

Module 18: IoT and OT Hacking

Scenario

The significant development of the paradigm of the Internet of Things (IoT) is contributing to the proliferation of devices in daily life. From smart homes to automated healthcare applications, IoT is ubiquitous. However, despite the potential of IoT to make our lives easier and more comfortable, we cannot underestimate its vulnerability to cyber-attacks. IoT devices lack basic security, which makes them prone to various cyber-attacks.

The objective of a hacker in exploiting IoT devices is to gain unauthorized access to users' devices and data. A hacker can use compromised IoT devices to build an army of botnets, which, in turn, is used to launch DDoS attacks.

Owing to a lack of security policies, smart devices are easy targets for hackers who can compromise these devices to spy on users' activities, misuse sensitive information (such as patients' health records, etc.), install ransomware to block access to the devices, monitor victims' activities using CCTV cameras, commit credit-card-related fraud, gain access to users' homes, or recruit the devices in an army of botnets to carry out DDoS attacks.

As an ethical hacker and penetration tester, you must have sound knowledge of hacking IoT and OT platforms using various tools and techniques. The labs in this module will provide you with real-time experience in performing footprinting and analyzing traffic between IoT and OT devices.

Objective

The objective of the lab is to perform IoT and OT platform hacking and other tasks that include, but are not limited to:

- Performing IoT and OT device footprinting
- Capturing and analyzing traffic between IoT devices

Overview of IoT and OT Hacking

Using the IoT and OT hacking methodology, an attacker acquires information using techniques such as information gathering, attack surface area identification, and vulnerability scanning, and uses such information to hack the target device and network.

The following are the various phases of IoT and OT device hacking:

- Information gathering
- Vulnerability scanning
- Launch attacks
- Gain remote access
- Maintain access

Lab Tasks

Ethical hackers or pen testers use numerous tools and techniques to hack the target IoT and OT platforms. Recommended labs that will assist you in learning various IoT platform hacking techniques include:

1. Perform footprinting using various footprinting techniques
 - Gather information using online footprinting tools
2. Capture and analyze IoT device traffic
 - Capture and analyze IoT traffic using Wireshark

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.

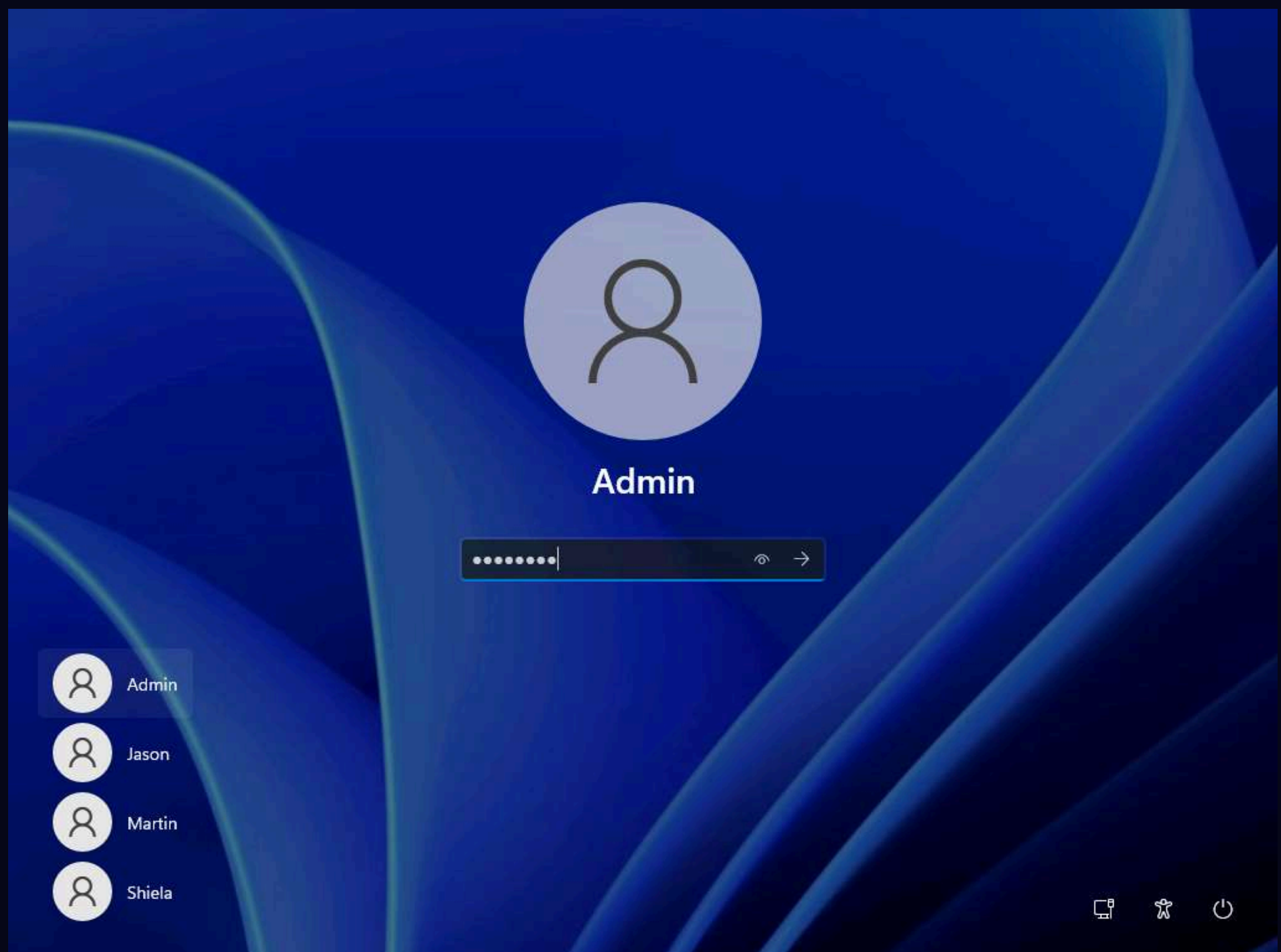
Note: In this task, we will focus on performing footprinting on the MQTT protocol, which is a machine-to-machine (M2M)/"Internet of Things" connectivity protocol. It is useful for connections with remote locations where a small code footprint is required and/or network bandwidth is at a premium.

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

1. Click **CEHv12 Windows 11** to switch to the **Windows 11** machine, click **Ctrl+Alt+Del**.
2. By default, **Admin** user profile is selected, type **Pa\$\$w0rd** in the **Password** field and press **Enter** to login.

Note: If **Welcome to Windows** wizard appears, click **Continue** and in **Sign in with Microsoft** wizard, click **Cancel**.

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



3. Launch any browser, here, we are using **Mozilla Firefox**. In the address bar of the browser place your mouse cursor and type **https://www.whois.com/whois/** and press **Enter**.

4. The **Whois Domain Lookup** page appears; type **www.oasis-open.org** in the search field and click **SEARCH**.

Note: 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.



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

Note: 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 Identity for everyone

Enter Domain or IP WHOIS

DOMAINS WEBSITE CLOUD HOSTING SERVERS EMAIL SECURITY WHOIS SUPPORT LOGIN

oasis-open.org Updated 6 days ago

Interested in similar domains?

Domain Information

Domain: oasis-open.org

Registrar: DNC Holdings, Inc.

Registered On: 1998-03-04

Expires On: 2023-03-03

Updated On: 2022-01-17

Status: clientDeleteProhibited
clientTransferProhibited
clientUpdateProhibited

Name Servers: dns2.stabletransit.com
dns1.stabletransit.com

Registrant Contact

Organization: OASIS Open

State: MA

oasis-open.com Buy Now

littleoasisopen.com Buy Now

oasisopenhouse.com Buy Now

oasisthird.com Buy Now

oasisdream.net Buy Now

oasisguest.net Buy Now

.space Sale
\$24.88 \$0.88

1:11 AM 4/18/2022

Whois oasis-open.org

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

State: MA

Country: US

Raw Whois Data

Domain Name: OASIS-OPEN.ORG
Registry Domain ID: D1849375-LROR
Registrar WHOIS Server: whois.directnic.com
Registrar URL: http://www.directnic.com
Updated Date: 2022-01-17T07:40:27Z
Creation Date: 1998-03-04T05:00:00Z
Registry Expiry Date: 2023-03-03T05:00:00Z
Registrar Registration Expiration Date:
Registrar: DNC Holdings, Inc.
Registrar IANA ID: 291
Registrar Abuse Contact Email: abuse@directnic.com
Registrar Abuse Contact Phone: +1.8778569598
Reseller:
Domain Status: clientDeleteProhibited https://icann.org/epp#clientDelete
Domain Status: clientTransferProhibited https://icann.org/epp#clientTra
Domain Status: clientUpdateProhibited https://icann.org/epp#clientUpdate
Registrant Organization: OASIS Open
Registrant State/Province: MA
Registrant Country: US
Name Server: DNS2.STABLETRANSIT.COM
Name Server: DNS1.STABLETRANSIT.COM
DNSSEC: unsigned
URL of the ICANN Whois Inaccuracy Complaint Form https://www.icann.org/
>>> Last update of WHOIS database: 2022-04-11T16:29:49Z <<<

For more information on Whois status codes, please visit https://icann.
Access to Public Interest Registry WHOIS information is provided to ass
The Registrar of Record identified in this output may have an RDNS serv
< >

https://shop.whois.com/optimized-wordpress-hosting.php

.TOP \$24.88 \$0.88
BUY NOW
*Offer ends 30th April 2022

On Sale!
.TOP
.TOP @ \$1.88 \$9.88

Introducing
WORDPRESS HOSTING
\$3.58 /mo
VIEW MORE

1:14 AM 4/18/2022

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

6. Now, open a new tab, and type <https://www.exploit-db.com/google-hacking-database> in the address bar, and press **Enter**.

7. The **Google Hacking Database** page appears; type **SCADA** in the **Quick Search** field and press **Enter**.

8. The result appears, which displays the Google dork related to SCADA, as shown in the screenshot.

The screenshot shows the Google Hacking Database (GHD) interface. The browser address bar displays <https://www.exploit-db.com/google-hacking-database>. The page title is "Google Hacking Database". A "Quick Search" field contains the text "SCADA". Below the search field, a table lists search results. The table has columns: "Date Added", "Dork", "Category", and "Author". The results are filtered to show 6 entries out of 7,341 total entries. The results are as follows:

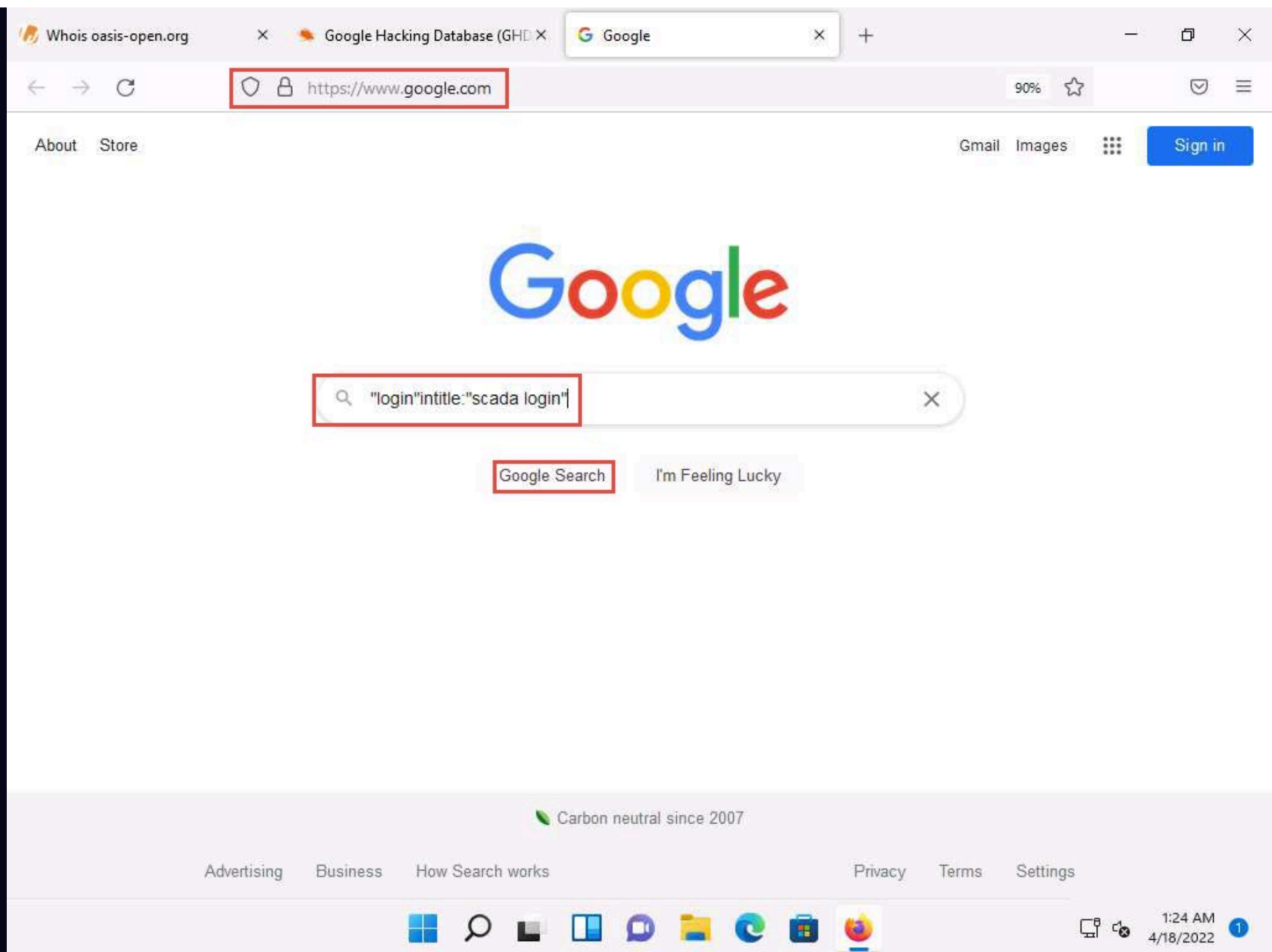
| Date Added | Dork | Category | Author |
|------------|--|--------------------------------|----------------------|
| 2021-10-04 | <code>intitle:"index of SCADA"</code> | Sensitive Directories | Romell Marin Cordoba |
| 2021-09-20 | <code>intitle:inurl:"SCADA login"</code> | Pages Containing Login Portals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

Below the table, it says "Showing 1 to 6 of 6 entries (filtered from 7,341 total entries)". Navigation links include "FIRST", "PREVIOUS", "1" (current page), "NEXT", and "LAST". At the bottom, there are sections for "Downloads", "Certifications", "Training", and "Professional Services".

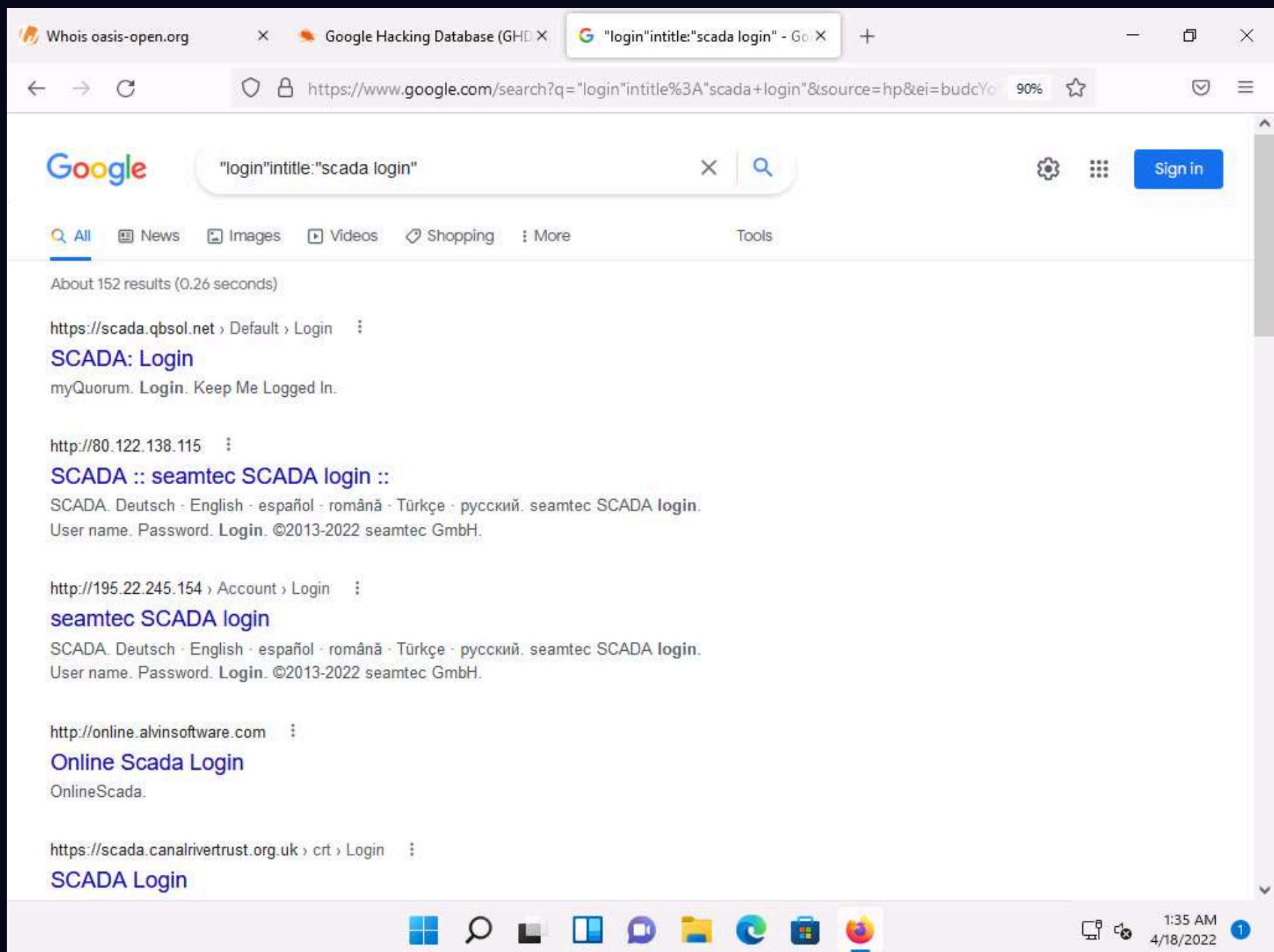
9. Now, we will use the dorks obtained in the previous step to query results in Google.

10. Open a new tab and type **<https://www.google.com>** in the address bar, and press **Enter**.

11. In the search field, type **"login" intitle:"scada login"** and click the **Google Search** button.



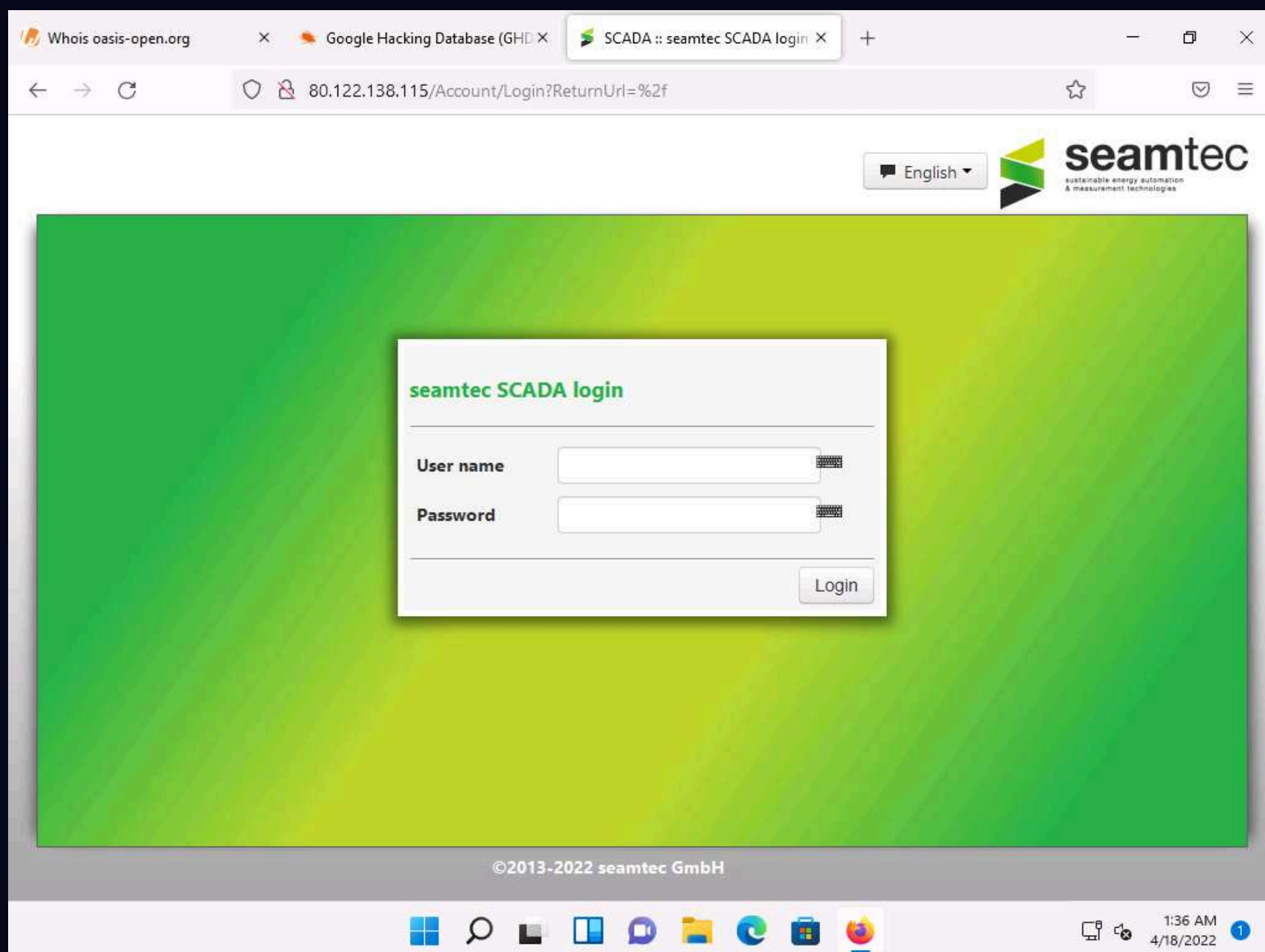
12. The search result appears; click any link (here, **SCADA :: seamtec SCADA login ::**).



Note: 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.

13. The **seamtec SCADA login** page appears, as shown in the screenshot.

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

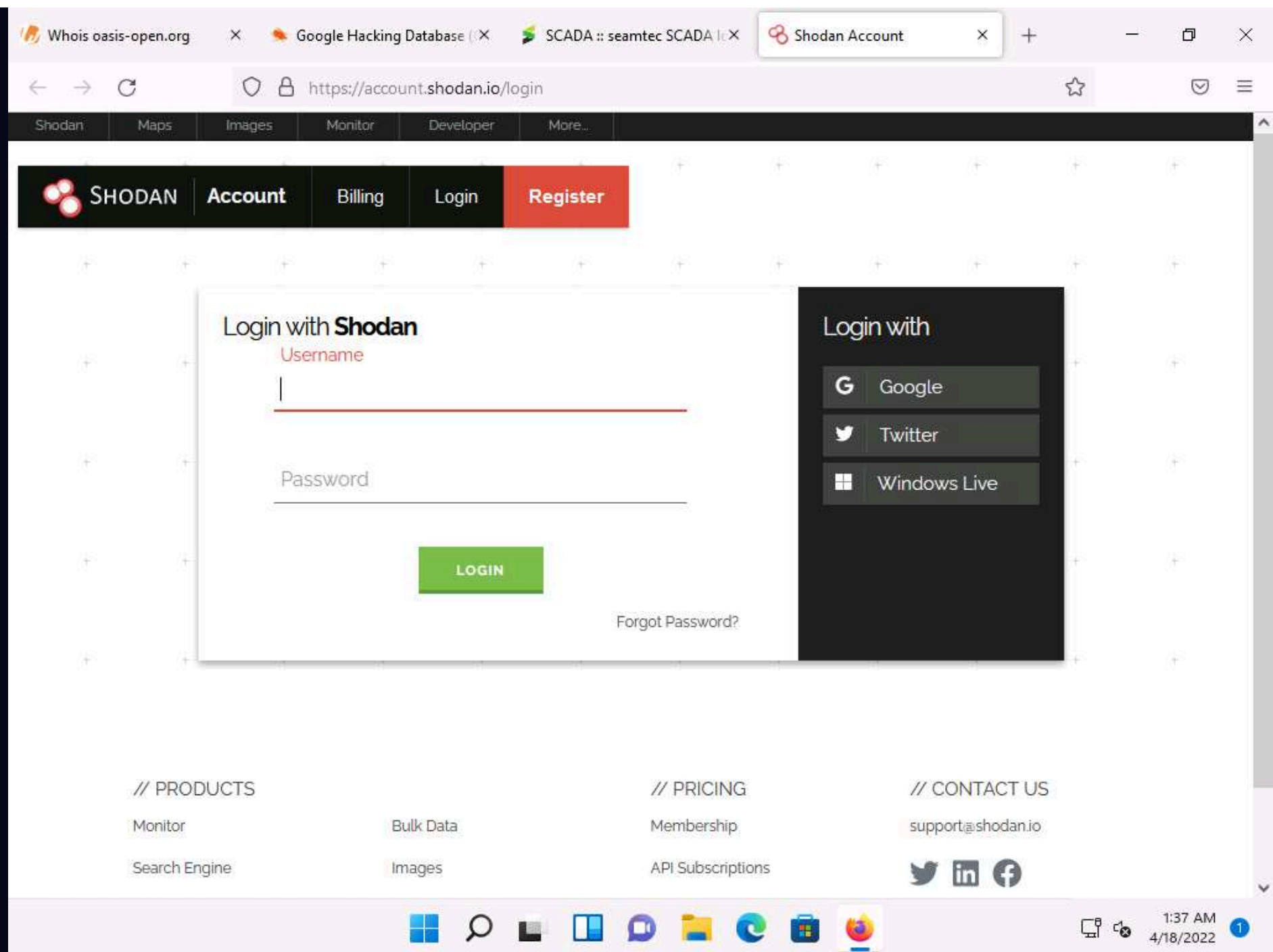


14. Similarly, you can use advanced search operators such as **intitle:"index of" scada** to search sensitive SCADA directories that are exposed on sites.

15. Now, in the browser window, open a new tab type **https://account.shodan.io/login** in the address bar, and press **Enter**.

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

Note: Go to the **Register** option to register yourself if you do not have an existing account.

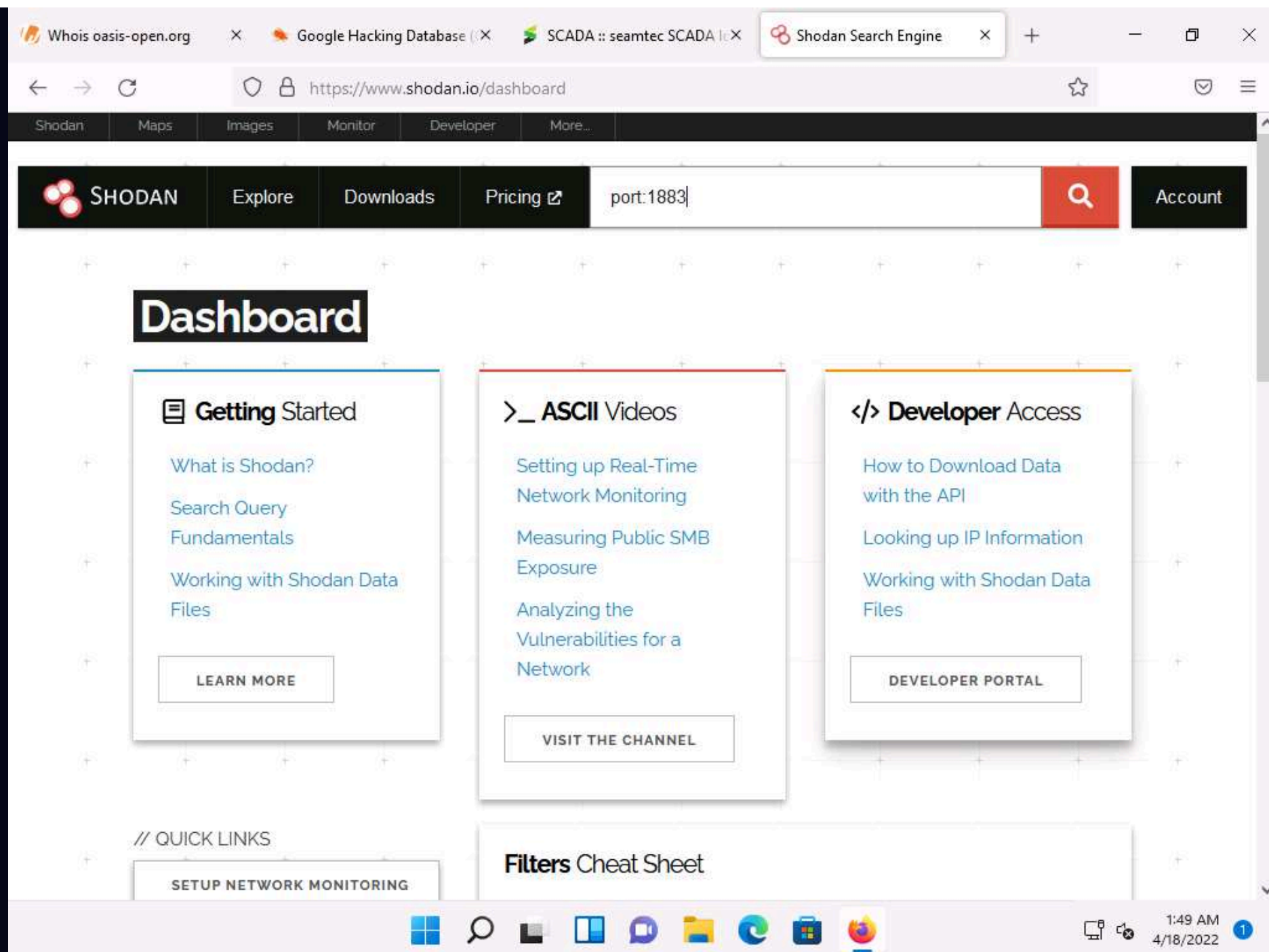


17. The **Account Overview** page appears, which displays the account-related information.

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

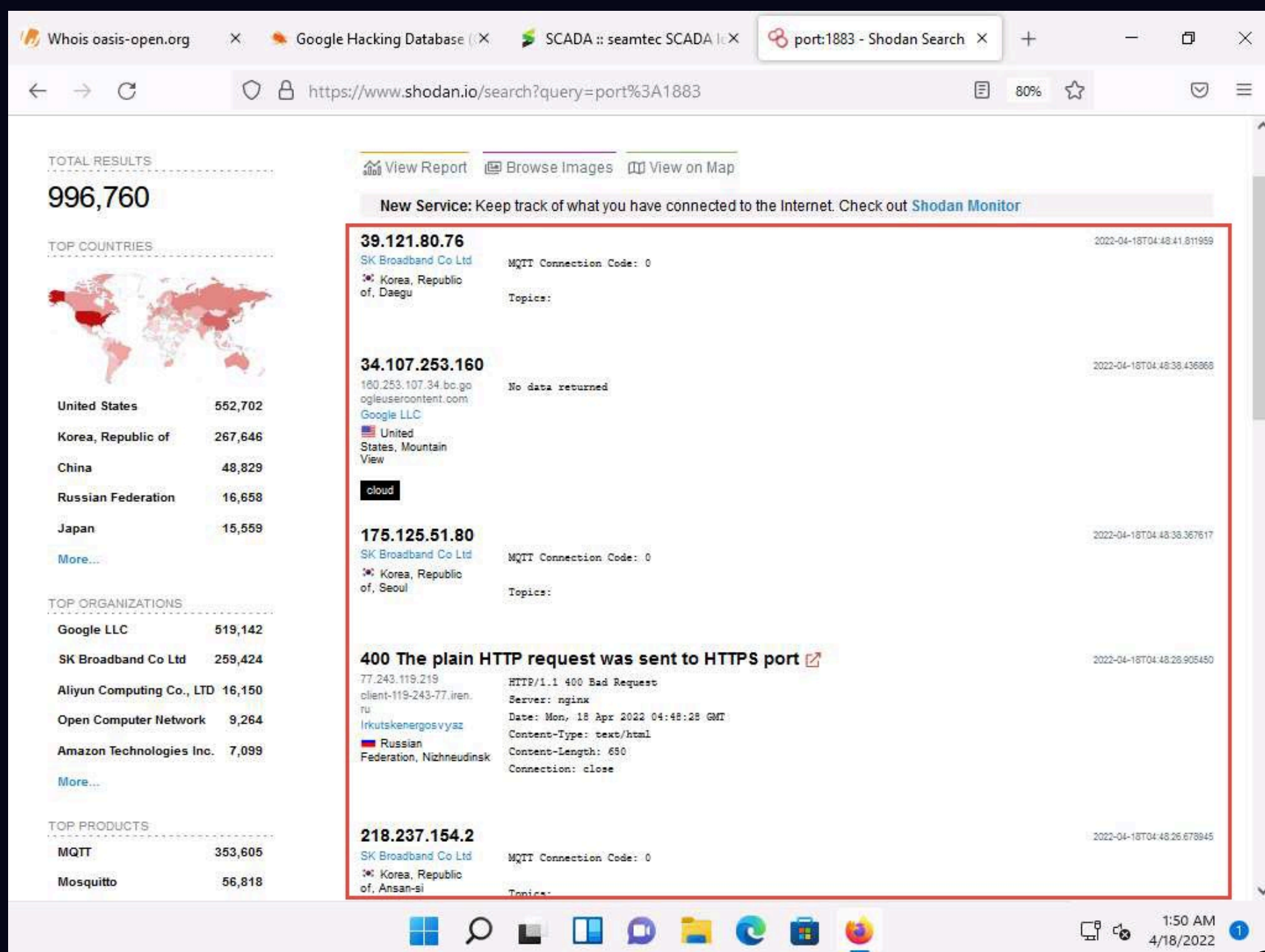
18. The **Shodan** main page appears; type **port:1883** in the address bar and press **Enter**.

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

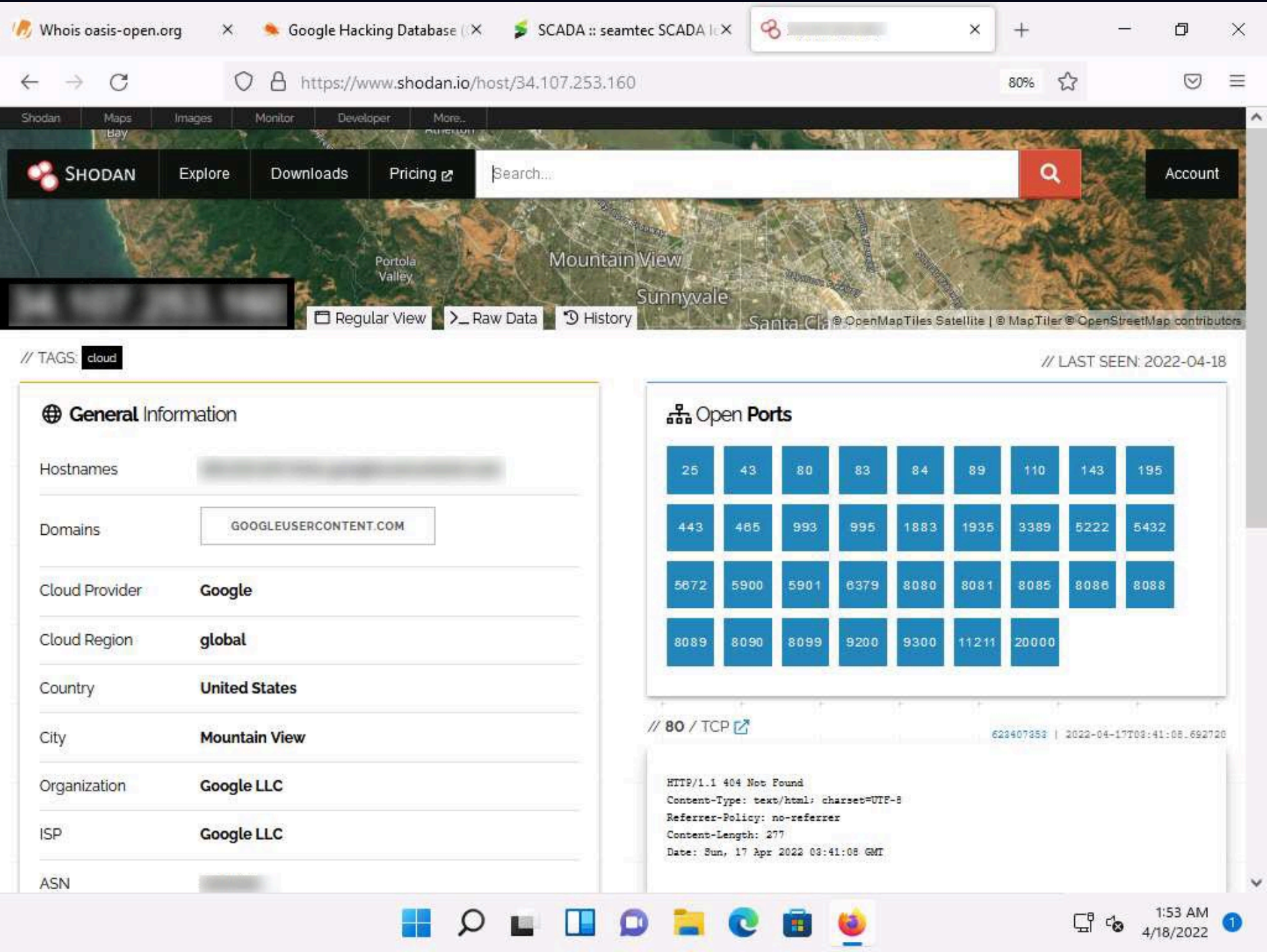


19. The result appears, displaying the list of IP addresses having port 1883 enabled, as shown in the screenshot.

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



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



22. 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"`

23. 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.

24. 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.

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

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.

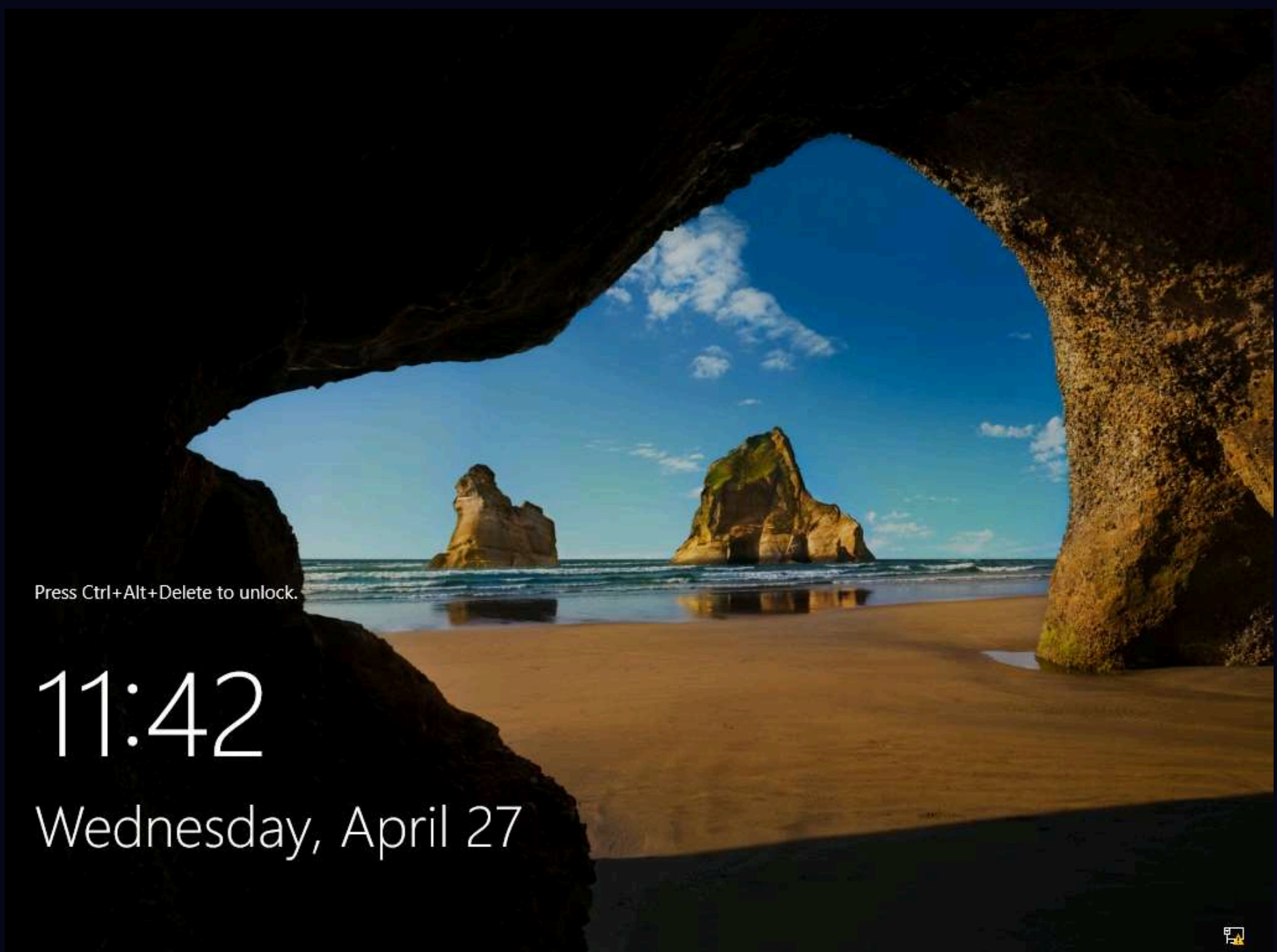
Task 1: Capture and Analyze IoT Traffic using Wireshark

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.

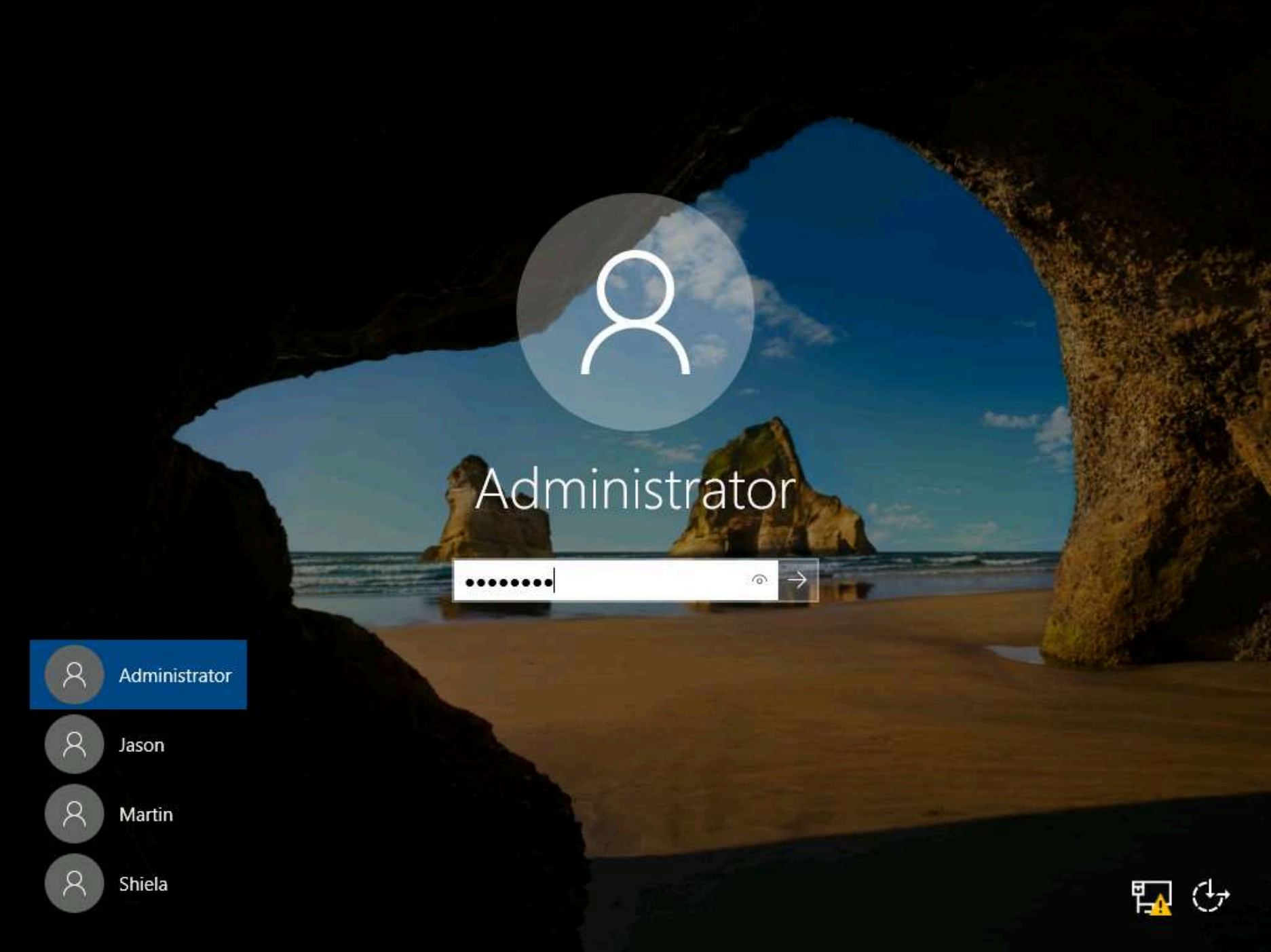
MQTT is a lightweight messaging protocol that uses a publish/subscribe communication pattern. Since the protocol is meant for devices with a low-bandwidth, it is considered ideal for machine-to-machine (M2M) communication or IoT applications. We can create virtual IoT devices over the virtual network 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 **CEHv12 Windows Server 2019** to launch **Windows Server 2019** machine, and then click **Ctrl+Alt+Del** link to login.

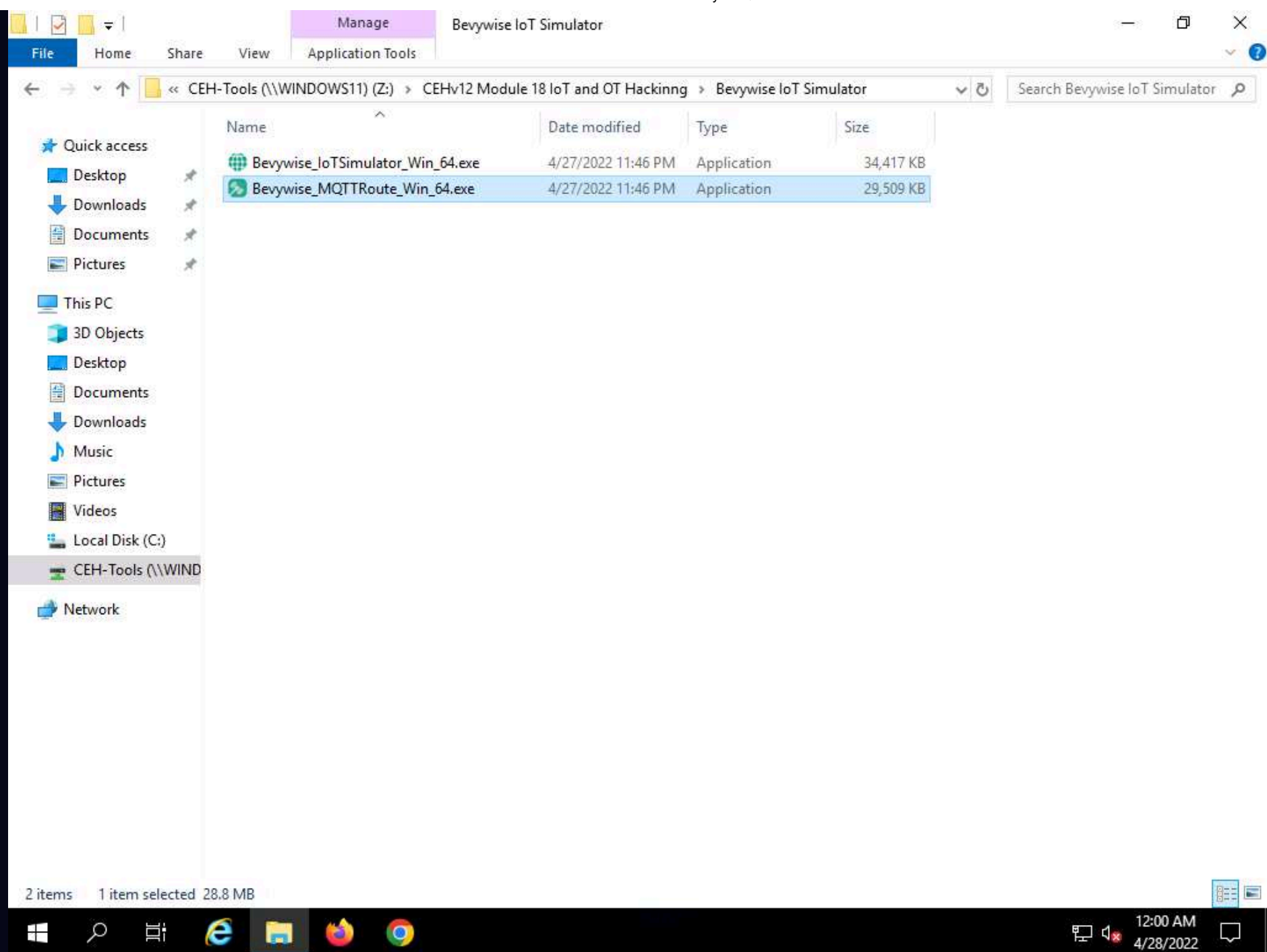


2. By default ****Administrator **** account is selected, type **Pa\$\$w0rd** in the Password field and press **Enter** to login.



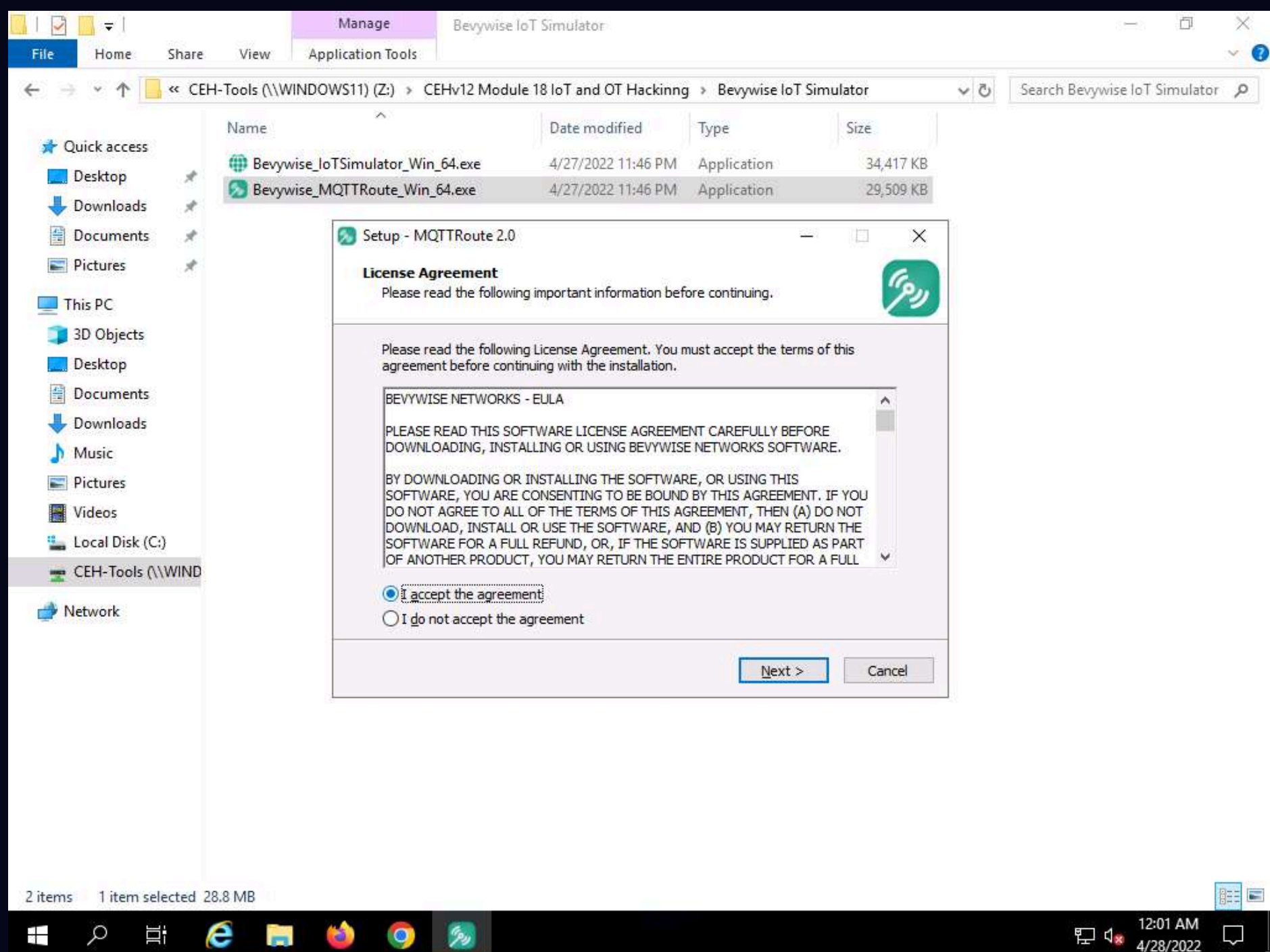
Note: If the network screen appears, click **Yes**.

- 3. Navigate to **Z:\CEH-Tools\CEHv12 Module 18 IoT and OT Hacking\Bevywise IoT Simulator** folder and double-click on the **Bevywise_MQTTRoute_Win_64.exe** file.



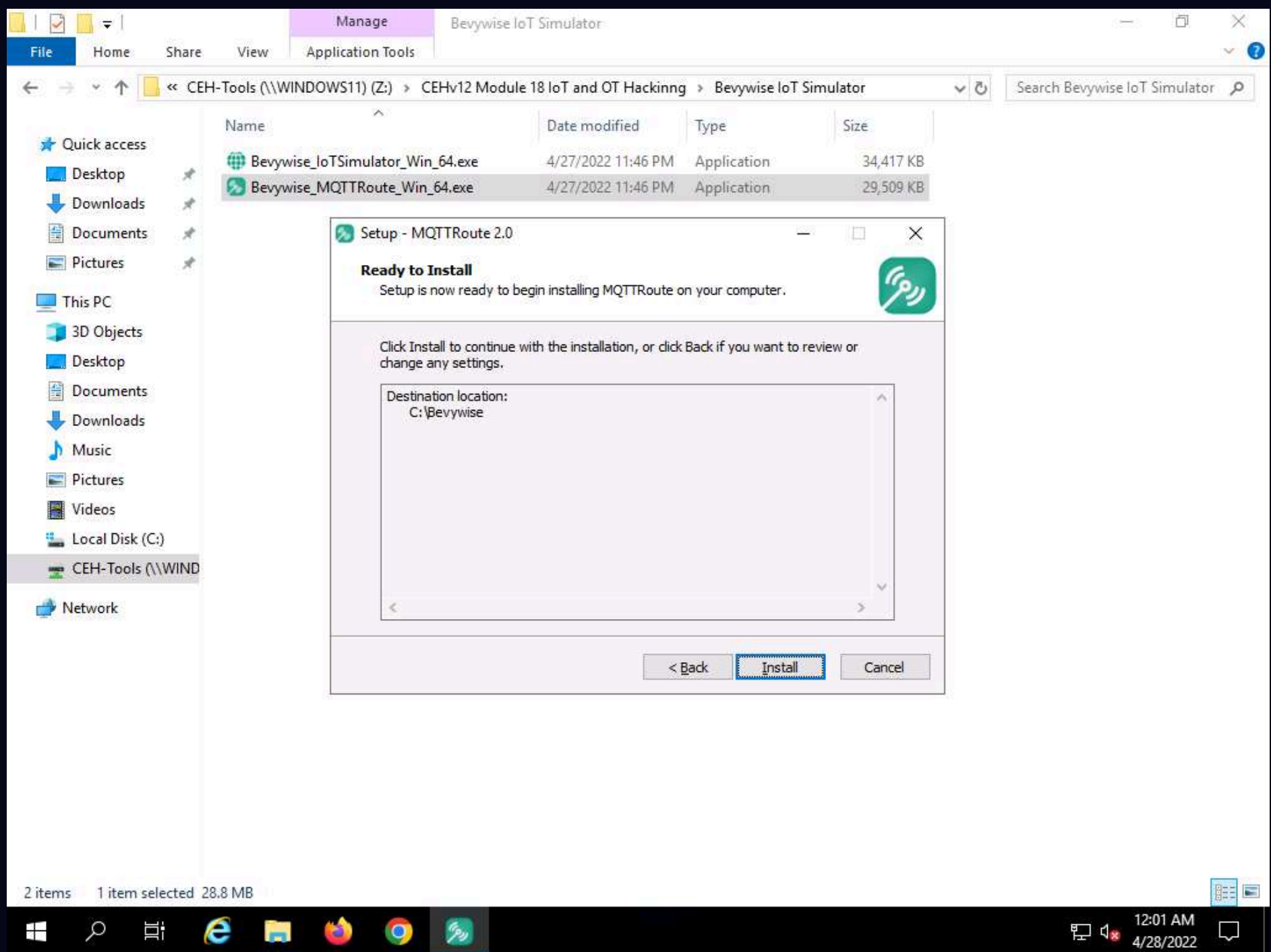
4. The **Open File - Security Warning** popup appears. Click **Run**.

5. The **Setup – MQTTRoute 2.0** window opens. Select **I accept the agreement** and click on **Next**.

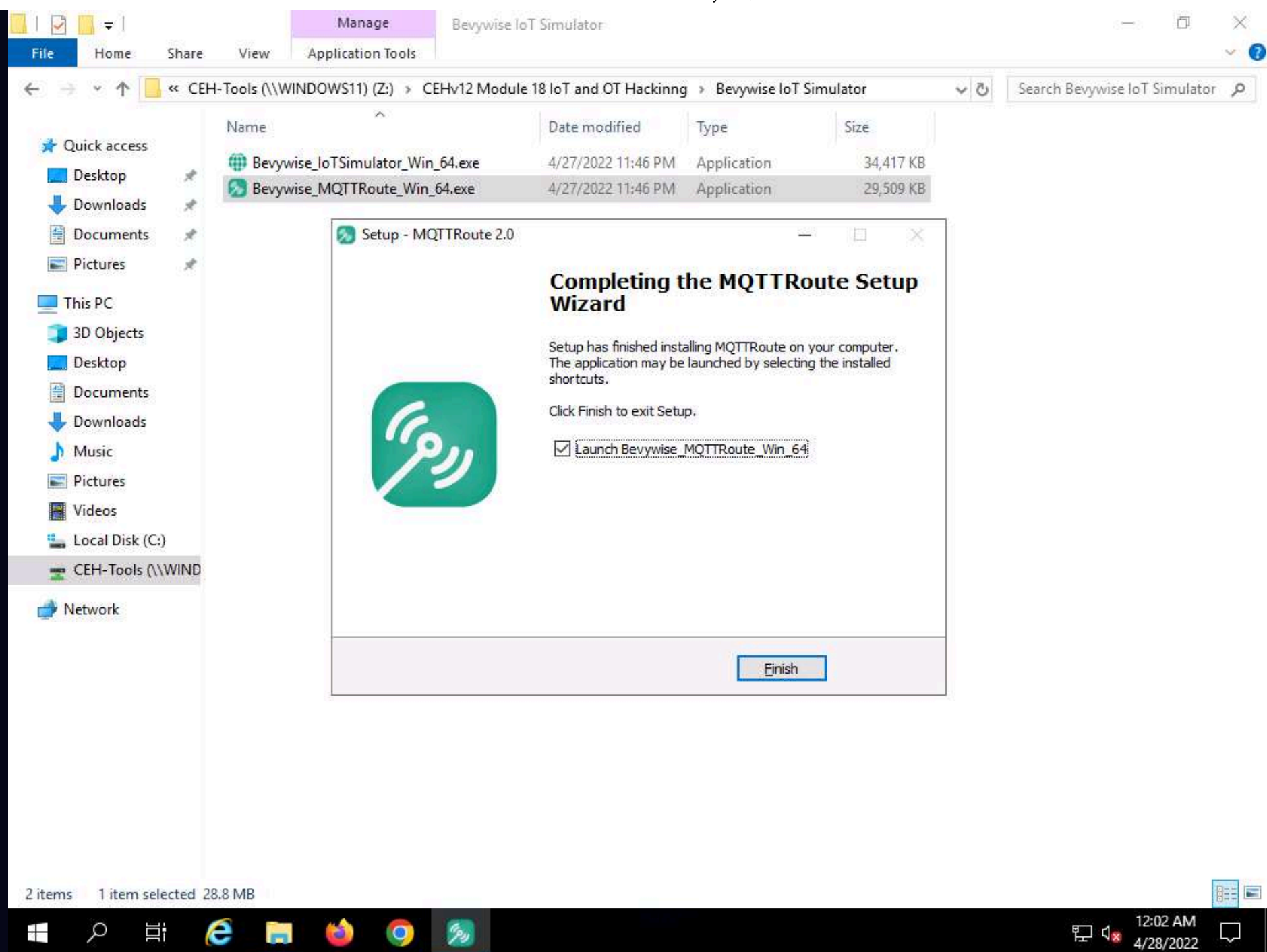


6. **Select Destination Location** page appears, without making any changes to the default installation location, click on **Next**.

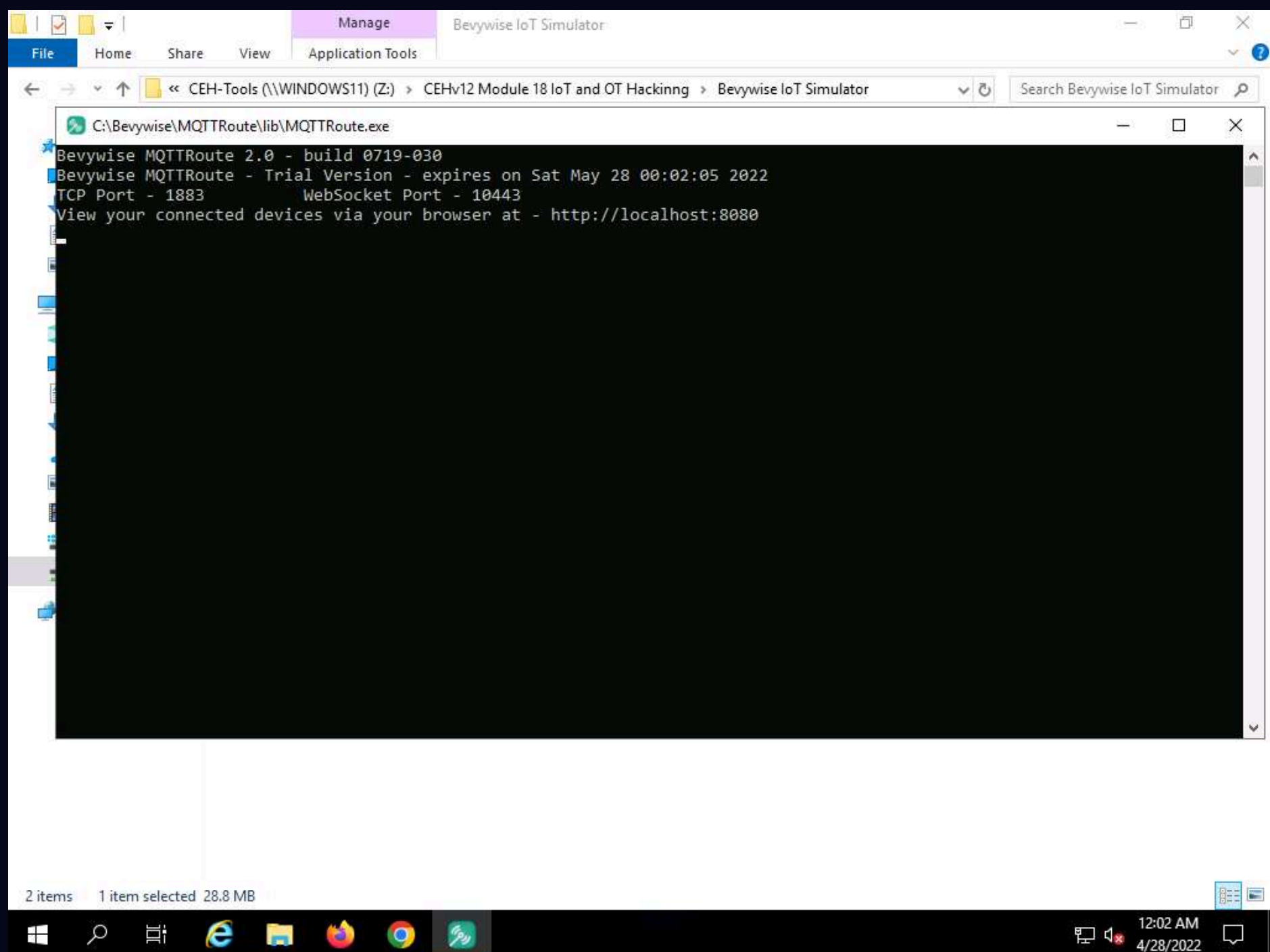
7. In the next window, click **Install** to complete the installation.



8. The installation completes, click on **Finish**. Ensure that **Launch Bevywise_MQTTRoute_Win_64** is checked.



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



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

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

12. Click **CEHv12 Windows Server 2022** to launch **Windows Server 2022** machine. Click **Ctrl+Alt+Del**.

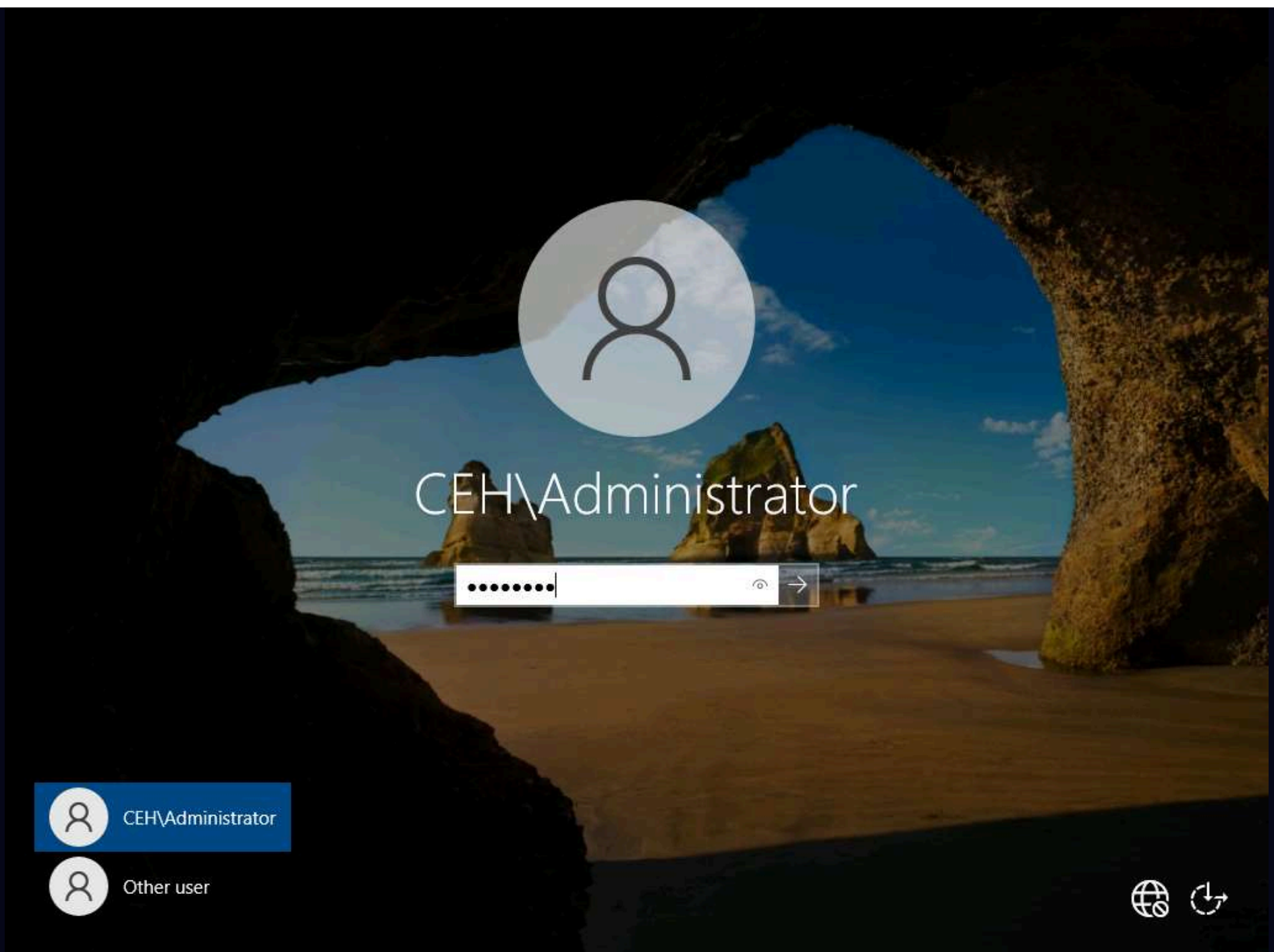


Press Ctrl+Alt+Delete to unlock.

12:05

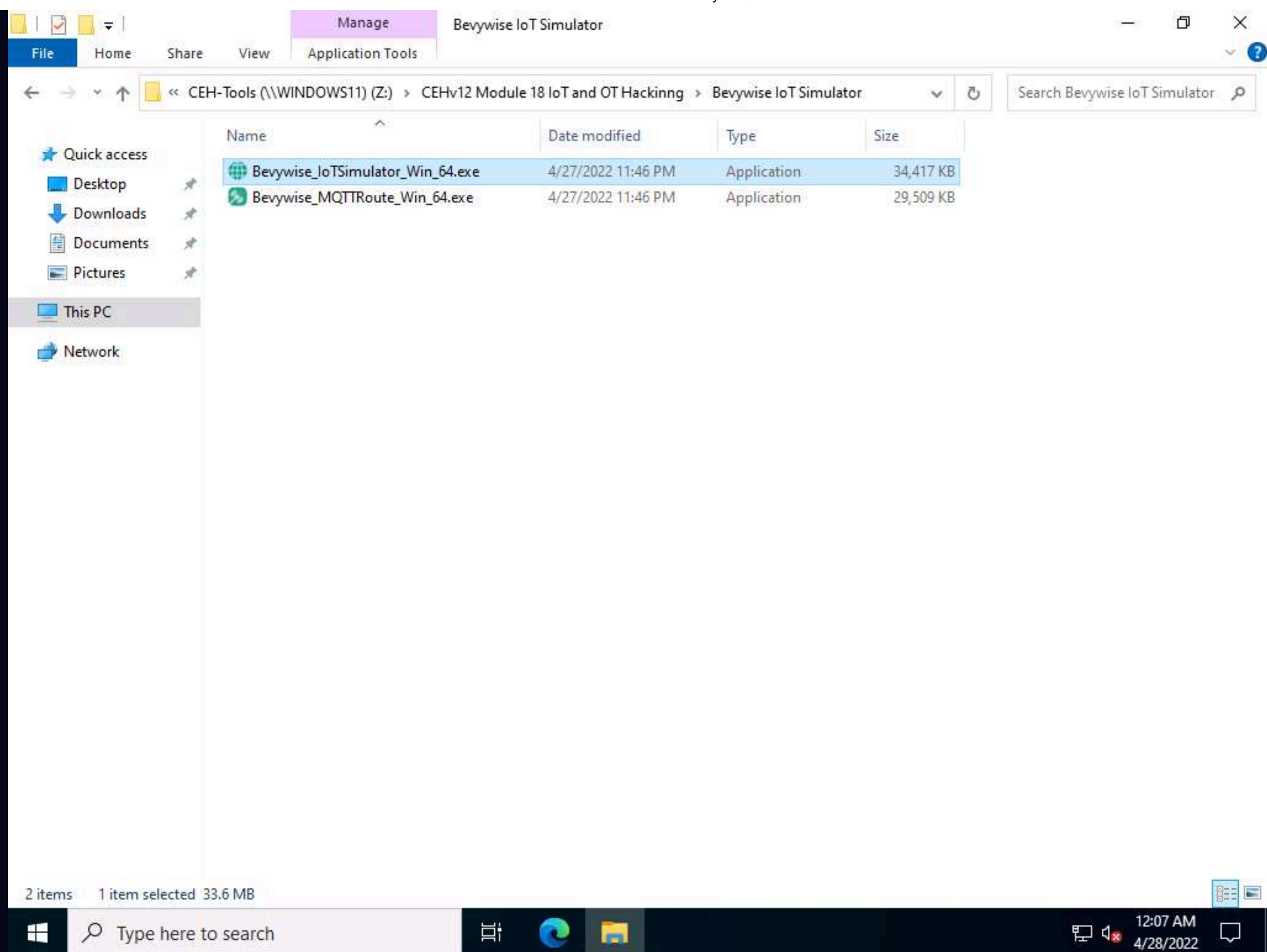
Thursday, April 28

13. By default **CEH\Administrator** account is selected, type **Pa\$\$w0rd** in the Password field and press **Enter** to login.



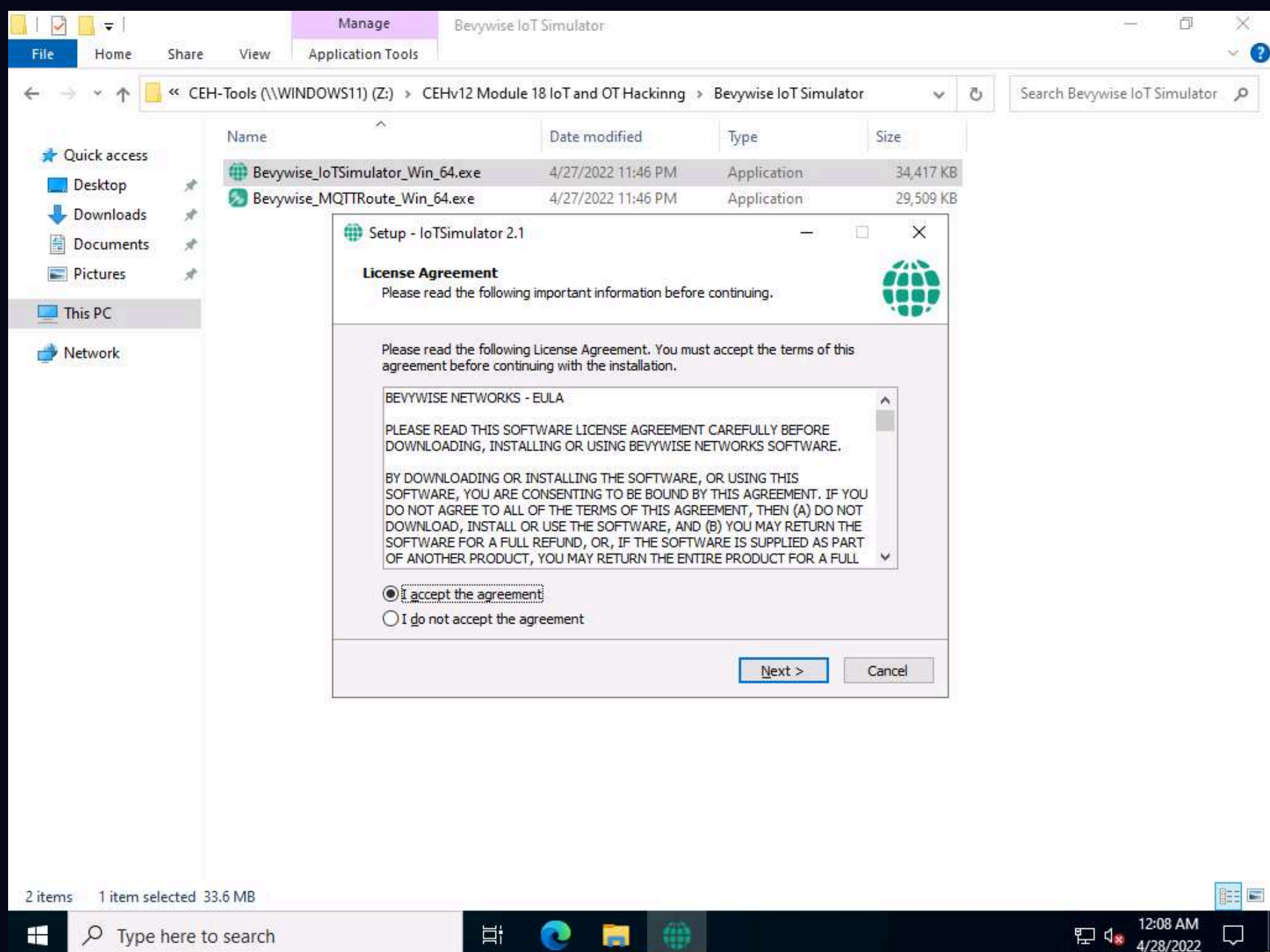
Note: If the network screen appears, click **Yes**.

14. Navigate to **Z:\CEH-Tools\CEHv12 Module 18 IoT and OT Hacking\Bevywise IoT Simulator** folder and double-click on the **Bevywise_IoTSimulator_Win_64.exe** file.



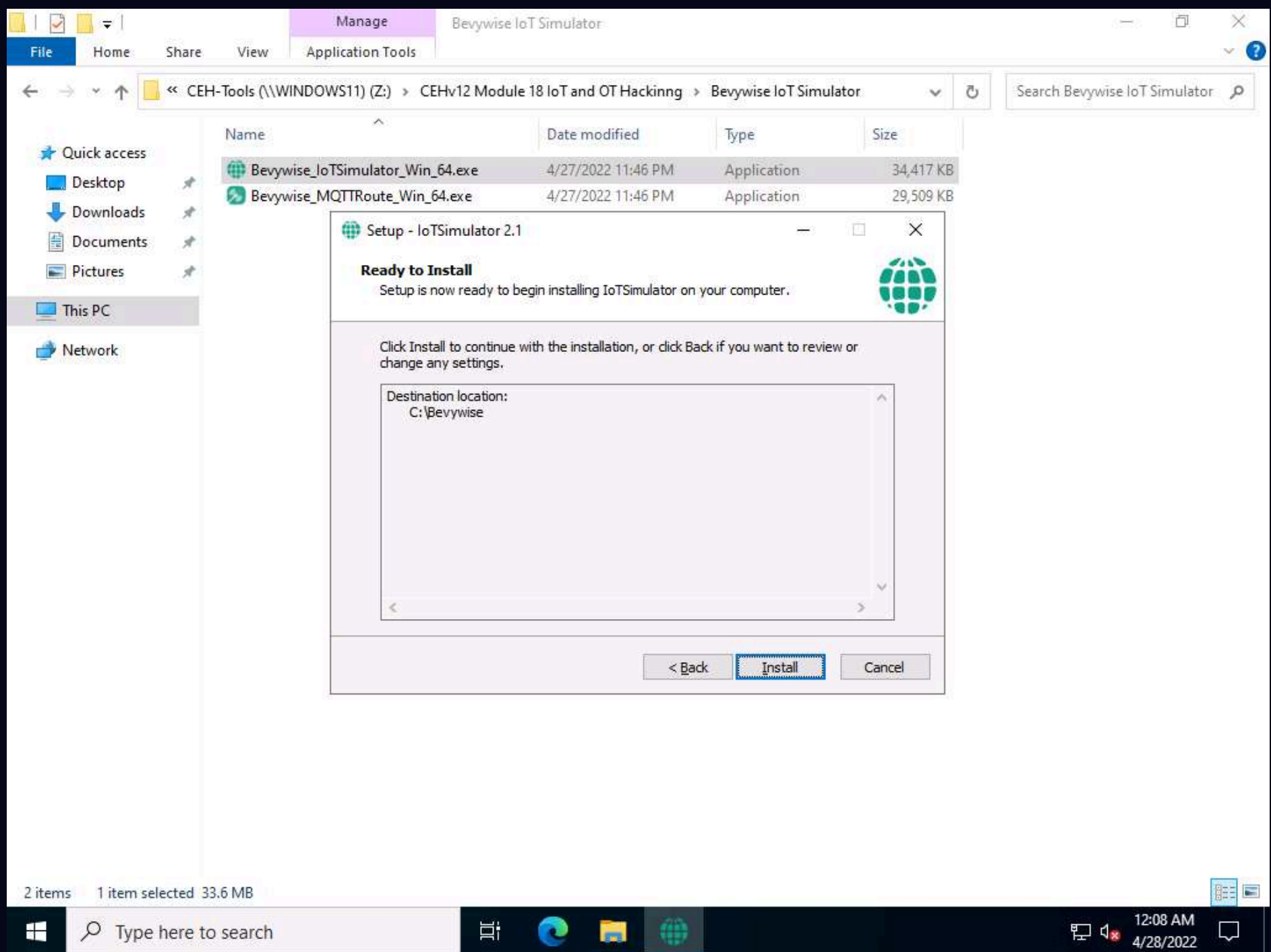
15. The **Open File - Security Warning** popup appears. Click **Run..**

16. The **Setup-IoTSimulator 2.1** setup wizard opens. Select **I accept the agreement** and click on **Next** to continue.

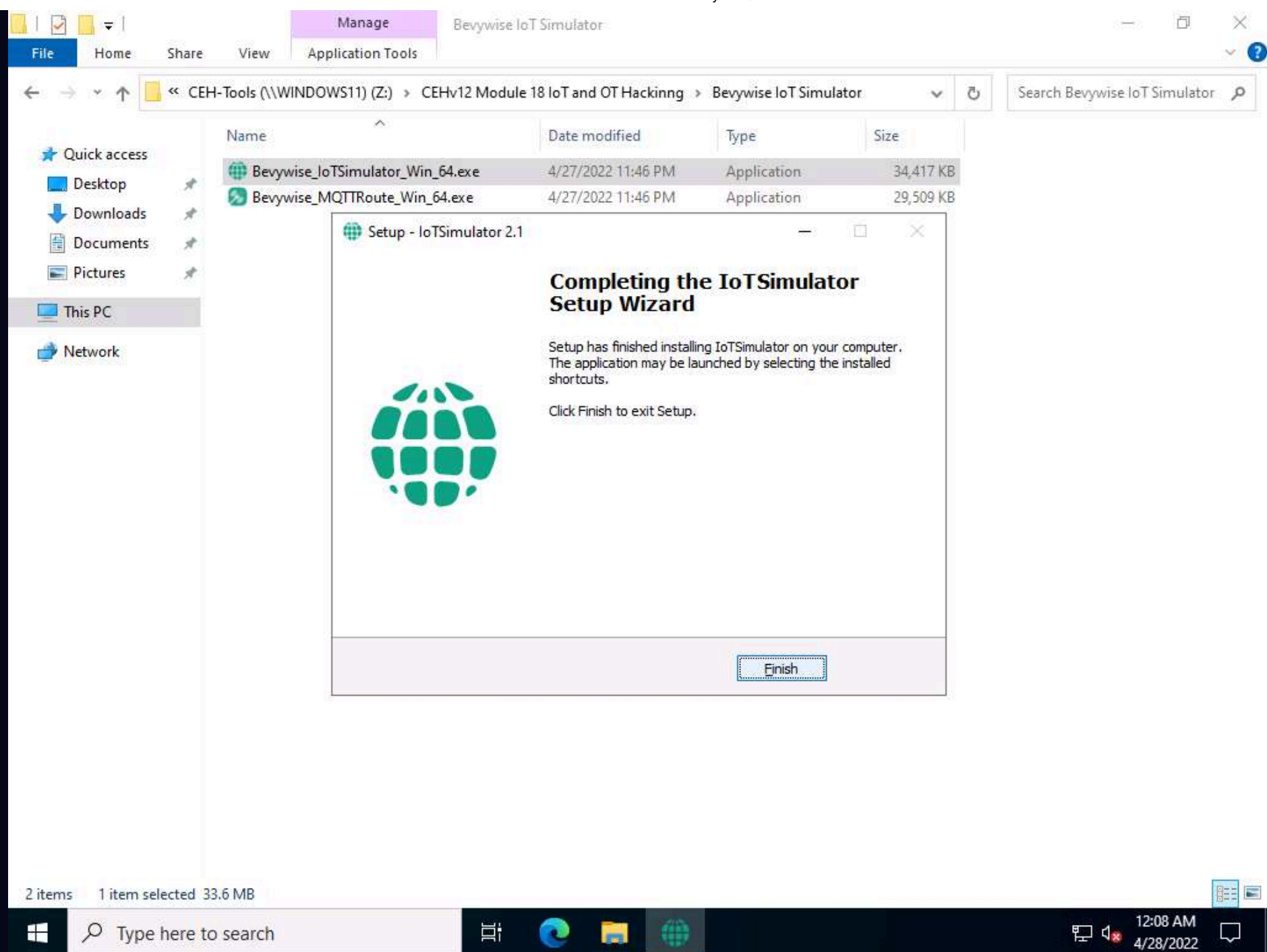


17. Keeping the default destination unchanged, click on **Next**.

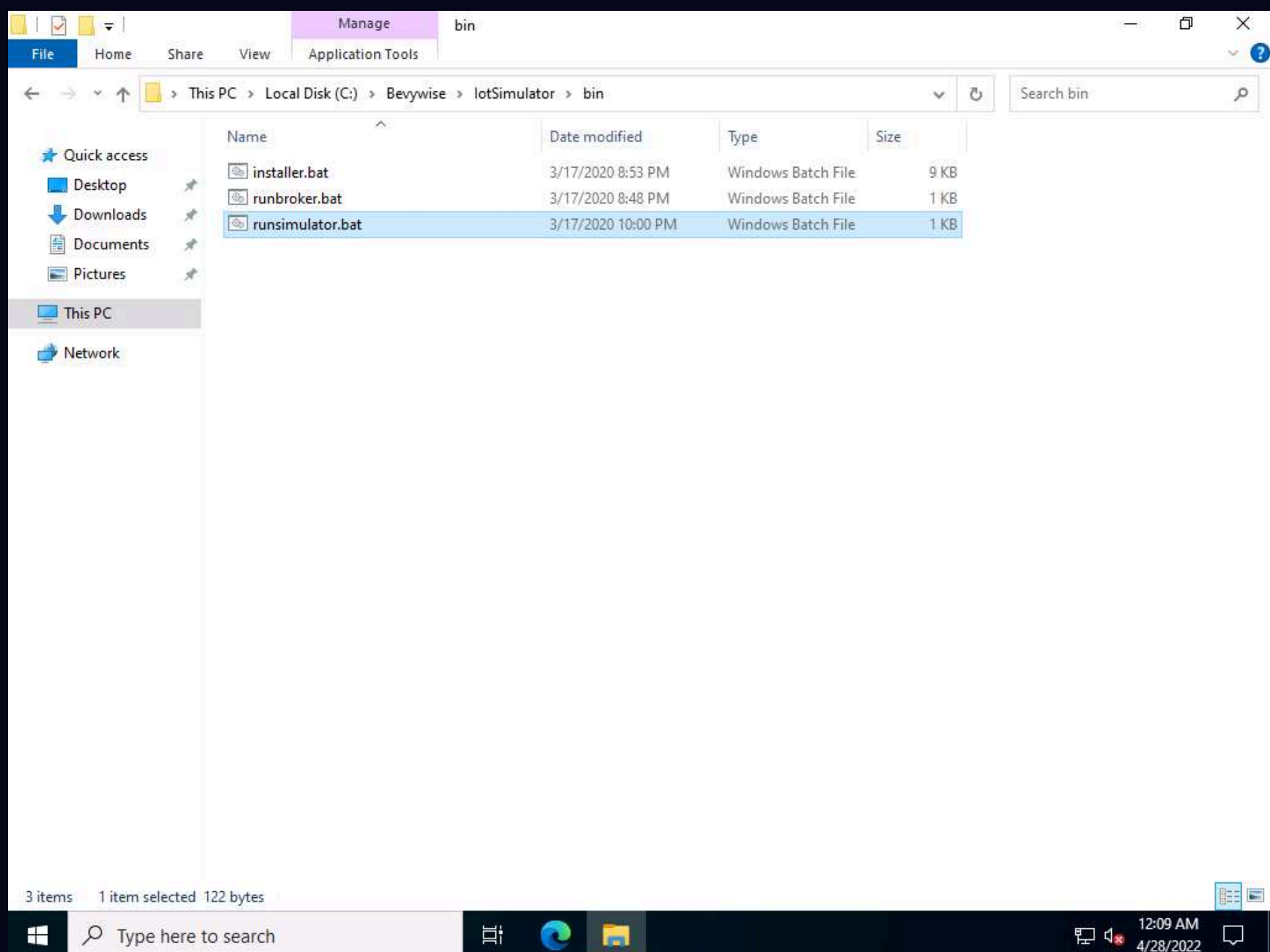
18. The Ready to install screen appears, click on **Install**



19. Click on **Finish** to complete the installation.



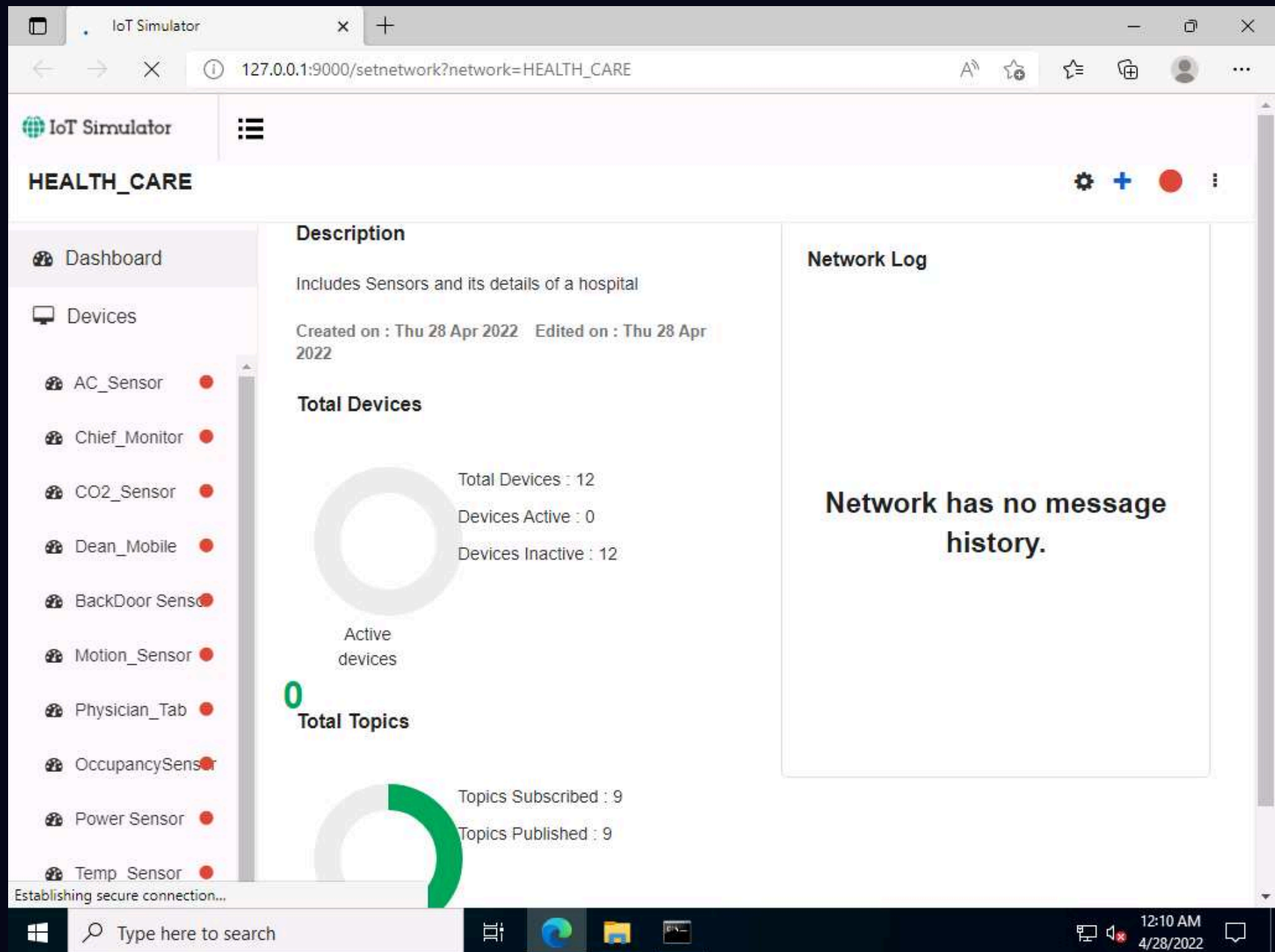
20. Bevywise IoT Simulator is installed successfully. To launch the **IoT simulator**, navigate to the **C:\Bevywise\IoT Simulator\bin** directory and double-click on the **runsimulator.bat** file.



21. 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**.

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

22. 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.



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

IoT Simulator

HEALTH_CARE

Dashboard

Devices

- AC_Sensor
- Chief_Monitor
- CO2_Sensor
- Dean_Mobile
- BackDoor Sensor
- Motion_Sensor
- Physician_Tab
- OccupancySensor
- Power Sensor
- Temp_Sensor

Includes Sensors and its details of a hospital

Created on : Thu 28 Apr 2022 Edited on : Thu 28 Apr 2022

Total Devices

Total Devices : 12
Devices Active : 0
Devices Inactive : 12

Active devices

Total Topics

Topics Subscribed : 9
Topics Published : 9

Network Log

Network has no message history.

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

IoT Simulator

HEALTH_CARE

Dashboard

Devices

- AC_Sensor
- Chief_Monitor
- CO2_Sensor
- Dean_Mobile
- BackDoor Sensor
- Motion_Sensor
- Physician_Tab
- OccupancySensor
- Power Sensor
- Temp_Sensor

Create New Network

Name: CEH_FINANCE_NETWORK

Description: CEH_FINANCE_NETWORK contains IoT devices for Finance department

Create

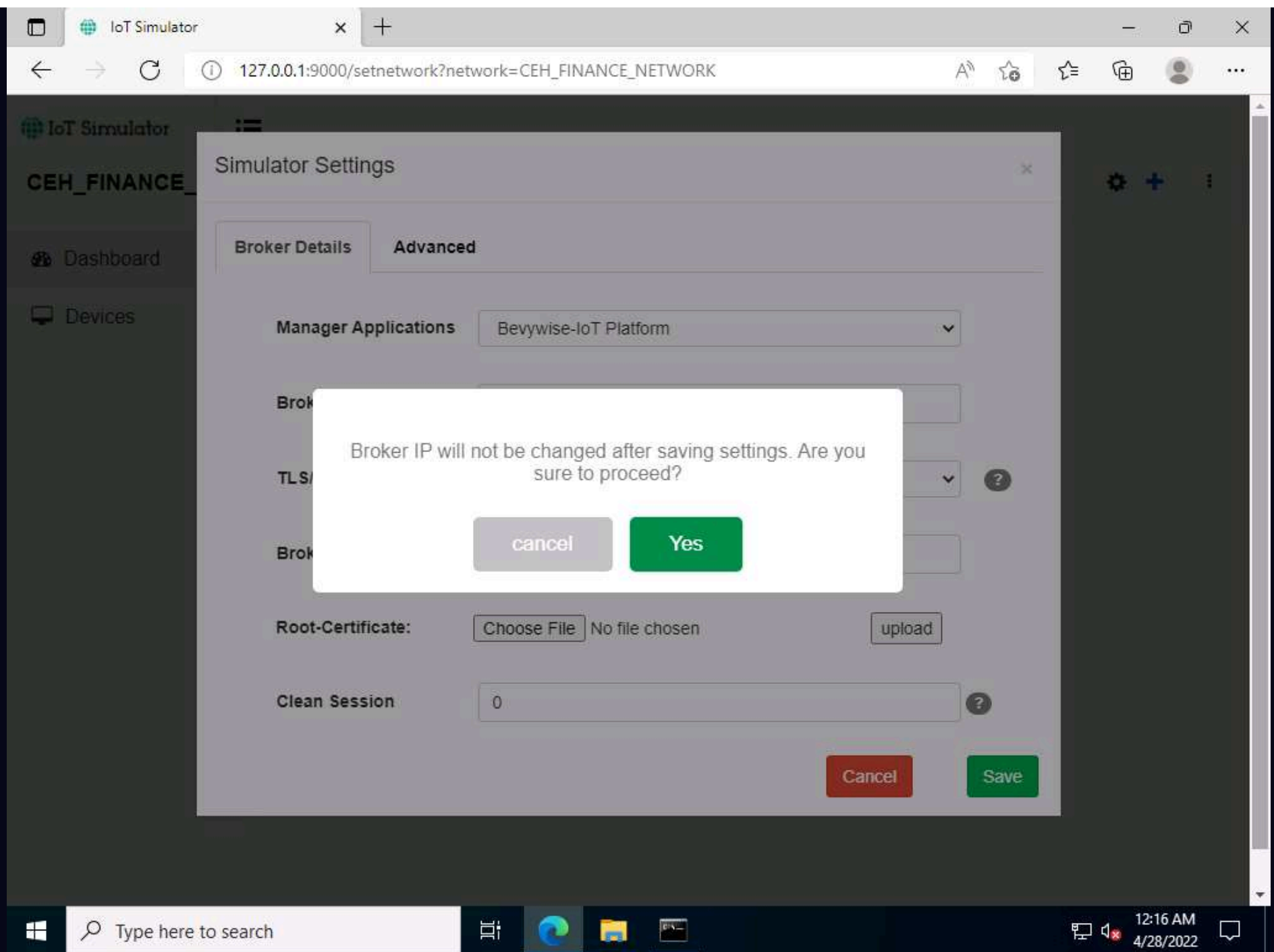
25. In the next screen, we will setup the **Simulator Settings**. Set the **Broker IP Address** as **10.10.1.19** (the IP address of the **Windows Server 2019**). Since we have installed the Broker on the web server, the created network will interact with the server using MQTT Broker. Do not change default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

The screenshot shows a web browser window with the URL `127.0.0.1:9000/setnetwork?network=CEH_FINANCE_NETWORK`. The page displays the 'IoT Simulator' interface with a sidebar containing 'Dashboard' and 'Devices'. A 'Simulator Settings' dialog box is open, showing the 'Advanced' tab. The settings are as follows:

| Field | Value |
|----------------------|-----------------------------------|
| Manager Applications | Bevywise-IoT Platform |
| Broker IP Address | 10.10.1.19 |
| TLS/SSL | Disabled |
| Broker Port | 1883 |
| Root-Certificate: | Choose File No file chosen upload |
| Clean Session | 0 |

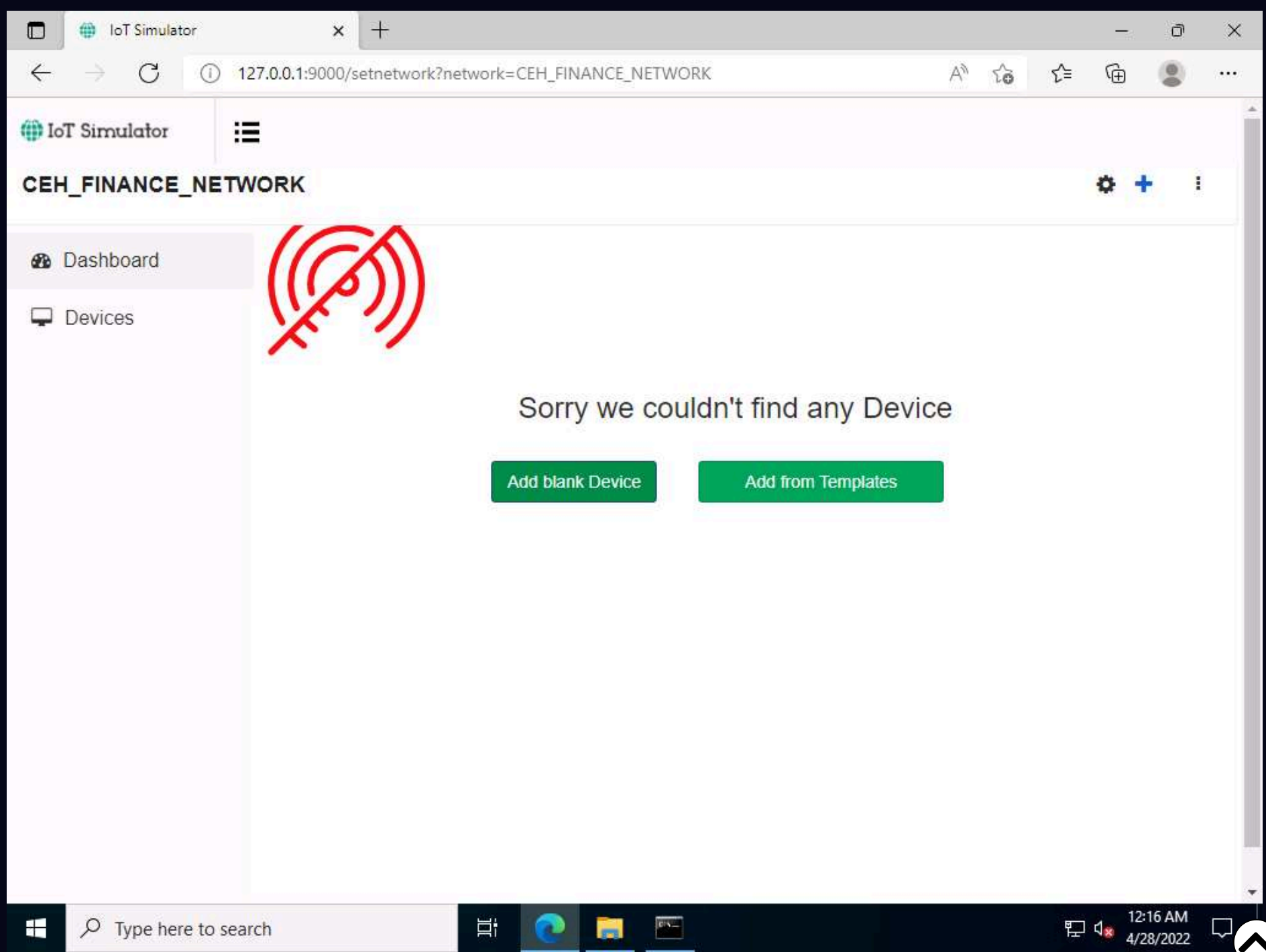
At the bottom of the dialog box, there are 'Cancel' and 'Save' buttons. The Windows taskbar at the bottom shows the time as 12:16 AM on 4/28/2022.

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



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

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



28. 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**.

The screenshot shows a web browser window with the URL `127.0.0.1:9000/setnetwork?network=CEH_FINANCE_NETWORK`. The page title is "IoT Simulator". A sidebar on the left contains links for "Dashboard" and "Devices". The main content area is titled "CEH_FINANCE_NETWORK". A "Create New Device" modal form is open in the center. The form has three input fields: "Device Name" with the value "Temperature_Sensor", "Device Id" with the value "TS1", and "Description" with the value "This is a Temperature Sensor IoT device". At the bottom right of the form are "Cancel" and "Save" buttons. The Windows taskbar at the bottom shows the search bar, task view, and several open applications. The system clock in the bottom right corner displays "12:17 AM 4/28/2022".

IoT Simulator

CEH_FINANCE_NETWORK

Dashboard

Devices

Create New Device

Device Name: Temperature_Sensor

Device Id: TS1

Description: This is a Temperature Sensor IoT device

Cancel Save

Type here to search

12:17 AM 4/28/2022

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

IoT Simulator

CEH_FINANCE_NETWORK

Dashboard

Devices

Temperature_Sens

Temperature_Sensor

Client id
TS1

Description
This is a Temperature Sensor IoT device

Authentication

Will

Device Log

Device has no message history.

Events

| Event Topic | Event Data |
|-------------|------------|
|-------------|------------|

30. 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.

IoT Simulator

CEH_FINANCE_NETWORK

Dashboard

Devices

Temperature_Sens

Temperature_Sensor

Client id
TS1

Description
This is a Temperature Sensor IoT device

Authentication

Will

Device Log

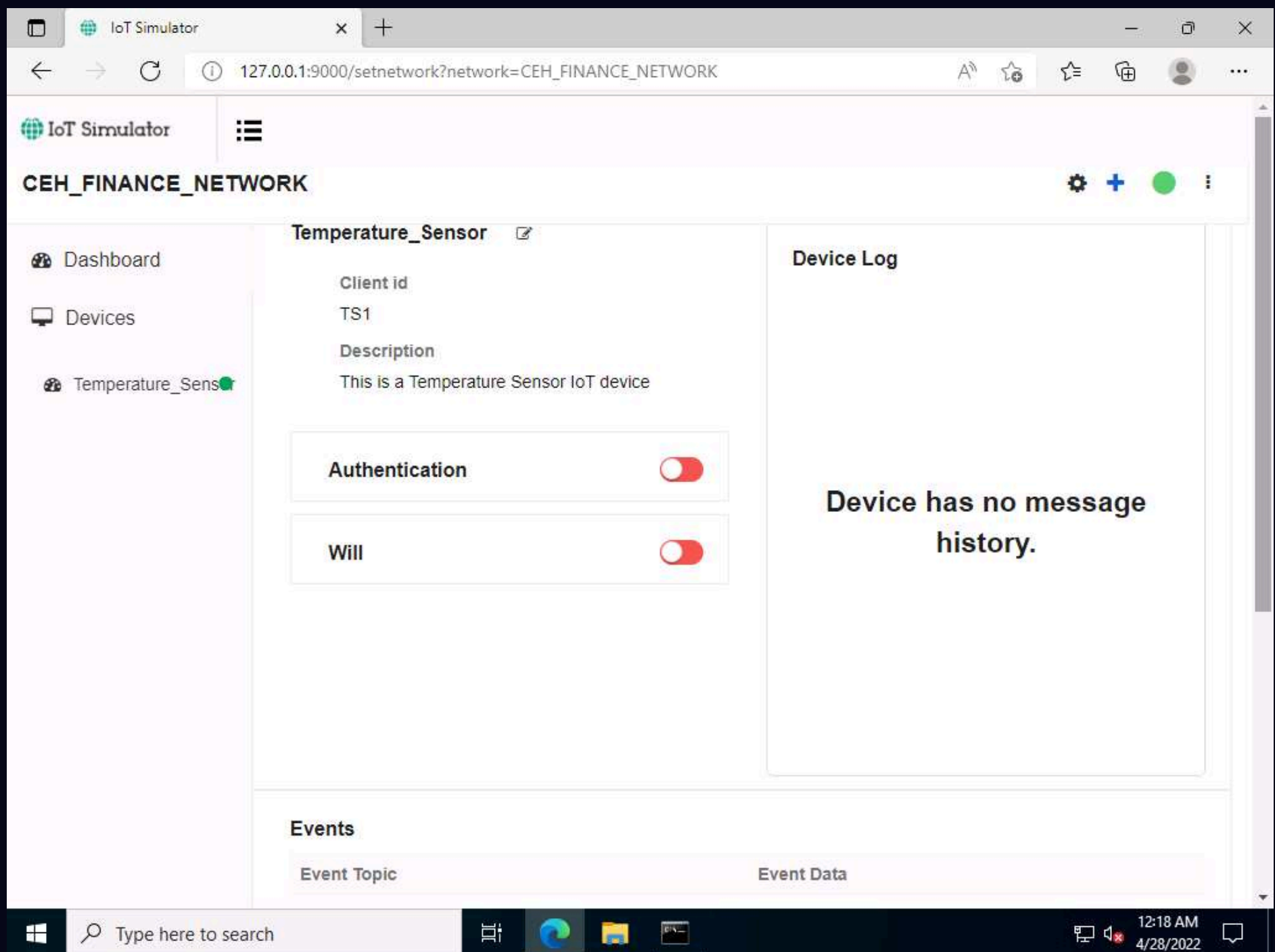
Device has no message history.

Events

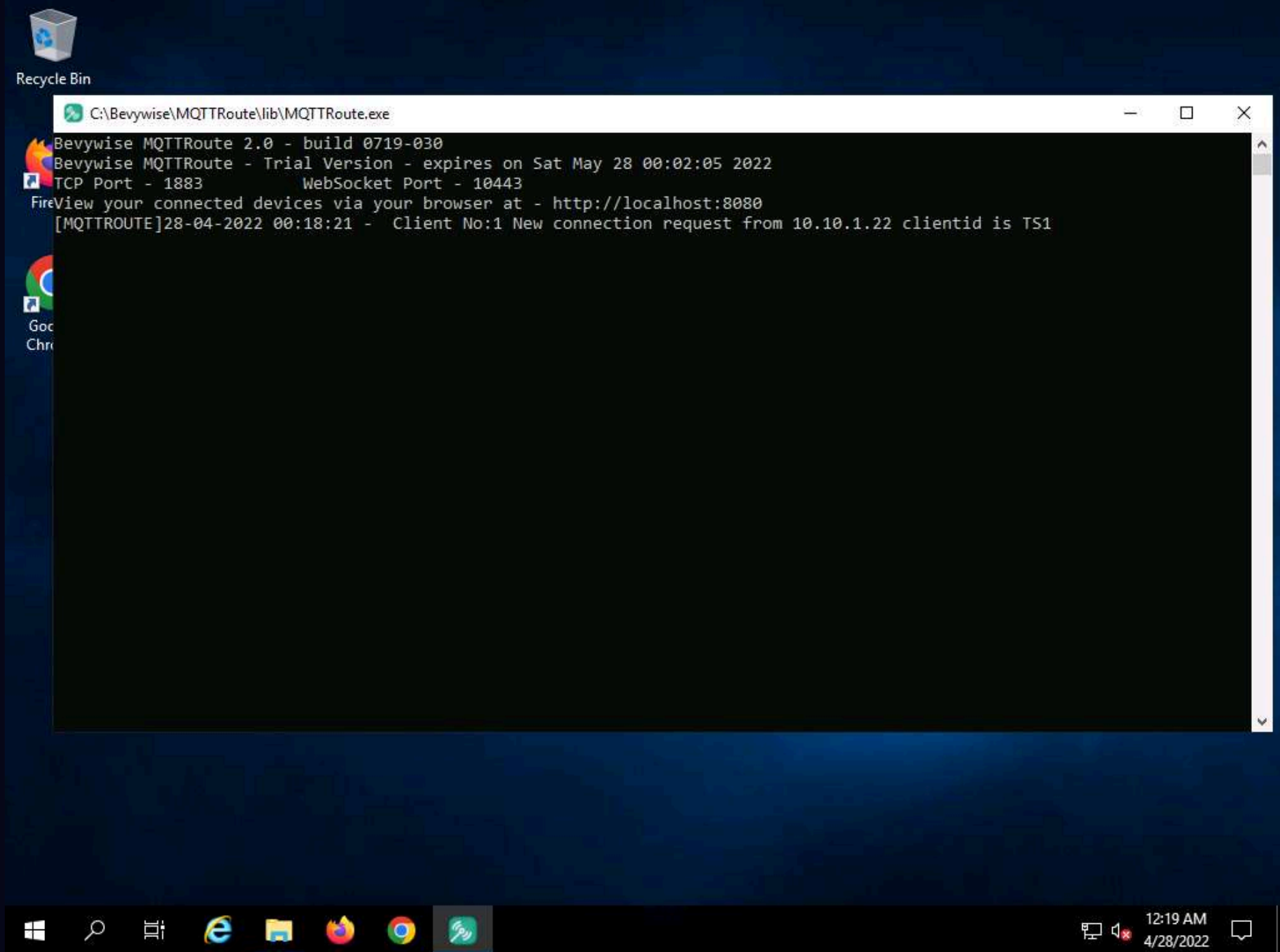
| Event Topic | Event Data |
|-------------|------------|
|-------------|------------|

Start Network

31. 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**.



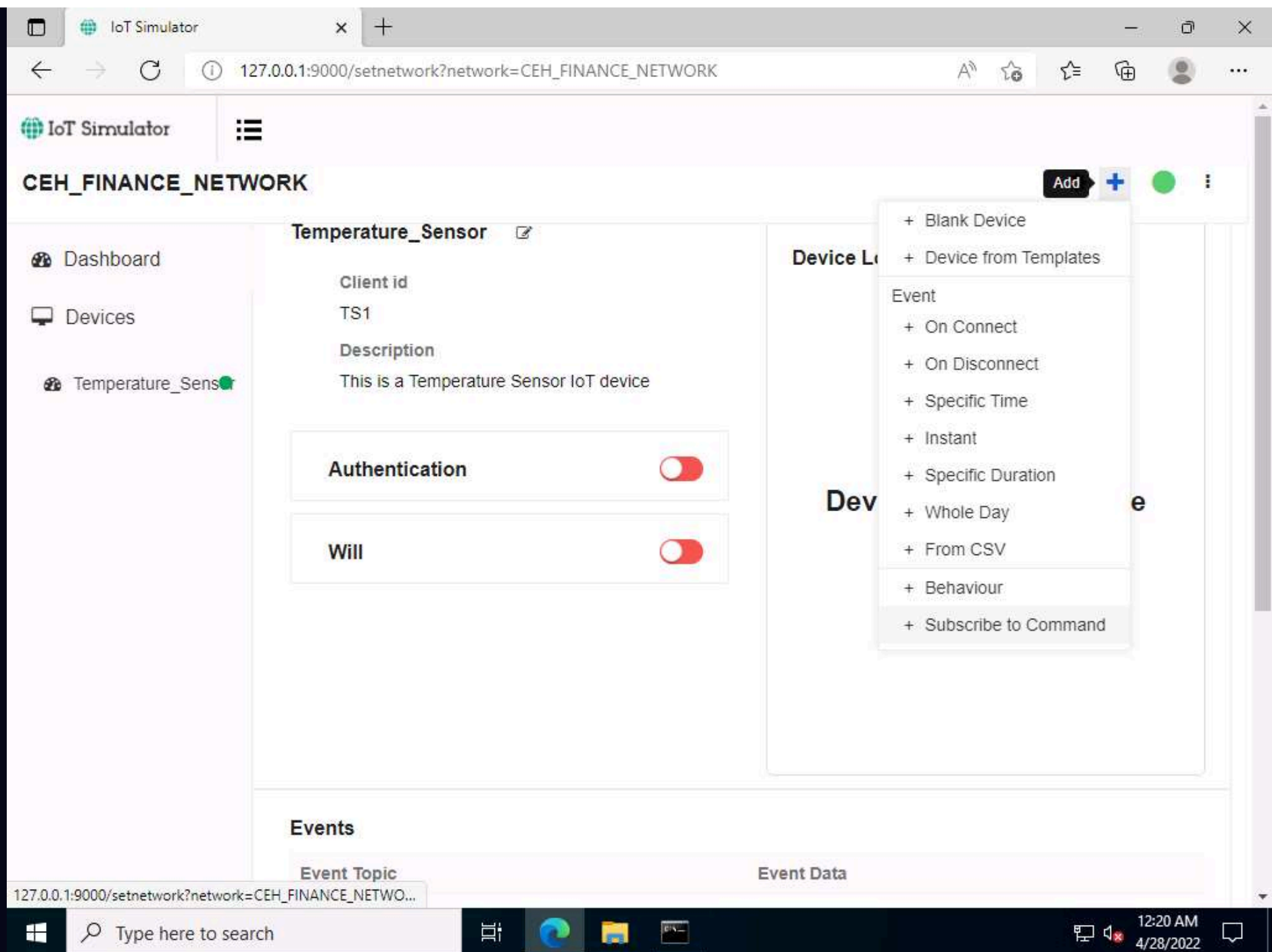
32. Next, switch to the **CEHv12 Windows Server 2019** machine. Since the Broker was **left running**, you can see a connection request from machine **10.10.1.22** for the device **TS1**.



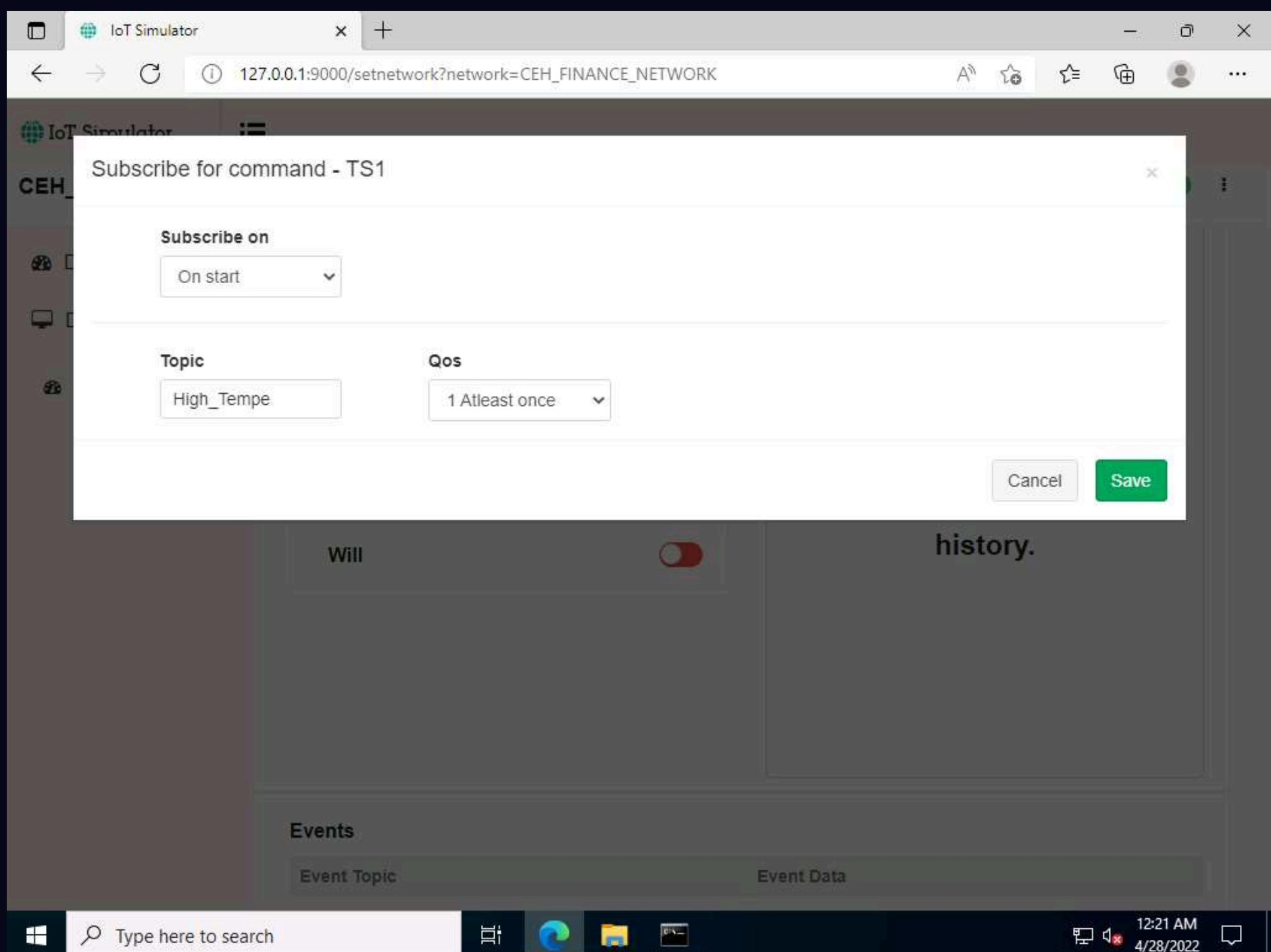
33. Switch back to **CEHv12 Windows Server 2022** machine.

34. Next, we will create the **Subscribe command** for the device Temperature_Sensor.

35. Click on the **Plus** icon in **the top right corner** and select the **Subscribe to Command** option.



36. 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**.



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

The screenshot shows the IoT Simulator web interface in a browser window. The address bar displays the URL `127.0.0.1:9000/setnetwork?network=CEH_FINANCE_NETWORK`. The page title is "CEH_FINANCE_NETWORK". On the left sidebar, there are links for "Dashboard", "Devices", and "Temperature_Sens" (which is highlighted with a green dot). The main content area is divided into three sections:

- Events**: A table with headers "Event Topic" and "Event Data". The content below the header is "No Event is configured".
- Subscribe to Commands**: A table with headers "Topic", "Qos", and "Time". It contains one row with the values "High_Tempe", "1-Atleast Once", and "On Start". There is a trash icon to the right of the row.
- Behaviour**: A table with headers "Command" and "Event". The content below the header is "No Behavior Simulation".

The Windows taskbar at the bottom shows the search bar with the text "Type here to search", and the system clock displays "12:21 AM 4/28/2022".

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

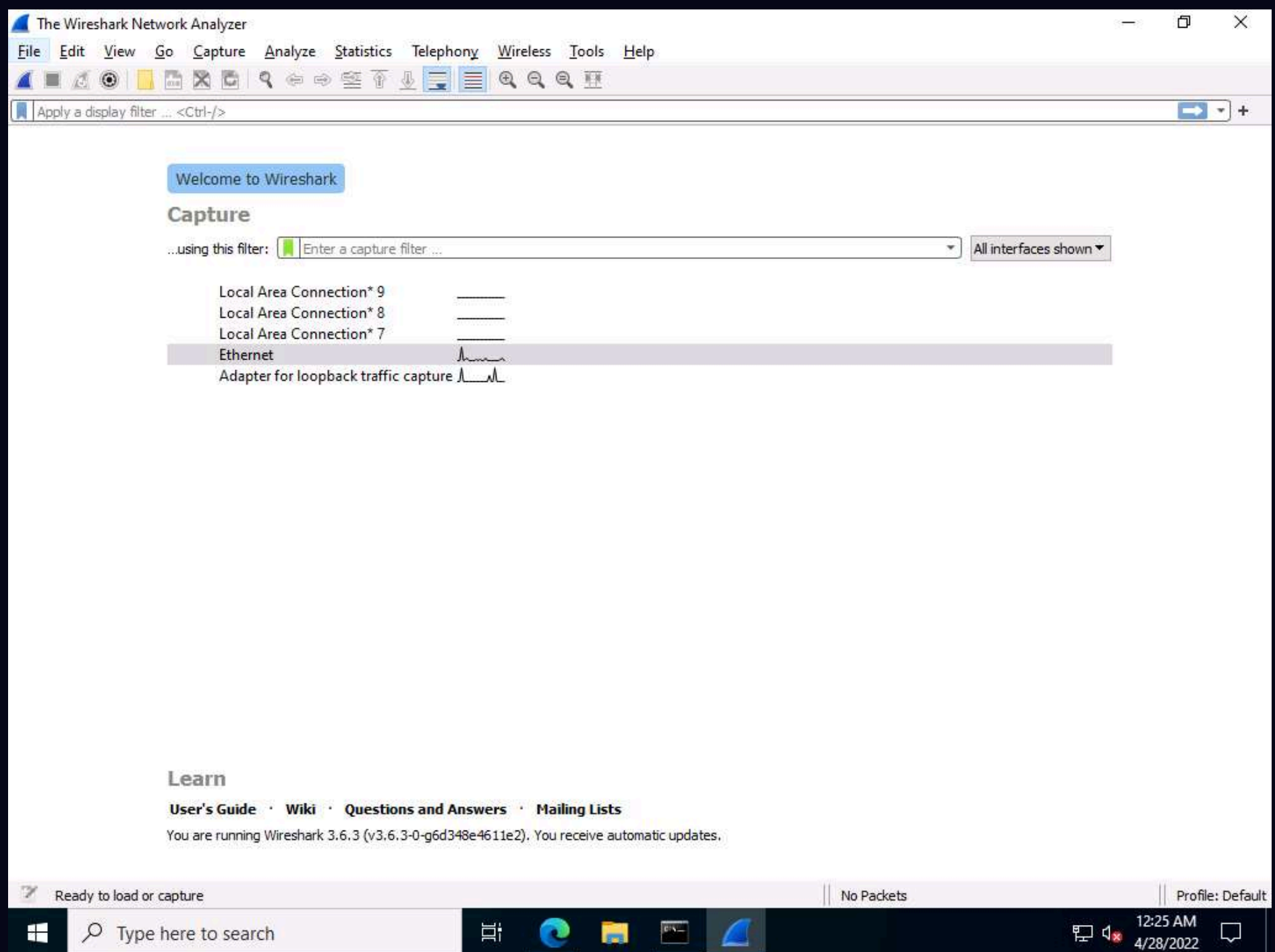
39. Minimise the Edge browser. Click on **Type here to search** at the bottom left of the desktop, type **wireshark** and select **Wireshark** from the results to launch the **Wireshark** from the application list.

The screenshot shows the Windows desktop with a blue background. On the left side, there are icons for the Recycle Bin, Google Chrome, and Firefox. A search window is open, displaying the results for the search term "wireshark". The results show "Best match" and "Wireshark App". The Windows taskbar at the bottom shows the search bar with the text "wireshark", and the system clock displays "12:24 AM 4/28/2022".

40. The Wireshark Application window appears, select the **Ethernet** as interface.

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

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

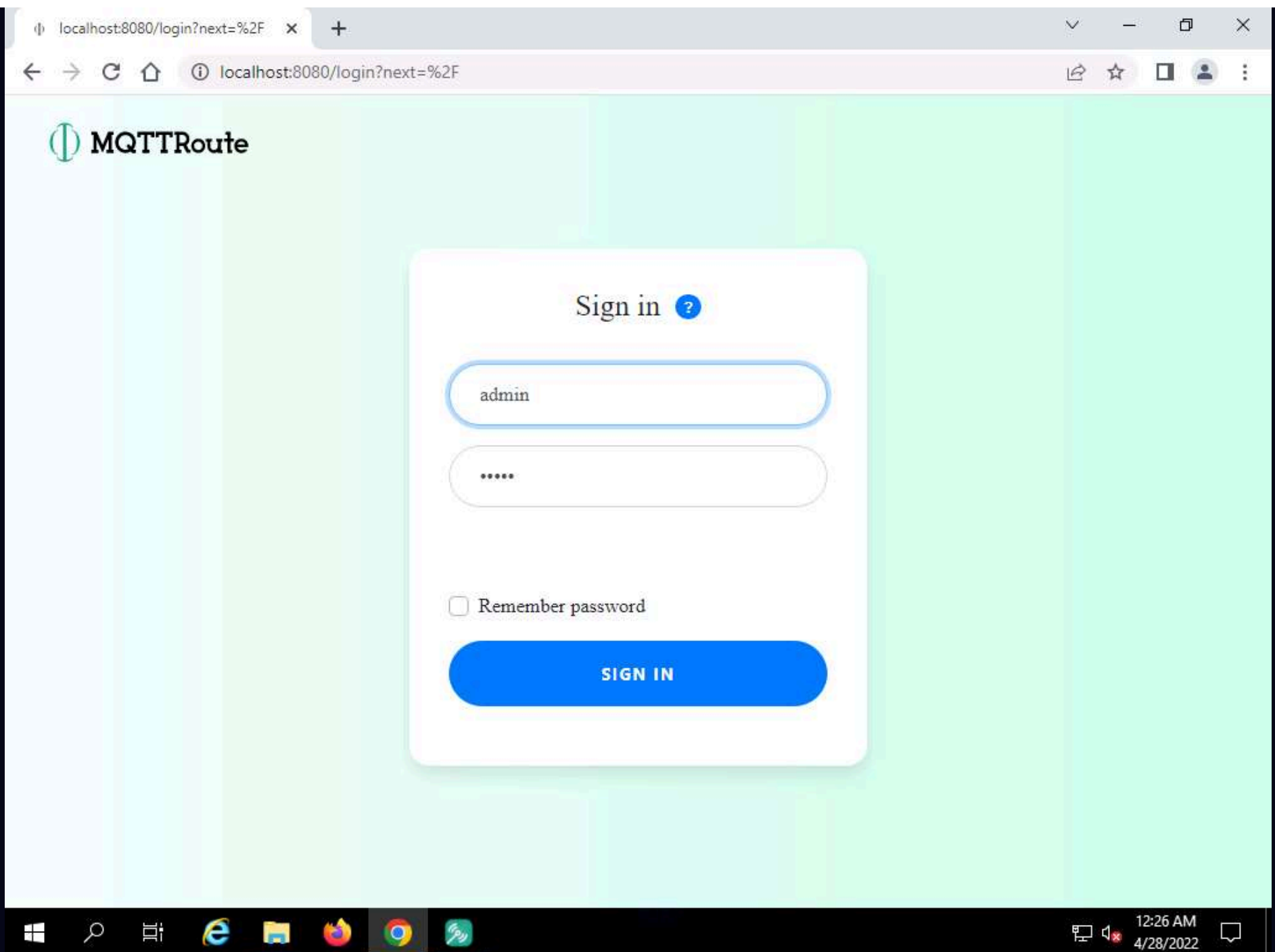


41. Click on the **Start Wireshark** icon to start the capturing packets, leave the Wireshark running.

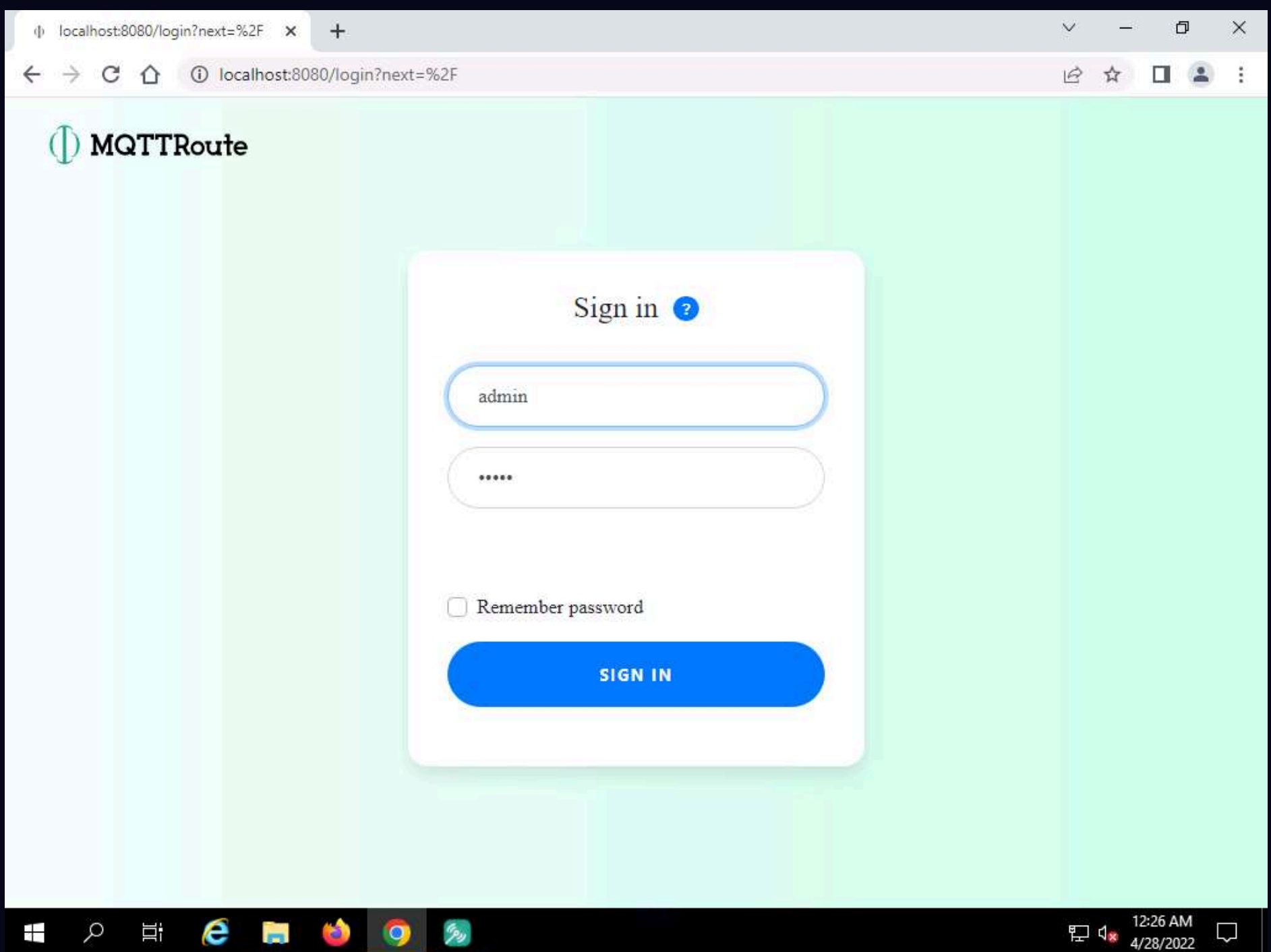
42. Leave the IoT simulator running and switch to the **CEHv12 Windows Server 2019** machine.

43. Minimise all opened applications and windows, Open Chrome browser, type **http://localhost:8080** and press **Enter**.

Note: Do not use Internet Explorer web browser to open the above URL.



44. As soon as you press **Enter**, the **MQTTRoute Sign in** page appears, keep the default credential unchanged and click on **SIGN IN**.



45. Navigate to **Devices** menu. You will be able to see the connected device **TS1** in the left pane.

MQTTRoute

Dashboard Devices Topic Message Rules Error Log Authentication MQTT Clients Tour

Devices List

TS1

| Device Property | Value |
|-----------------|---------------------|
| Client Name | TS1 |
| From IP Address | 10.10.1.22 |
| Connected On | 28 Apr 2022 0:18:21 |
| WILL Topic | NIL |
| WILL Message | NIL |
| WILL Retain | NIL |

Topic: Select Topic

Message: [Input Field]

Send

Messages Received

| Topic | Message | QoS |
|-------|---------|-----|
| NIL | NIL | NIL |

Subscribed Topics

| Subscribed Topics | QoS |
|-------------------|-----|
|-------------------|-----|

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

47. Go to the **Command Send** section, select **Topic** as **High_Tempe**, type **Alert for High Temperature** and click on the **Send** button.

MQTTRoute

Dashboard Devices Topic Message Rules Error Log Authentication MQTT Clients Tour

Devices List

TS1

Device Details Connection Status :Online

| Device Property | Value |
|-----------------|---------------------|
| Client Name | TS1 |
| From IP Address | 10.10.1.22 |
| Connected On | 28 Apr 2022 0:18:21 |
| WILL Topic | NIL |
| WILL Message | NIL |
| WILL Retain | NIL |

Command Send

Topic: High_Tempe

Message: Alert for High Temperature

Send

Messages Received

| Topic | Message | QoS |
|-------|---------|-----|
| NIL | NIL | NIL |

Subscribed Topics

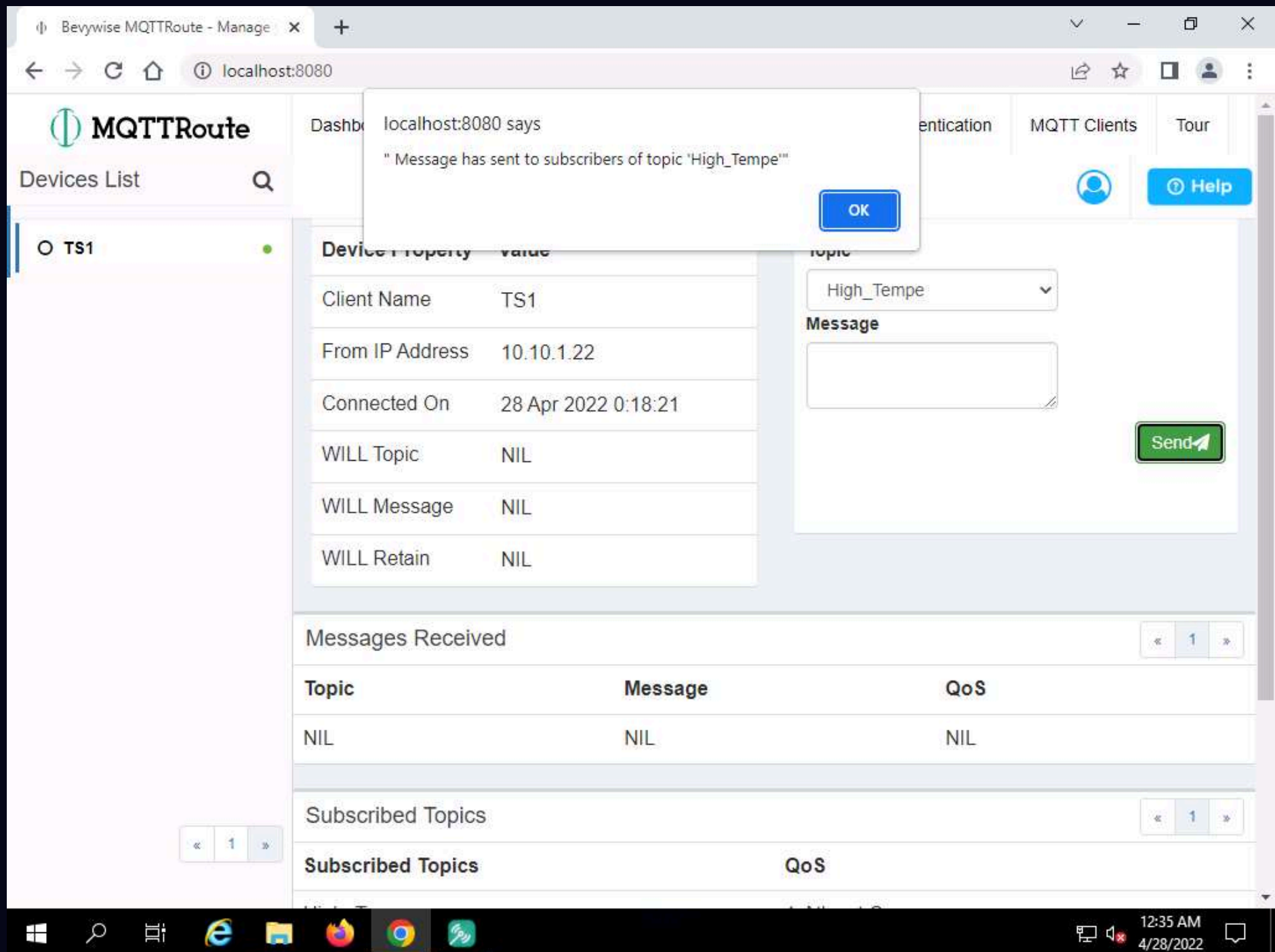
| Subscribed Topics | QoS |
|-------------------|----------------|
| High_Tempe | 1-Atleast Once |

Messages Published

| Time | Topic | Message |
|------|-------|---------|
|------|-------|---------|

NO MESSAGE LOG

48. The alert popup appears, then click on **OK**.



49. The message has been sent to the device using this topic.

50. Next, switch to **CEHv12 Windows Server 2022** machine.

51. 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.

IoT Simulator

CEH_FINANCE_NETWORK

Dashboard

Devices

Temperature_Sens

Temperature_Sensor

Client id
TS1

Description
This is a Temperature Sensor IoT device

Authentication

Will

Device Log

Command received by Device TS1
Thu Apr 28 00:34:47 2022

Events

Event Topic

Event Data

52. You can see the alert message **"Alert for High Temperature"**

IoT Simulator

CEH_FINANCE_NETWORK

Dashboard

Devices

Temperature_Sens

Temperature_Sensor

Client id
TS1

Description
This is a Temperature Sensor IoT device

Authentication

Will

Device Log

Command received by Device TS1
Thu Apr 28 00:34:47 2022

Topic
High_Tempe

Message
"Alert for High Temperature"

Events

Event Topic

Event Data

53. To verify the communication, we have executed **Wireshark** application, switch to the Wireshark traffic capturing window.

54. Type **mqtt** under the **filter** field and press **Enter**. To display only the MQTT protocol packets.

The screenshot shows the Wireshark interface with the filter field set to **mqtt**. The packet list displays several MQTT packets, including Ping Requests and Responses, and a Publish Message. The packet details pane shows the structure of the selected packet, including Ethernet II, Internet Protocol Version 4, Transmission Control Protocol, and MQ Telemetry Transport Protocol.

| No. | Time | Source | Destination | Protocol | Length | Info |
|------|------------|------------|-------------|----------|--------|-------------------------------------|
| 92 | 55.375177 | 10.10.1.22 | 10.10.1.19 | MQTT | 56 | Ping Request |
| 93 | 55.375596 | 10.10.1.19 | 10.10.1.22 | MQTT | 56 | Ping Response |
| 225 | 114.075899 | 10.10.1.22 | 10.10.1.19 | MQTT | 56 | Ping Request |
| 226 | 114.077353 | 10.10.1.19 | 10.10.1.22 | MQTT | 56 | Ping Response |
| 336 | 171.798721 | 10.10.1.22 | 10.10.1.19 | MQTT | 56 | Ping Request |
| 337 | 171.798904 | 10.10.1.19 | 10.10.1.22 | MQTT | 56 | Ping Response |
| 454 | 229.566708 | 10.10.1.22 | 10.10.1.19 | MQTT | 56 | Ping Request |
| 455 | 229.566990 | 10.10.1.19 | 10.10.1.22 | MQTT | 56 | Ping Response |
| 570 | 288.319887 | 10.10.1.22 | 10.10.1.19 | MQTT | 56 | Ping Request |
| 571 | 288.320127 | 10.10.1.19 | 10.10.1.22 | MQTT | 56 | Ping Response |
| 676 | 345.830189 | 10.10.1.22 | 10.10.1.19 | MQTT | 56 | Ping Request |
| 677 | 345.830443 | 10.10.1.19 | 10.10.1.22 | MQTT | 56 | Ping Response |
| 802 | 404.300820 | 10.10.1.22 | 10.10.1.19 | MQTT | 56 | Ping Request |
| 803 | 404.301034 | 10.10.1.19 | 10.10.1.22 | MQTT | 56 | Ping Response |
| 913 | 461.756481 | 10.10.1.22 | 10.10.1.19 | MQTT | 56 | Ping Request |
| 914 | 461.756762 | 10.10.1.19 | 10.10.1.22 | MQTT | 56 | Ping Response |
| 1034 | 520.313102 | 10.10.1.22 | 10.10.1.19 | MQTT | 56 | Ping Request |
| 1035 | 520.313585 | 10.10.1.19 | 10.10.1.22 | MQTT | 56 | Ping Response |
| 1148 | 576.746486 | 10.10.1.19 | 10.10.1.22 | MQTT | 96 | Publish Message (id=2) [High_Tempe] |
| 1149 | 576.746978 | 10.10.1.22 | 10.10.1.19 | MQTT | 58 | Publish Ack (id=2) |
| 1151 | 576.756810 | 10.10.1.22 | 10.10.1.19 | MQTT | 58 | Publish Received (id=2) |

Frame 92: 56 bytes on wire (448 bits), 56 bytes captured (448 bits) on interface \Device\NPF_{6FB33290-5CC6-4897-B630-59BC293DB8D9}, id 0
 Ethernet II, Src: Microsof_01:80:02 (00:15:5d:01:80:02), Dst: Microsof_75:f7:7f (00:15:5d:75:f7:7f)
 Internet Protocol Version 4, Src: 10.10.1.22, Dst: 10.10.1.19
 Transmission Control Protocol, Src Port: 64264, Dst Port: 1883, Seq: 1, Ack: 1, Len: 2
 MQ Telemetry Transport Protocol, Ping Request

0000 00 15 5d 75 f7 7f 00 15 5d 01 80 02 08 00 45 02 ..]u...].....E
 0010 00 2a 66 e9 40 00 80 06 00 00 0a 0a 01 16 0a 0a *f.@.....
 0020 01 13 fb 08 07 5b 05 a6 d3 c7 90 24 6b b2 50 18[...\$k.P
 0030 04 02 16 59 00 00 c0 00 ...Y....

MQ Telemetry Transport Protocol: Protocol | Packets: 1509 · Displayed: 31 (2.1%) | Profile: Default

55. 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.

56. Under the **MQ Telemetry Transport Protocol** nodes, you can observe details such as **Msg Len**, **Topic Length**, **Topic**, and **Message**.

57. Publish Message can be used to obtain the message sent by the MQTT client to the broker.

The screenshot shows the Wireshark interface with the following details:

- Packet List:** A table of captured packets. Packet 1619 is selected, showing it is an MQTT Ping Response from 10.10.1.19 to 10.10.1.22.
- Packet Details:**
 - Acknowledgment Number:** 19 (relative ack number)
 - Acknowledgment number (raw):** 94819289
 - 0101 = Header Length:** 20 bytes (5)
 - Flags:** 0x018 (PSH, ACK)
 - Window:** 8212
 - [Calculated window size:** 8212]
 - [Window size scaling factor:** -1 (unknown)]
 - Checksum:** 0x9794 [unverified]
 - [Checksum Status:** Unverified]
 - Urgent Pointer:** 0
 - [Timestamps]**
 - [SEQ/ACK analysis]**
 - TCP payload (42 bytes)**
 - [PDU Size:** 42]
 - MQ Telemetry Transport Protocol, Publish Message**
 - [Expert Info (Note/Protocol):** Unknown version (missing the CONNECT packet?)]
 - Header Flags:** 0x32, Message Type: Publish Message, QoS Level: At least once delivery (Acknowledged deliver)
 - Msg Len:** 40
 - Topic Length:** 10
 - Topic:** High_Tempe
 - Message Identifier:** 2
 - Message:** 416c657274206666f7220486967682054656d7065726174757265
- Packet Bytes:** Hex and ASCII representation of the packet data.
- Status Bar:** MQ Telemetry Transport Protocol: Protocol | Packets: 1928 · Displayed: 35 (1.8%) | Profile: Default

Note: 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.

58. 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.

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

Packet List:

| No. | Time | Source | Destination | Protocol | Length | Info |
|------|------------|------------|-------------|----------|--------|-------------------------------------|
| 1148 | 576.746486 | 10.10.1.19 | 10.10.1.22 | MQTT | 96 | Publish Message (id=2) [High_Tempe] |
| 1149 | 576.746978 | 10.10.1.22 | 10.10.1.19 | MQTT | 58 | Publish Ack (id=2) |
| 1151 | 576.756810 | 10.10.1.22 | 10.10.1.19 | MQTT | 58 | Publish Received (id=2) |
| 1152 | 576.757609 | 10.10.1.19 | 10.10.1.22 | MQTT | 58 | Publish Release (id=2) |
| 1153 | 576.757665 | 10.10.1.22 | 10.10.1.19 | MQTT | 58 | Publish Complete (id=2) |
| 1155 | 577.778010 | 10.10.1.22 | 10.10.1.19 | MQTT | 56 | Ping Request |
| 1156 | 577.778251 | 10.10.1.19 | 10.10.1.22 | MQTT | 56 | Ping Response |
| 1276 | 636.334100 | 10.10.1.22 | 10.10.1.19 | MQTT | 56 | Ping Request |
| 1277 | 636.334405 | 10.10.1.19 | 10.10.1.22 | MQTT | 56 | Ping Response |

Packet Details:

- Sequence Number: 61 (relative sequence number)
- Sequence Number (raw): 2418306030
- [Next Sequence Number: 65 (relative sequence number)]
- Acknowledgment Number: 27 (relative ack number)
- Acknowledgment number (raw): 94819297
- 0101 = Header Length: 20 bytes (5)
- Flags: 0x018 (PSH, ACK)
- Window: 8212
- [Calculated window size: 8212]
- [Window size scaling factor: -1 (unknown)]
- Checksum: 0x3f75 [unverified]
- [Checksum Status: Unverified]
- Urgent Pointer: 0
- [Timestamps]
- [SEQ/ACK analysis]
- TCP payload (4 bytes)
- [PDU Size: 4]
- MQ Telemetry Transport Protocol, Publish Release**
 - [Expert Info (Note/Protocol): Unknown version (missing the CONNECT packet?)]
 - Header Flags: 0x62, Message Type: Publish Release
 - Msg Len: 2
 - Message Identifier: 2

Packet Bytes:

```

0000  00 15 5d 01 80 02 00 15 5d 75 f7 7f 08 00 45 02  ..].....]u....E-
0010  00 2c 7d 5f 40 00 80 06 67 2e 0a 0a 01 13 0a 0a  .,}_@...g.....
  
```

MQ Telemetry Transport Protocol: Protocol | Packets: 2015 · Displayed: 37 (1.8%) | Profile: Default

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

60. 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.

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

Wireshark packet capture analysis of MQTT traffic. The packet list shows a 'Publish Complete' packet (No. 1153) from 10.10.1.22 to 10.10.1.19. The packet details pane shows the 'MQ Telemetry Transport Protocol, Publish Complete' node expanded, displaying fields like 'Msg Len: 2' and 'Message Identifier: 2'. The packet bytes pane shows the raw data in hexadecimal and ASCII.

| No. | Time | Source | Destination | Protocol | Length | Info |
|------|------------|------------|-------------|----------|--------|-------------------------|
| 1153 | 576.757665 | 10.10.1.22 | 10.10.1.19 | MQTT | 58 | Publish Complete (id=2) |
| 1155 | 577.778010 | 10.10.1.22 | 10.10.1.19 | MQTT | 56 | Ping Request |
| 1156 | 577.778251 | 10.10.1.19 | 10.10.1.22 | MQTT | 56 | Ping Response |
| 1276 | 636.334100 | 10.10.1.22 | 10.10.1.19 | MQTT | 56 | Ping Request |
| 1277 | 636.334405 | 10.10.1.19 | 10.10.1.22 | MQTT | 56 | Ping Response |

Destination Port: 1883
 [Stream index: 1]
 [Conversation completeness: Incomplete (12)]
 [TCP Segment Len: 4]
 Sequence Number: 27 (relative sequence number)
 Sequence Number (raw): 94819297
 [Next Sequence Number: 31 (relative sequence number)]
 Acknowledgment Number: 65 (relative ack number)
 Acknowledgment number (raw): 2418306034
 0101 = Header Length: 20 bytes (5)
 > Flags: 0x018 (PSH, ACK)
 Window: 1026
 [Calculated window size: 1026]
 [Window size scaling factor: -1 (unknown)]
 Checksum: 0x165b [unverified]
 [Checksum Status: Unverified]
 Urgent Pointer: 0
 > [Timestamps]
 > [SEQ/ACK analysis]
 TCP payload (4 bytes)
 [PDU Size: 4]
 > MQ Telemetry Transport Protocol, Publish Complete
 > [Expert Info (Note/Protocol): Unknown version (missing the CONNECT packet?)]
 > Header Flags: 0x70, Message Type: Publish Complete
 Msg Len: 2
 Message Identifier: 2

0000 00 15 5d 75 f7 7f 00 15 5d 01 80 02 08 00 45 02 ..]u....].....E-
 0010 00 2c 66 fd 40 00 80 06 00 00 0a 0a 01 16 0a 0a .,f.@.....

MQ Telemetry Transport Protocol: Protocol | Packets: 2129 · Displayed: 39 (1.8%) | Profile: Default

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

62. 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.
63. Under the **MQ Telemetry Transport Protocol** nodes, you can observe details such as **Message Type**, **Msg Len** and **Message Identifier**.

Wireshark interface showing MQTT packet capture details. The packet list displays several MQTT messages between 10.10.1.19 and 10.10.1.22. The selected packet (No. 1151) is a 'Publish Received' message. The packet details pane shows TCP segment information and MQTT-specific fields.

| No. | Time | Source | Destination | Protocol | Length | Info |
|------|------------|------------|-------------|----------|--------|-------------------------------------|
| 1148 | 576.746486 | 10.10.1.19 | 10.10.1.22 | MQTT | 96 | Publish Message (id=2) [High_Tempe] |
| 1149 | 576.746978 | 10.10.1.22 | 10.10.1.19 | MQTT | 58 | Publish Ack (id=2) |
| 1151 | 576.756810 | 10.10.1.22 | 10.10.1.19 | MQTT | 58 | Publish Received (id=2) |
| 1152 | 576.757609 | 10.10.1.19 | 10.10.1.22 | MQTT | 58 | Publish Release (id=2) |
| 1153 | 576.757665 | 10.10.1.22 | 10.10.1.19 | MQTT | 58 | Publish Complete (id=2) |

Packet Details (Selected Packet: 1151):

- [Conversation completeness: Incomplete (12)]
- [TCP Segment Len: 4]
- Sequence Number: 23 (relative sequence number)
- Sequence Number (raw): 94819293
- [Next Sequence Number: 27 (relative sequence number)]
- Acknowledgment Number: 61 (relative ack number)
- Acknowledgment number (raw): 2418306030
- 0101 = Header Length: 20 bytes (5)
- Flags: 0x018 (PSH, ACK)
- Window: 1026
- [Calculated window size: 1026]
- [Window size scaling factor: -1 (unknown)]
- Checksum: 0x165b [unverified]
- [Checksum Status: Unverified]
- Urgent Pointer: 0
- [Timestamps]
- [SEQ/ACK analysis]
- TCP payload (4 bytes)
- [PDU Size: 4]
- MQ Telemetry Transport Protocol, Publish Received
 - [Expert Info (Note/Protocol): Unknown version (missing the CONNECT packet?)]
 - Header Flags: 0x50, Message Type: Publish Received
 - 0101 = Message Type: Publish Received (5)
 - 0000 = Reserved: 0
 - Msg Len: 2
 - Message Identifier: 2

Packet Bytes:

```

0000  00 15 5d 75 f7 7f 00 15 5d 01 80 02 08 00 45 02  ..]u.... ].....E
0010  00 2c 66 fc 40 00 80 06 00 00 0a 0a 01 16 0a 0a  .,f.@.....
  
```

MQ Telemetry Transport Protocol: Protocol | Packets: 3268 · Displayed: 59 (1.8%) | Profile: Default

64. Similarly you can select **Ping Request**, **Ping Response** and **Publish Ack** packets and observe the details.

65. 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.

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