



SECURITY+ V4 LAB SERIES

Lab 13: IoT Management

Document Version: **2023-02-27**

Material in this Lab Aligns to the Following	
CompTIA Security+ (SY0-601) Exam Objectives	2.6: Explain the security implications of embedded and specialized systems
All-In-One CompTIA Security+ Sixth Edition ISBN-13: 978-1260464009 Chapters	14: Embedded and Specialized Systems

Copyright © 2023 Network Development Group, Inc.
www.netdevgroup.com

NETLAB+ is a registered trademark of Network Development Group, Inc.
KALI LINUX™ is a trademark of Offensive Security.
Microsoft®, Windows®, and Windows Server® are trademarks of the Microsoft group of companies.
VMware is a registered trademark of VMware, Inc.
SECURITY ONION is a trademark of Security Onion Solutions LLC.
Android is a trademark of Google LLC.
pfSense® is a registered mark owned by Electric Sheep Fencing LLC ("ESF").
All trademarks are property of their respective owners.

Contents

Introduction	3
Objective	3
Lab Topology	4
Lab Settings	5
1 Start the Service and Create Accounts	6
1.1 Start ThingsBoard Service	6
1.2 Create Different Accounts.....	9
2 Create and Add an IoT Device.....	15
2.2 Add a New Rule Chain	15
2.3 Add New Device	23
3 Testing the Device.....	26

Introduction

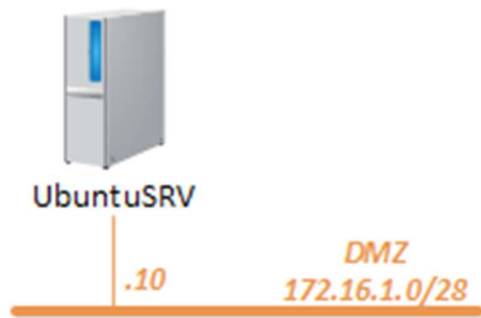
The Internet of Things (IoT) has become part of our daily lives. The more devices involved, the harder it is to secure them. In this lab, you will learn about IoT management.

Objective

In this lab, you will perform the following tasks:

- Be able to host your own IoT Central server
- Be able to send and monitor the IoT traffic

Lab Topology



Lab Settings

The information in the table below will be needed in order to complete the lab. The task sections below provide details on the use of this information.

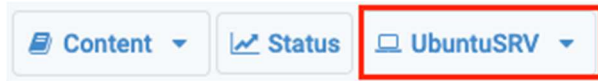
Virtual Machine	IP Address	Account (if needed)	Password (if needed)
UbuntuSRV	172.16.1.10	sysadmin	NDGlabpass123!

1 Start the Service and Create Accounts

1.1 Start ThingsBoard Service

In this section, you will start the ThingsBoard service.

1. Click on the **UbuntuSRV** tab to access the *UbuntuSRV*.



2. Log in to the *UbuntuSRV* as username `sysadmin`, password `NDGLabpass123!`.
3. Click the **Terminal** icon to start a *Terminal*.

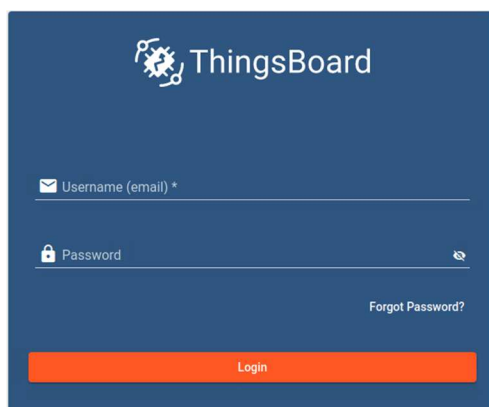


4. In the *Terminal*, type the following command to start the *ThingsBoard* server. When prompted for a password, type `NDGLabpass123!`

```
sysadmin@ubuntusrv:~$ sudo docker run -it -p 9090:9090 -p 1883:1883 -p 7070:7070 -p 5683-5688:5683-5688/udp -v ~/.mytb-data:/data -v ~/.mytb-logs:/var/log/thingsboard --name mytb --restart always thingsboard/tb-postgres
```

```
sysadmin@ubuntusrv:~$ sudo docker run -it -p 9090:9090 -p 1883:1883 -p 7070:7070 -p 5683-5688:5683-5688/udp -v ~/.mytb-data:/data -v ~/.mytb-logs:/var/log/thingsboard --name mytb --restart always thingsboard/tb-postgres
```

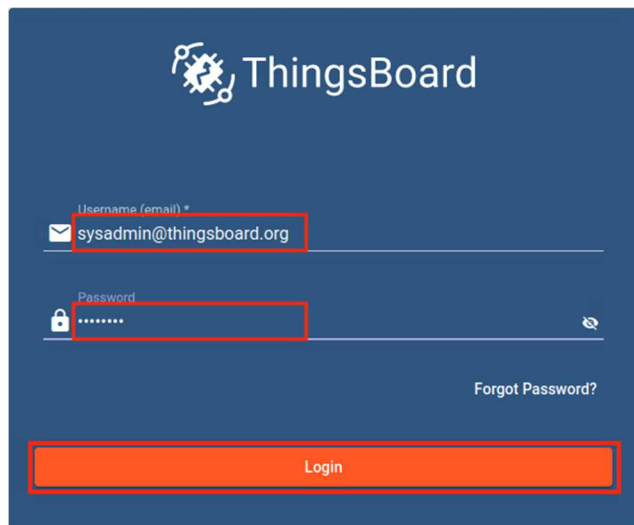
5. After a few minutes, the *ThingsBoard* should finish booting. You can check it by visiting the address `http://localhost:9090` in a browser. When the server is ready, you should see the login below.



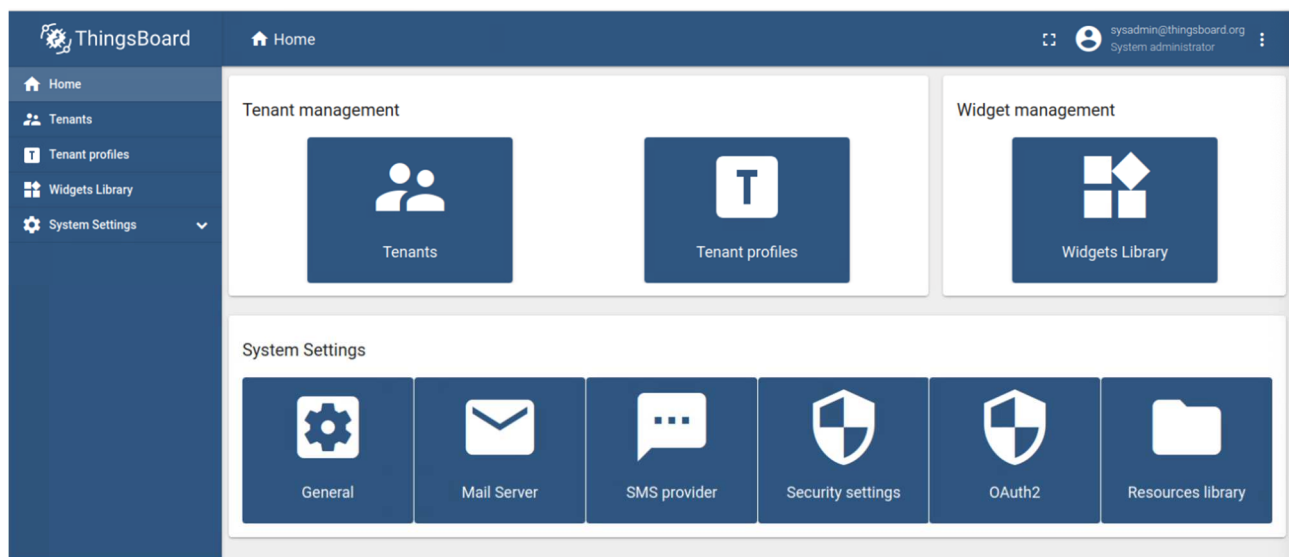
6. Fill in the default login information.

Username: `sysadmin@thingsboard.org`

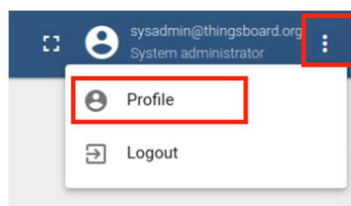
Password: `sysadmin`



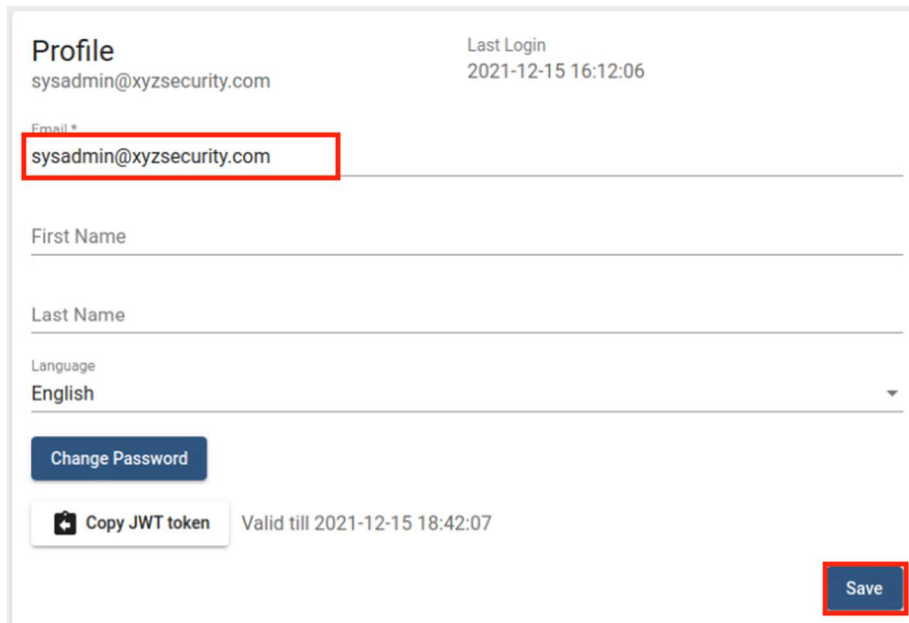
7. Once you log in, you should see something like the image below; you may need to change the window's size if the window is too small.



8. Click the menu button (three vertical dots) at the upper-right corner, and select **Profile**.

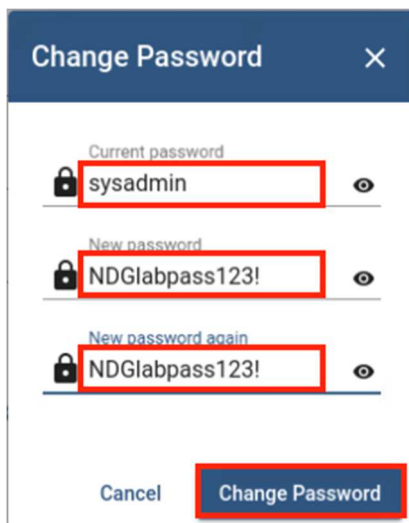


9. For security purposes, we will change the default login credentials. Click and change the *Email address* to `sysadmin@xyzsecurity.com` then click the **Save** button to save the change.



The image shows a 'Profile' form for a user named 'sysadmin@xyzsecurity.com'. The 'Email' field is highlighted with a red box and contains the text 'sysadmin@xyzsecurity.com'. Other fields include 'First Name', 'Last Name', and 'Language' (set to 'English'). A 'Change Password' button is visible. Below the form, there is a 'Copy JWT token' button and a token validity notice: 'Valid till 2021-12-15 18:42:07'. A 'Save' button is highlighted with a red box in the bottom right corner.

10. Then, click the **Change Password** button. In the pop-up window, fill in the current password `sysadmin` and type the new password `NDGlabpass123!` twice, then click **Change Password** to change it.



The image shows a 'Change Password' pop-up window. It has three input fields: 'Current password' with the value 'sysadmin', 'New password' with the value 'NDGlabpass123!', and 'New password again' with the value 'NDGlabpass123!'. Each field has a lock icon and an eye icon. At the bottom, there are 'Cancel' and 'Change Password' buttons. The 'Change Password' button is highlighted with a red box.

- Once you are brought back to the *Profile* page, click the menu at the upper-right corner again, and click **Logout** to log out of the current session. Notice the user email address is already changed.



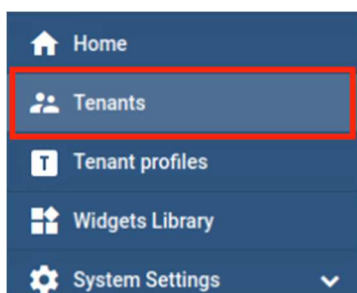
- Log back in using the new credentials.
username: `sysadmin@xyzsecurity.com`
password: `NDGlabpass123!`



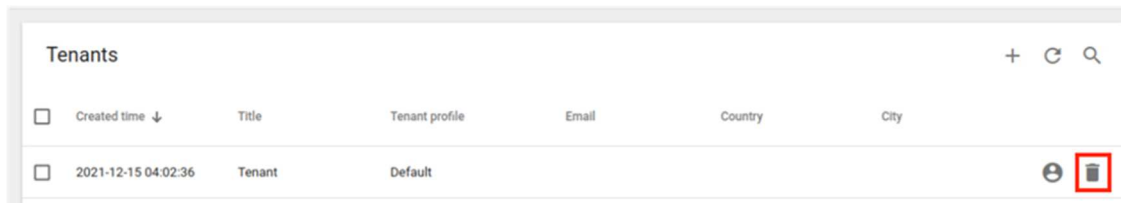
1.2 Create Different Accounts

In this section, you will make changes to the existing account and add necessary accounts.

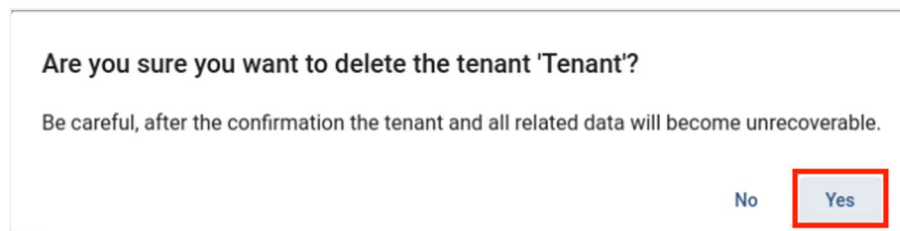
- First, we will start by creating a *Tenant*. Click **Tenants** on the left side.



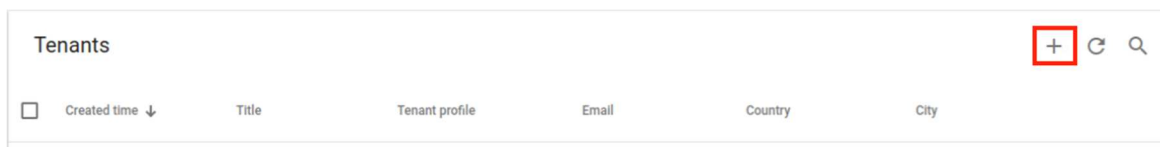
- To the right side, we will first delete the default *Tenant* account (again, for security purposes). Click the **trash can** icon.



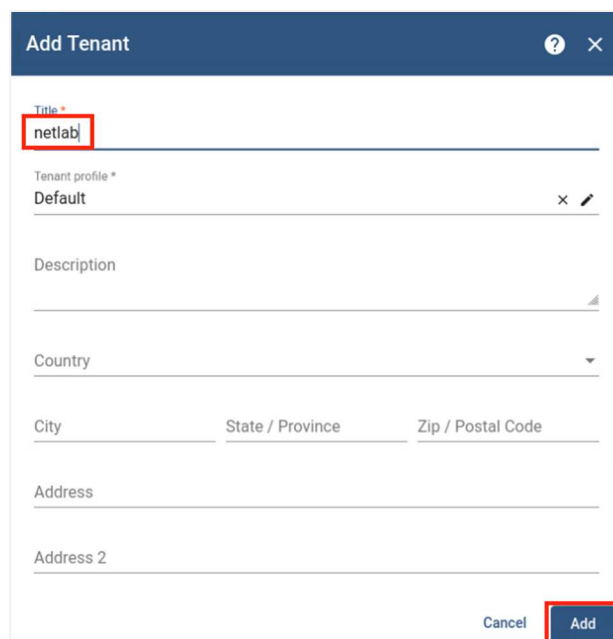
- A pop-up question will appear; click **Yes** to confirm.



- As the default *Tenant* is gone, we will create a new *Tenant*. Click the **+** button in the upper-right corner.

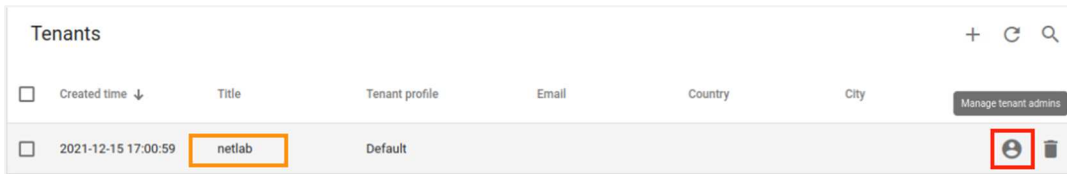



- A window will pop-up. For practice purposes, we will only fill in the *Title* field and leave the *Tenant profile* as **Default**. Type netLab as the *Title* and click the **Add** button.



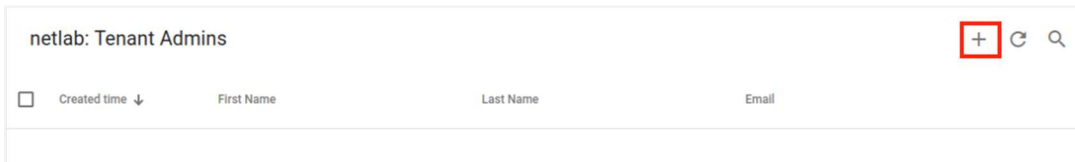
The "Add Tenant" form has the following fields: Title (with "netlab" entered and highlighted by a red box), Tenant profile (set to "Default"), Description, Country, City, State / Province, Zip / Postal Code, Address, and Address 2. At the bottom right, there are "Cancel" and "Add" buttons. The "Add" button is highlighted with a red box.

6. You should see that the **netlab** *Tenant* is created. Click the **Manage tenant admins** button.



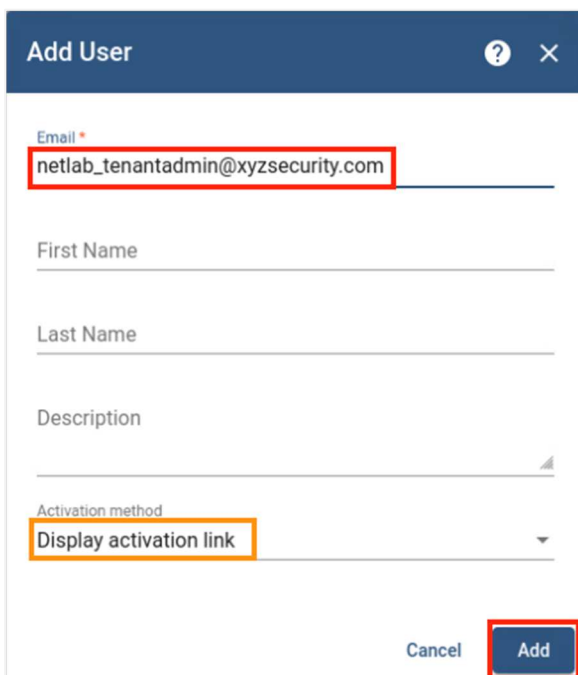
	Created time ↓	Title	Tenant profile	Email	Country	City	
<input type="checkbox"/>	2021-12-15 17:00:59	netlab	Default				

7. On the new page, click the **+** button to create a *Tenant Admin*.



	Created time ↓	First Name	Last Name	Email
--	----------------	------------	-----------	-------

8. In the pop-up window, fill the email field as **netlab_tenantadmin@xyzsecurity.com** leave the **Activation method** as the default setting, and click the **Add** button.



Add User

Email *

netlab_tenantadmin@xyzsecurity.com

First Name

Last Name

Description

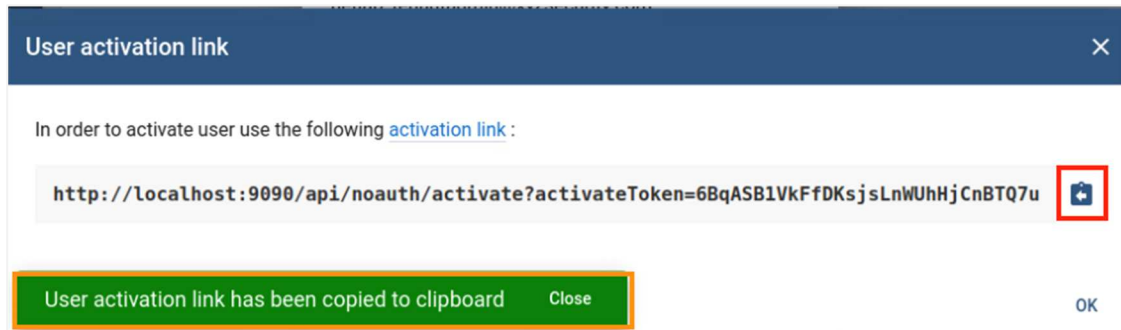
Activation method

Display activation link

Cancel

Add

9. After you click the **Add** button, a new window will show. Click the **Copy** button to copy the activation link. It will prompt with, *User activation link has been copied to clipboard*, click **Close**.



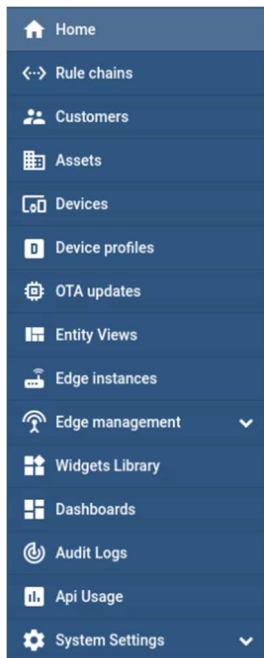
10. Start a new tab in the browser, and paste the copied link to the address bar. Press **Enter** to visit that page.



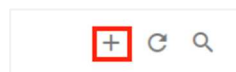
11. You will be asked to create a password for the account; type **NDGlabpass123!** and then click **Create Password**.



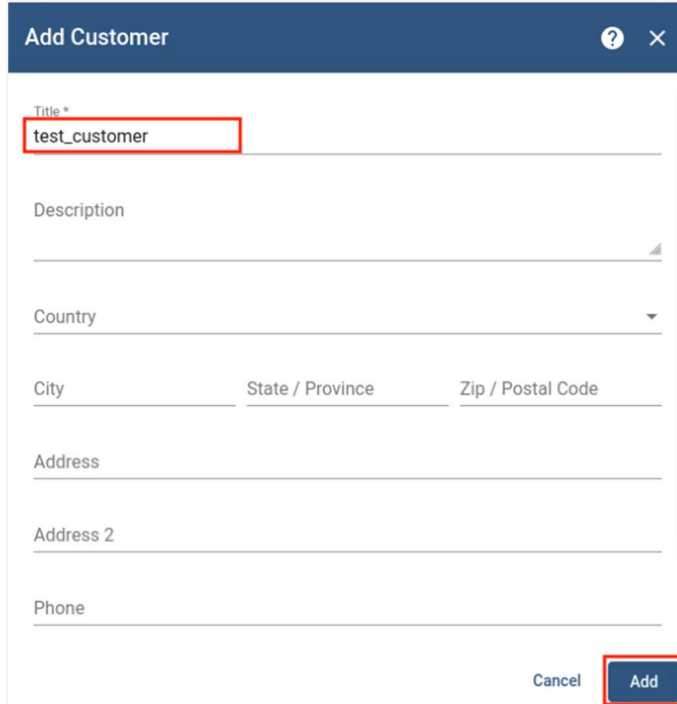
12. You will be brought to the main admin page, and the upper-right corner should show that you are logged in as the *netlab_tenantadmin@xyzsecurity.com* account. To the left side, there will be more options showing than before.



13. Click the **Customers** button, then on the right side, click the + to add a new customer.



14. In the pop-up window, we will fill the title field with `test_customer` as our customer and leave the rest blank. Click the **Add** button.

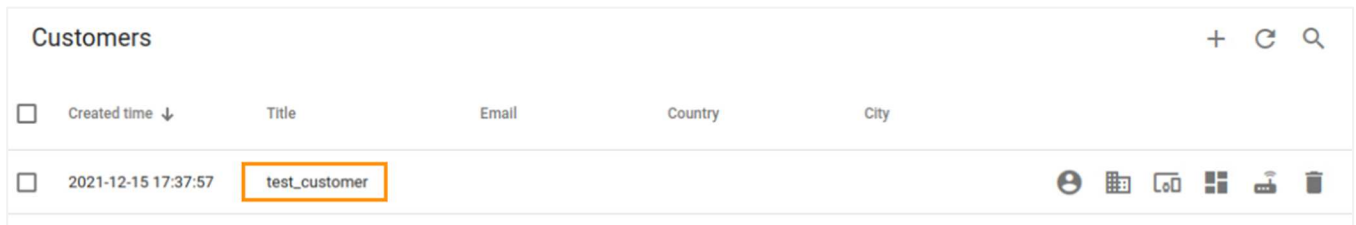


The image shows a pop-up window titled "Add Customer" with a close button (X) and a help button (?). The form contains the following fields:

- Title *: `test_customer` (highlighted with a red box)
- Description
- Country
- City
- State / Province
- Zip / Postal Code
- Address
- Address 2
- Phone

At the bottom right, there are two buttons: "Cancel" and "Add" (highlighted with a red box).

15. The customer should be created and shown on the *Customers* page. Leave the window open and proceed to the next step.



The image shows a table titled "Customers" with a header row and one data row. The header row has columns: Created time ↓, Title, Email, Country, and City. The data row has the following values:

	Created time ↓	Title	Email	Country	City
<input type="checkbox"/>	2021-12-15 17:37:57	<code>test_customer</code> (highlighted with an orange box)			

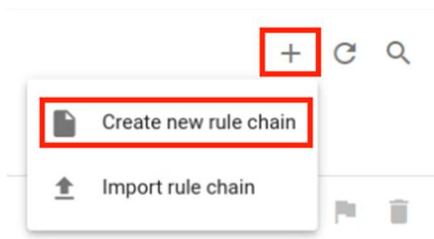
At the bottom right of the table, there are several icons: a person, a calendar, a document, a window, a network, and a trash can.

2 Create and Add an IoT Device

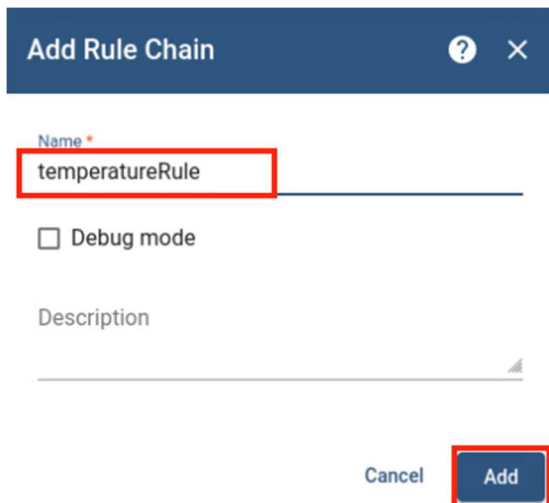
In this section, you will create and add a device to the ThingsBoard server.

2.2 Add a New Rule Chain








1. We will create a rule chain first. This will be used in the new device. Click on the **Rule Chains** button from the left panel. Then click the **+** button at the upper-right corner and select the **Create new rule chain** option.



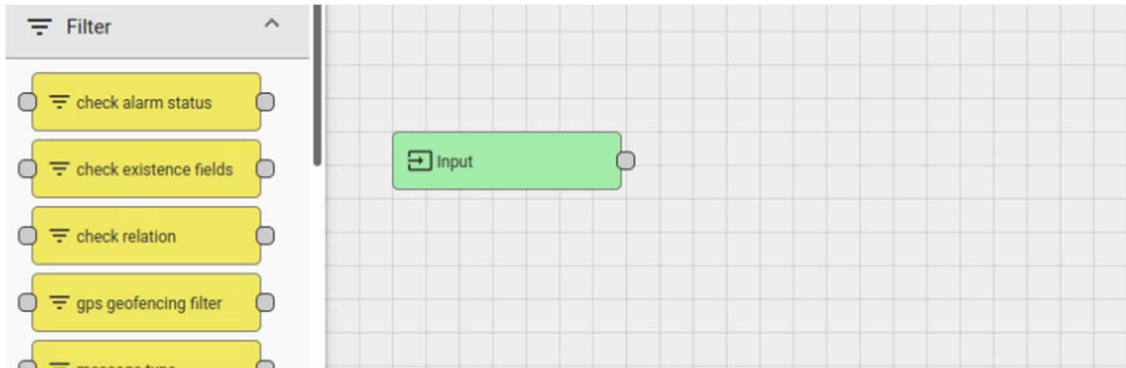
2. In the pop-up window, type `temperatureRule`. Then, click **Add** to confirm.



3. You will have two rule chains; click the **Open rule chain** button on the right side of the **temperatureRule**.

<input type="checkbox"/>	Created time ↓	Name	Root	Open rule chain
<input type="checkbox"/>	2021-12-16 03:35:47	temperatureRule	<input type="checkbox"/>	   
	2021-12-15 17:01:00	Root Rule Chain	<input checked="" type="checkbox"/>	   

4. You will see something like this:



5. To the left side, find **message type switch**, click and drag it to the grids on the right. When prompted, type check POST telemetry as the *Name*. Then, click **Add**.



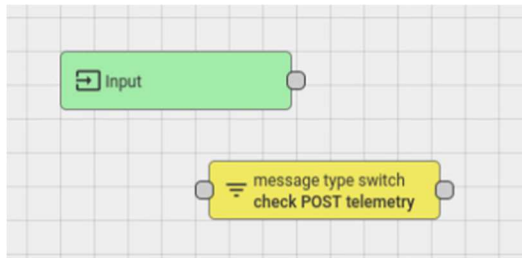
Add rule node: message type switch
?
×

Name *
☐ Debug mode

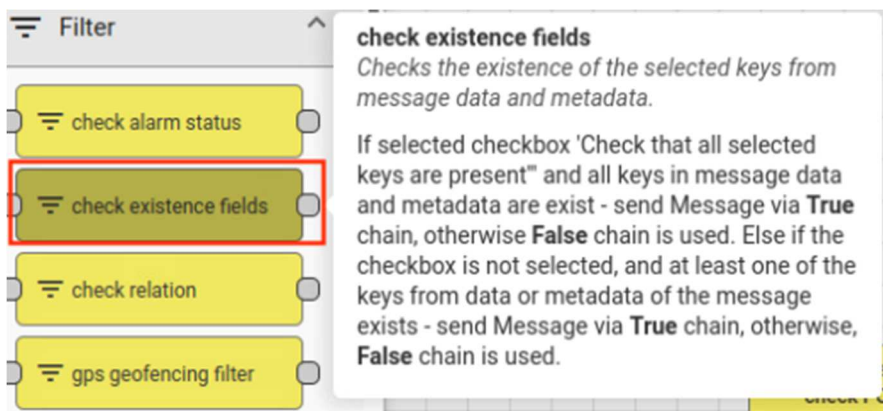
Description

Cancel
Add

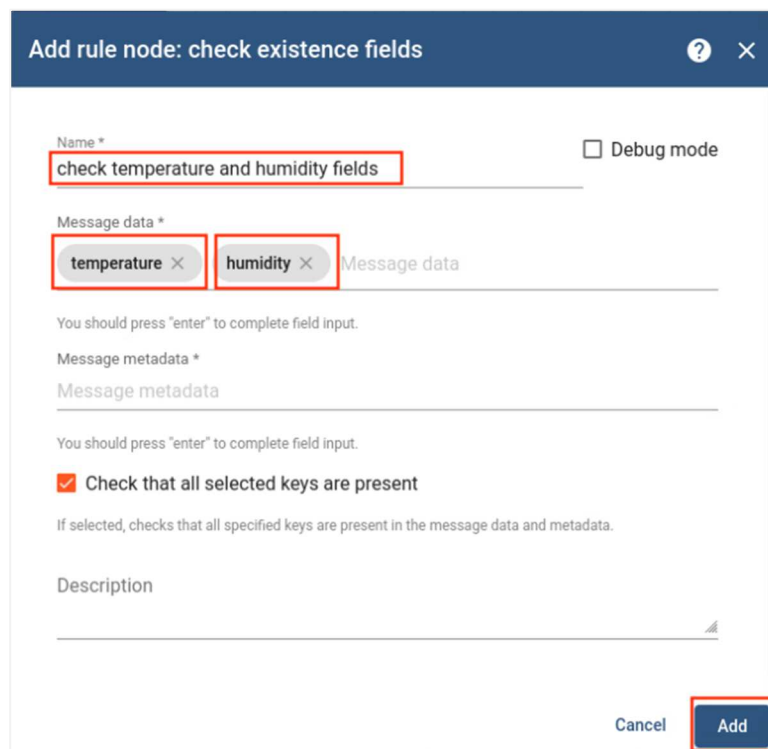
6. You should have something like this now:



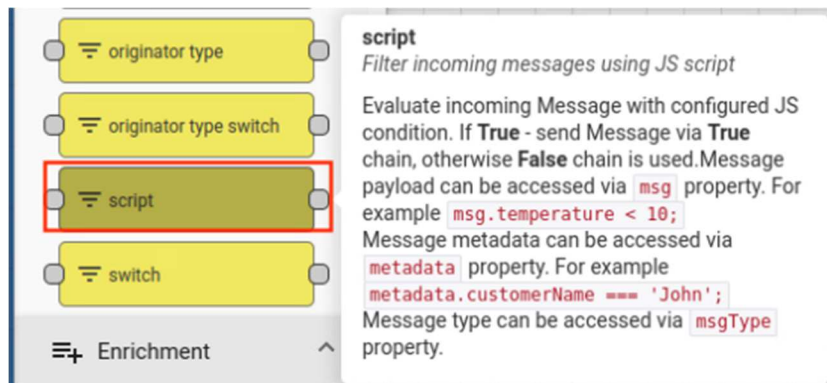
7. Next, find **check existence fields** and drag it to the grids.



8. When prompted, type **check temperature and humidity fields** as the *Name*. In the **Message data** area, type **temperature** and press **Enter**; type **humidity** and press **Enter**. You should have something like what is shown below. Leave the *Message metadata* empty and click **Add** to confirm.



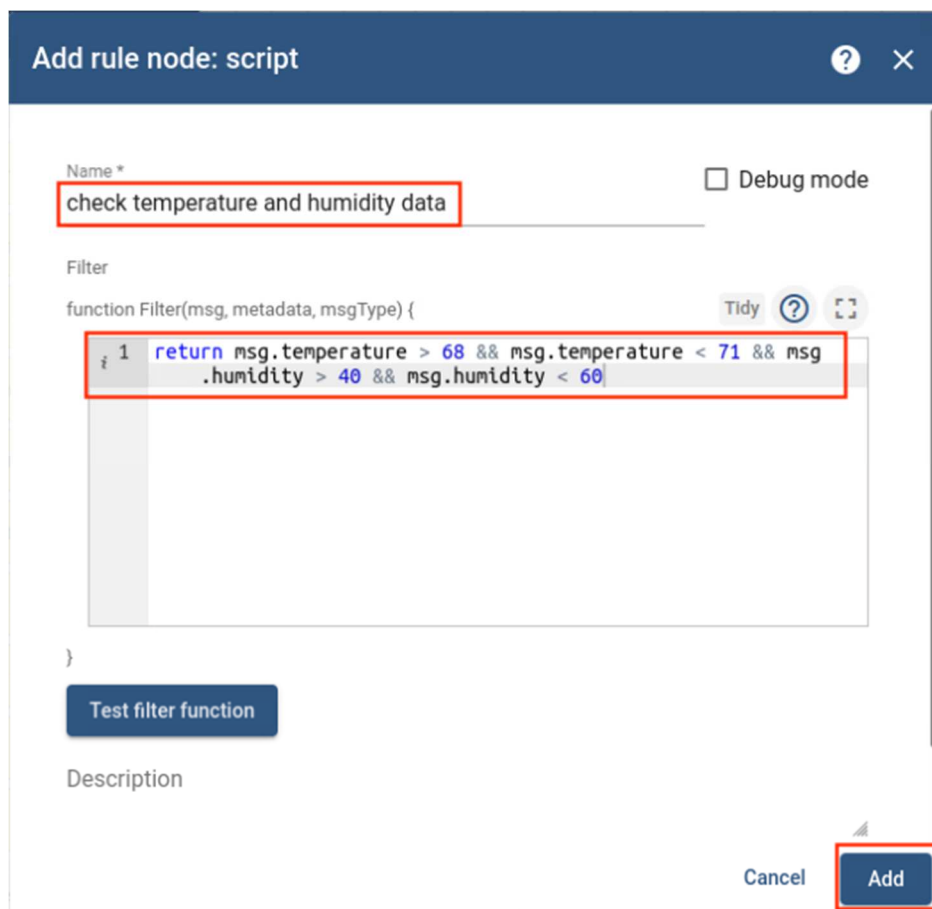
9. Next, find **script** and drag it to the grids.



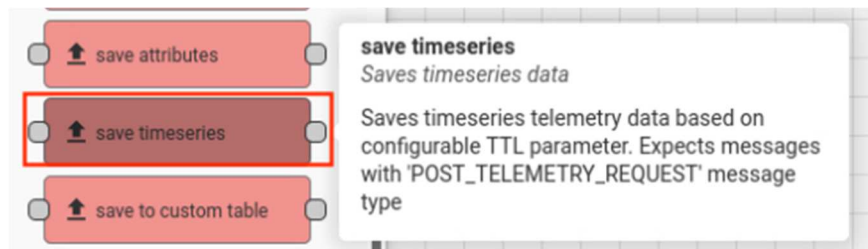
10. When prompted, type check temperature and humidity data as the *Name*, then change the script in the *function Filter* area. When finished, click **Add** to confirm.

The script should be:

```
return msg.temperature > 68 && msg.temperature < 71 && msg.humidity > 40 && msg.humidity < 60
```



11. Once again, on the left side, find **save timeseries** and drag it to the grids. When prompted, type **save the entry** as the *Name*. Change the *Default TTL in seconds* to 3 and click **Add** to confirm.



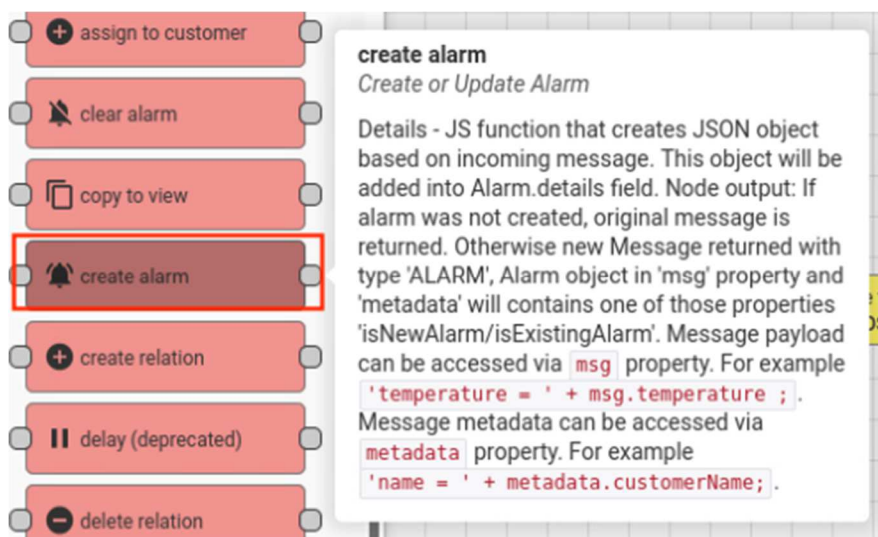
Add rule node: save timeseries [?] [X]

Name * ☐ Debug mode

Default TTL in seconds *

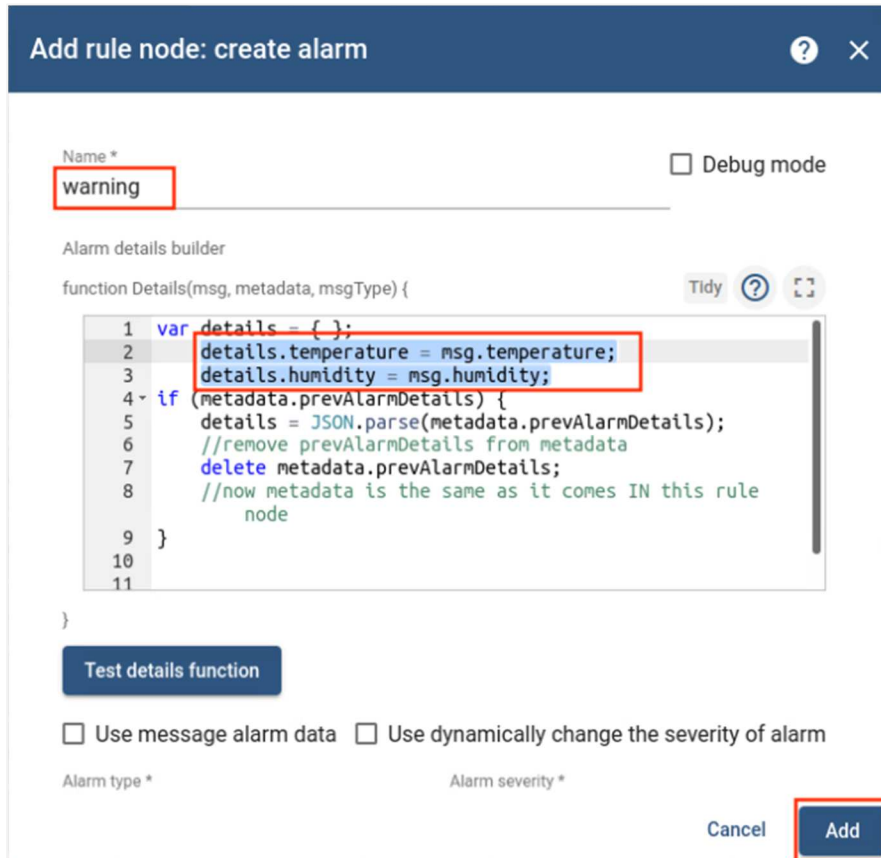
Description

12. For the last time, on the left side, find **create alarm** and drag it to the grids.



13. When prompted, type **warning** as the *Name*. Then, add the following script to the *function Details* area. When finished, click **Add** to confirm.

```
details.temperature = msg.temperature;
details.humidity = msg.humidity;
```



Add rule node: create alarm

Name * **warning** ☐ Debug mode

Alarm details builder

function Details(msg, metadata, msgType) {

```

1 var details = { };
2 details.temperature = msg.temperature;
3 details.humidity = msg.humidity;
4 if (metadata.prevAlarmDetails) {
5   details = JSON.parse(metadata.prevAlarmDetails);
6   //remove prevAlarmDetails from metadata
7   delete metadata.prevAlarmDetails;
8   //now metadata is the same as it comes IN this rule
  node
9 }
10
11 }

```

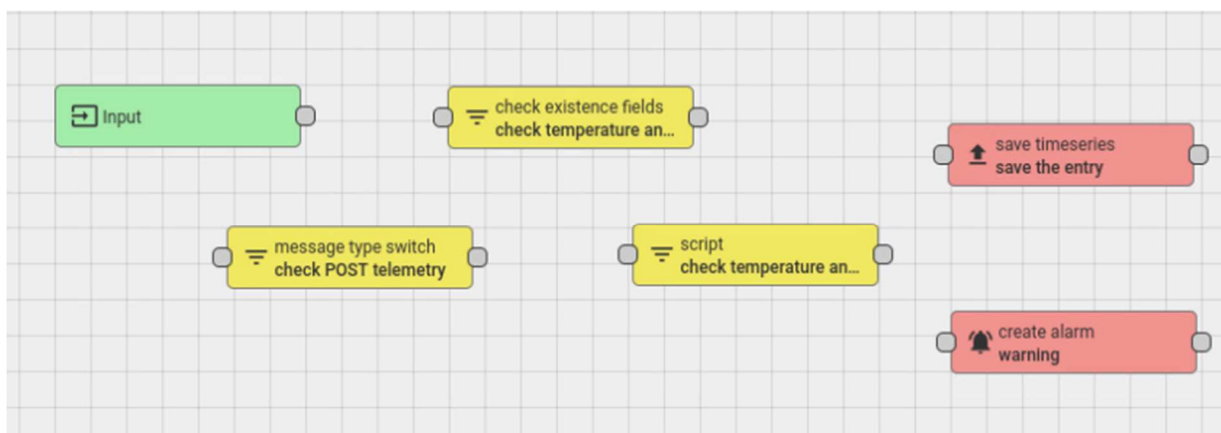
Test details function

☐ Use message alarm data ☐ Use dynamically change the severity of alarm

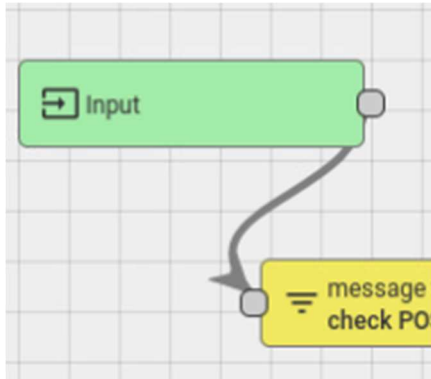
Alarm type * Alarm severity *

Add

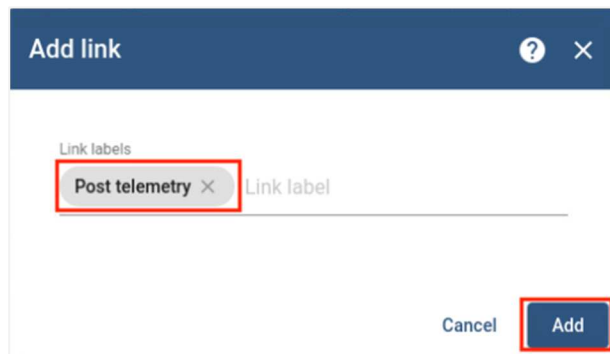
14. Your grid area should look something like this. The next step is to connect them to make a chain.



15. Click on the round connection point to the right side of **Input**. Click and drag it to the connection point on the left side of the **message type switch**. After you release the mouse, it should create a link with the arrow pointing to the direction of the data flow.



16. Use the same method to create a link between **message type switch** and **check existence fields**. When prompted, select **Post telemetry** as the *Link label*.



17. Then, add a link between **check existence fields** and **script**. When prompted, select **True** as the *Link label*. Click **Add** to confirm.



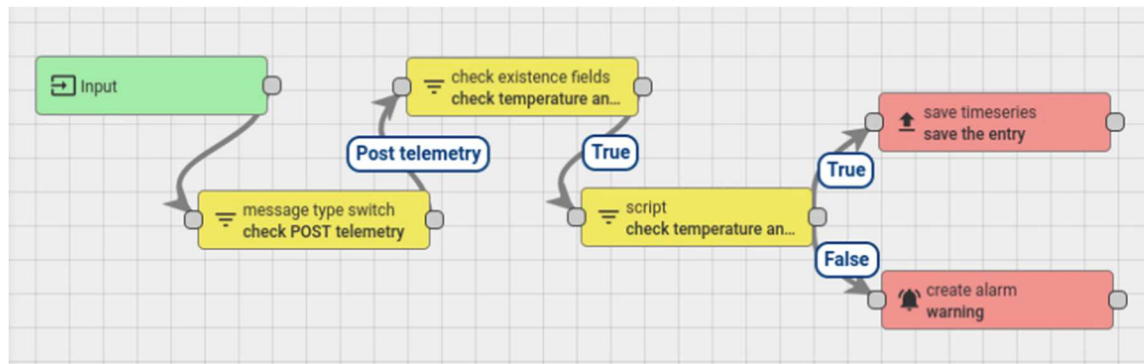
18. Add a link from **script** to **save timeseries**. When prompted, select **True**. Click **Add** to confirm.



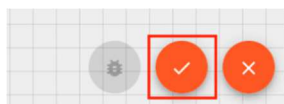
19. Add a link from **script** to **create alarm**. When prompted, select **False**. Click **Add** to confirm.



20. With all links added, everything should look like this:



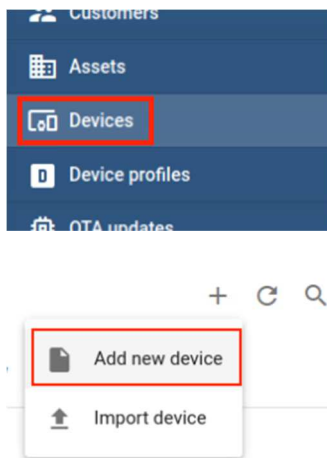
21. Save the changes by clicking the orange checkmark button at the lower-right corner.



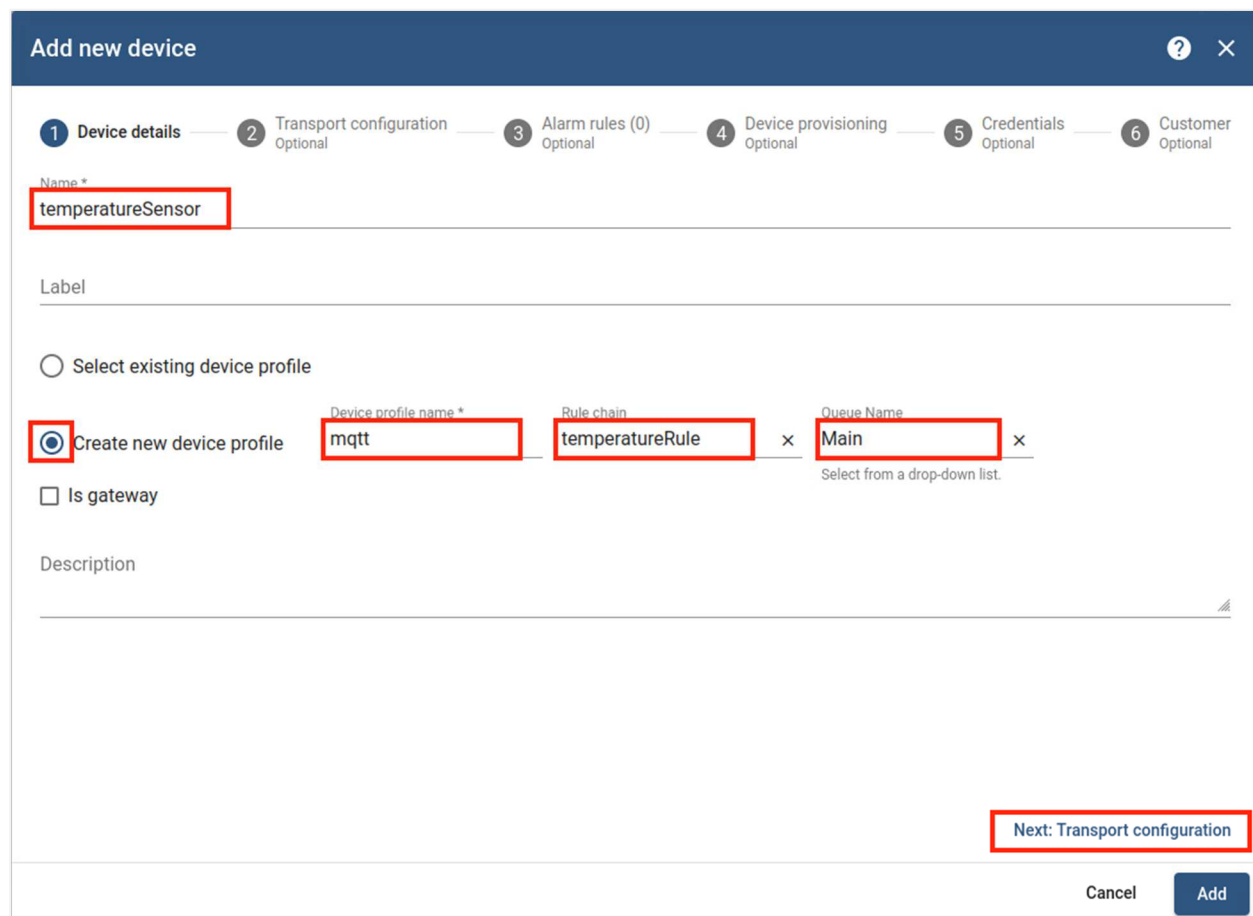
22. Leave the window open and proceed to the next step.

2.3 Add New Device

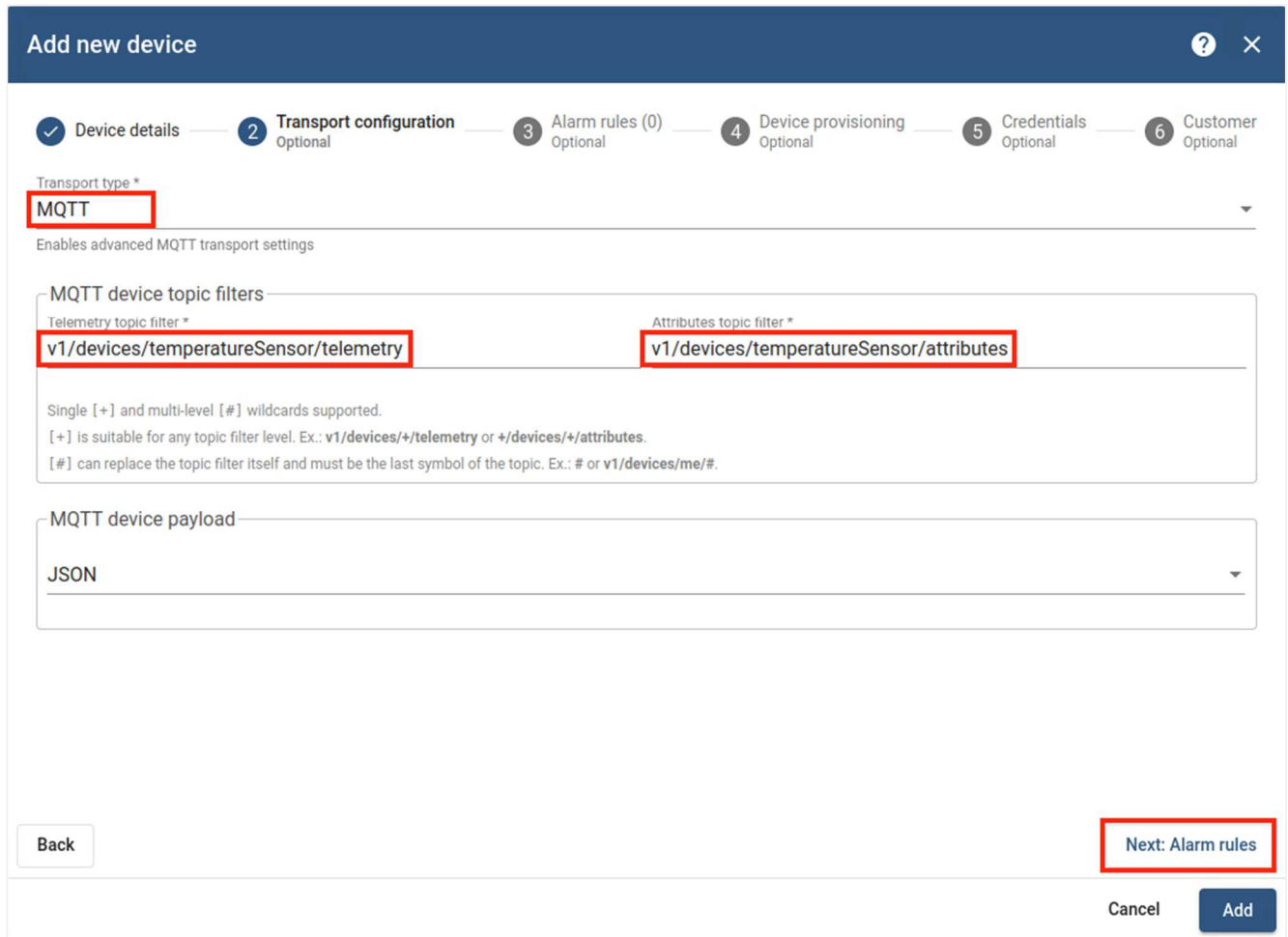
1. Click on *Devices* on the left side. Then, click the + button in the upper-right corner and select the **Add new device** option.



2. The *Add new device* window will appear. Type **temperatureSensor** as the *Name*. Then, select to check **Create new device profile**. Type **mqtt** as the *Device profile name*. Select **temperatureRule** for the *Rule chain* option, and **Main** for the *Queue Name*. Double-check everything; once done, you can then click the **Next: Transport configuration** button.



- On the *Transport Configuration* page, click to change the **Transport type** to **MQTT**. Then, change the *Telemetry topic filter* to `v1/devices/temperatureSensor/telemetry` and change the *Attributes topic filter* to `v1/devices/temperatureSensor/attributes`. These paths will be used for the MQTT message later on. Click **Next: Alarm rules** to continue.



Add new device

1 Device details — 2 **Transport configuration** Optional — 3 Alarm rules (0) Optional — 4 Device provisioning Optional — 5 Credentials Optional — 6 Customer Optional

Transport type *
MQTT

Enables advanced MQTT transport settings

MQTT device topic filters

Telemetry topic filter *
v1/devices/temperatureSensor/telemetry

Attributes topic filter *
v1/devices/temperatureSensor/attributes

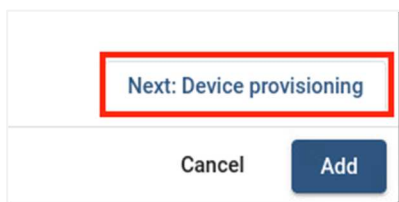
Single [+] and multi-level [#] wildcards supported.
[+] is suitable for any topic filter level. Ex.: v1/devices/+/telemetry or +/devices/+/attributes.
[#] can replace the topic filter itself and must be the last symbol of the topic. Ex.: # or v1/devices/me/#.

MQTT device payload
JSON

Back **Next: Alarm rules**

Cancel Add

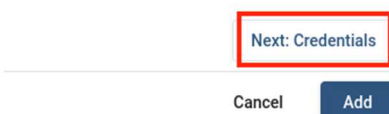
- On the *Alarm rules* page, we will not add any rules. Click **Next: Device provisioning** to continue.



Next: Device provisioning

Cancel Add


- On the *Device provisioning* page, do not change anything, click **Next: Credentials** to continue.



Next: Credentials

Cancel Add

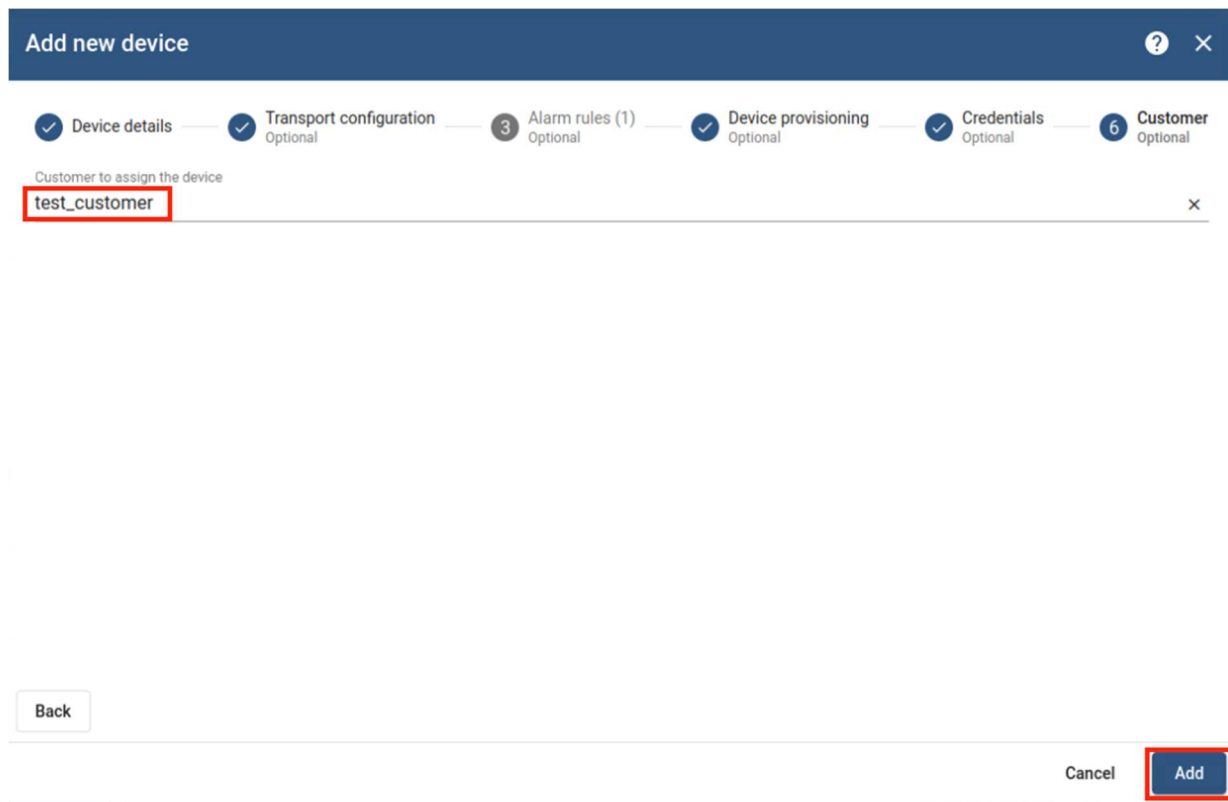
6. On the *Credentials* page, we will not add credentials. Click **Next: Customer** to continue.



Next: Customer

Cancel Add

7. Click to select **test_customer** in the *Customer to assign the device* field. Click **Add** to add the device.



Add new device

Device details ✓ Transport configuration ✓ Optional Alarm rules (1) 3 Optional Device provisioning ✓ Optional Credentials ✓ Optional Customer 6 Optional

Customer to assign the device

test_customer

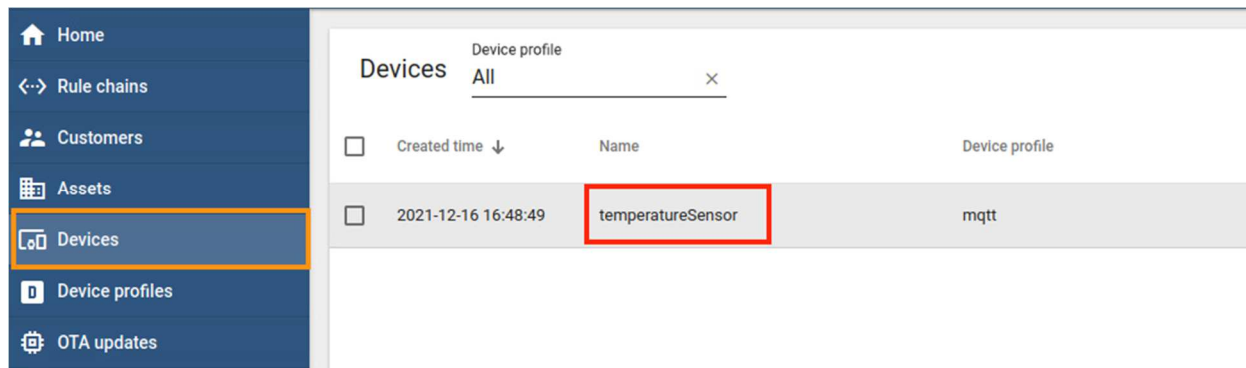
Back

Cancel Add

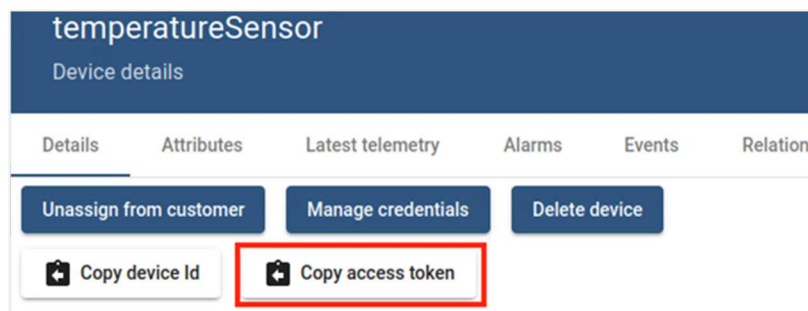
8. Leave the *ThingsBoard* window open and proceed to the next section.

3 Testing the Device

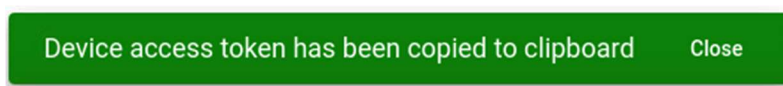
1. A quick way to test the device we created is to send an MQTT message to the ThingsBoard server from the terminal. First, we will get the access token from our temperatureSensor device. With the ThingsBoard still open, make sure you are still on the *Devices* page. Then, click on the **temperatureSensor** once.



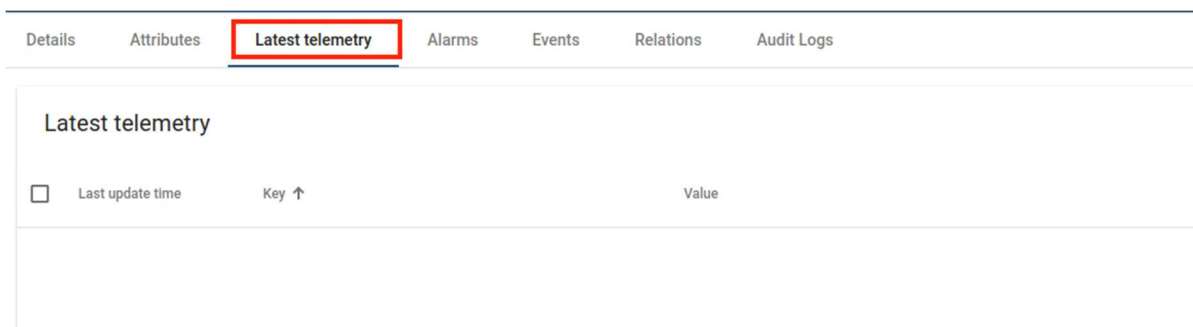
2. A panel will slide open from the right side. Click on **Copy access token**.



3. A pop-up message will show quickly at the lower-right corner, indicating that the *Device access token has been copied to clipboard*. Click **Close**.



4. Once the token is copied, click **Latest telemetry** and we will observe the data from this tab. Right now, there is no telemetry at all.



5. Start a new terminal and enter the following command. Replace the `$ACCESS_TOKEN` with the token we just copied. Press **Enter** to send the message to the *ThingsBoard* server.

```
sysadmin@ubuntu:~$ mosquitto_pub -d -q 1 -h "127.0.0.1" -t "v1/devices/temperatureSensor/telemetry" -u "$ACCESS_TOKEN" -m '{"temperature":69,"humidity":56}'
```

```
sysadmin@ubuntu:~$ mosquitto_pub -d -q 1 -h "127.0.0.1" -t "v1/devices/temperatureSensor/telemetry" -u "DHX2eR1A5o8SxYQGHPMT" -m '{"temperature":69,"humidity":56}'
Client null sending CONNECT
Client null received CONNACK (0)
Client null sending PUBLISH (d0, q1, r0, m1, 'v1/devices/temperatureSensor/telemetry', ... (28 bytes)
Client null received PUBACK (Mid: 1, RC:0)
Client null sending DISCONNECT
sysadmin@ubuntu:~$
```



The `$ACCESS_TOKEN` must be replaced for the message to work. The token in the example will be different from yours.

6. Immediately, you should see a telemetry show up in the *ThingsBoard*, under the **Latest telemetry** tab.

Details	Attributes	Latest telemetry	Alarms	Events	Relations	Audit Logs
Latest telemetry						
<input type="checkbox"/>	Last update time	Key ↑	Value			
<input type="checkbox"/>	2021-12-16 17:11:31	humidity	56			
<input type="checkbox"/>	2021-12-16 17:11:31	temperature	69			

7. Alternatively, you could move the *Terminal* window to show the *Latest telemetry* in the background. Then, change the humidity to 50 and run the command again.

Latest telemetry

<input type="checkbox"/>	Last update time	Key ↑	Value
<input type="checkbox"/>	2021-12-16 17:15:35	humidity	50
<input type="checkbox"/>	2021-12-16 17:15:35	temperature	69

```

sysadmin@ubuntu:~$ mosquitto_pub -d -q 1 -h "127.0.0.1" -t "v1/devices/temperatureSensor/telemetry" -u "DHX2eR1A5o8SxYQGHPMt" -m '{"temperature":69,"humidity":56}'
Client null sending CONNECT
Client null received CONNACK (0)
Client null sending PUBLISH (d0, q1, r0, m1, 'v1/devices/temperatureSensor/telemetry', ..
. (28 bytes))
Client null received PUBACK (Mid: 1, RC:0)
Client null sending DISCONNECT
sysadmin@ubuntu:~$ mosquitto_pub -d -q 1 -h "127.0.0.1" -t "v1/devices/temperatureSensor/telemetry" -u "DHX2eR1A5o8SxYQGHPMt" -m '{"temperature":69,"humidity":50}'
Client null sending CONNECT
Client null received CONNACK (0)
Client null sending PUBLISH (d0, q1, r0, m1, 'v1/devices/temperatureSensor/telemetry', ..
. (28 bytes))
Client null received PUBACK (Mid: 1, RC:0)
Client null sending DISCONNECT
sysadmin@ubuntu:~$

```

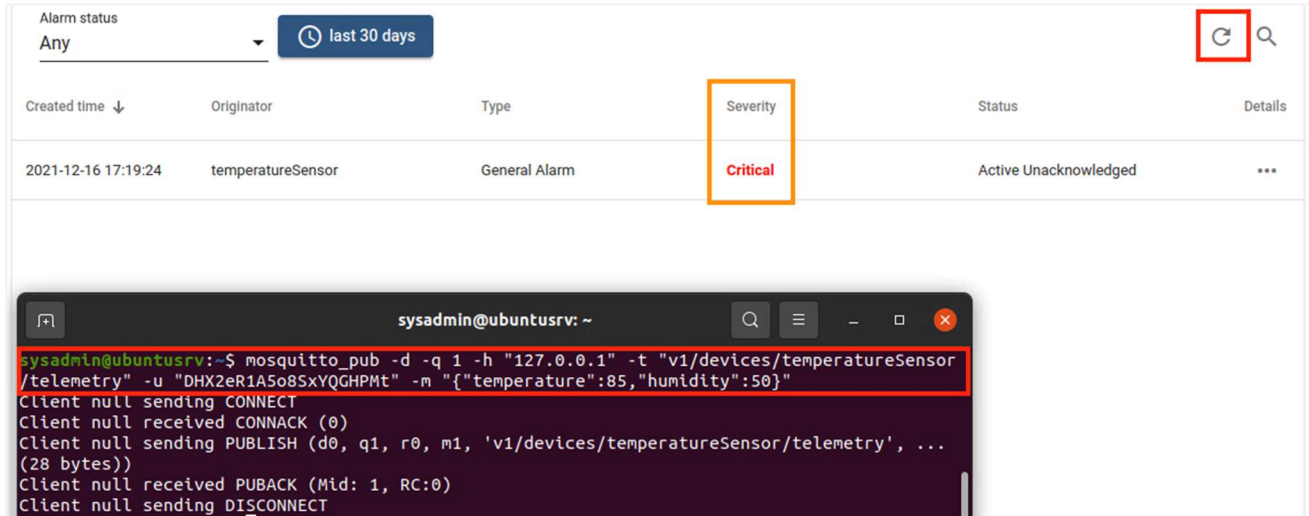
8. Now, we will set a higher temperature to create an alarm. Go back to the *ThingsBoard* and click on the *Alarms* tab. There is nothing right now.

Details
Attributes
Latest telemetry
Alarms
Events
Relations
Audit Logs

Alarm status
Any
last 30 days

Created time ↓	Originator	Type	Severity
----------------	------------	------	----------

9. Switch back to the *Terminal window*; we will send another MQTT message but with a temperature of 85. An alarm will show up with a *Severity level* of **Critical**. If the *Alarm* page does not refresh itself, click the **Refresh** button.



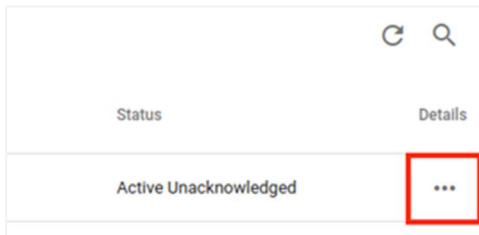
Alarm status: Any last 30 days Refresh Search

Created time ↓	Originator	Type	Severity	Status	Details
2021-12-16 17:19:24	temperatureSensor	General Alarm	Critical	Active Unacknowledged	...

```

sysadmin@ubuntu: ~
sysadmin@ubuntu:~$ mosquitto_pub -d -q 1 -h "127.0.0.1" -t "v1/devices/temperatureSensor/telemetry" -u "DHX2eR1A5o8SxYQGHPMt" -m '{"temperature":85,"humidity":50}'
Client null sending CONNECT
Client null received CONNACK (0)
Client null sending PUBLISH (d0, q1, r0, m1, 'v1/devices/temperatureSensor/telemetry', ... (28 bytes))
Client null received PUBACK (Mid: 1, RC:0)
Client null sending DISCONNECT
  
```

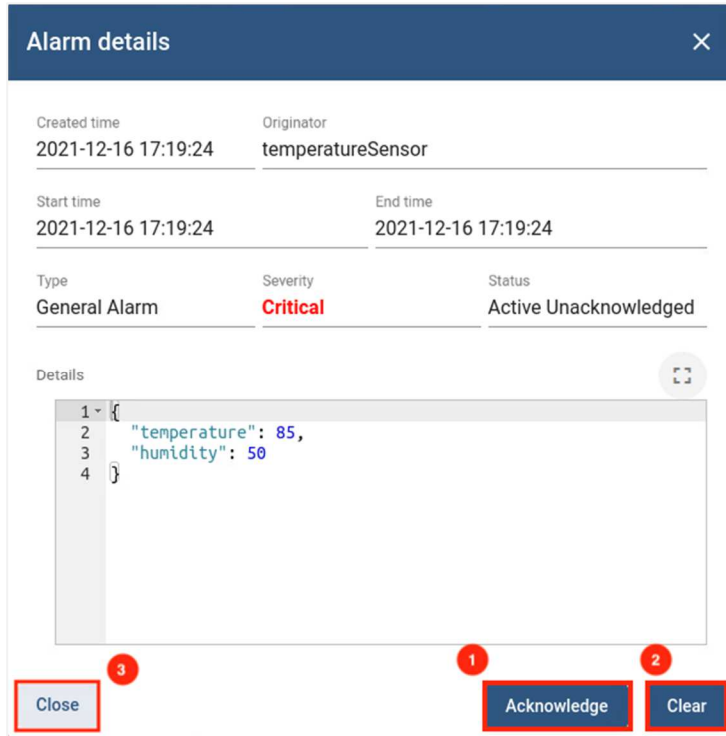
10. We can acknowledge the alarm and even clear the alarm once it has been dealt with. Click the **Details** button.



Refresh Search

Status	Details
Active Unacknowledged	...

11. In the pop-up window, click **Acknowledge**, then click **Clear**. Once cleared, you can click the **Close** to exit the pop-up window.



The "Alarm details" pop-up window displays the following information:

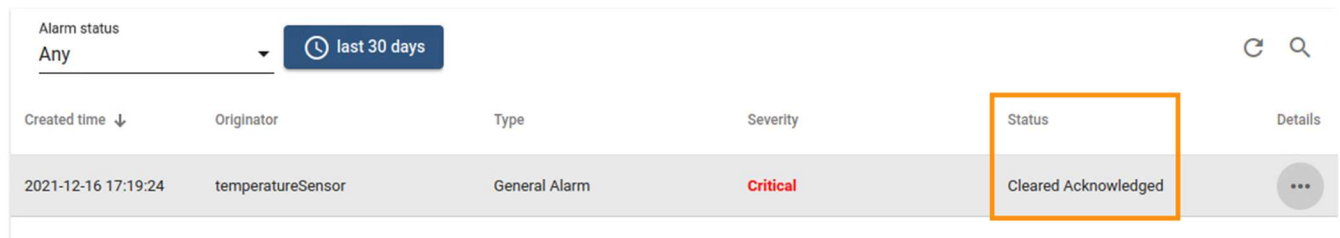
- Created time:** 2021-12-16 17:19:24
- Originator:** temperatureSensor
- Start time:** 2021-12-16 17:19:24
- End time:** 2021-12-16 17:19:24
- Type:** General Alarm
- Severity:** Critical
- Status:** Active Unacknowledged

The **Details** section shows a JSON object:

```
1 {  
2   "temperature": 85,  
3   "humidity": 50  
4 }
```

At the bottom, there are three buttons: **Close** (labeled 3), **Acknowledge** (labeled 1), and **Clear** (labeled 2).

12. Now the *Status* should indicate **Cleared Acknowledged**.



The "Alarm status" table shows the following data:

Created time ↓	Originator	Type	Severity	Status	Details
2021-12-16 17:19:24	temperatureSensor	General Alarm	Critical	Cleared Acknowledged	...

The **Status** column is highlighted with an orange box, showing the value **Cleared Acknowledged**.

13. The lab is now finished; you may end the reservation.