
InGateway Documentation

Release 0.0.1

zhangning

May 06, 2023

1	InGateway Documentation Site Navigation	1
1.1	InGateway902 Docker user manual	1
1.2	Azure IoT Edge User Manual	22

InGateway Documentation Site Navigation

InGateway902 series edge computing gateway supports manage docker images. You can publish your docker images to IG902 to quickly deploy and run applications developed by yourself.

Docker is an open source application container engine that allows developers to package their applications and dependencies into a portable container and then publish it to any popular Linux machine or Windows machine. It can also be virtualized. The container is completely with the sandbox mechanism, there will be no interface between each other.

1.1 InGateway902 Docker user manual

InGateway902 series edge computing gateway (IG902 for short) supports manage docker images. You can publish your docker images to IG902 to quickly deploy and run applications developed by yourself. In order to introduce how to use IG902's Docker environment, this document will demonstrate how to run an Nginx image on IG902. This image is used for open source reverse proxy server for HTTP, HTTPS, SMTP, POP3 and IMAP protocols, and load balancer, HTTP cache And web server. Docker is an open source application container engine that allows developers to package their applications and dependencies into a portable container and then publish it to any popular Linux machine or Windows machine. It can also be virtualized. The container is completely with the sandbox mechanism, there will be no interface between each other.

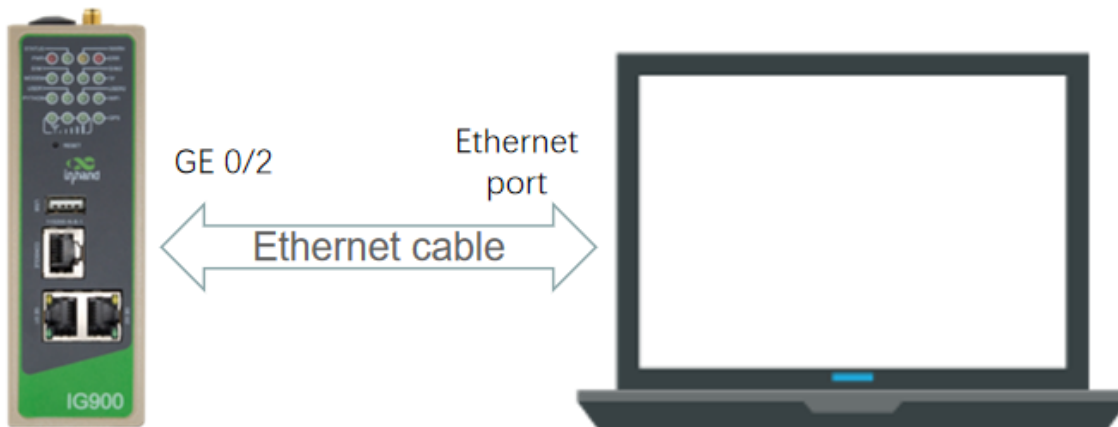
- *1. Prepare IG902 Hardware and Network Environment*
 - *1.1 Connect IG902 to the Power Source and to a PC with a Network Cable*
 - *1.2 Access the IG902*

- *1.3 Connect IG902 to the Internet*
 - *1.4 Update the firmware*
- *2. Enable and configure Docker manager*
 - *2.1 Install Docker SDK and enable Docker manager*
 - *2.2 Configure Docker Manager–Portainer*
 - * *2.2.1 Access Portainer*
 - * *2.2.2 Add docker image*
 - * *2.2.3 Configure and deploy container*
- *Appendix*
 - *Use Serial port for communication in container*
 - *Set the container to run permanently*
 - *Run Ubuntu in IG902*
 - *Build the image through a container (create an image to save the container configuration)*
 - *How to download docker images from gitlab / github*
- *FAQ*
 - *Q1: It prompted succeed after pulling image on the “Images” page, but the image is not shown on the “Images” page.*

1.1.1 1. Prepare IG902 Hardware and Network Environment

1.1 Connect IG902 to the Power Source and to a PC with a Network Cable

Connect IG902 to the power source and to a PC with an Ethernet cable according to the topology diagram.



1.2 Access the IG902

Access the IG902 by referring to [Access the IG902](#).

1.3 Connect IG902 to the Internet

Connect IG902 to the Internet by referring to [Connect IG902 to the Internet](#).

1.4 Update the firmware

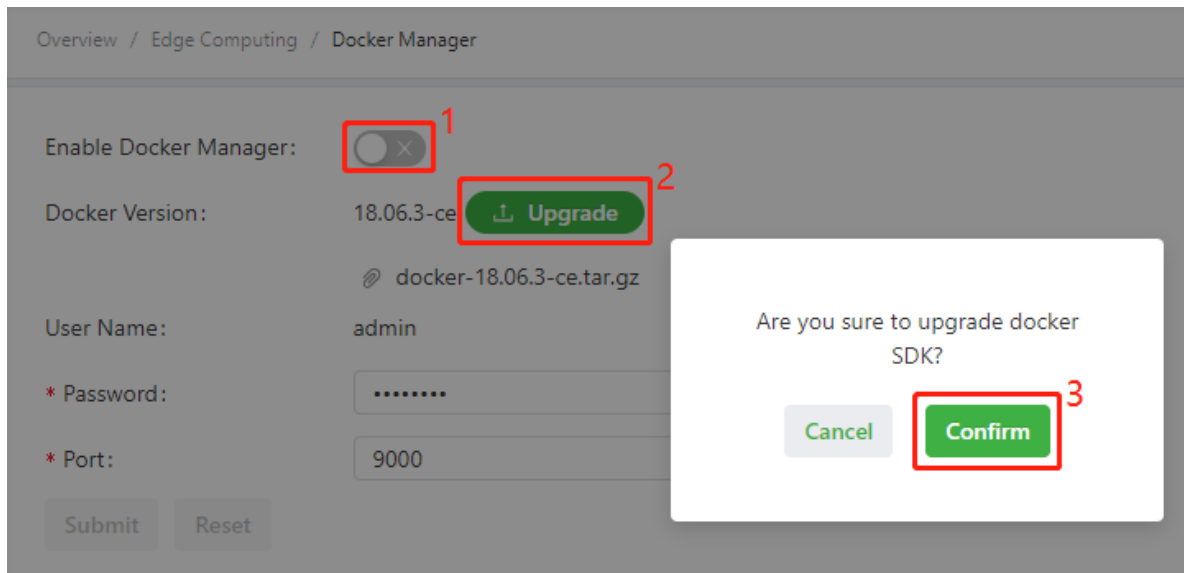
To obtain the latest firmware version of IG902 and updated functions, please visit the [Resource](#). To update the IG902 firmware, see [Update the IG902 software version](#). (The firmware version should be 2.0.0.r12057 and above)

1.1.2 2. Enable and configure Docker manager

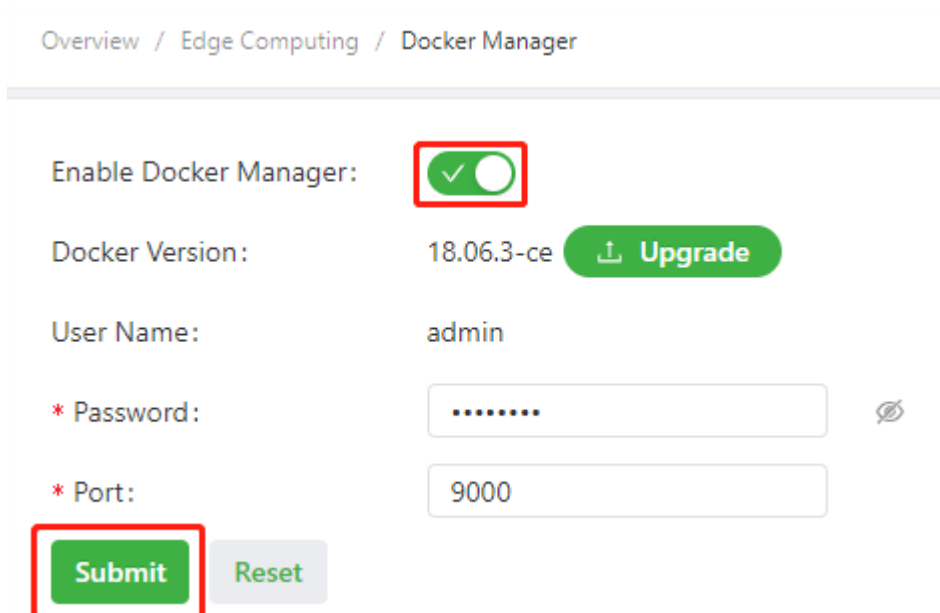
2.1 Install Docker SDK and enable Docker manager

The Docker SDK integrates the operating environment and docker image manager required to run the docker image. Before using Docker, you must install the Docker SDK. To obtain the Docker SDK, please visit the [Resource](#).

- Step 1: If you already have the Docker SDK, choose Edge Computing > Docker Manager page of IG902, close the Docker Manager and import the Docker SDK.



- Step 2: After importing, IG902 will automatically install the Docker SDK. The installation process usually takes 1-2 minutes. Please be patient. After successful installation, select Enable Docker Manager and click Submit.



- Step 3: Then you can modify the port number and login password to access the Docker manager.

Overview / Edge Computing / Docker Manager

Enable Docker Manager: ☒

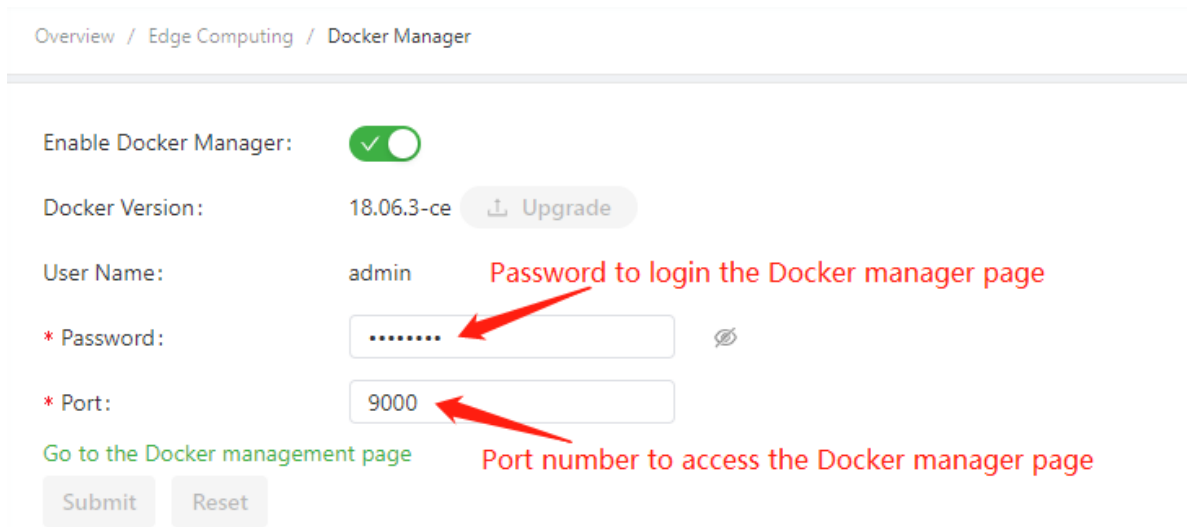
Docker Version: 18.06.3-ce [Upgrade](#)

User Name: admin

* Password: [Show/Hide](#)

* Port:

[Go to the Docker management page](#)

The screenshot shows the 'Docker Manager' configuration page. It has a breadcrumb trail 'Overview / Edge Computing / Docker Manager'. The 'Enable Docker Manager' toggle is turned on. The 'Docker Version' is '18.06.3-ce' with an 'Upgrade' button. The 'User Name' is 'admin'. The 'Password' field is masked with dots, and the 'Port' field contains '9000'. A green link 'Go to the Docker management page' is present. At the bottom are 'Submit' and 'Reset' buttons. Two red arrows are overlaid: one points from the text 'Password to login the Docker manager page' to the password input field, and another points from the text 'Port number to access the Docker manager page' to the port input field.

2.2 Configure Docker Manager–Portainer

IG902 uses Portainer to build, manage and maintain Docker images and containers. For a detailed introduction and instructions on Portainer, please see the [Portainer official website](#). This document will show you how to add and deploy an Nginx docker image on IG902.

2.2.1 Access Portainer

- Step 1: Click Portainer's access button, and Portainer will prompt you to enter your username and password. At this time, copy the user name and the set password from the Edge Computing > Docker Manager page of IG902 and click Login.

Overview / Edge Computing / Docker Manager

Enable Docker Manager: ☒

Docker Version: 18.06.3-ce [Upgrade](#)

User Name: admin


* Password: [Show/Hide](#)

* Port:

[Go to the Docker management page](#)

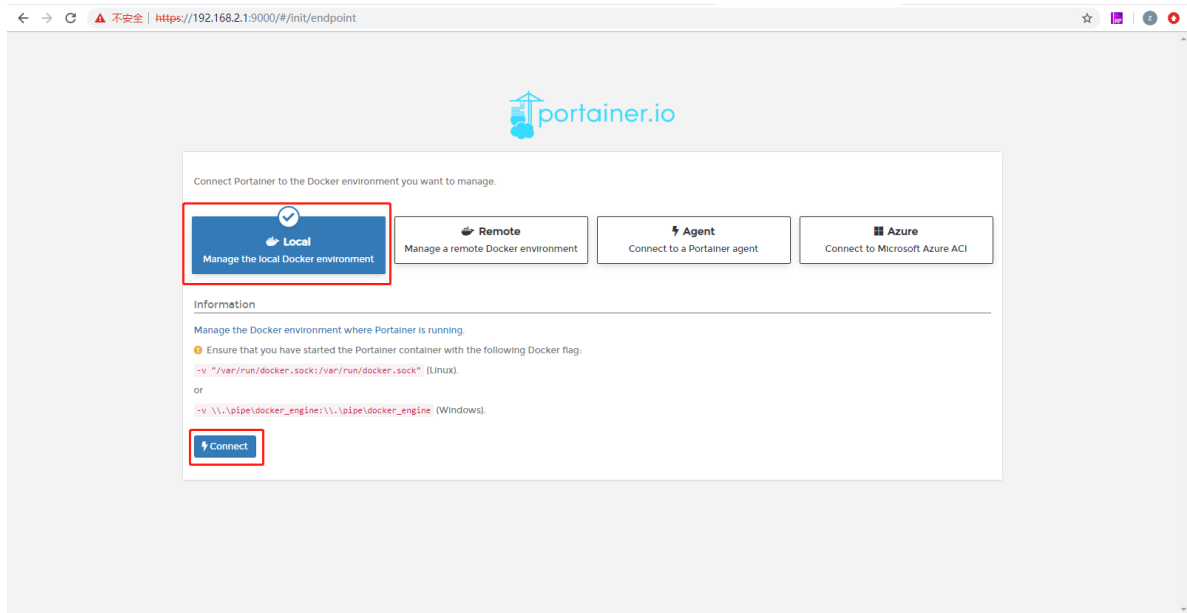
[Submit](#) [Reset](#)

← → ↻ 不安全 | https://192.168.2.1:9000/#/auth

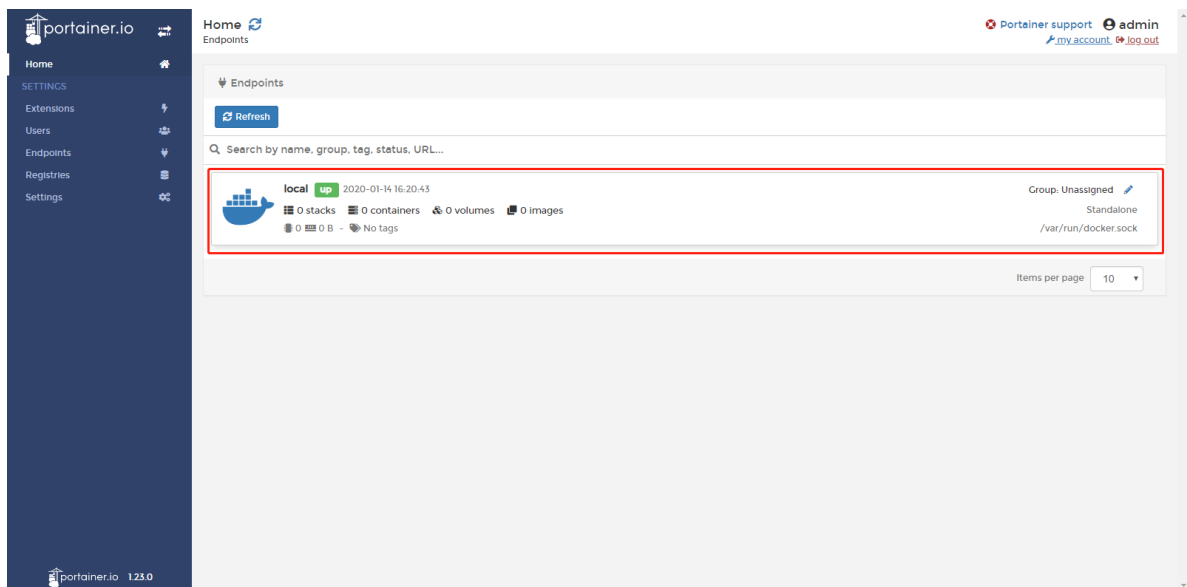


[Login](#)

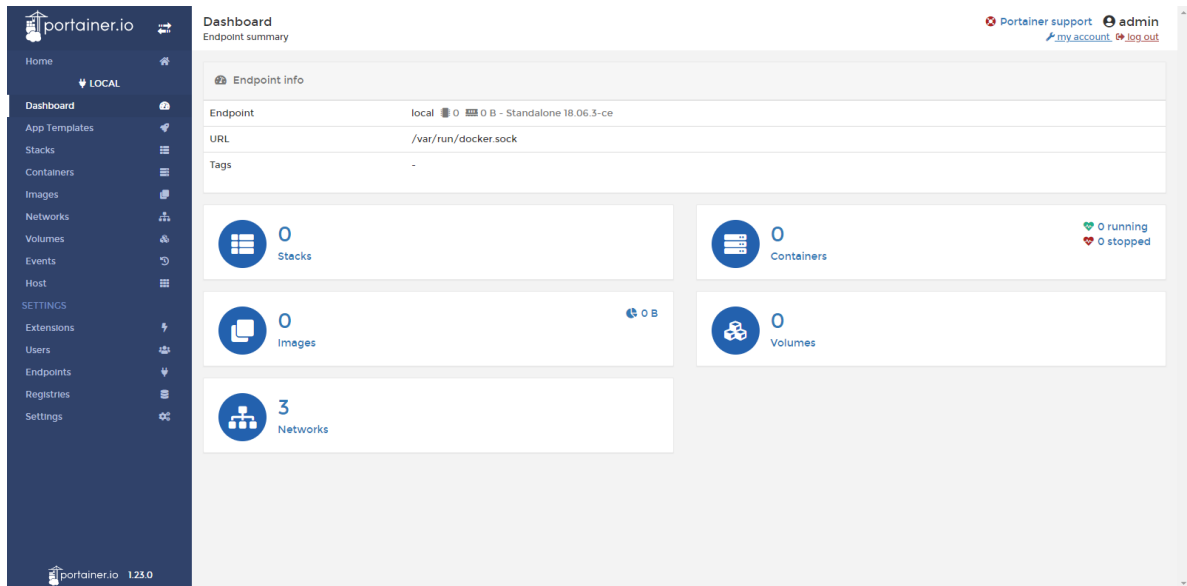
- Step 2: After the login is successful, as shown in the figure below, select Local to use the Portainer to manage the docker image on the IG902, and then click Connect.



- Step 3: On the Home page of Portainer, select local to manage the docker image on IG902.



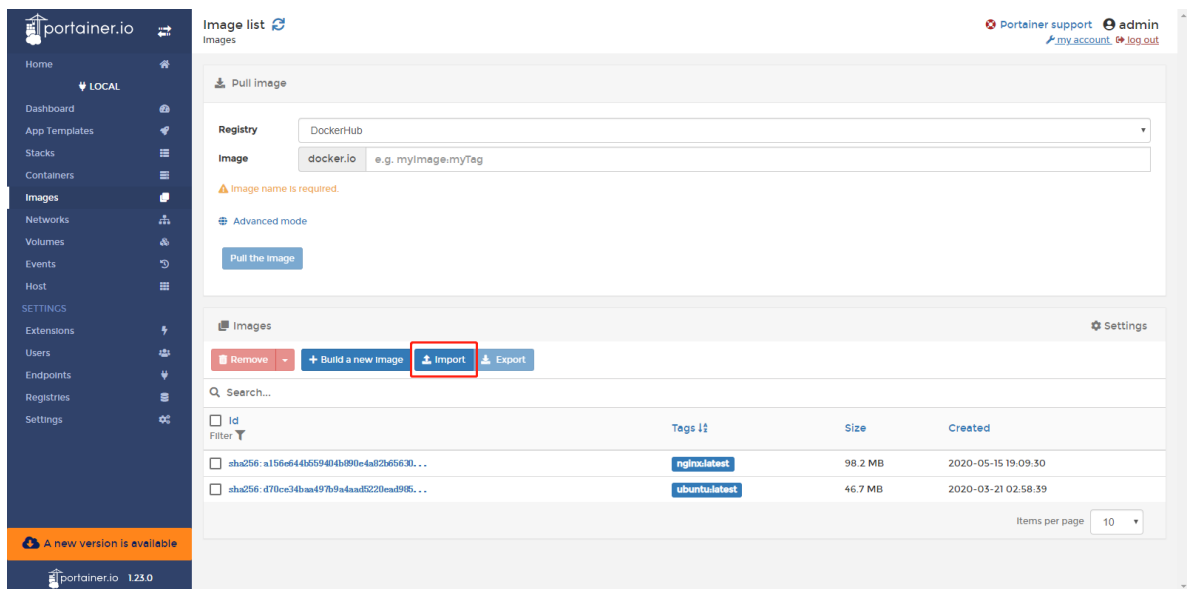
Then you will jump to the local dashboard, where you can get an overview of the IG902's containers and images.



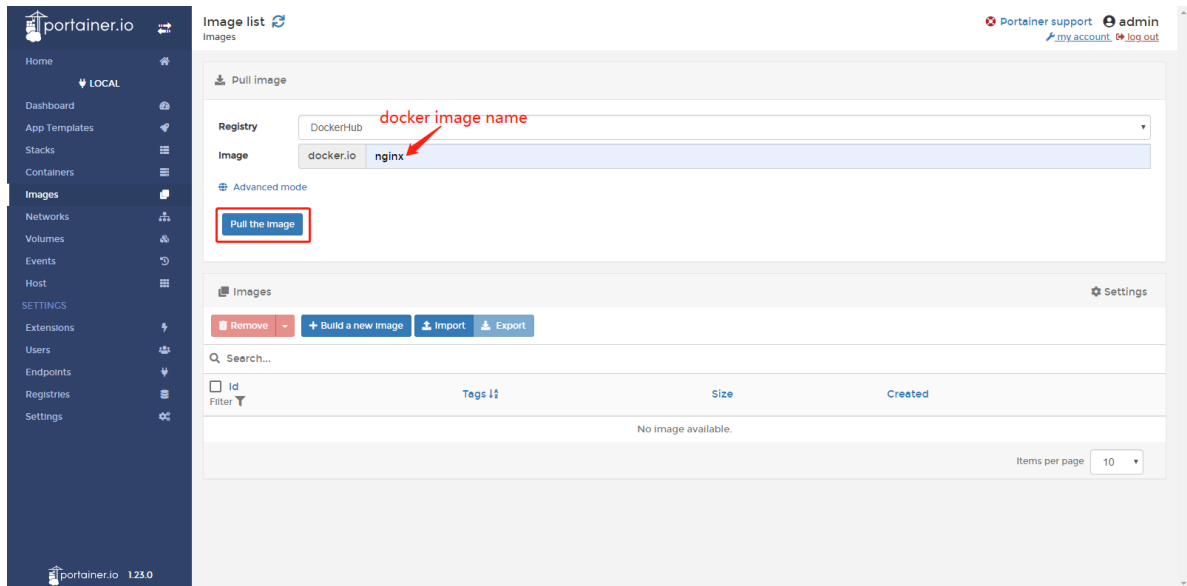
2.2.2 Add docker image

There are two ways to add docker images for Portainer:

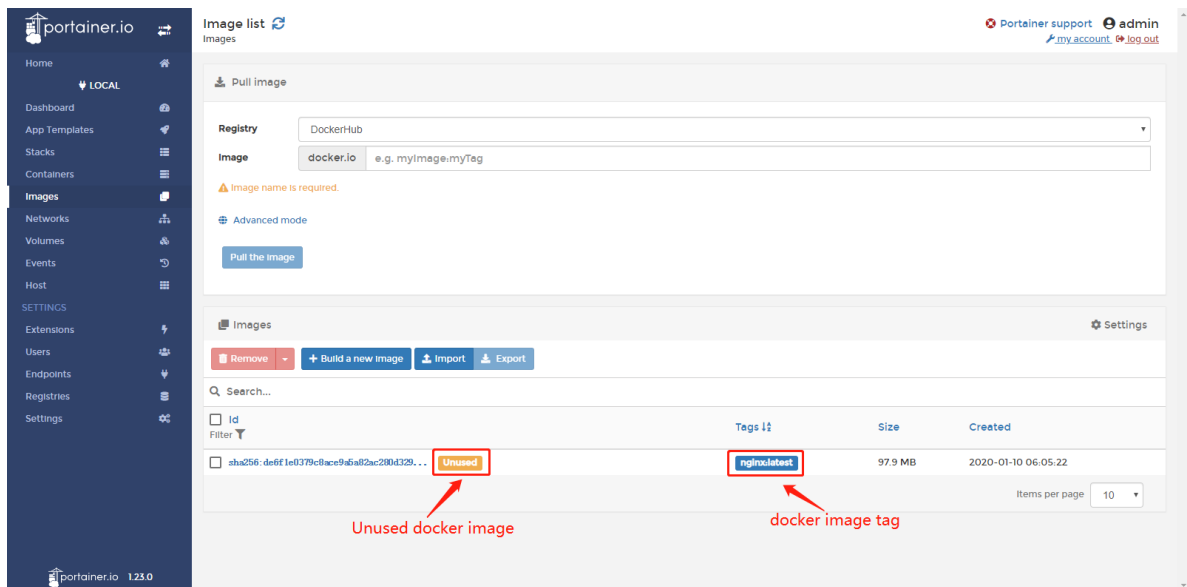
- Method 1: Enter the Local > Images page of Portainer. click “Import” to import the image.



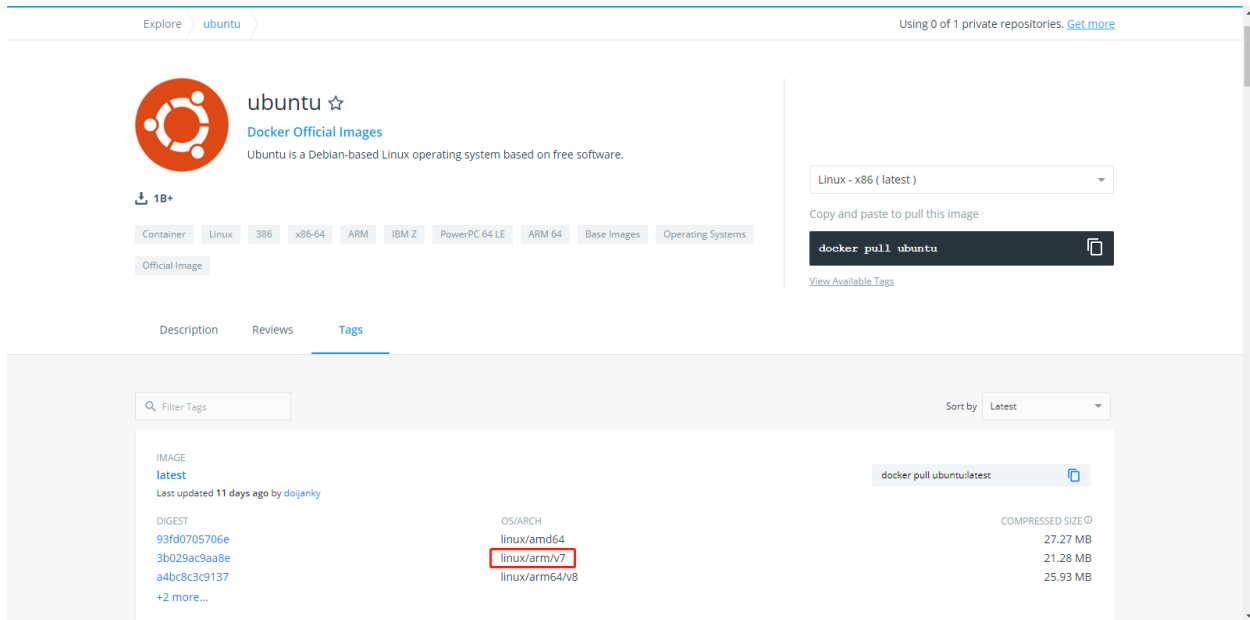
- Method 2: Choose Local > Images page of Portainer and download the nginx docker image from DockerHub. (The time required to download the image varies depending on the size of the image; please be patient when the docker image is large)



After the docker image is downloaded, you can see the corresponding docker image information in Local > Images as shown below:

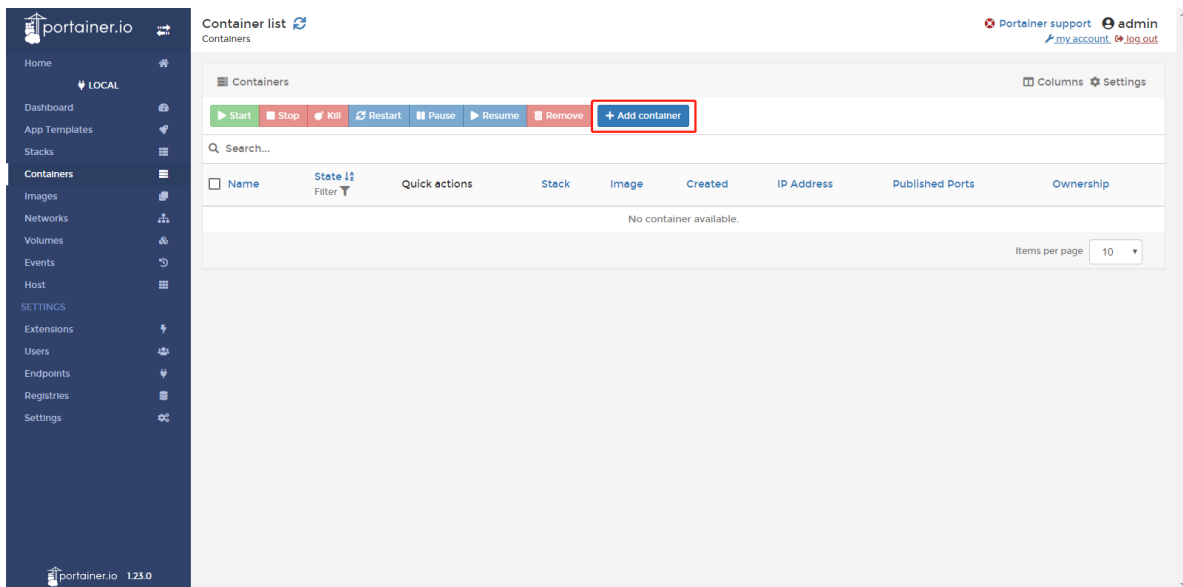


Note: The architecture of IG902 CPU is linux/arm/v7. Only images that support linux/arm/v7 architecture can run normally in IG902. Images of other architectures, such as window/amd64, may not be imported, pulled, or run successfully in IG902.

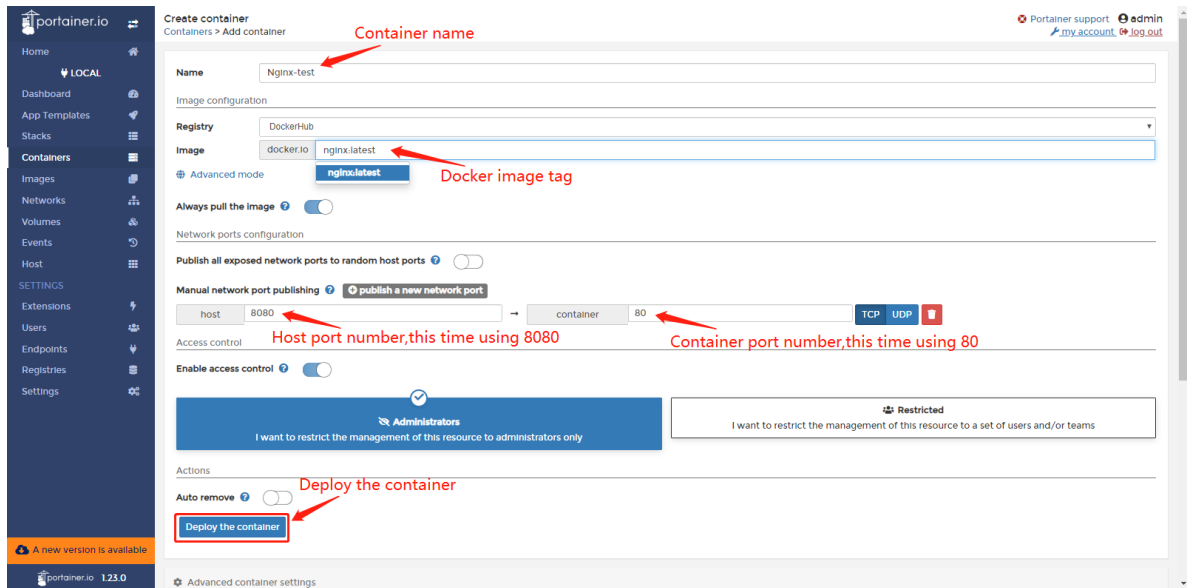


2.2.3 Configure and deploy container

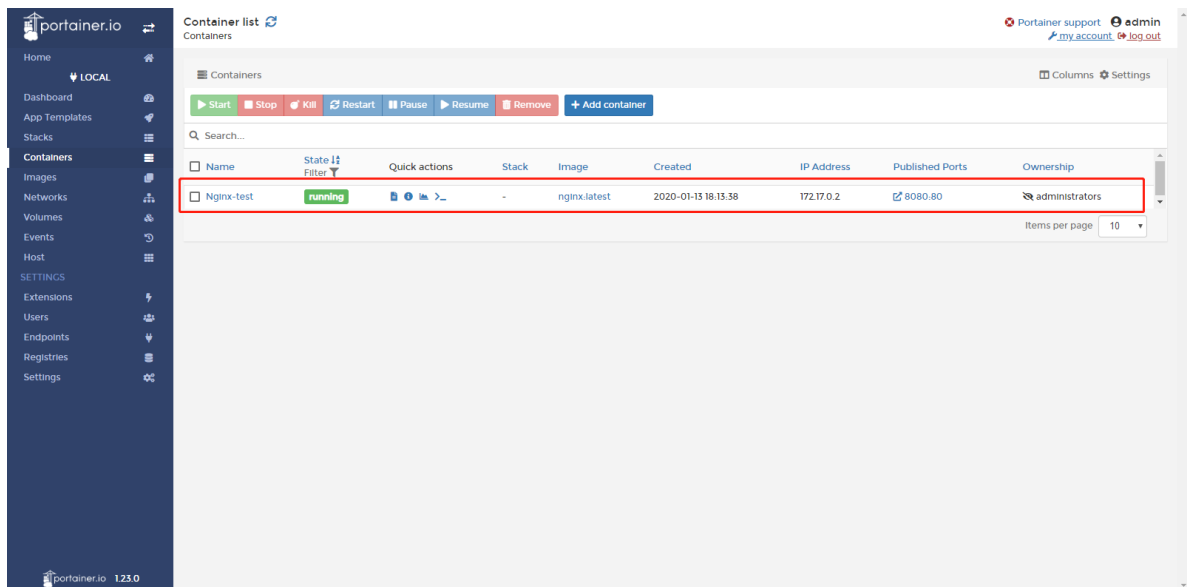
- Step 1: Choose Local > Containers page of Portainer and click Add container to add a new container.



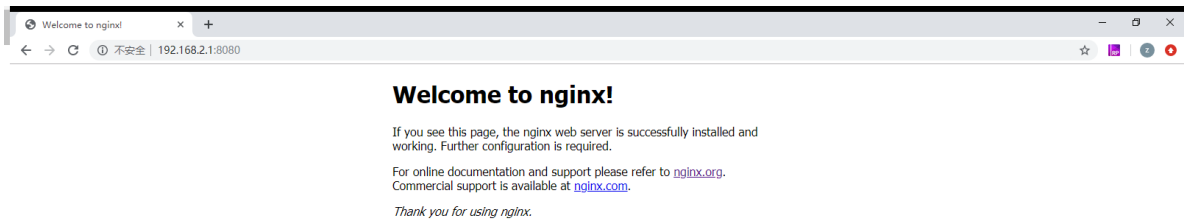
- Step 2: Configure the operating parameters for the container and deploy the container.



- Step 3: The container will run automatically after deployment. You can view the container running status on Portainer's Local > Containers page.



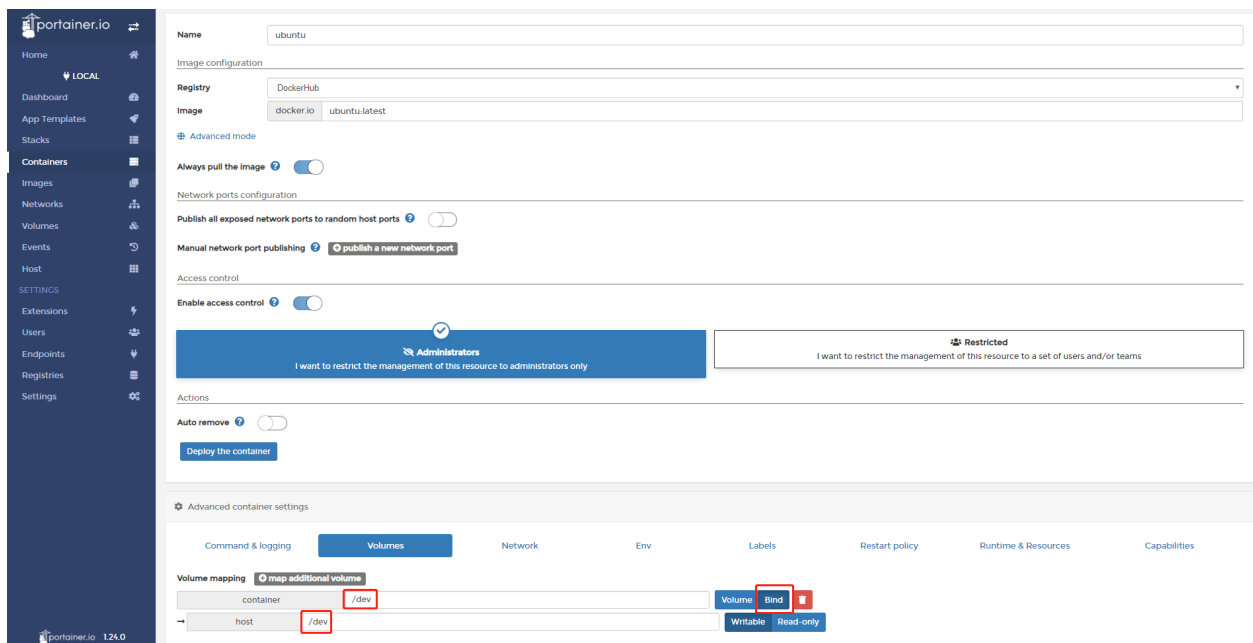
- Step 4: After entering the Nginx access link (IP address + port number of IG902) configured in the container in the browser, you can see the Nginx welcome page. This shows that the Nginx docker image has been running on the IG902 normally. Now, you have completed adding and deploying an Nginx docker image on the IG902.



1.1.3 Appendix

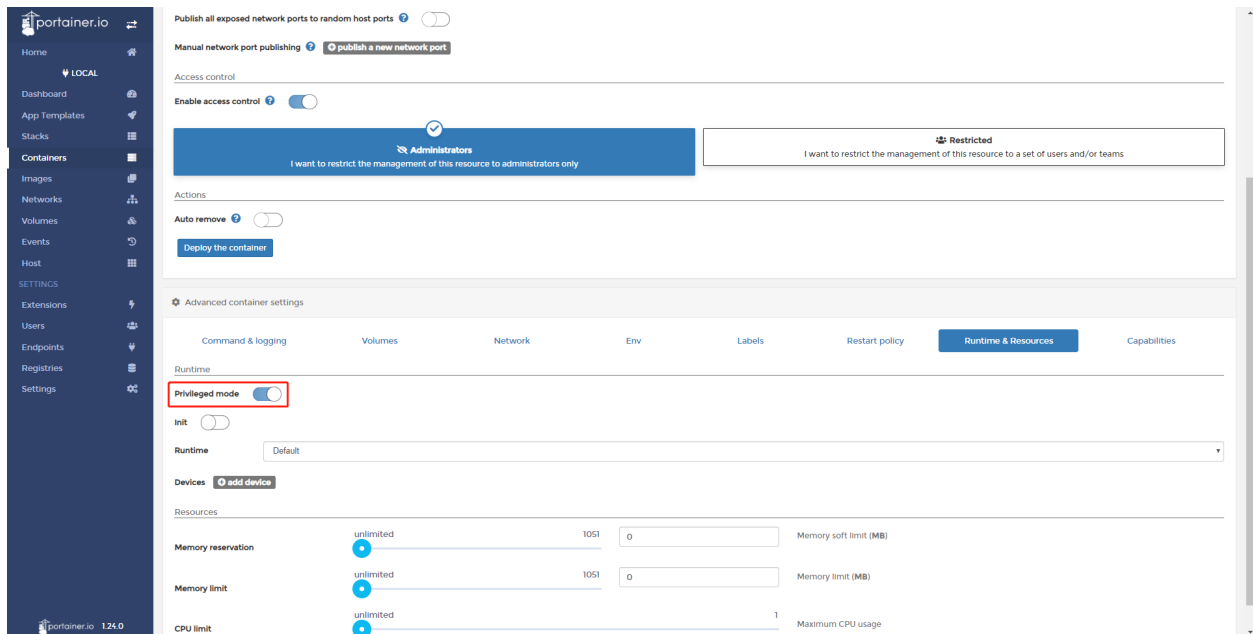
Use Serial port for communication in container

When deploying the container, add **Volume Mapping** to the **Advanced Container Settings > Volumes** page of the Portainer. The following figure maps the files in the dev directory of IG902 to the dev directory in the container (The corresponding interface file is included in the dev directory of IG902).

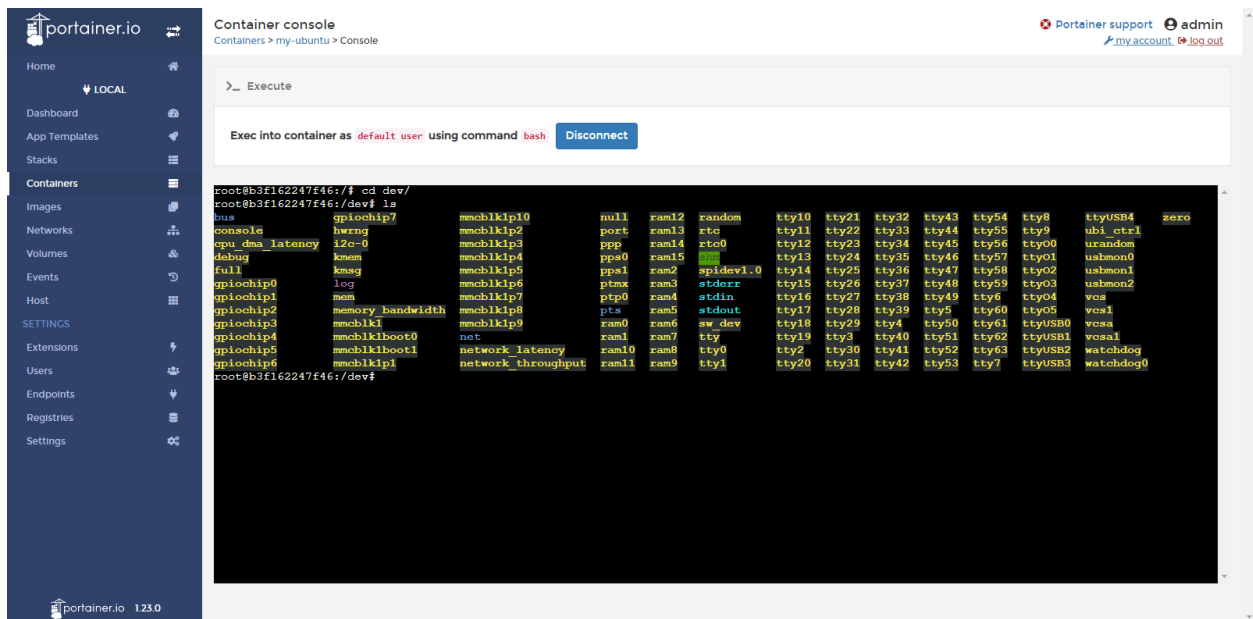


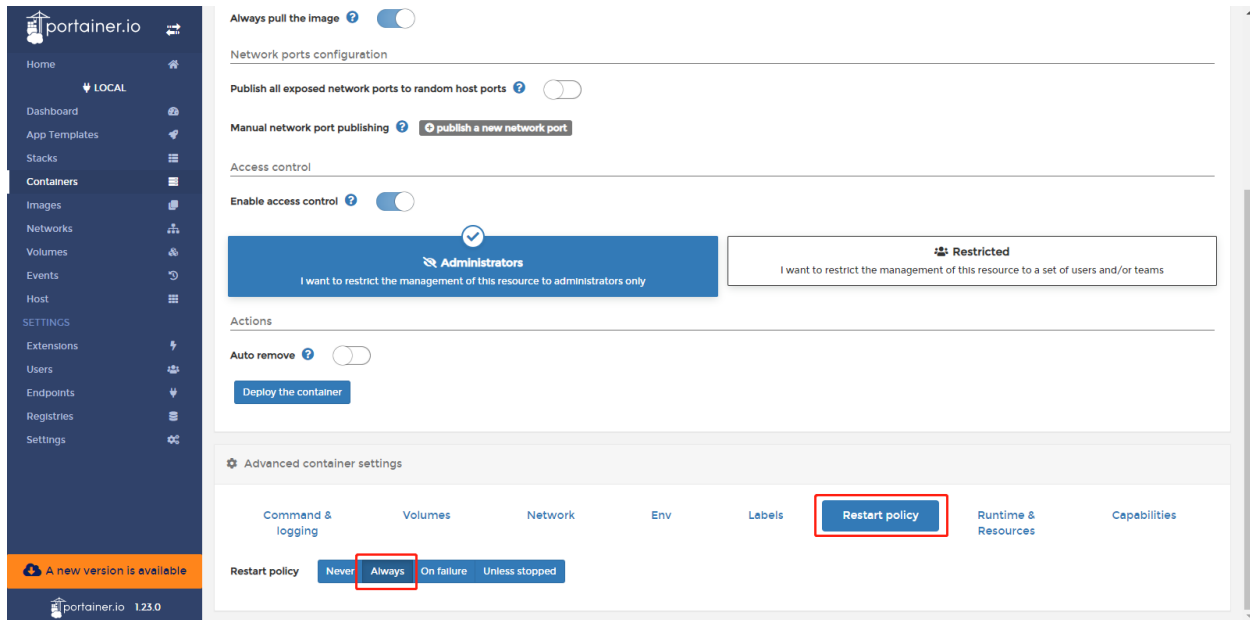
Enable **Privileged mode** on the **Advanced Container Settings > Runtime & Resources** page of the

Portainer (if it is not enabled, using the serial port will prompt that there is no operation authority).



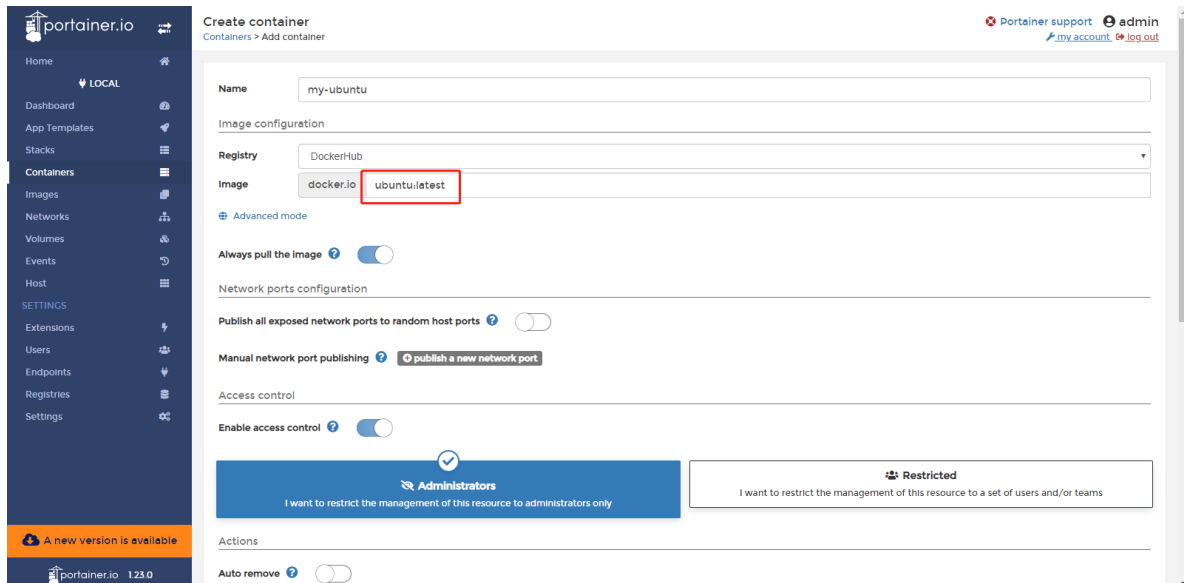
At this time you can deploy the container. Enter the dev directory of the container's console, you can find the interface files such as `tty01` and `tty03`.





Run Ubuntu in IG902

- Step 1: Pull the Ubuntu image on the **Local > Images** page of the Portainer, as shown below:



- Step 2: Go to **Local > Containers** page of the Portainer, click **Add container** to add a new container. Select the Ubuntu image downloaded in the previous step as the containers image. At the same time, click **Advanced Container Settings > Command & Logging**, check **Interactive & TTY** in the **Console**. After the configuration is complete, click **Deploy the container** to deploy the container.

The top screenshot shows the Portainer.io 'Image list' page. The left sidebar contains navigation links: Home, LOCAL, Dashboard, App Templates, Stacks, Containers, Images, Networks, Volumes, Events, Host, SETTINGS, Extensions, Users, Endpoints, Registries, and Settings. The main content area has a 'Pull image' form with 'Registry' set to 'DockerHub' and 'Image' set to 'docker.io e.g. myImage:myTag'. Below the form is a table of images:

Id	Tags	Size	Created
sha256:a156e644b659404b890e4a82b65630...	nginx:latest	98.2 MB	2020-05-15 19:09:30
sha256:d78ec34bae497b9a4ad5220ead9165...	ubuntu:latest	46.7 MB	2020-05-21 02:58:39

The bottom screenshot shows the 'Advanced container settings' page for a container. The left sidebar is the same as the top screenshot. The main content area has a 'Deploy the container' button and a 'Command & logging' tab. The 'Command & logging' tab shows the following settings:

- Command: e.g. /usr/bin/nginx -t -c /mynginx.conf
- Entry Point: e.g. /bin/sh -c
- Working Dir: e.g. /myapp
- User: e.g. nginx
- Console: ☒ Interactive & TTY (-i -t) (highlighted with a red box), ☐ Interactive (-i), ☐ None
- Logging Driver: Default logging driver

- Step 3: After deployment, on the **Local > Containers** page of Portainer, you can see that the container is running. Click on **Exec Console** to log in to the console.

The screenshot shows the Portainer.io interface. On the left is a sidebar with navigation links: Home, LOCAL, Dashboard, App Templates, Stacks, Containers (selected), Images, Networks, Volumes, Events, Host, SETTINGS, Extensions, Users, Endpoints, Registries, and Settings. The main area is titled 'Container list' and shows a table of containers. The 'my-ubuntu' container is highlighted with a red box around its 'running' status and the 'Exec Console' button. The table has columns: Name, State, Quick actions, Stack, Image, Created, IP Address, Published Ports, and Ownership. The 'test' container is also visible with a 'stopped' status.

Name	State	Quick actions	Stack	Image	Created	IP Address	Published Ports	Ownership
my-ubuntu	running	i l x e	-	ubuntu.latest	2020-06-29 15:54:20	172.17.0.2	-	administrators
test	stopped	i l x e	Exec Console	nginx.latest	2020-05-29 11:46:41	172.17.0.4	9090:80	administrators

Click on **Connect** in **Execute** page, and then enter the container to run the corresponding command.

The screenshot shows the 'Container console' page for the 'my-ubuntu' container. The 'Execute' section is active, showing a 'Command' field with '/bin/bash' and a 'User' field with 'root'. The 'Connect' button is highlighted with a red box.

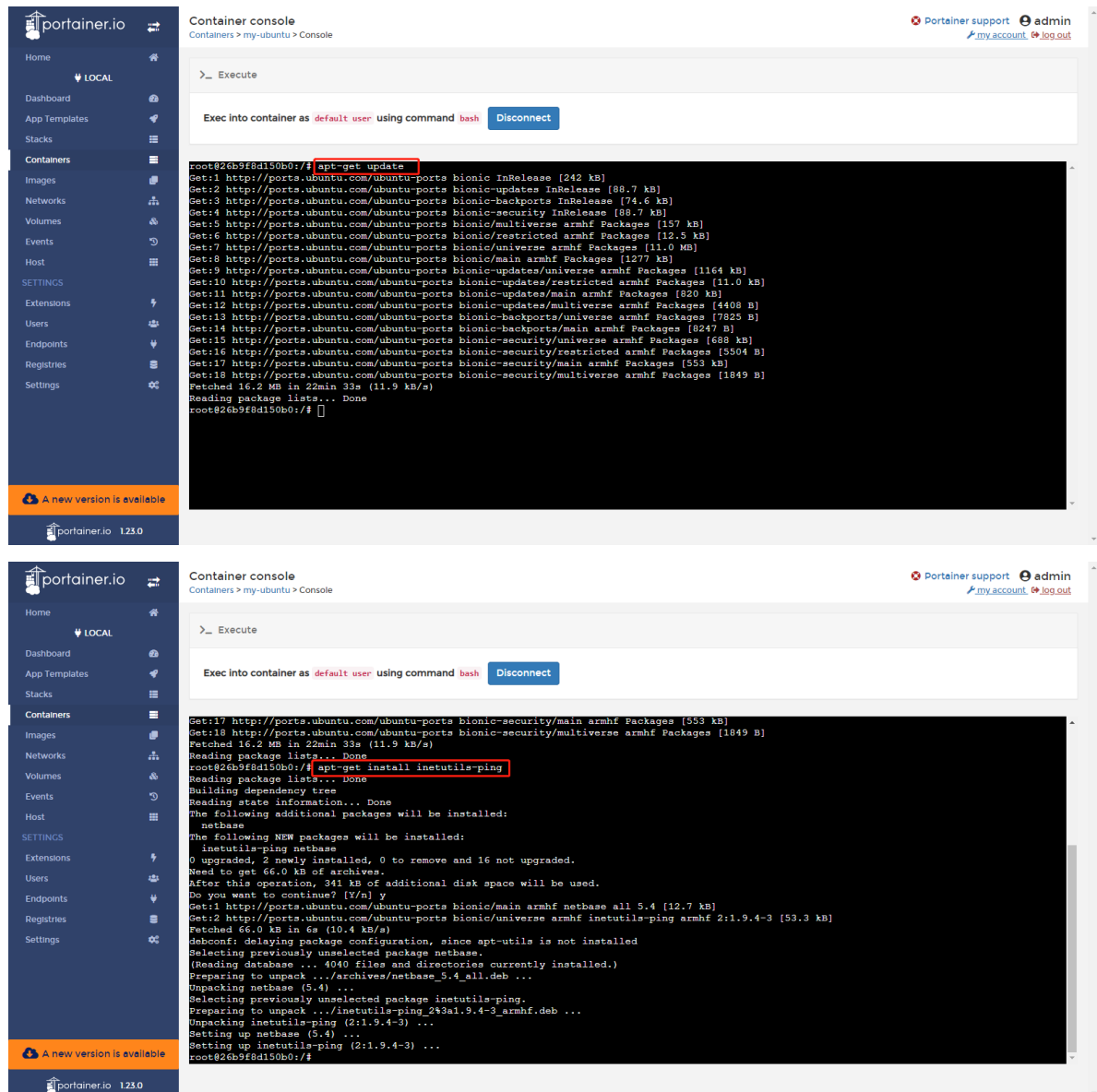
The screenshot shows the 'Container console' page for the 'my-ubuntu' container. The 'Execute' section is active, showing a 'Command' field with 'bash' and a 'User' field with 'default user'. The 'Disconnect' button is visible. Below the input fields, the terminal output shows the command 'ls' being executed, resulting in a list of files and directories: 'root@26b9f8d150b0:/# ls', 'proc', 'sys', and 'dev'.

Build the image through a container (create an image to save the container configuration)

When the corresponding development or operating environment has been configured in the container, if you need to save the environment configuration, you can create a new image based on the container's changes. The method is as follows (take the ping tool installation in the Ubuntu container as an example):

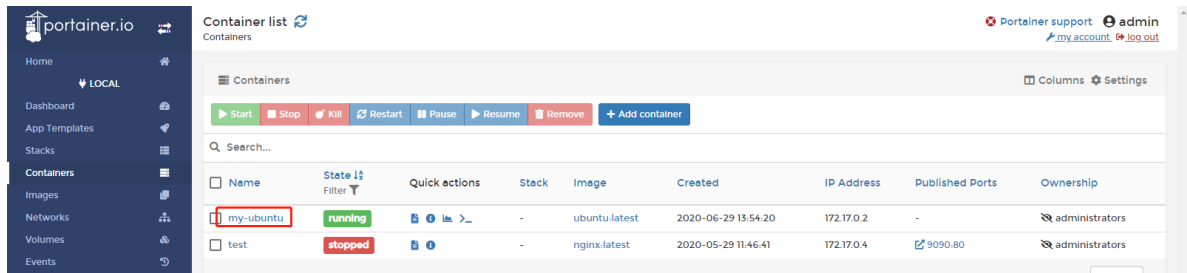
- Step 1: Configure the development or runtime environment of the container.

Run the `apt-get update` and `apt-get install inetutils-ping` commands to install the ping tool.

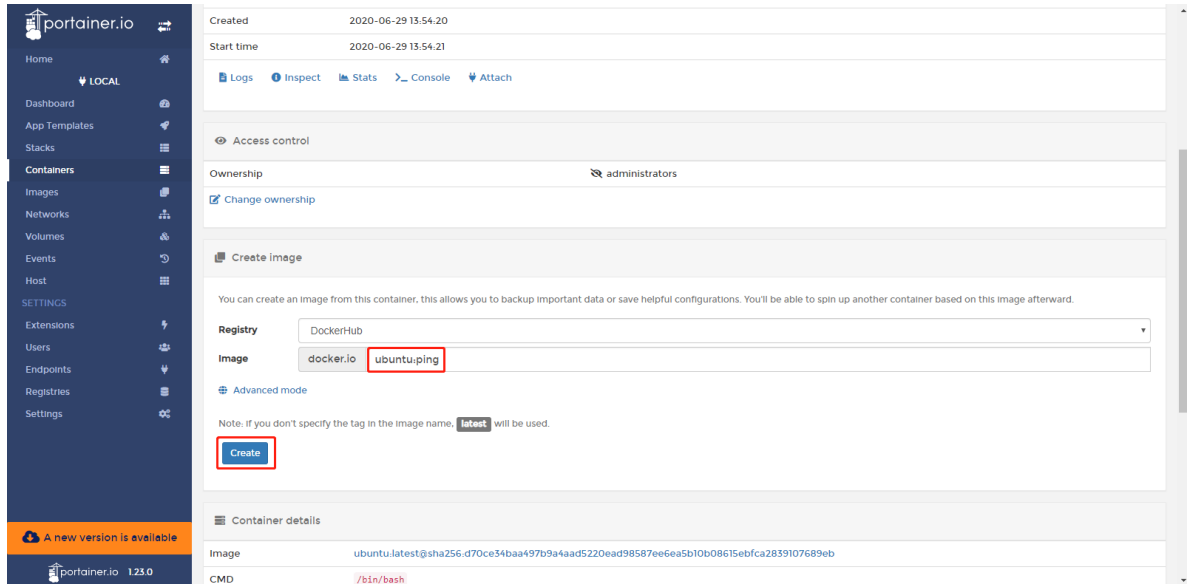


- Step 2: Create an image based on the container.

Click on the container name to enter the details of the container page.

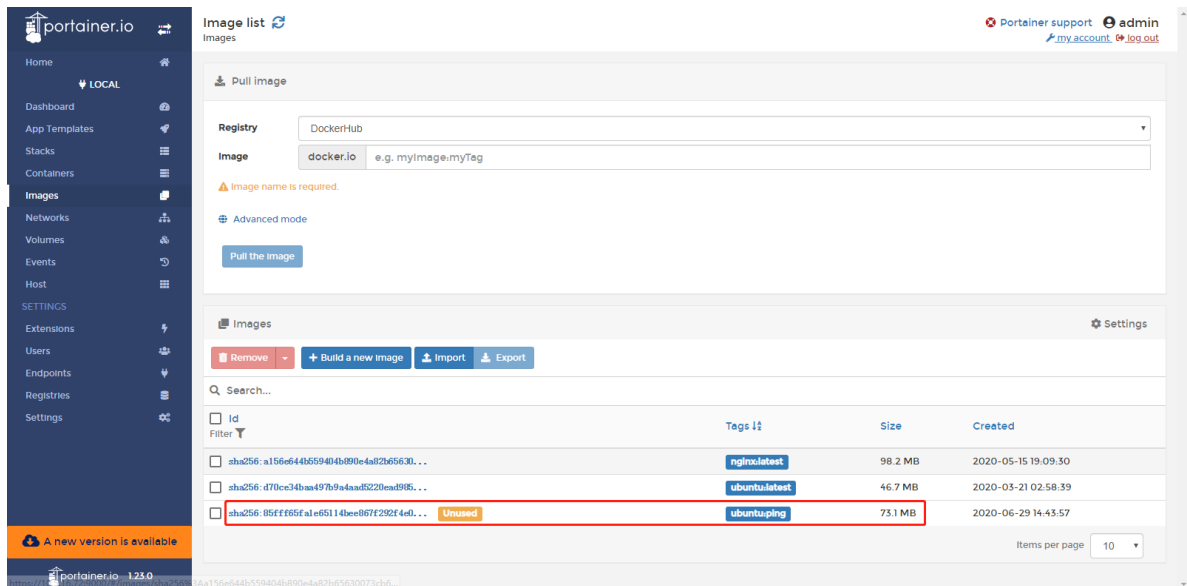


Configure the name of the image in the **Create Image** on the details page and click on **Create**.



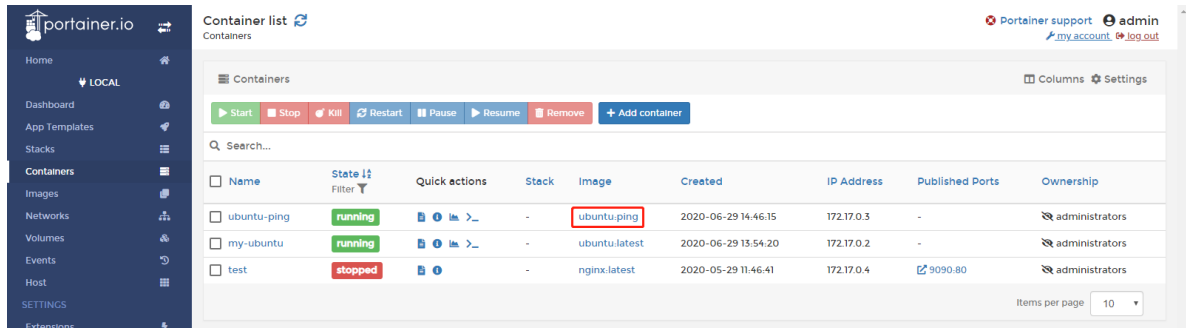
- Step 3: Use the created image to deploy the container.

After the image is created, you can view it in **Local > Images** page of Portainer.

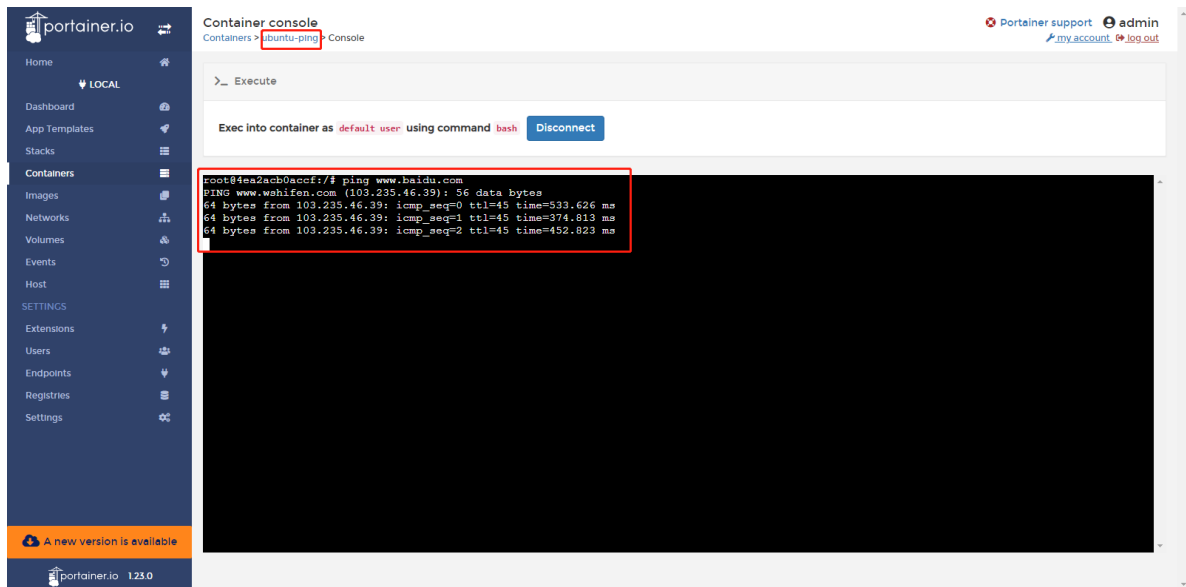


Then deploy an ubuntu container using the image on the **Local > Containers** page of Portainer. As

shown below:

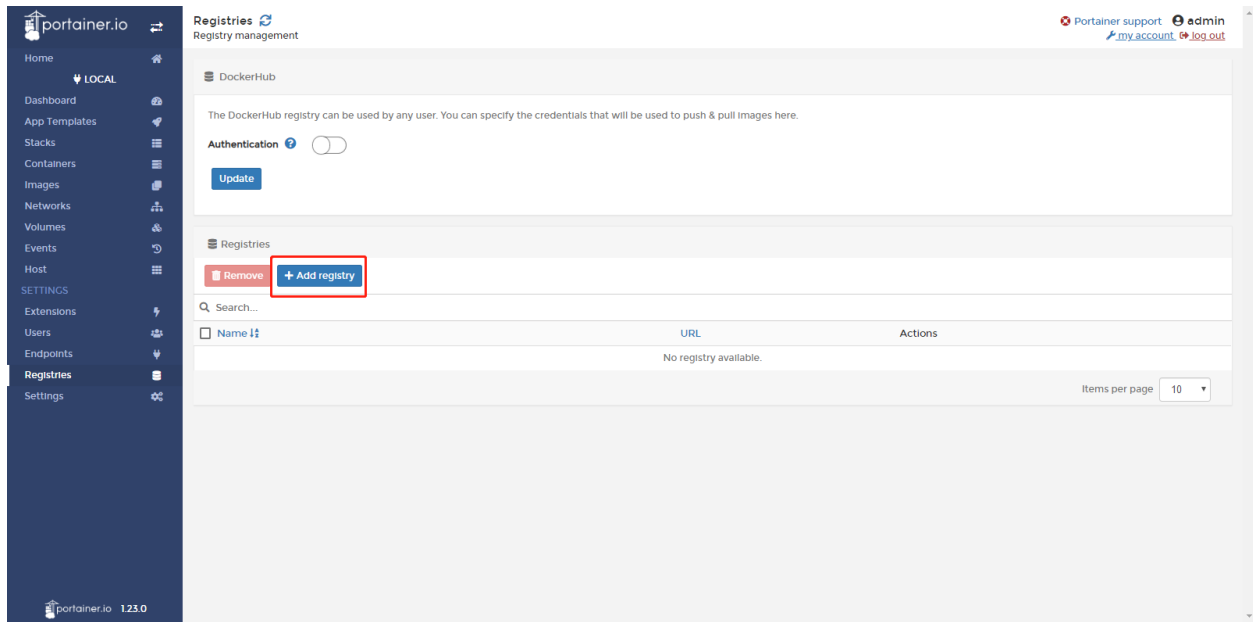


Log in to the console of the container, you can use your ping command.

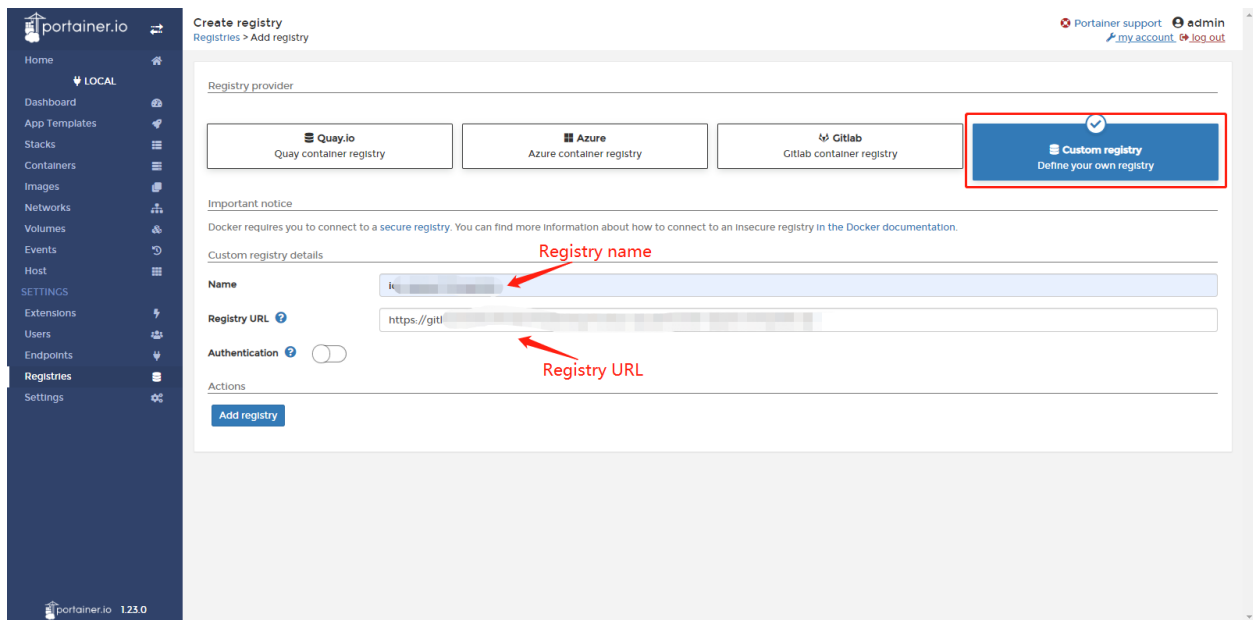


How to download docker images from gitlab / github

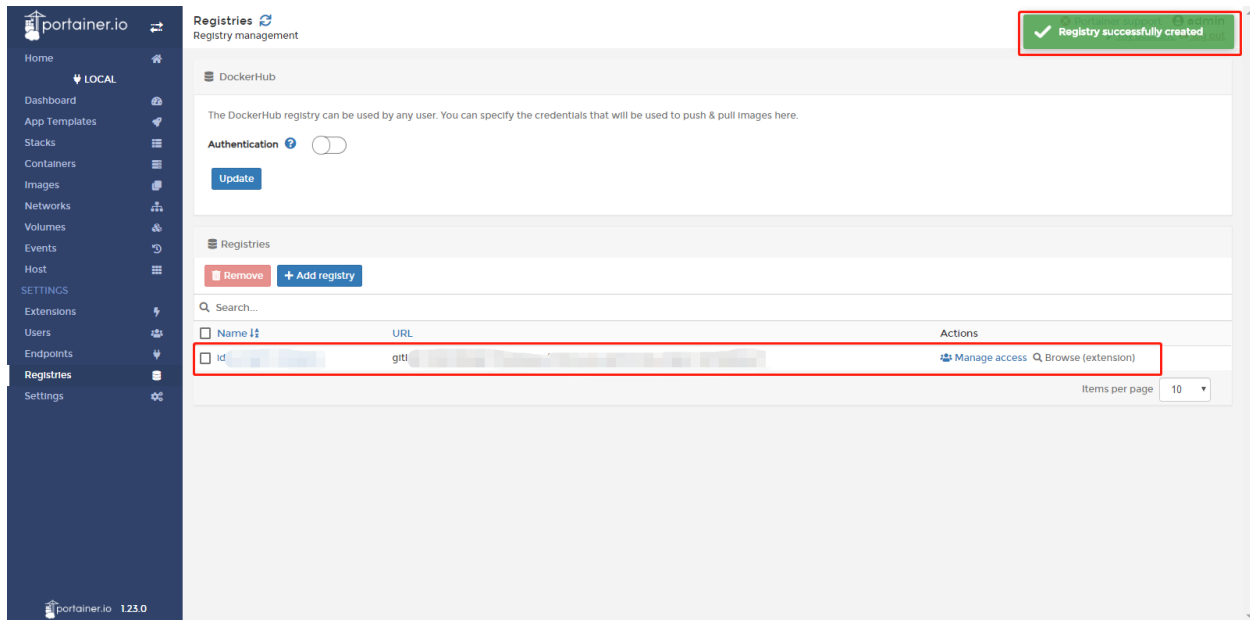
Choose Local > Registries page of Portainer and click Add registry to add a docker mirror repository (must be a public repository).



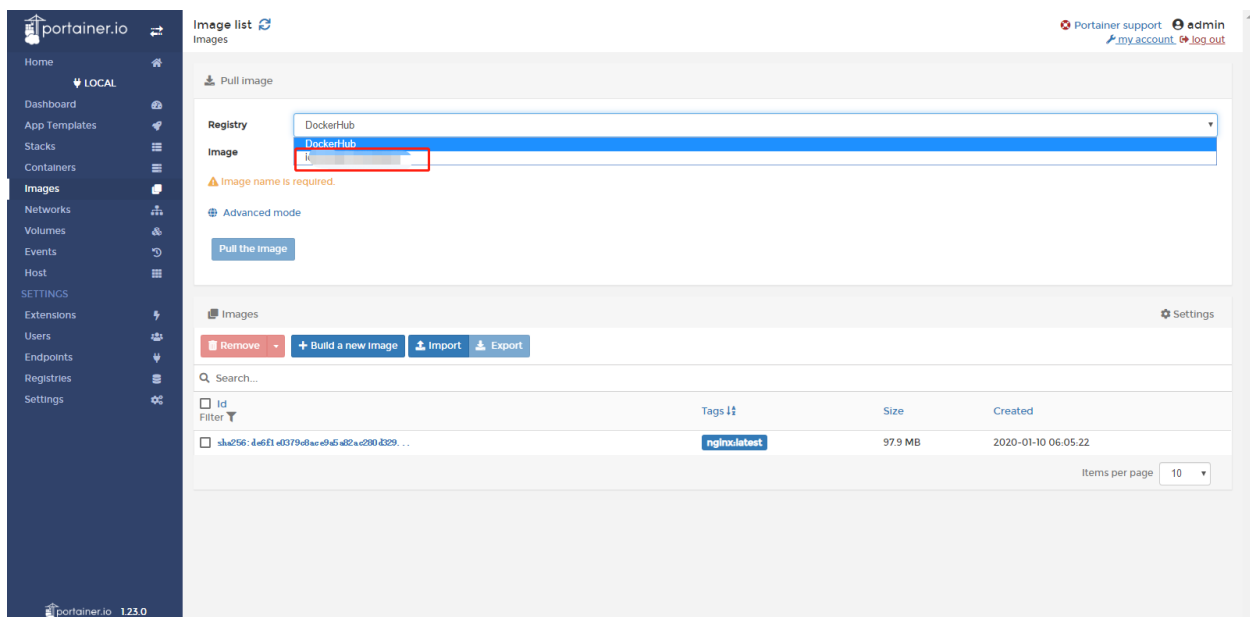
Then select Custom registry and configure the mirror repository information. After configuration, click Add registry.



After the mirror repository is successfully added, you can see the web page as shown below:



After the addition is successful, you can select the configured image repository when pulling the docker image.

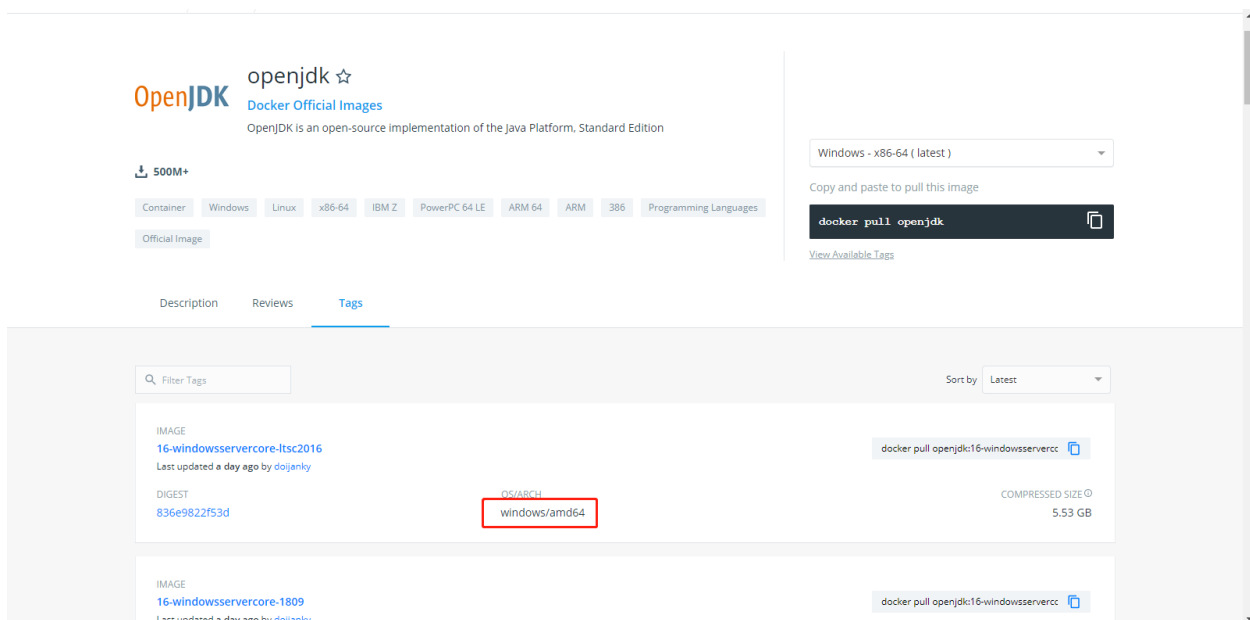
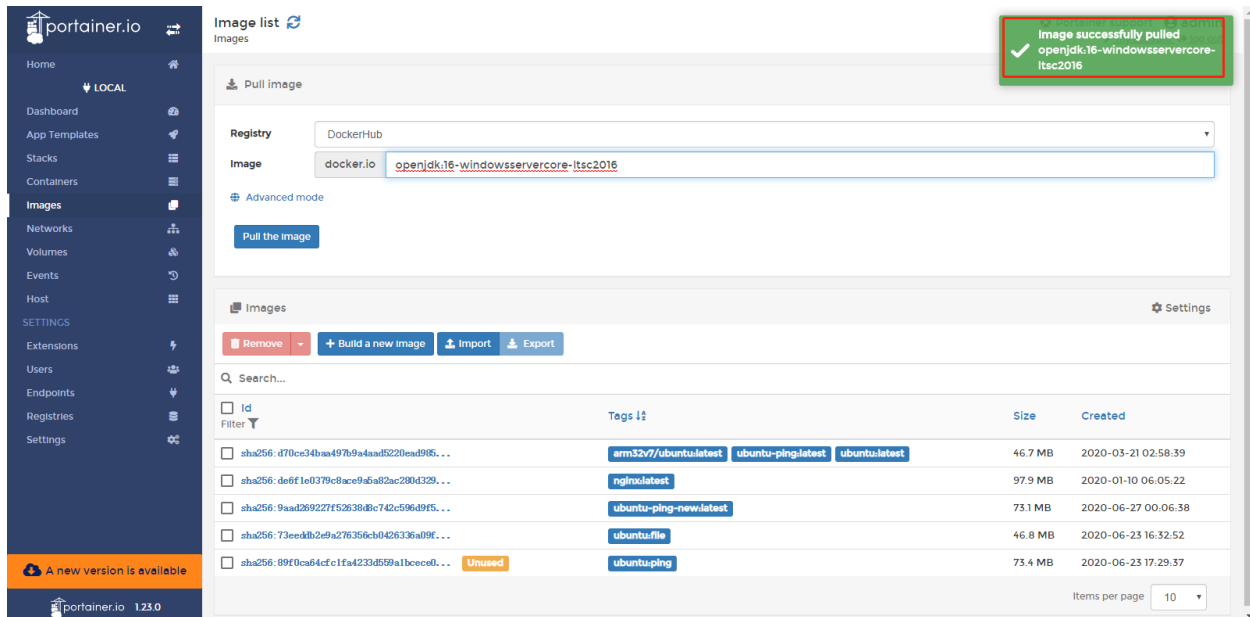


1.1.4 FAQ

Q1: It prompted succeed after pulling image on the “Images” page, but the image is not shown on the “Images” page.

A1: That is because the architecture of IG902 CPU is linux/arm/v7. Only images that support linux/arm/v7 architecture can run normally in IG902. Images of other architectures, such as window/amd64, may not be

imported, pulled, or run successfully in IG902.



1.2 Azure IoT Edge User Manual

Azure IoT Edge integrates cloud analysis and custom service logic into devices to help organizations focus on service insights instead of data management. It packages service logic into standard containers, scales out IoT solutions, deploys these containers to devices, and then monitors these devices on the cloud. For more information about Azure IoT Edge, see [About Azure IoT Edge module](#). The InHand IG902-series products provide Azure IoT Edge SDK to support Azure IoT Edge and enable you to quickly develop and complete tasks and securely and efficiently deploy services. This SDK manages the Azure IoT Edge runtime to manage

the Azure cloud platform deployment and run the IoT Edge module (docker image) on the IoT Edge device (IG902). This document describes how to deploy and run an IoT Edge module that simulates telemetry data and sends it to IoT Hub on IG902 through the Azure platform by using the Azure IoT Edge SDK.

- *1. Environment Preparations*
 - *1.1 Configure the Azure IoT environment*
 - *1.2 Configure the IG902 environment*
 - * *1.2.1 Configure IG902 to connect to the Internet*
 - * *1.2.2 Update the IG902 software version*
 - *1.3 Modify the configuration file of Azure IoT Edge*
- *2. Run Azure IoT Edge*
- *3. Configure and Deploy the Module*

1.2.1 1. Environment Preparations

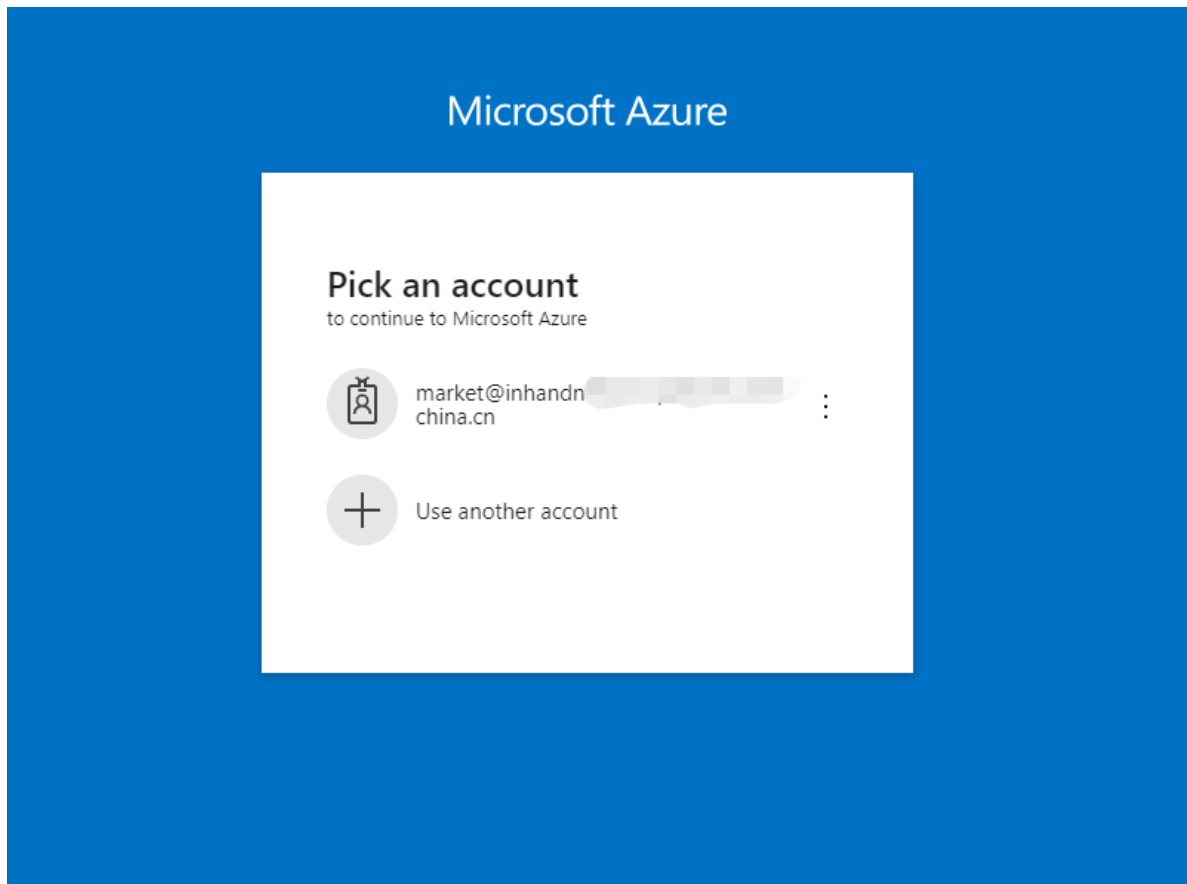
You need to prepare the following items before start (as for how to get the IG902 software version, please visit the [Resource](#).):

- Azure IoT account
- IG902 firmware version: v2.0.0.r12644 or later
- Docker SDK version: 18.06.3-ce or later
- Azure IoT Edge SDK version: 1.0.4 or later
- IG902-series products

1.1 Configure the Azure IoT environment

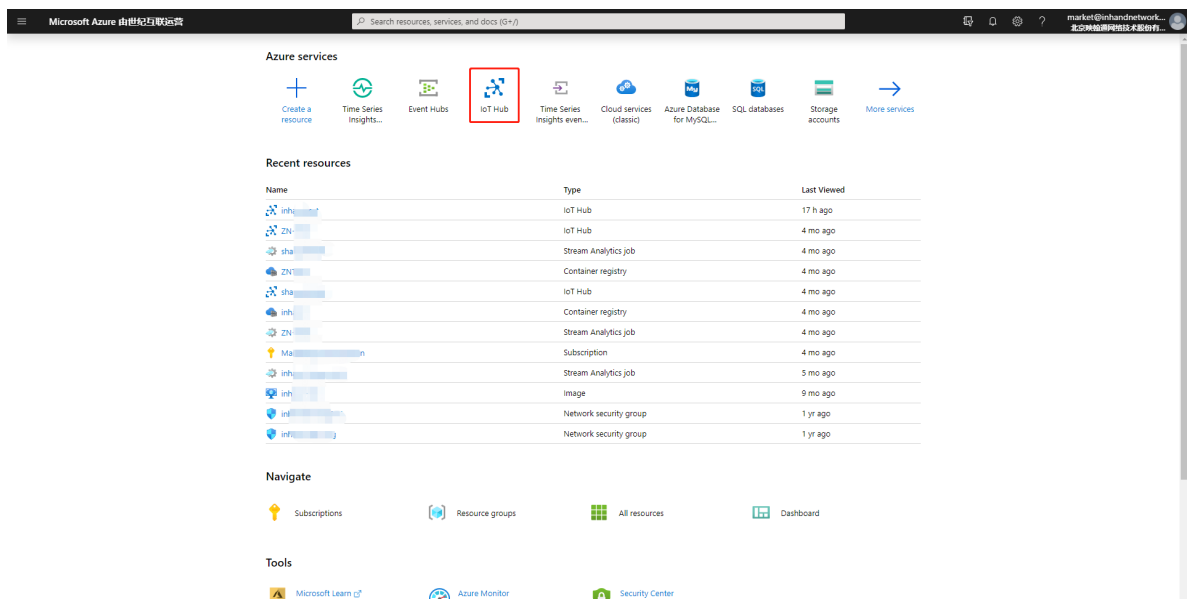
If you have deployed the corresponding IoT Hub and IoT Edge devices on Azure IoT, skip this section.

- Step 1: Log in to Azure IoT
Visit <https://portal.azure.cn/> and log in to Azure.



- Step 2: Add IoT Hub

After successful login, the page is as follows. Select **IoT Hub**.



Click **Add** to create an IoT Hub.

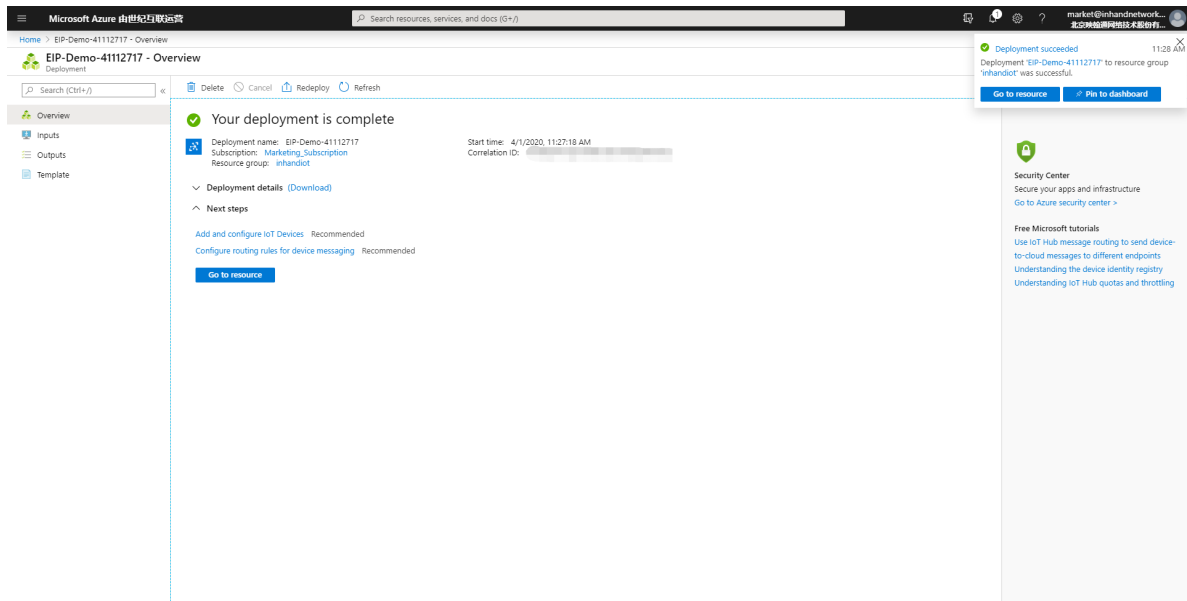
The screenshot displays the Microsoft Azure IoT Hub management interface. The top section shows a list of IoT Hubs with columns for Name, Type, Resource group, Location, and Subscription. The 'Add' button is highlighted with a red box. The bottom section shows the 'Create new IoT Hub' form with fields for Subscription, Resource group, Region, and IoT hub name, all of which are filled out. The 'Review + create' button is highlighted with a red box.

Name	Type	Resource group	Location	Subscription
IoT Hub	IoT Hub	inhandiot	China North	Marketing_Subscription
IoT Hub	IoT Hub	inhandiot	China North	Marketing_Subscription
IoT Hub	IoT Hub	ZH-TEST	China North	Marketing_Subscription

Subscription * Marketing_Subscription
Resource group * inhandiot
Region * China North
IoT hub name * EIP-Demo

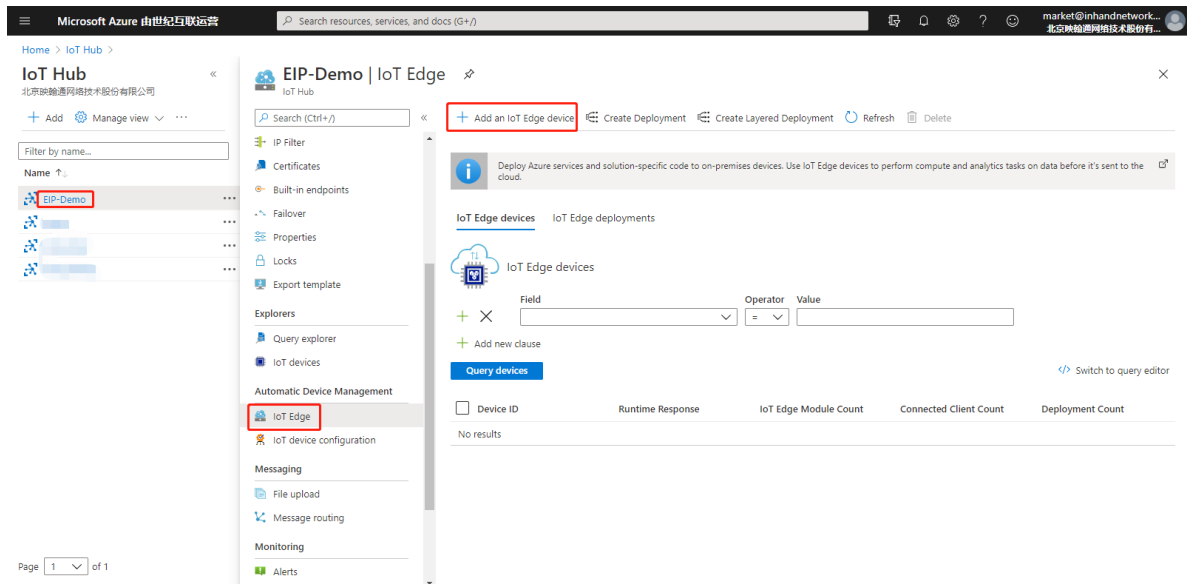
Review + create

After the IoT Hub is created, the page is as follows:




- Step 3: Add an IoT Edge device

On the **IoT Hub** page, click the target IoT Hub. On the **IoT Edge** that appears, click **Add an IoT Edge device**.





Configure the parameters and click **Save**.


 Microsoft Azure 由世纪互联运营


Search resources, services, and docs (G


Home > IoT Hub > EIP-Demo | IoT Edge >

Create a device

 Find Certified for Azure IoT devices in the Device Catalog 


Device ID * 

EIP-demo-edge 


Authentication type 

Symmetric key


X.509 Self-Signed

Primary key 


Enter your primary key

Secondary key 

Enter your secondary key


Auto-generate keys 

☒

Connect this device to an IoT hub 

Enable

Disable

Child devices 

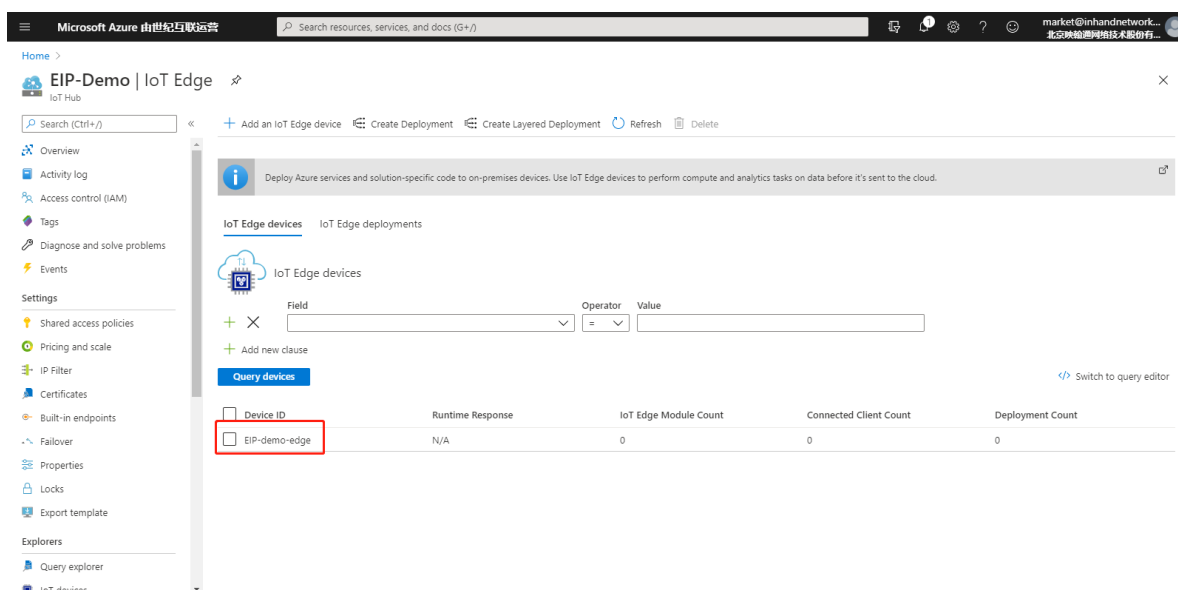
0

Choose child devices

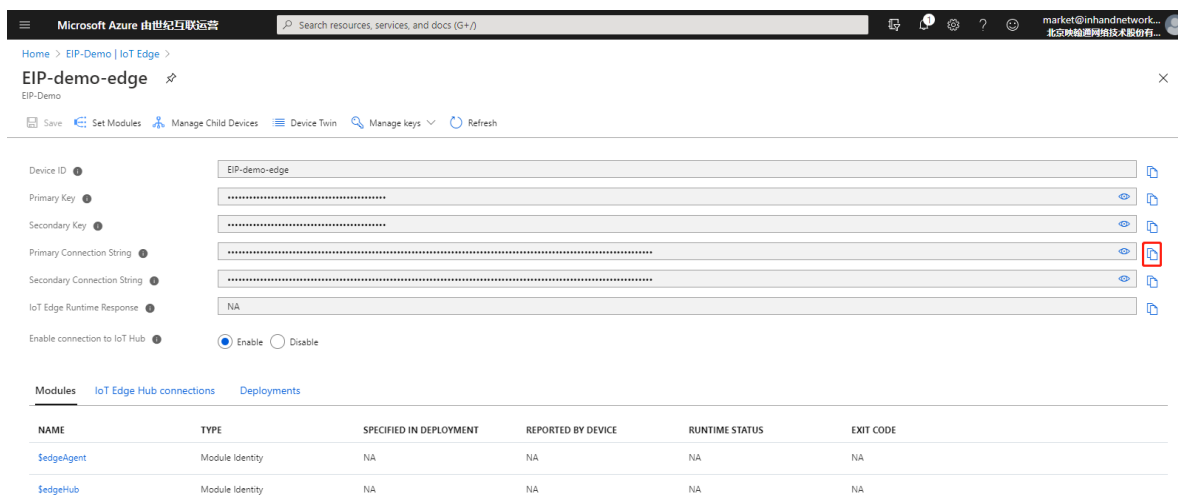
Save

- Step 4: Copy the connection string of the IoT Edge device

After the IoT Edge device is created, the page is as follows:



Click **Device ID** of the IoT Edge device. On the details page of the IoT Edge device that appears, copy the **Primary Connection String** parameter for subsequent use.



1.2 Configure the IG902 environment

1.2.1 Configure IG902 to connect to the Internet

As for how to configure IG902 to connect to the Internet, see [Connect IG902 to the Internet](#).

1.2.2 Update the IG902 software version

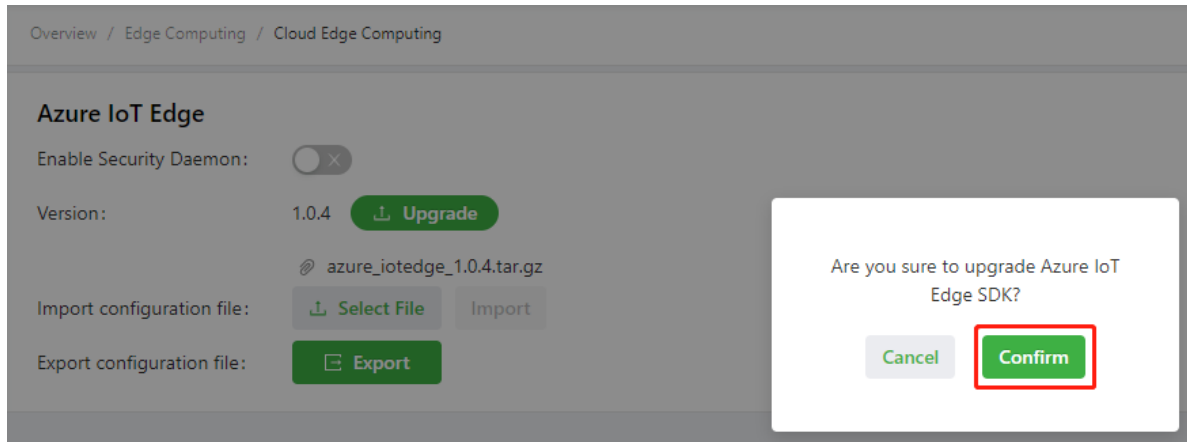
- Update the IG902 firmware version

As for how to update the IG902 firmware version, see [Update the IG902 software version](#).

- Update the IG902 Docker SDK

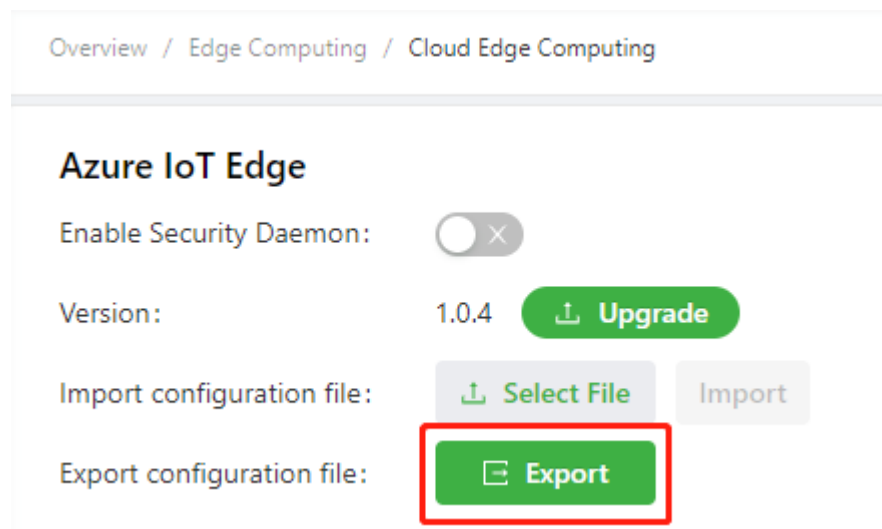
As for how to update the IG902 Docker SDK, see [Install Docker SDK](#).

- Update the IG902 Azure IoT Edge SDK Choose **Edge Computing >> Cloud Edge Computing**, uncheck **Enable Security Daemon**, click **Upgrade**, select the Azure IoT Edge SDK file, and click **Confirm**.



1.3 Modify the configuration file of Azure IoT Edge

On the **Edge Computing >> Cloud Edge Computing** page, click **Export** to export the configuration file of Azure IoT Edge.



Modify the `device_connection_string` parameter in the configuration file of Azure IoT Edge and save the modification. This string is the **Primary Connection String** of the IoT Edge device that you copied in the step *Copy the connection string of the IoT Edge device*.

```

31 # .....symmetric_key..... Optional. This entry should only be specified when
32 # .....provisioning devices configured for symmetric key
33 # .....attestation: Device specific symmetric key
34 # .....identity_cert..... Optional. The Edge device identity X.509 certificate
35 # .....entry should only be specified when provisioning
36 # .....an Edge device configured for X.509 attestation
37 # .....The value should be specified as a URI
38 # .....Ex. when specifying a PEM encoded certificate file, the URI
39 # .....should be specified as file:///path/identity_certificate.pem
40 # .....identity_pk..... Optional. The Edge device identity private key
41 # .....entry should only be specified when provisioning
42 # .....an Edge device configured for X.509 attestation
43 # .....The value should be specified as a URI
44 # .....Ex. when specifying a PEM encoded private key file, the URI
45 # .....should be specified as file:///path/identity_key.pem
46 #
47 # External Settings
48 # .....endpoint..... Required. Value of the endpoint used to retrieve device specific
49 # .....information such as its IoT hub connection information
50 # .....
51 #
52 # Manual provisioning configuration
53 provisioning:
54   source: "manual"
55   device_connection_string: "HostName=EIF-Demo,
56
57 # DPS-TPM provisioning configuration
58 provisioning:
59   source: "dps"
60   global_endpoint: "https://global.azure-devices-provisioning.net"
61   scope_id: "<SCOPE_ID>"
62   attestation:
63     method: "tpm"
64     registration_id: "<REGISTRATION_ID>"
65
66 # DPS-symmetric-key provisioning configuration
67 provisioning:
68   source: "dps"
69   global_endpoint: "https://global.azure-devices-provisioning.net"
70   scope_id: "<SCOPE_ID>"
71   attestation:
72     method: "symmetric-key"
73     registration_id: "<REGISTRATION_ID>"
74     symmetric_key: "<SYMMETRIC_KEY>"
75
76 # DPS X.509 provisioning configuration

```

Import the modified configuration file of Azure IoT Edge.

Overview / Edge Computing / Cloud Edge Computing

Azure IoT Edge

Enable Security Daemon: ☐

Version: 1.0.4 [Upgrade](#)

Import configuration file: [Select File](#) [Import](#)

config.yaml

Export configuration file: [Export](#)

1.2.2 2. Run Azure IoT Edge

Choose **Edge Computing >> Docker Manager** and check **Enable Docker Manager**.

Overview / Edge Computing / Docker Manager

Enable Docker Manager:



Docker Version:

18.06.3-ce

⬇️ Upgrade

User Name:

admin

* Password:

.....



* Port:

9000

[Go to the Docker management page](#)

Submit

Reset

Choose **Edge Computing** >> **Cloud Edge Computing**, and check **Enable Security Daemon**.

Overview / Edge Computing / Cloud Edge Computing

Azure IoT Edge

Enable Security Daemon:



Version:

1.0.4

⬇️ Upgrade

Import configuration file:

⬇️ Select File

Import

Export configuration file:

📄 Export

After the security daemon is enabled, the Azure IoT Edge daemon pulls an image to create a container named **edgeAgent**. This process takes for about 20 minutes because the image is large. You can choose **LOCAL** >> **Containers** on portainer to check whether the **edgeAgent** container is running. If the **edgeAgent** container is running, Azure IoT Edge is in normal operating state.

Overview / Edge Computing / Docker Manager

Enable Docker Manager:



Docker Version:

18.06.3-ce

Upgrade

User Name:

admin

* Password:

.....



* Port:

9000

[Go to the Docker management page](#)

Submit

Reset

Connect Portainer to the Docker environment you want to manage.

Local
Manage the local Docker environment

Remote
Manage a remote Docker environment

Agent
Connect to a Portainer agent

Azure
Connect to Microsoft Azure ACI

Information

Manage the Docker environment where Portainer is running.

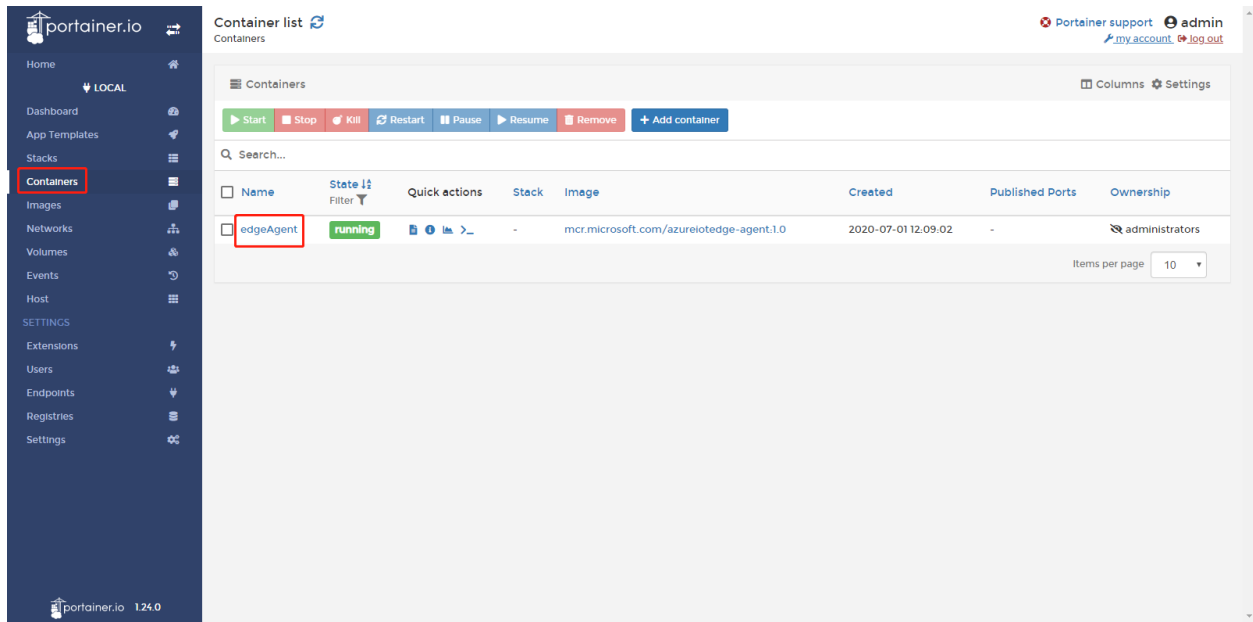
ⓘ Ensure that you have started the Portainer container with the following Docker flag:

`-v "/var/run/docker.sock:/var/run/docker.sock"` (Linux).

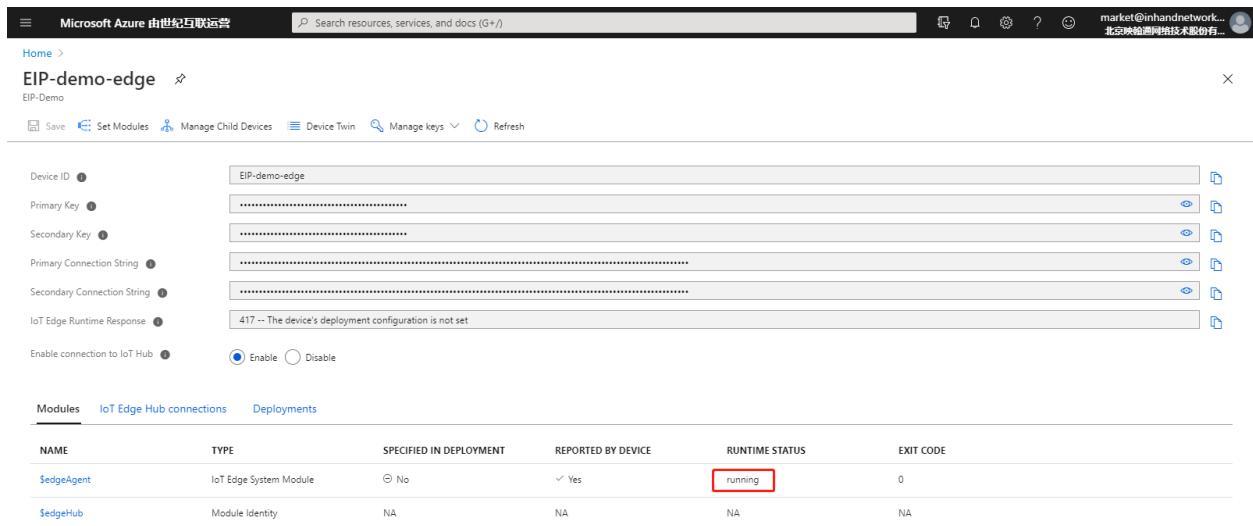
or

`-v "\\.\pipe\docker_engine:\\.\pipe\docker_engine"` (Windows).

Connect



Then, **RUNTIME STATUS** of \$edgeAgent on the details page of the IoT Edge device is running.



1.2.3 3. Configure and Deploy the Module

- Step 1: Add the IoT Edge module

On the details page of the IoT Edge device, click **Set Modules**.

Microsoft Azure 由世纪互联运营

Home > IoT Hub > EIP-Demo | IoT Edge > EIP-demo-edge

EIP-demo-edge

Save Set Modules Manage Child Devices Device Twin Manage keys Refresh

Device ID: EIP-demo-edge

Primary Key: [Redacted]

Secondary Key: [Redacted]

Primary Connection String: [Redacted]

Secondary Connection String: [Redacted]

IoT Edge Runtime Response: 417 -- The device's deployment configuration is not set

Enable connection to IoT Hub: ☒ Enable ☐ Disable

NAME	TYPE	SPECIFIED IN DEPLOYMENT	REPORTED BY DEVICE	RUNTIME STATUS	EXIT CODE
SedgeAgent	IoT Edge System Module	No	Yes	running	0
SedgeHub	Module Identity	NA	NA	NA	NA

On the **Set modules on device** page, click **Add** and select **IoT Edge Module** to add the IoT Edge module.

Microsoft Azure 由世纪互联运营

Home > IoT Hub > EIP-Demo | IoT Edge > EIP-demo-edge

Set modules on device: EIP-demo-edge

Modules Routes Review + create

Container Registry Credentials

You can specify credentials to container registries hosting module images. Listed credentials are used to retrieve modules with a matching URL. The Edge Agent will report error code 500 if it can't find a container registry setting for a module.

NAME	ADDRESS	USER NAME	PASSWORD
Name	Address	User name	Password

IoT Edge Modules

An IoT Edge module is a Docker container you can deploy to IoT Edge devices. It communicates with other modules and sends data to the IoT Edge runtime. Using this UI you can import Azure Service IoT Edge modules or specify the settings for an IoT Edge module. Setting modules on each device will be counted towards the quota and throttled based on the IoT Hub tier and units. For example, for S1 tier, modules can be set 10 times per second if no other updates are happening in the IoT Hub.

Learn more

+ Add Runtime Settings

+ IoT Edge Module

+ Marketplace Module

+ Azure Stream Analytics Module

Review + create < Previous Next: Routes >

In the window **Add IoT Edge Module** that appears, configure the module name and image URL. This document takes the `mcr.microsoft.com/azureiotedge-simulated-temperature-sensor:1.0` image as an example. This image, provided by Microsoft, simulates telemetry data and sends the data to IoT Hub. As for how to deploy the module, see [Develop and deploy the Python IoT Edge module for the Linux device](#).

Microsoft Azure 由世纪互联运营

Home > IoT Hub > EIP-Demo > IoT Edge > EIP-demo-edge >

Set modules on device: EIP-demo-edge

EIP-Demo

Modules Routes Review + create

Add IoT Edge Module

Specify the settings for an IoT Edge custom module.
[Learn more](#)

IoT Edge Module Name *

EIP-demo-edge-module

Image URI *

mcr.microsoft.com/azureiotedge-simulated-temperature-sensor:1.0

Restart Policy

always

Desired Status

running

Review + create < Previous Next Add Cancel

After the module is added, the page is as follows:

Microsoft Azure 由世纪互联运营

Home > EIP-demo-edge >

Set modules on device: EIP-demo-edge

EIP-Demo

Modules Routes Review + create

Container Registry Credentials

You can specify credentials to container registries hosting module images. Listed credentials are used to retrieve modules with a matching URL. The Edge Agent will report error code 500 if it can't find a container registry setting for a module.

NAME	ADDRESS	USER NAME	PASSWORD
Name	Address	User name	Password

IoT Edge Modules

An IoT Edge module is a Docker container you can deploy to IoT Edge devices. It communicates with other modules and sends data to the IoT Edge runtime. Using this UI you can import Azure Service IoT Edge modules or specify the settings for an IoT Edge module. Setting modules on each device will be counted towards the quota and throttled based on the IoT Hub tier and units. For example, for S1 tier, modules can be set 10 times per second if no other updates are happening in the IoT Hub.
[Learn more](#)

+ Add ⚙ Runtime Settings

NAME	DESIRED STATUS
EIP-demo-edge-module	running

Review + create < Previous Next: Routes >

The edgeHub container listens to the port 443 and is mapped to the port 443 of the host (IG902) by default. Generally, the IG902's port 443 is listened to and occupied by other programs. Therefore, you need to modify the mapping port of the edgeHub container to ensure that edgeHub can be started. Click **Runtime Setting**. In the window **Runtime Settings** that appears, modify **HostPort** to

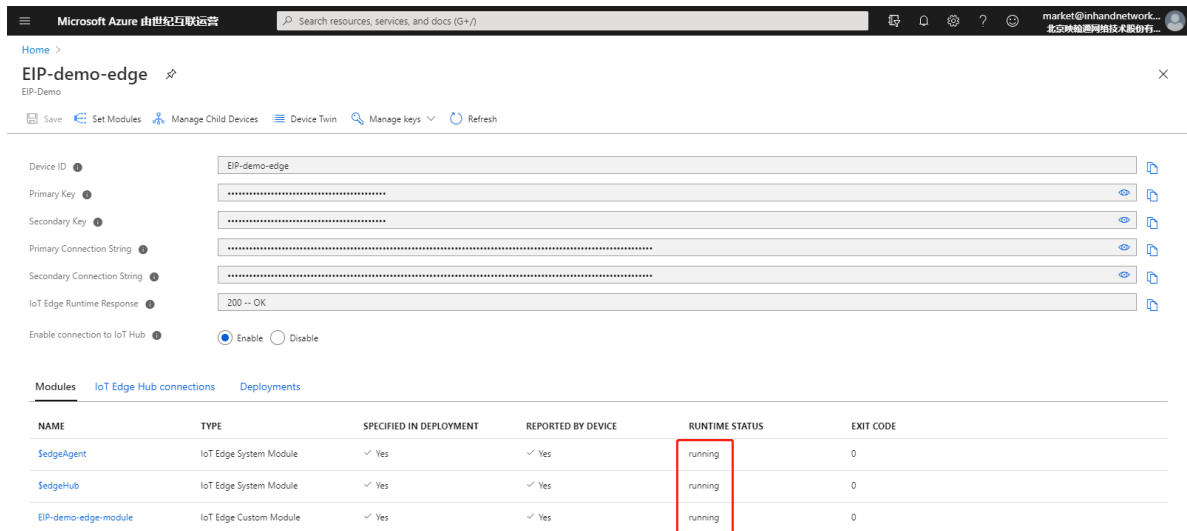
another port, such as 444. After modification, click **Save**.

The screenshot shows the 'Set modules on device: EIP-demo-edge' page in the Microsoft Azure portal. The 'Runtime Settings' sidebar is open, displaying the configuration for the 'Edge Hub' module. The 'Image' field is set to 'mcr.microsoft.com/azureiotedge-hub:1.0'. The 'Store and forward configuration - time to live (seconds)' is set to 7200. The 'Create Options' section shows a JSON configuration for 'HostConfig' with 'PortBindings' for ports 443, 5671, and 8883. The 'HostPort' for port 443 is highlighted with a red box and labeled '2'.

Then, click **Review + create** and click **Create** after confirmation. The IoT Edge module is added.

The screenshot shows the 'Set modules on device: EIP-demo-edge' page in the Microsoft Azure portal. The 'Review + create' button is highlighted with a red box. The 'IoT Edge Modules' table shows the 'EIP-demo-edge-module' with a status of 'running'.

The `edgeHub` and `mcr.microsoft.com/azureiotedge-simulated-temperature-sensor:1.0` are automatically deployed to IG902 and run on IG902. You can view the deployed containers on the details page of the IoT Edge device. If **RUNTIME STATUS** is **running**, the container has been deployed and runs normally. (The total size of two images is about 400 MB. It takes about 20 or more minutes to deploy them.)



Device ID: EIP-demo-edge

Primary Key: [Redacted]

Secondary Key: [Redacted]

Primary Connection String: [Redacted]

Secondary Connection String: [Redacted]

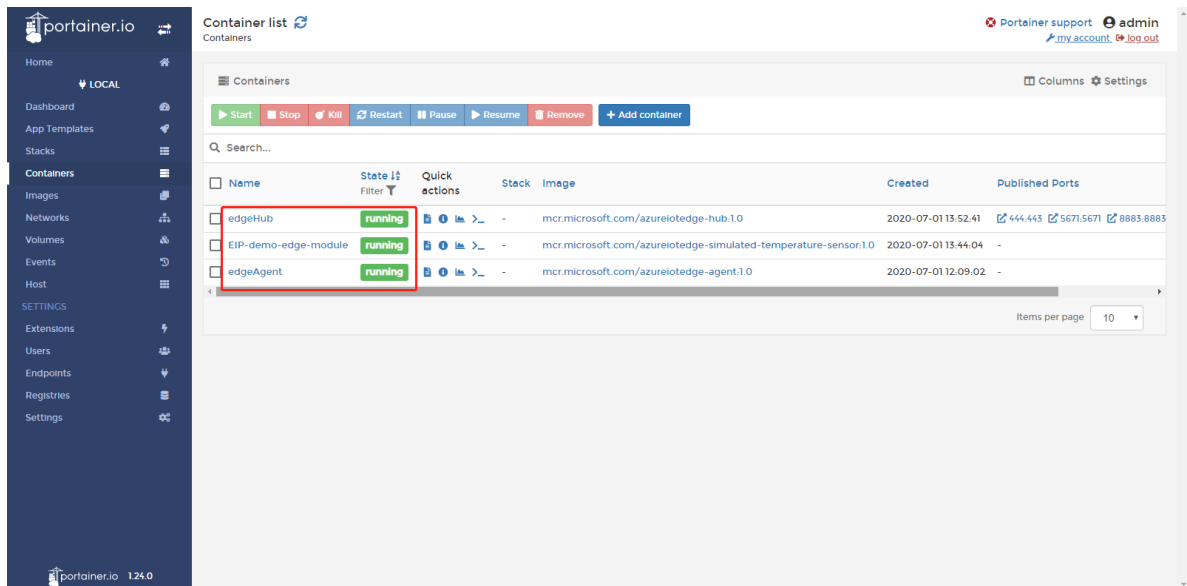
IoT Edge Runtime Response: 200 -- OK

Enable connection to IoT Hub: ☒ Enable ☐ Disable

NAME	TYPE	SPECIFIED IN DEPLOYMENT	REPORTED BY DEVICE	RUNTIME STATUS	EXIT CODE
SedgeAgent	IoT Edge System Module	✓ Yes	✓ Yes	running	0
SedgeHub	IoT Edge System Module	✓ Yes	✓ Yes	running	0
EIP-demo-edge-module	IoT Edge Custom Module	✓ Yes	✓ Yes	running	0

- Step 2: View the container running status

Access portainer and choose **LOCAL >> Containers**. It can be seen that three containers are already running.



Container list

Name	State	Quick actions	Stack	Image	Created	Published Ports
edgeHub	running	[Icons]	-	mcr.microsoft.com/azureiotedge-hub:1.0	2020-07-01 13:52:41	444.443, 5671.5671, 8883.8883
EIP-demo-edge-module	running	[Icons]	-	mcr.microsoft.com/azureiotedge-simulated-temperature-sensor:1.0	2020-07-01 13:44:04	-
edgeAgent	running	[Icons]	-	mcr.microsoft.com/azureiotedge-agent:1.0	2020-07-01 12:09:02	-

Click **Logs** of the **EIP-demo-edge-module** container to view its running logs. If the logs are as follows, the container runs normally, that is, it simulates the telemetry data and sends the data to IoT Hub.

The top screenshot shows the Portainer.io 'Container list' page. It features a sidebar with navigation options like Home, Dashboard, App Templates, Stacks, Containers, Images, Networks, Volumes, Events, Host, SETTINGS, Extensions, Users, Endpoints, Registries, and Settings. The main area displays a table of containers:

Name	State	Quick actions	Stack	Image	Created	Published Ports
edgeHub	running	[Icons]	-	mcr.microsoft.com/azureiotedge-hub:1.0	2020-07-01 13:52:41	5671:5671, 8883:8883, 4444:4443
EIP-demo-edge-module	running	[Icons]	-	mcr.microsoft.com/azureiotedge-simulated-temperature-sensor:1.0	2020-07-01 13:44:04	-
edgeAgent	running	[Icons]	-	mcr.microsoft.com/azureiotedge-agent:1.0	2020-07-01 12:09:02	-

The bottom screenshot shows the 'Logs' page for the 'EIP-demo-edge-module' container. It includes controls for 'Wrap lines', 'Display timestamps', 'Fetch' (set to 'All logs'), 'Search' (with a filter), 'Lines' (set to 100), and 'Actions' (Copy, Copy selected lines, Unselect). The log output shows the container starting and sending telemetry messages to IoT Hub:

```
cs:line 79
at Microsoft.Azure.Devices.Edge.ModuleUtil.ModuleUtil.CreateModuleClientAsync(TransportType transportType, ITransientErrorDetectionStrategy transientErrorDetectionStrategy, RetryStrate
@ retryStrategy, ILogger logger) in /home/vsts/work/1/s/edge-modules/ModuleLib/ModuleUtil.cs:line 41
at SimulatedTemperatureSensor.Program.MainAsync() in /home/vsts/work/1/s/edge-modules/SimulatedTemperatureSensor/src/Program.cs:line 68
--- End of inner exception stack trace ---
at System.Threading.Tasks.Task`1.GetResultCore(Boolean waitCompletionNotification)
at System.Threading.Tasks.Task`1.get_Result()
at SimulatedTemperatureSensor.Program.Main() in /home/vsts/work/1/s/edge-modules/SimulatedTemperatureSensor/src/Program.cs:line 37
[2020-07-01 05:55:12 +00:00]: Starting Module
SimulatedTemperatureSensor.Main() started.
Initializing simulated temperature sensor to send 500 messages, at an interval of 5 seconds.
To change this, set the environment variable MessageCount to the number of messages that should be sent (set it to -1 to send unlimited messages).
Information: Trying to initialize module client using transport type [Amqp_Tcp_Only].
Information: Successfully initialized module client of transport type [Amqp_Tcp_Only].
07/01/2020 05:59:24: Sending message: 1, Body: [{"machine":{"temperature":22.192162529538461,"pressure":1.1358159843777993,"ambient":{"temperature":21.41624249560836,"humidity":
25},"timeCreated":"2020-07-01T05:59:24.0978326Z"}}]
07/01/2020 05:59:38: Sending message: 2, Body: [{"machine":{"temperature":22.878451198981352,"pressure":1.2140007695042048,"ambient":{"temperature":21.334552988332955,"humidity":
25},"timeCreated":"2020-07-01T05:59:38.5587204Z"}}]
07/01/2020 05:59:45: Sending message: 3, Body: [{"machine":{"temperature":23.0453997517874555,"pressure":1.233019978390772,"ambient":{"temperature":21.486635411617641,"humidity":2
6},"timeCreated":"2020-07-01T05:59:45.6147837Z"}}]
```

At this time, you have deployed and run an IoT Edge module that simulates telemetry data and sends it to IoT Hub on IG902 through the Azure platform.