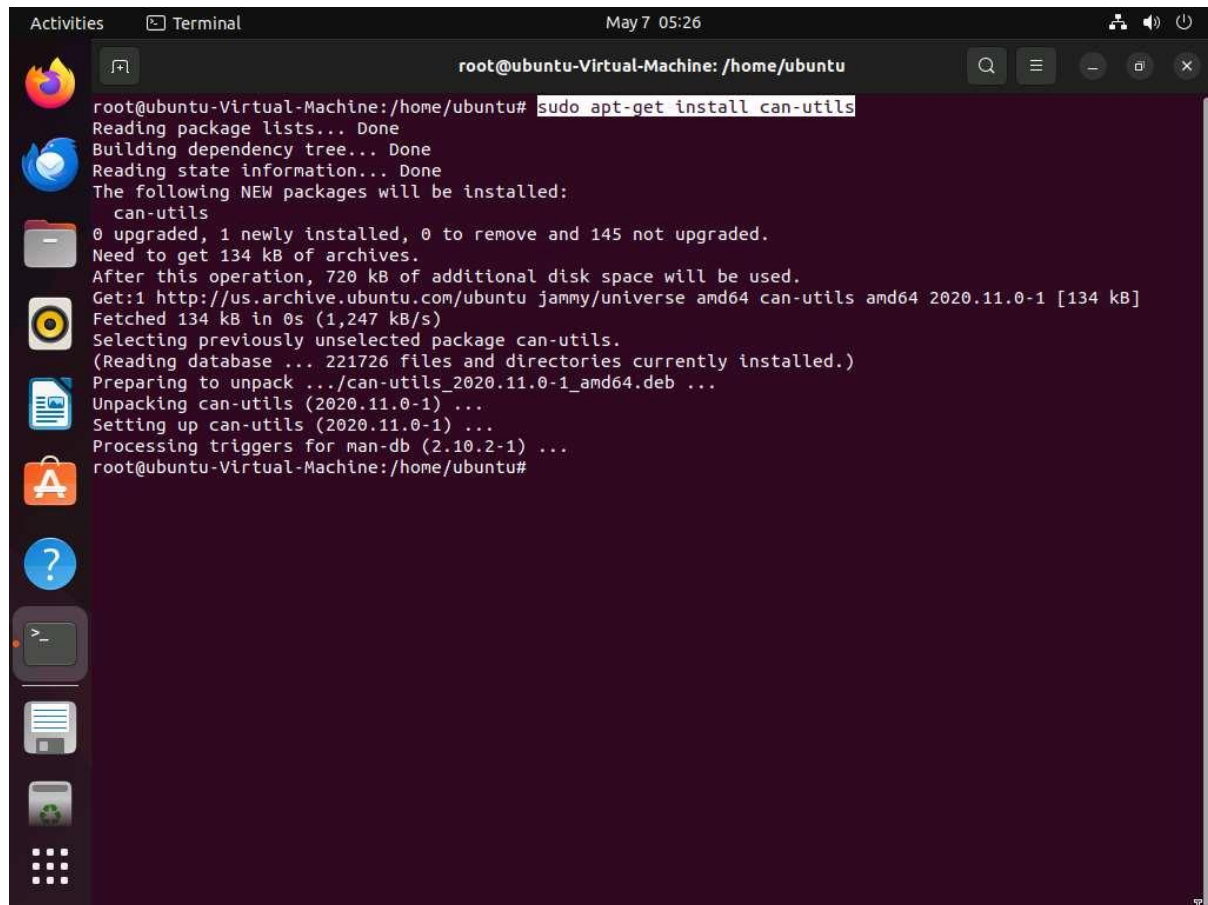
The Controller Area Network (CAN) protocol is a robust communication system that allows microcontrollers and devices to interact without a central computer. It uses a message-based approach for reliable data exchange, even in noisy environments. CAN is widely used in automotive industry due to its reliability and simplicity. In modern vehicles, CAN protocol is central to system communication, enabling connections between engine controls, brakes, and infotainment units. However, this interconnectivity can be exploited by hackers to manipulate vehicle functions, posing safety risks.

Here, we are using the ICSim tool to simulate CAN protocol and demonstrate how attackers sniff the transmitted packets and perform replay attack to gain basic control over the target.

1. Click [Ubuntu](#) to switch to the **Ubuntu** machine and login with **Ubuntu/toor**.
2. In the **Ubuntu** machine, open a **Terminal** window and execute **sudo su** to run the programs as a root user (When prompted, enter the password **toor**).

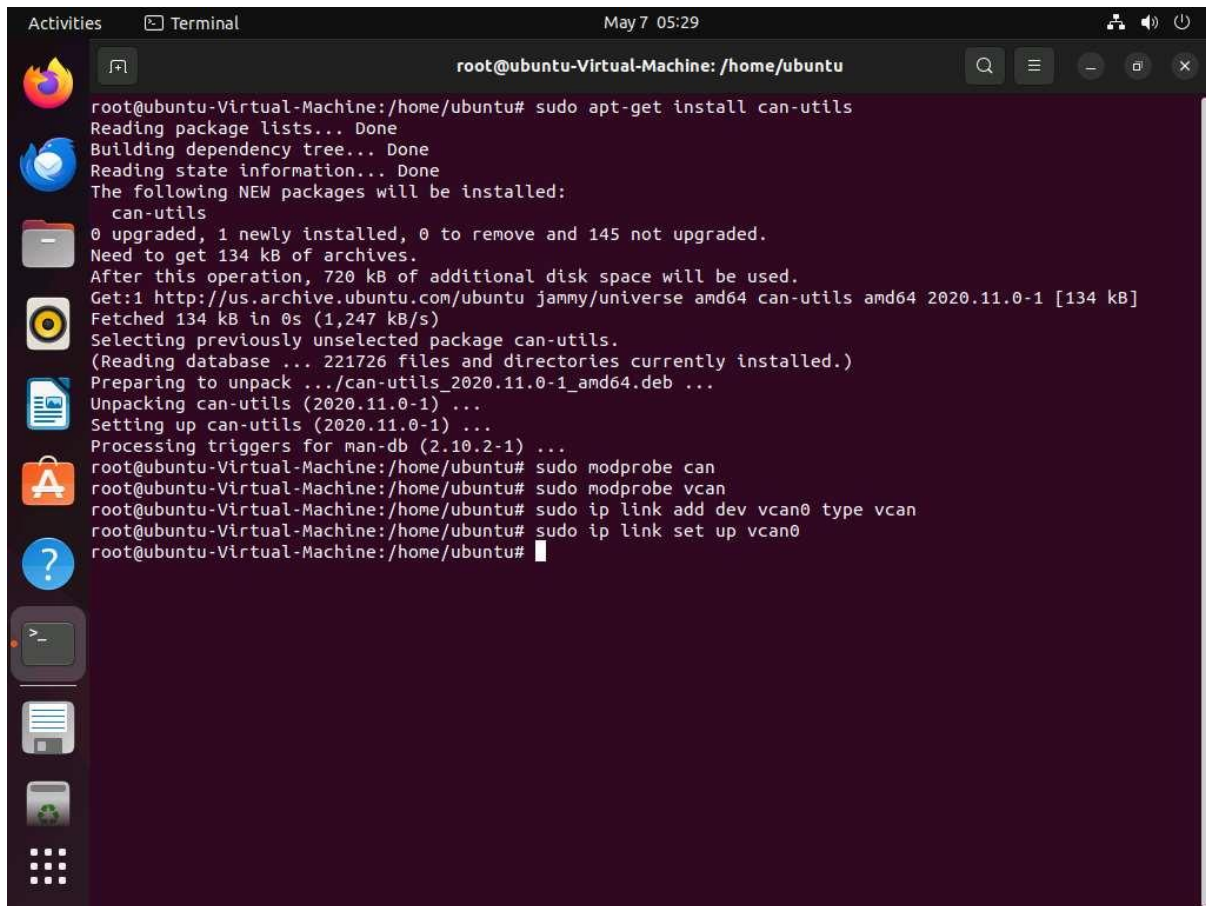
3. Run **sudo apt-get install can-utils** to install CAN utility

While installing if prompted **Do you want to continue?**, type **Y** and press **Enter**.

A screenshot of a terminal window titled 'Terminal' with a timestamp of 'May 7 05:26'. The terminal shows the command 'sudo apt-get install can-utils' being executed. The output indicates that the package 'can-utils' is being installed, with details about disk space requirements and the source of the package. The terminal prompt is 'root@ubuntu-Virtual-Machine: /home/ubuntu#'.

```
root@ubuntu-Virtual-Machine:/home/ubuntu# sudo apt-get install can-utils
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  can-utils
0 upgraded, 1 newly installed, 0 to remove and 145 not upgraded.
Need to get 134 kB of archives.
After this operation, 720 kB of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu jammy/universe amd64 can-utils amd64 2020.11.0-1 [134 kB]
Fetched 134 kB in 0s (1,247 kB/s)
Selecting previously unselected package can-utils.
(Reading database ... 221726 files and directories currently installed.)
Preparing to unpack .../can-utils_2020.11.0-1_amd64.deb ...
Unpacking can-utils (2020.11.0-1) ...
Setting up can-utils (2020.11.0-1) ...
Processing triggers for man-db (2.10.2-1) ...
root@ubuntu-Virtual-Machine:/home/ubuntu#
```

4. Now, to setup a virtual CAN interface issue following commands:
 - **sudo modprobe can**
 - **sudo modprobe vcan**
 - **sudo ip link add dev vcan0 type vcan**
 - **sudo ip link set up vcan0**



```
root@ubuntu-Virtual-Machine: /home/ubuntu# sudo apt-get install can-utils
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  can-utils
0 upgraded, 1 newly installed, 0 to remove and 145 not upgraded.
Need to get 134 kB of archives.
After this operation, 720 kB of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu jammy/universe amd64 can-utils amd64 2020.11.0-1 [134 kB]
Fetched 134 kB in 0s (1,247 kB/s)
Selecting previously unselected package can-utils.
(Reading database ... 221726 files and directories currently installed.)
Preparing to unpack .../can-utils_2020.11.0-1_amd64.deb ...
Unpacking can-utils (2020.11.0-1) ...
Setting up can-utils (2020.11.0-1) ...
Processing triggers for man-db (2.10.2-1) ...
root@ubuntu-Virtual-Machine:/home/ubuntu# sudo modprobe can
root@ubuntu-Virtual-Machine:/home/ubuntu# sudo modprobe vcan
root@ubuntu-Virtual-Machine:/home/ubuntu# sudo ip link add dev vcan0 type vcan
root@ubuntu-Virtual-Machine:/home/ubuntu# sudo ip link set up vcan0
root@ubuntu-Virtual-Machine:/home/ubuntu#
```

5. To check whether Virtual CAN interface is setup successfully, run **ifconfig**. Here, **vcan0** interface is present which confirms that our Virtual CAN interface is setup successfully.

```
Activities Terminal May 7 05:30
root@ubuntu-Virtual-Machine: /home/ubuntu

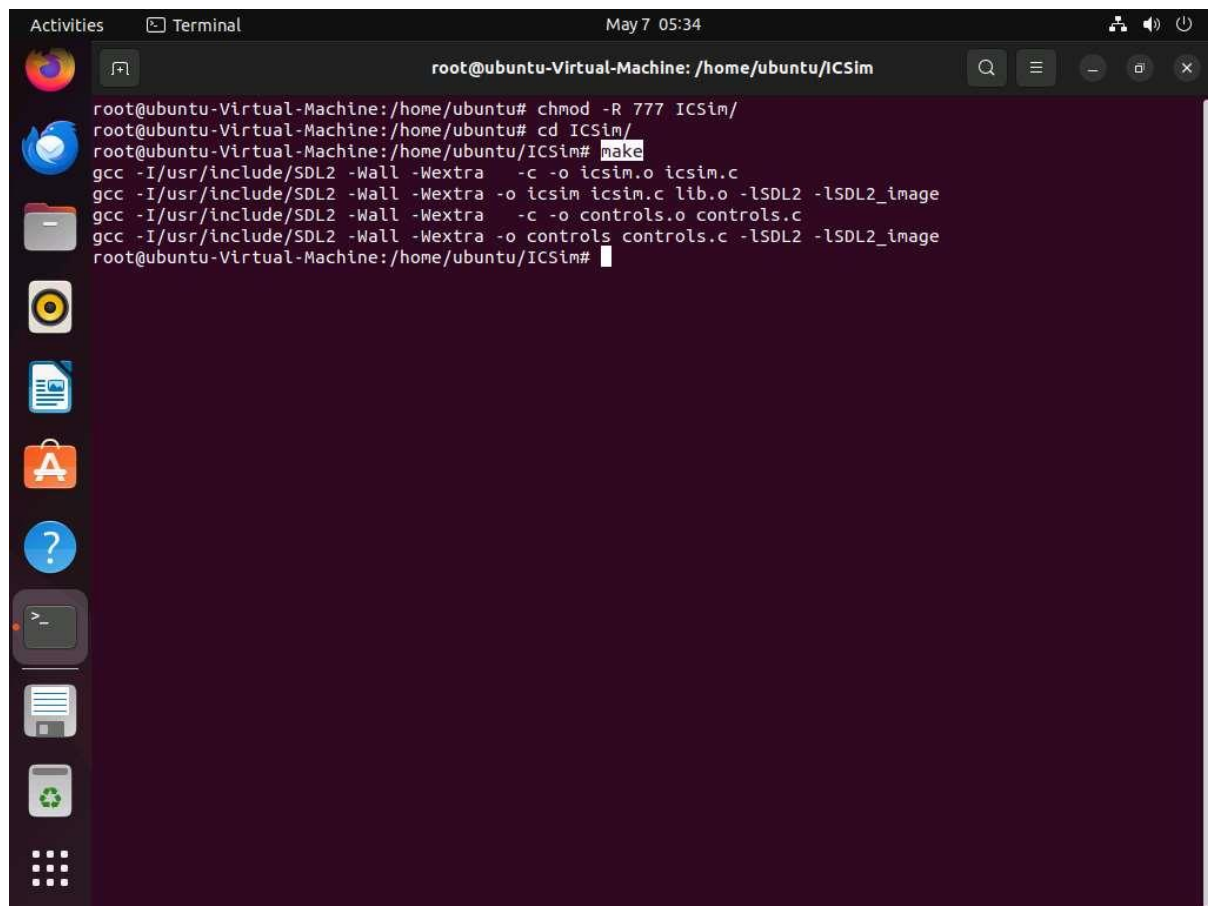
Fetched 134 kB in 0s (1,247 kB/s)
Selecting previously unselected package can-utils.
(Reading database ... 221726 files and directories currently installed.)
Preparing to unpack .../can-utils_2020.11.0-1_amd64.deb ...
Unpacking can-utils (2020.11.0-1) ...
Setting up can-utils (2020.11.0-1) ...
Processing triggers for man-db (2.10.2-1) ...
root@ubuntu-Virtual-Machine:/home/ubuntu# sudo modprobe can
root@ubuntu-Virtual-Machine:/home/ubuntu# sudo modprobe vcan
root@ubuntu-Virtual-Machine:/home/ubuntu# sudo ip link add dev vcan0 type vcan
root@ubuntu-Virtual-Machine:/home/ubuntu# sudo ip link set up vcan0
root@ubuntu-Virtual-Machine:/home/ubuntu# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.10.1.9 netmask 255.255.255.0 broadcast 10.10.1.255
    inet6 fe80::53d4:50b3:593b:c961 prefixlen 64 scopeid 0x20<link>
    ether 02:15:5d:49:c8:70 txqueuelen 1000 (Ethernet)
    RX packets 261127 bytes 391230870 (391.2 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 18442 bytes 1415499 (1.4 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 267 bytes 26937 (26.9 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 267 bytes 26937 (26.9 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

vcan0: flags=193<UP,RUNNING,NOARP> mtu 72
    unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00 txqueuelen 1000 (UNSPEC)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@ubuntu-Virtual-Machine:/home/ubuntu#
```

6. Run **chmod -R 777 ICSim** to give permissions to the ICSim folder.
7. Now, run **cd ICSim** to navigate to ICSim directory and execute **make** command to create two executable files for IC Simulator and CANBus Control Panel.

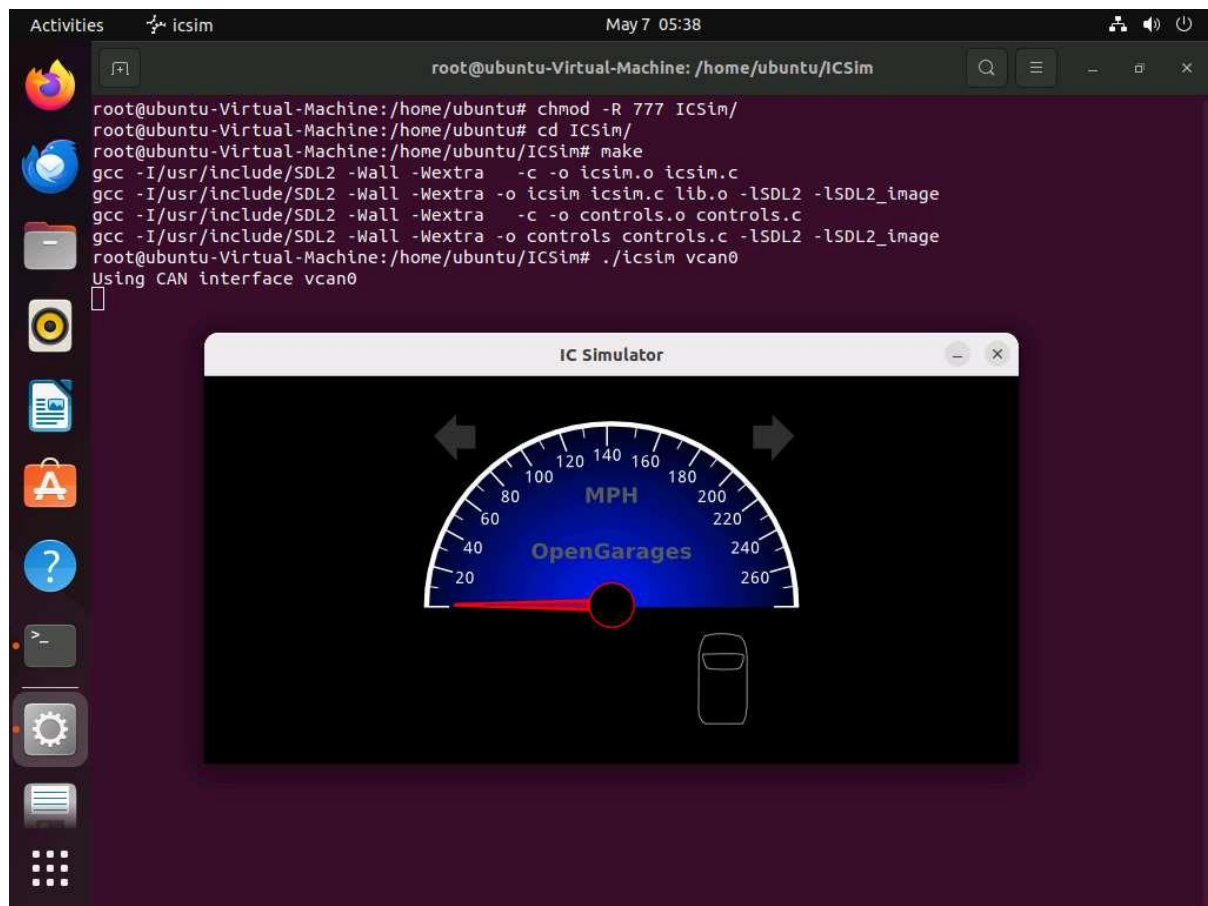


The screenshot shows a terminal window titled "Terminal" with the date and time "May 7 05:34". The prompt is "root@ubuntu-Virtual-Machine: /home/ubuntu/ICSim". The user has executed the following commands:

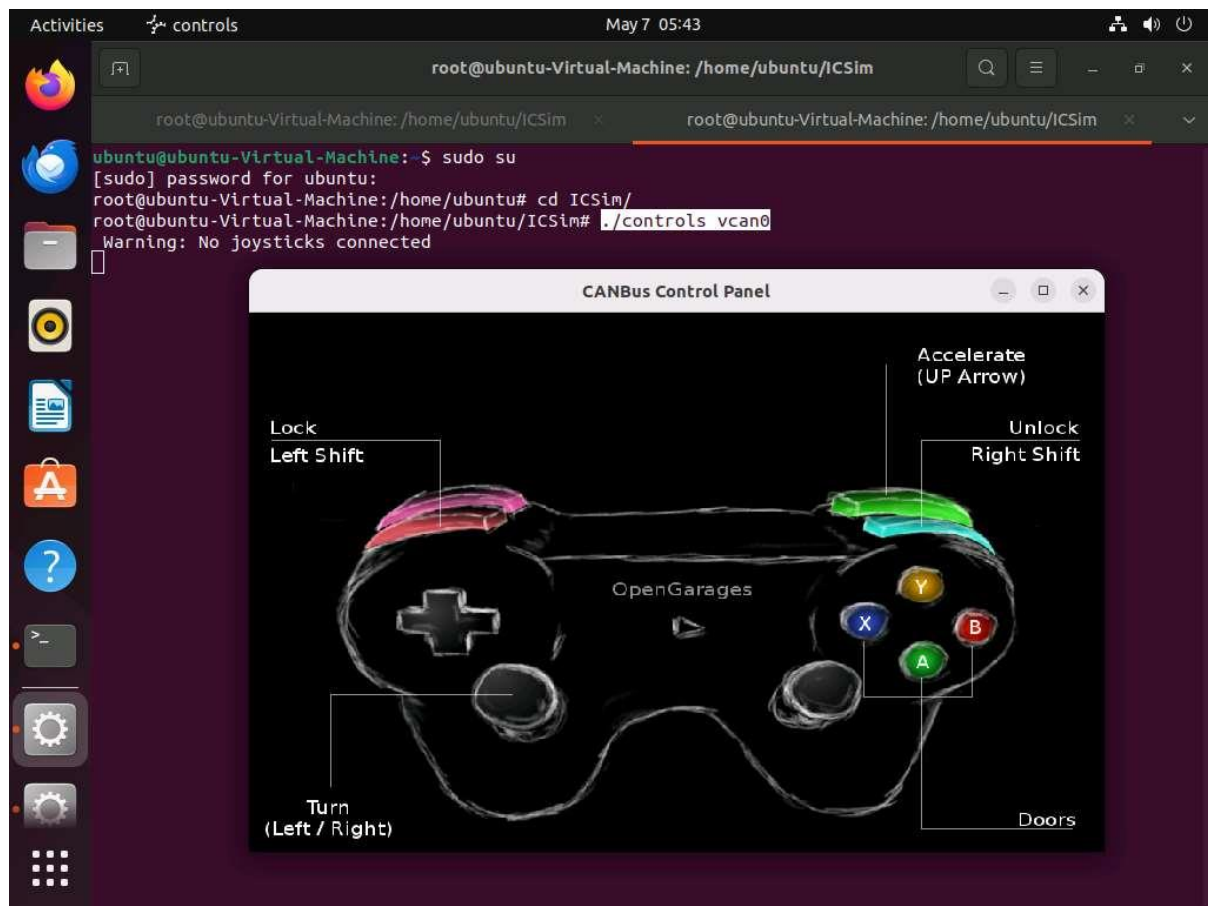
```
root@ubuntu-Virtual-Machine:/home/ubuntu# chmod -R 777 ICSim/
root@ubuntu-Virtual-Machine:/home/ubuntu# cd ICSim/
root@ubuntu-Virtual-Machine:/home/ubuntu/ICSim# make
gcc -I/usr/include/SDL2 -Wall -Wextra -c -o icsim.o icsim.c
gcc -I/usr/include/SDL2 -Wall -Wextra -o icsim icsim.c lib.o -lSDL2 -lSDL2_image
gcc -I/usr/include/SDL2 -Wall -Wextra -c -o controls.o controls.c
gcc -I/usr/include/SDL2 -Wall -Wextra -o controls controls.c -lSDL2 -lSDL2_image
root@ubuntu-Virtual-Machine:/home/ubuntu/ICSim#
```

The terminal window is part of a desktop environment with a sidebar on the left containing icons for various applications and system utilities.

8. Run `./icsim vcan0` to start the ICSim simulator. You will see the IC Simulator interface as shown in the screenshot.



9. Open a new terminal tab and execute **sudo su** to run the programs as a root user (When prompted, enter the password **toor**). Navigate to ICSim directory to do so run **cd ICSim/**.
10. Execute **./controls vcan0** to start the CANBus Control Panel. You will see the CANBus Control Panel interface as shown in the screenshot.



11. Now, we will start sniffer to capture the traffic sent to the ICSim Simulator by CANBus control panel simulator. To do so, open a new terminal tab and execute **sudo su** to run the programs as a root user (When prompted, enter the password **toor**). Navigate to ICSim directory to do so run **cd ICSim/**.
12. Execute **cansniffer -c vcan0** to start sniffing on the vcan0 interface. Leave this sniffer on.

```

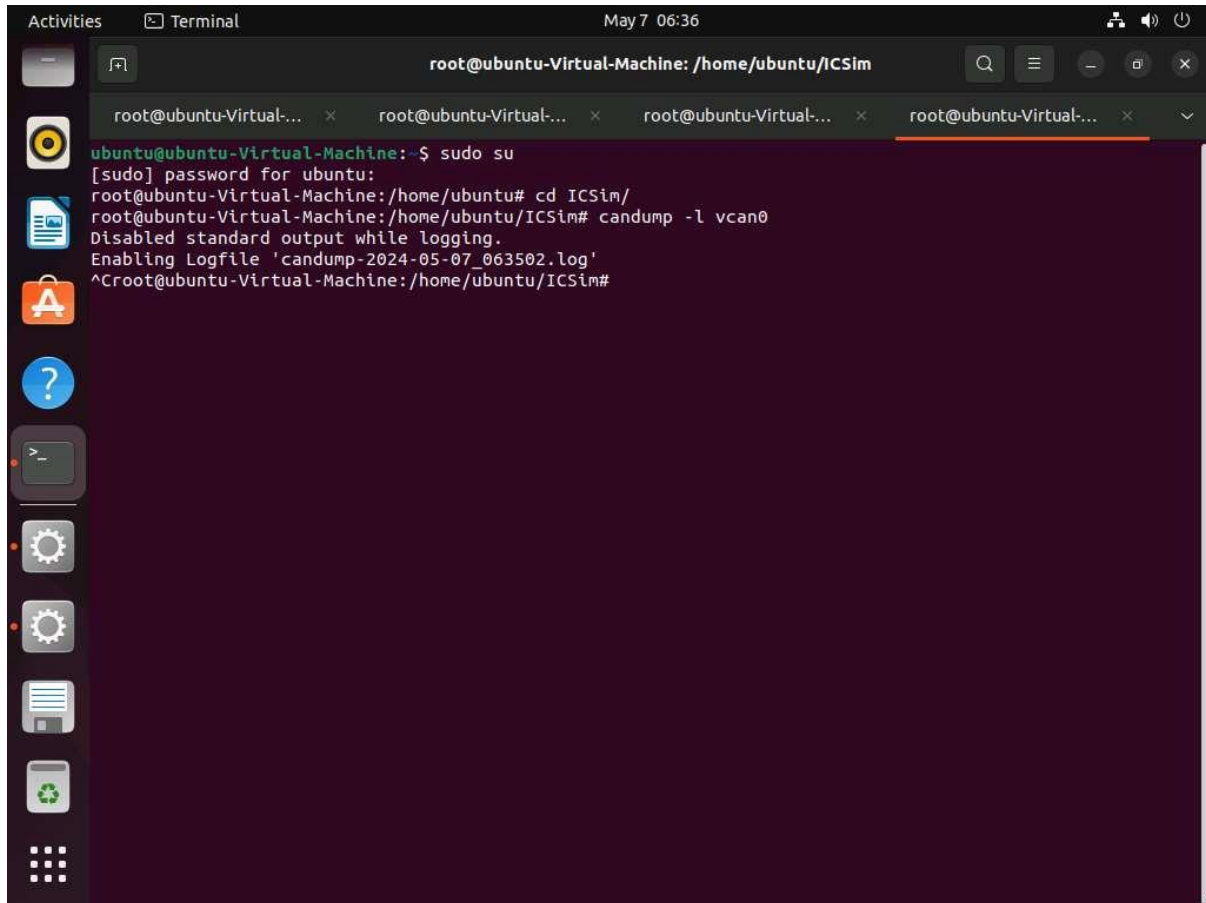
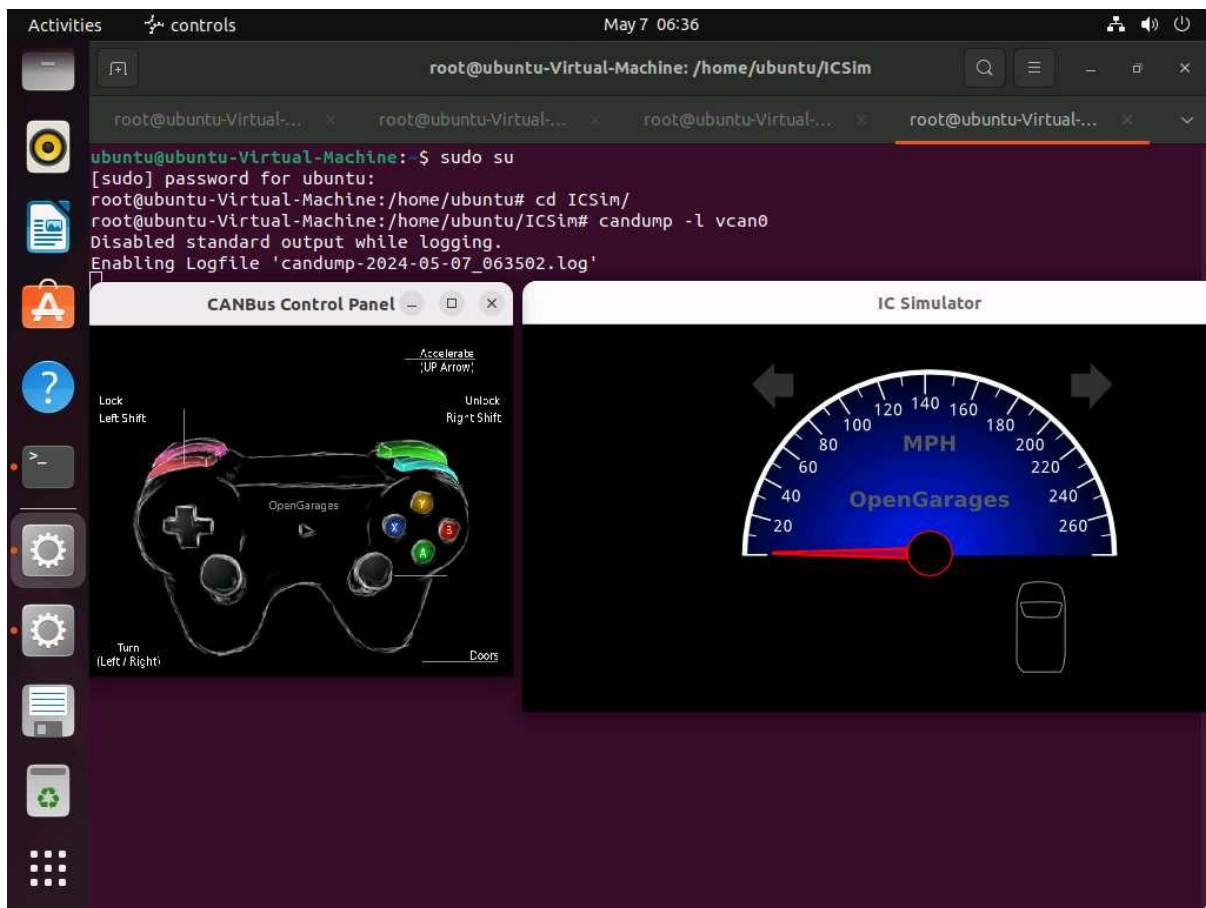
root@ubuntu-Virtual-Machine: /home/ubuntu/ICSim
root@ubuntu-Virtual-Machine: /h... x root@ubuntu-Virtual-Machine: /h... x root@ubuntu-Virtual-Machine: /h... x
ubuntu@ubuntu-Virtual-Machine:~$ sudo su
[sudo] password for ubuntu:
root@ubuntu-Virtual-Machine:/home/ubuntu# cd ICSim/
root@ubuntu-Virtual-Machine:/home/ubuntu/ICSim# cansniffer -c vcan0
00|ms | ID | data ... < vcan0 # l=20 h=100 t=500 slots=1 >
99999 | 143 | 68 68 00 D1 kk..

```

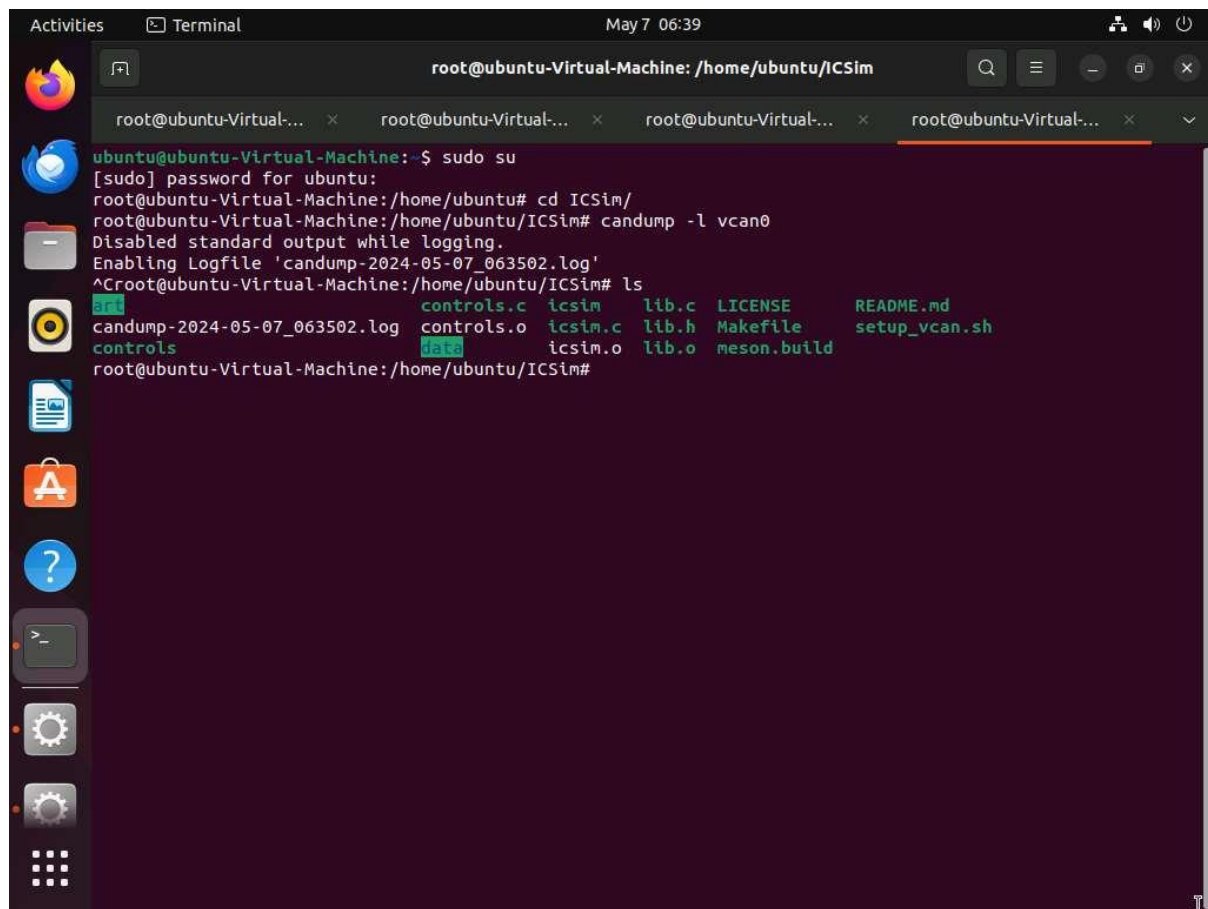
13. Open a new terminal and execute **sudo su** to run the programs as a root user (When prompted, enter the password toor). Navigate to ICSim directory to do so run **cd ICSim/**. To capture the logs run **candump -i vcan0**.
14. After starting to capture the logs, open ICSim and Controller simulator and perform functions such as acceleration, turning left/right, opening and locking doors so that logs are generated. Once you are done, terminate the ongoing process by pressing **Ctrl + C**.

Use the following keys to perform various functions

| ICSim Functions | Keys |
|-------------------------------|---------------------------------------|
| Accelerate | Up arrow |
| Left/Right Turn | Left arrow/ Right arrow |
| Unlock Rear Left/Right doors | Right Shift + X / Right Shift + Y |
| Unlock Front Left/Right doors | Right Shift +A / Right Shift + B |
| Lock all doors | Hold Right Shift key + Tap Left Shift |
| Unlock all doors | Hold Left Shift key + Tap Right Shift |



15. Now verify if you have obtained the log file by executing **ls** command.
The **.log** file has been generated as shown in the screenshot.

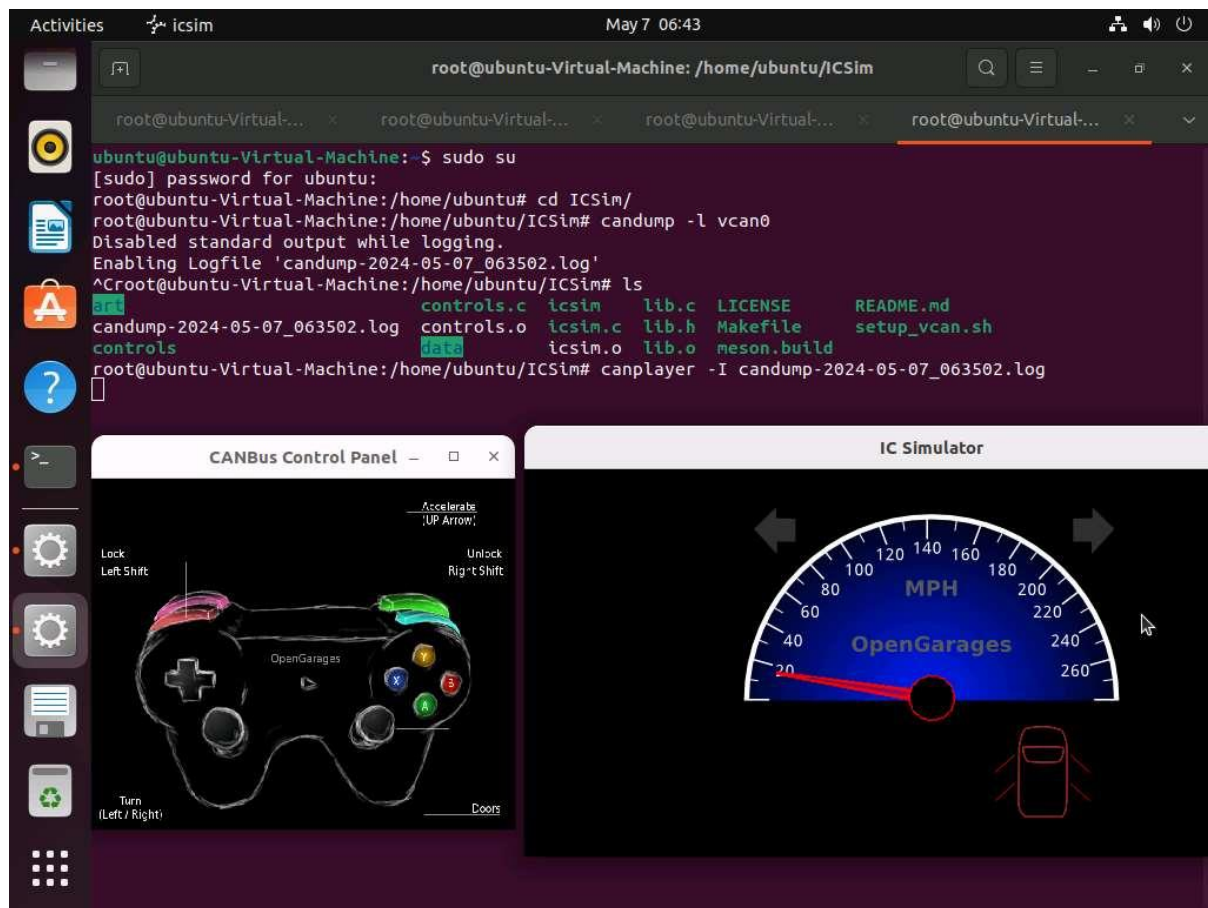


```
root@ubuntu-Virtual-Machine: /home/ubuntu/ICSim
root@ubuntu-Virtual-Machine:~$ sudo su
[sudo] password for ubuntu:
root@ubuntu-Virtual-Machine:/home/ubuntu# cd ICSim/
root@ubuntu-Virtual-Machine:/home/ubuntu/ICSim# candump -l vcan0
Disabled standard output while logging.
Enabling Logfile 'candump-2024-05-07_063502.log'
^Croot@ubuntu-Virtual-Machine:/home/ubuntu/ICSim# ls
art          controls.c  icsim      lib.c       LICENSE    README.md
candump-2024-05-07_063502.log  controls.o  icsim.c    lib.h       Makefile   setup_vcan.sh
controls     data       icsim.o    lib.o       meson.build
```

16. Now, to perform replay attack, run **canplayer -I candump-2024-05-07_063502.log** and press enter.

Once the log file is executed, you can see the movements that were performed while creating the log file in real time in IC Simulator and CANBus control panel simulator.

The log file name might vary while performing lab.



17. This concludes the demonstration of performing replay attack to exploit CAN protocol.

18. Close all open windows and document all the acquired information.

Question 18.3.1.1

In Ubuntu machine install ICSim simulator, start a CAN sniffer and perform functions such as acceleration, turning left/right, opening and locking doors in the simulator to generate the logs. Perform replay attack using the sniffed log file. Enter the interface that is used while sniffing the can traffic in Ubuntu.