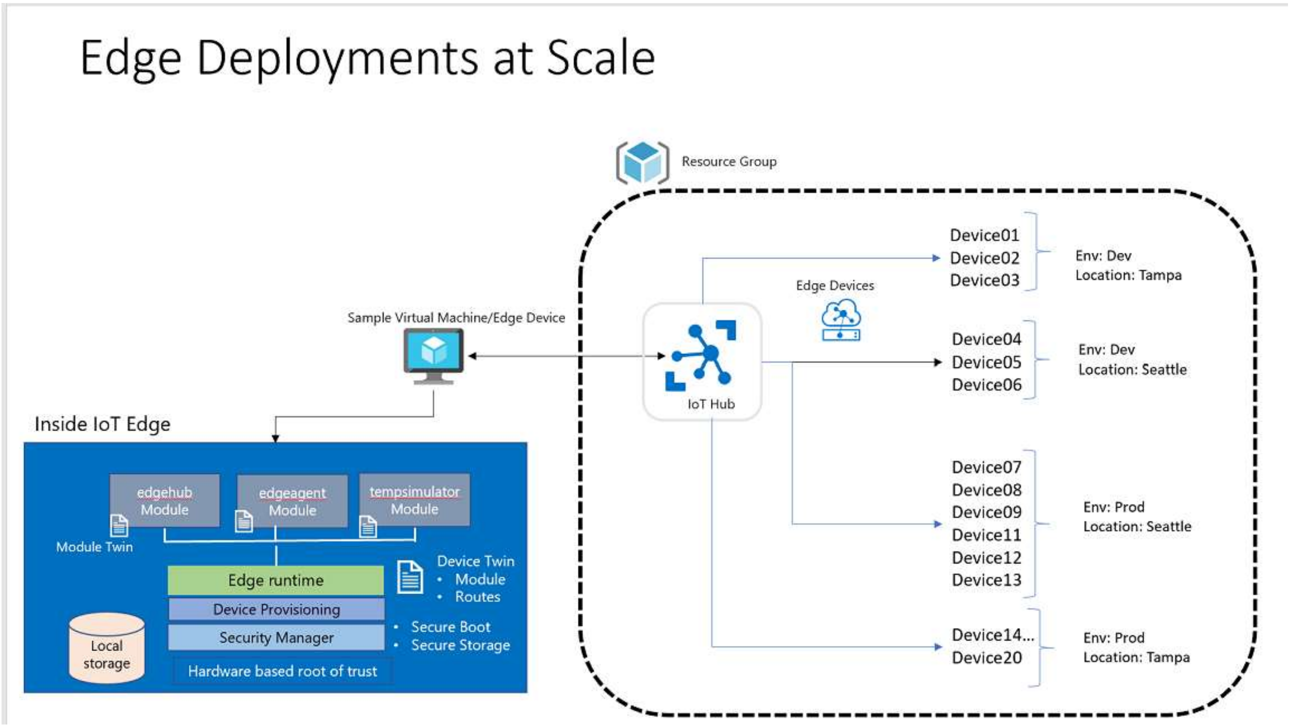
mpram Device Update ... 3 days ago History

..	
media	3 days ago
readme.md	3 days ago

readme.md

Deploy IoT Edge modules at Scale

Scenario

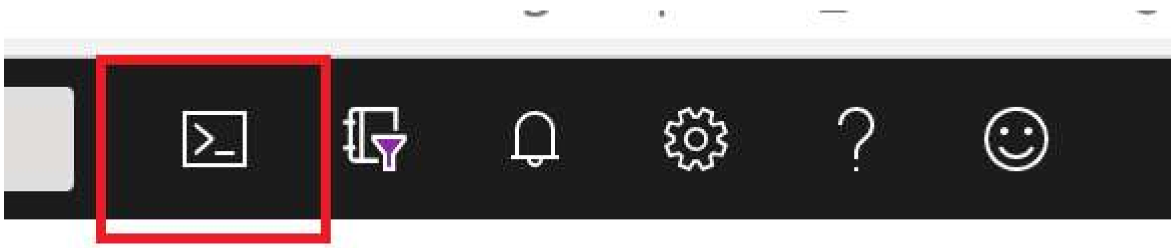


Content - Hands-on Lab:

Exercise 1: Setting up your Environment

Task 1: Create IoT Hub

First step we will create the services we need for this architecture using CLI. For that, open cloud shell, clicking in the following icon, top right in Azure Portal:



1. Run the az extension add command to add the Microsoft Azure IoT Extension for Azure CLI to your CLI shell. The IOT Extension adds IoT Hub, IoT Edge, and IoT Device Provisioning Service (DPS) specific commands to Azure CLI.

```
az extension add --name azure-iot
```

If you never used CLI before you will be prompted to mount an storage account, click **Create Storage** to continue. If you used before, you will skip this step.

Run the following commmand to create an Azure IoT Hub, make sure to replace the name of your Resource Group and assign a name to your IoT Hub similar to **iotacademySUFFIX**

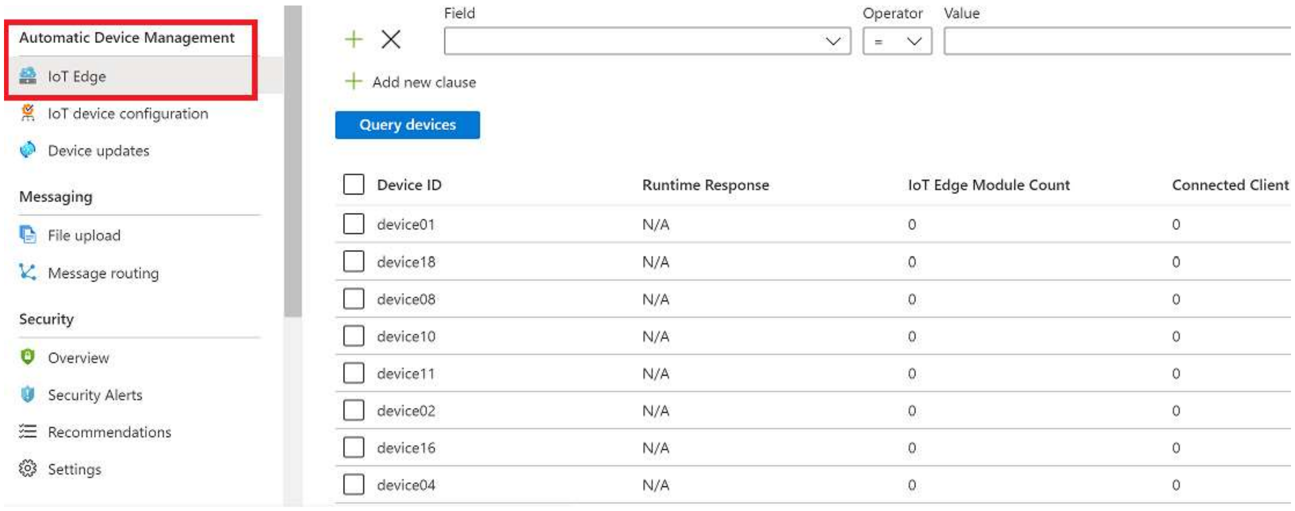
```
az iot hub create --name <YOUR_HUB_NAME_HERE> --resource-group <YOUR_RG_HERE> --sku
```

After a few minutes you should receive a provisioned states succeed message.

2. Once the resource is ready, we will use the below command to create 20 edge devices. Make sure to replace the <YOUR_HUB_NAME_HERE> with the name of your current IoT Hub before running the command:

```
az iot hub device-identity create -n <YOUR_HUB_NAME_HERE>    -d device01 --edge-enab
az iot hub device-identity create -n <YOUR_HUB_NAME_HERE>    -d device02 --edge-enab
az iot hub device-identity create -n <YOUR_HUB_NAME_HERE>    -d device03 --edge-enab
az iot hub device-identity create -n <YOUR_HUB_NAME_HERE>    -d device04 --edge-enab
az iot hub device-identity create -n <YOUR_HUB_NAME_HERE>    -d device05 --edge-enab
az iot hub device-identity create -n <YOUR_HUB_NAME_HERE>    -d device06 --edge-enab
az iot hub device-identity create -n <YOUR_HUB_NAME_HERE>    -d device07 --edge-enab
az iot hub device-identity create -n <YOUR_HUB_NAME_HERE>    -d device08 --edge-enab
az iot hub device-identity create -n <YOUR_HUB_NAME_HERE>    -d device09 --edge-enab
az iot hub device-identity create -n <YOUR_HUB_NAME_HERE>    -d device10 --edge-enab
az iot hub device-identity create -n <YOUR_HUB_NAME_HERE>    -d device11 --edge-enab
az iot hub device-identity create -n <YOUR_HUB_NAME_HERE>    -d device12 --edge-enab
az iot hub device-identity create -n <YOUR_HUB_NAME_HERE>    -d device13 --edge-enab
az iot hub device-identity create -n <YOUR_HUB_NAME_HERE>    -d device14 --edge-enab
az iot hub device-identity create -n <YOUR_HUB_NAME_HERE>    -d device15 --edge-enab
az iot hub device-identity create -n <YOUR_HUB_NAME_HERE>    -d device16 --edge-enab
az iot hub device-identity create -n <YOUR_HUB_NAME_HERE>    -d device17 --edge-enab
az iot hub device-identity create -n <YOUR_HUB_NAME_HERE>    -d device18 --edge-enab
az iot hub device-identity create -n <YOUR_HUB_NAME_HERE>    -d device19 --edge-enab
az iot hub device-identity create -n <YOUR_HUB_NAME_HERE>    -d device20 --edge-enab
```

3. After a few minutes you should see all the new devices in your Azure IoT Hub:

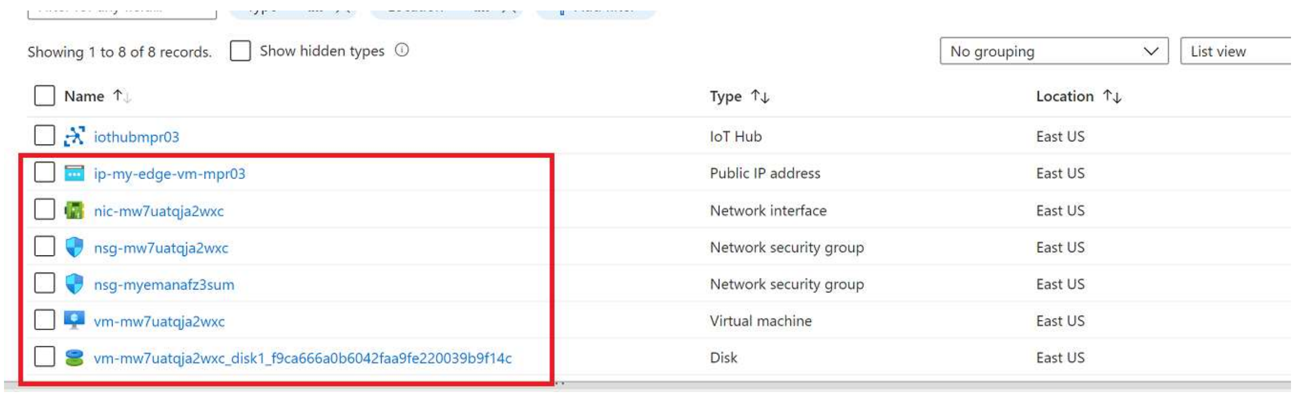


Device ID	Runtime Response	IoT Edge Module Count	Connected Client
device01	N/A	0	0
device18	N/A	0	0
device08	N/A	0	0
device10	N/A	0	0
device11	N/A	0	0
device02	N/A	0	0
device16	N/A	0	0
device04	N/A	0	0

4. In the next step we will create a virtual machine as an edge device, before running the script makesure to replace your **Resource Group**, your **IoT Hub** and assign a suffix to the dns **my-edge-vm-SUFFIX**

```
az deployment group create \
--resource-group <YOUR_RG_HERE> \
--template-uri "https://aka.ms/iotedge-vm-deploy" \
--parameters dnsLabelPrefix='my-edge-vm-SUFFIX' \
 --parameters adminUsername='academyuser'  \
--parameters deviceConnectionString=$(az iot hub device-identity connection-string s
--parameters authenticationType='password' \
 --parameters adminPasswordOrKey="IoTAcademy01!"
```

After a few minutes you should see your VM provisioned in the portal.



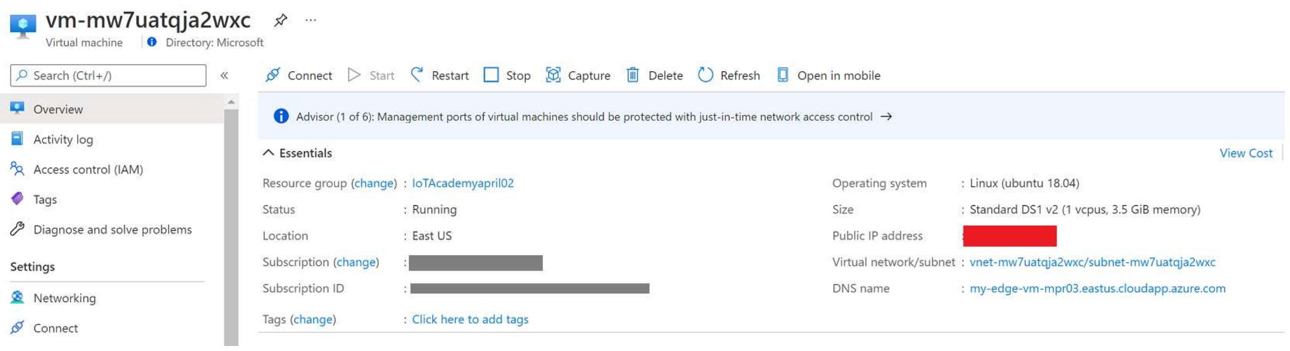
Name	Type	Location
iothubmpr03	IoT Hub	East US
ip-my-edge-vm-mpr03	Public IP address	East US
nic-mw7uatqja2wxc	Network interface	East US
nsg-mw7uatqja2wxc	Network security group	East US
nsg-myemanafz3sum	Network security group	East US
vm-mw7uatqja2wxc	Virtual machine	East US
vm-mw7uatqja2wxc_disk1_f9ca666a0b6042faa9fe220039b9f14c	Disk	East US

Task 2: Connect Virtual Machine to IoT Hub

In this step we will connect the virtual machine just created in previous step to Azure IoT Hub, assigning the device01.

1. Launch putty locally, copy the IP of the virtual machine in the overview tab

You will find the public IP for your VM as shown below:



Advisor (1 of 6): Management ports of virtual machines should be protected with just-in-time network access control →

Essentials

Resource group (change)	: IoTAcademyapril02	Operating system	: Linux (ubuntu 18.04)
Status	: Running	Size	: Standard DS1 v2 (1 vcpus, 3.5 GiB memory)
Location	: East US	Public IP address	: [redacted]
Subscription (change)	: [redacted]	Virtual network/subnet	: vnet-mw7uatqja2wxc/subnet-mw7uatqja2wxc
Subscription ID	: [redacted]	DNS name	: my-edge-vm-mpr03.eastus.cloudapp.azure.com
Tags (change)	: Click here to add tags		

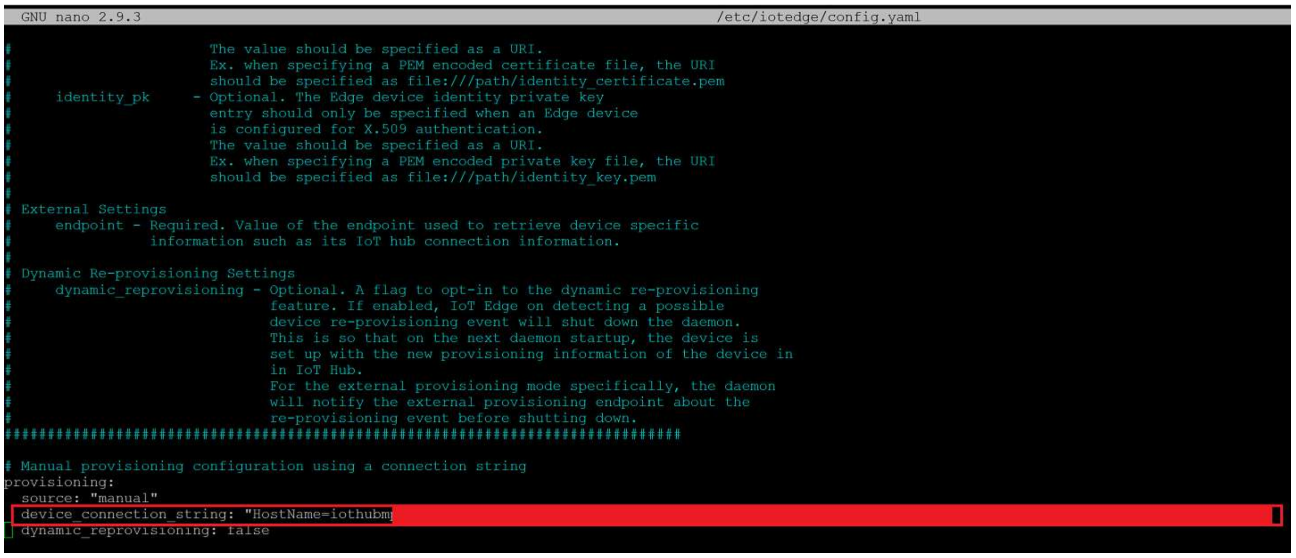
Copy and paste the IP in the **Host Name** section in Putty, then click **Yes** to continue. Now you will see the **Login** steps, use the credentials from the script creation you applied in previous step.

2. Once login run the following command to edit the connection string to your device

```
sudo nano /etc/iotedge/config.yaml
```

Note: Validate if you already have the connection string in your config.yaml, if you run succesfully the vm creation script, the device connection string should be assigned already, if you dont see it, then follow the next steps; otherwise skip it and move to restart your edge deamon.

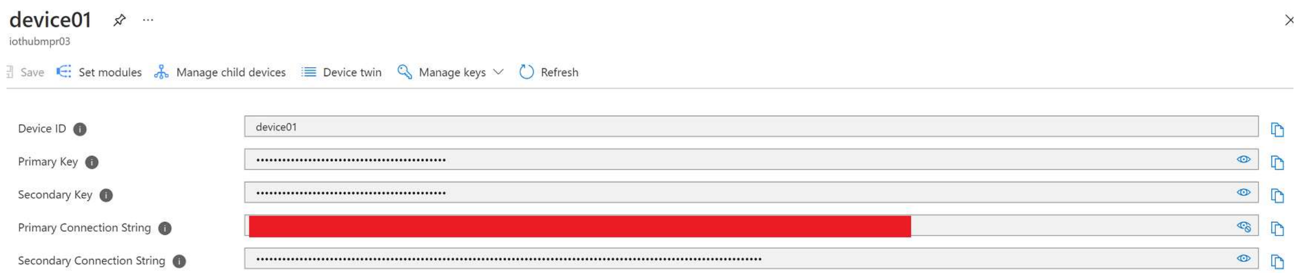
Now you can replace the connection string from Azure portal to the config.yaml file in your device:



```
  GNU nano 2.9.3                                                    /etc/iotedge/config.yaml
#                       The value should be specified as a URI.
#                       Ex. when specifying a PEM encoded certificate file, the URI
#                       should be specified as file:///path/identity_certificate.pem
#     identity_pk     - Optional. The Edge device identity private key
#                       entry should only be specified when an Edge device
#                       is configured for X.509 authentication.
#                       The value should be specified as a URI.
#                       Ex. when specifying a PEM encoded private key file, the URI
#                       should be specified as file:///path/identity_key.pem
#
# External Settings
#     endpoint - Required. Value of the endpoint used to retrieve device specific
#                information such as its IoT hub connection information.
#
# Dynamic Re-provisioning Settings
#     dynamic_reprovisioning - Optional. A flag to opt-in to the dynamic re-provisioning
#                              feature. If enabled, IoT Edge on detecting a possible
#                              device re-provisioning event will shut down the daemon.
#                              This is so that on the next daemon startup, the device is
#                              set up with the new provisioning information of the device in
#                              in IoT Hub.
#                              For the external provisioning mode specifically, the daemon
#                              will notify the external provisioning endpoint about the
#                              re-provisioning event before shutting down.
###############################################################################

# Manual provisioning configuration using a connection string
provisioning:
  source: "manual"
  device_connection_string: "HostName=iothubm[redacted]
  dynamic_reprovisioning: false
```

The connection string to paste in the section above you will find it in Azure IoT Hub, **Automatic Device Management** section, **IoT Edge**, click in **device01**, copy **Primary Connection String**


device01
iothubmpr03

Device ID	device01
Primary Key	•••
Secondary Key	•••
Primary Connection String	[redacted]
Secondary Connection String	••

After you replace the connection string, **Crtl+X** to save the changes and **Y** to confirm

Restart your edge deamon with the below command:

```
sudo systemctl restart iotedge
```

After a few minutes you should be able to see the edgeAgent container running in your Virtual machine executing the following command:

```
sudo iotedge list
```

Exercise 2: Assigning Tags

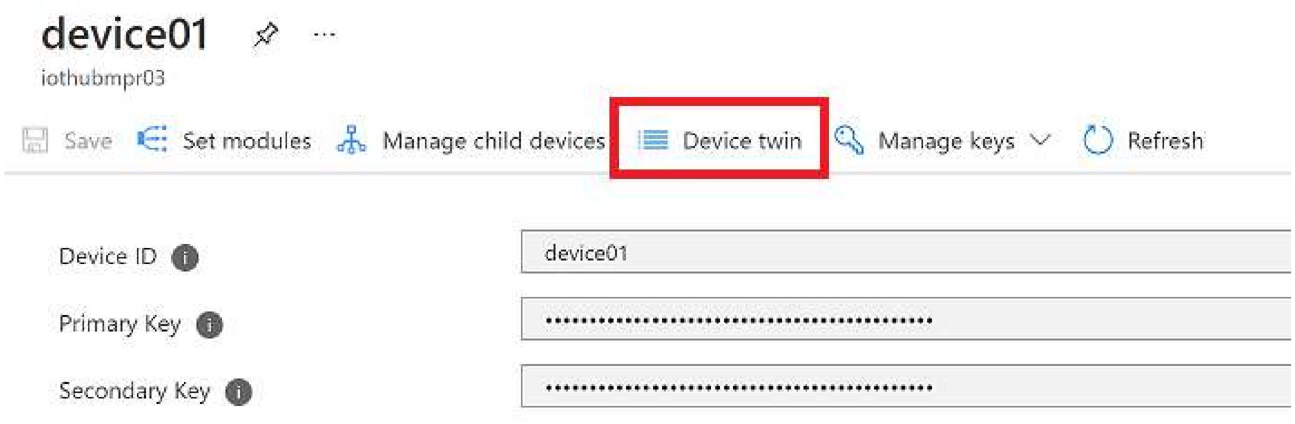
In this exercise you will learn how to assign tags to your devices using different tools. According to our architecture we will assign Tags based on the following distribution:

- Devices: 1-3 Env: Dev, Location: Tampa
- Devices: 4-6 Env: Dev, Location: Seattle
- Devices: 7-13 Env: Prod, Location: Seattle
- Devices: 14-20 Env: Prod, Location: Tampa

Task 1: Device Twins

In this first task you will assign task using the Azure Portal, modifiying the device twin of the edge device.

1. Go to Azure IoT Hub, **Automatic Device Management**, then **IoT Edge** select **device01**:



Device ID: device01

Primary Key: ••

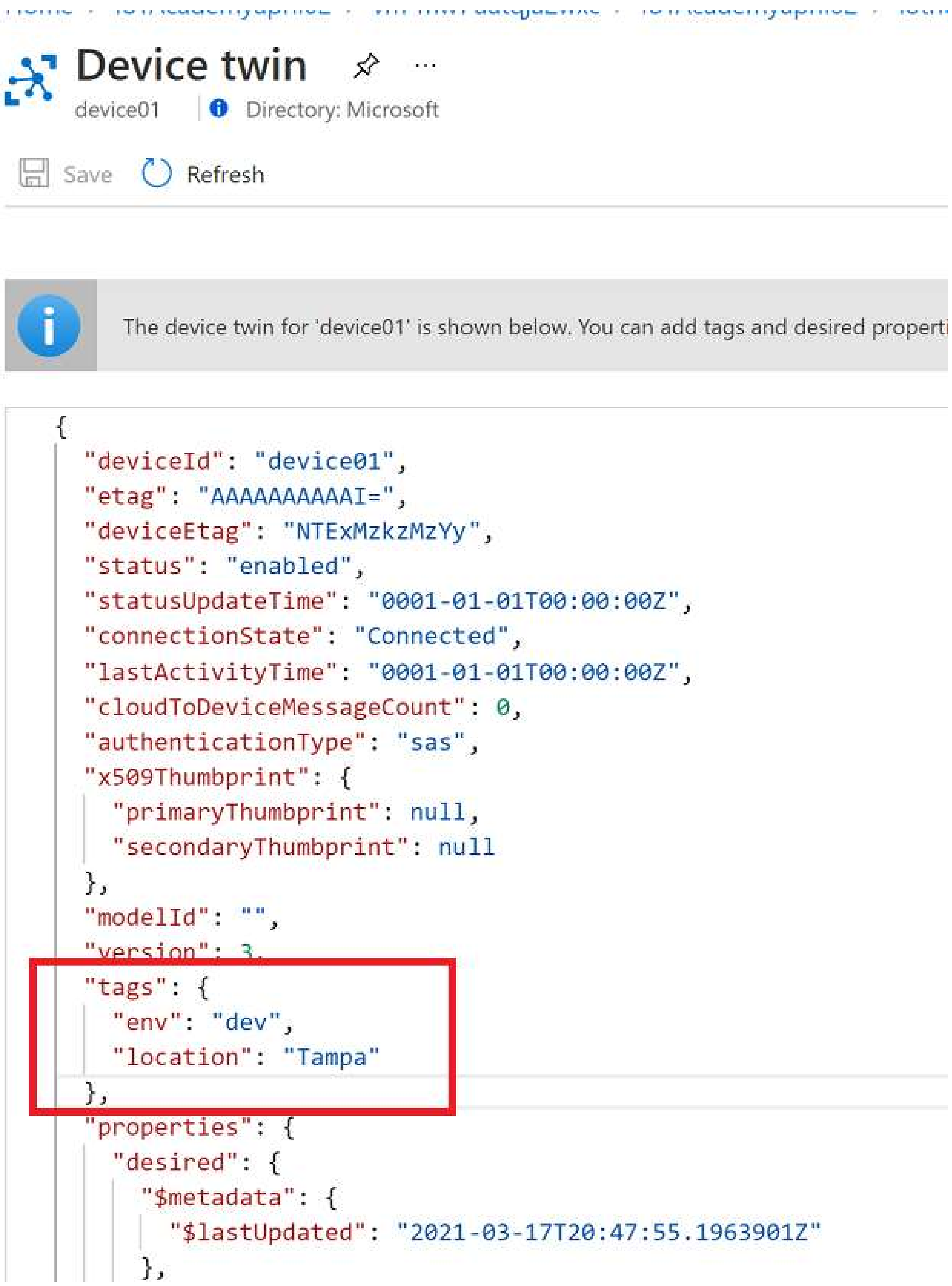
Secondary Key: ••

Once you open the devie twin you can add your tags, copy and paste the following json right below **version** section:

```
"tags": {
  "env": "dev",
  "location": "Tampa"
},
```

Then click **Save** on the top

Your Device Twin should look like the below screen:



The device twin for 'device01' is shown below. You can add tags and desired properti

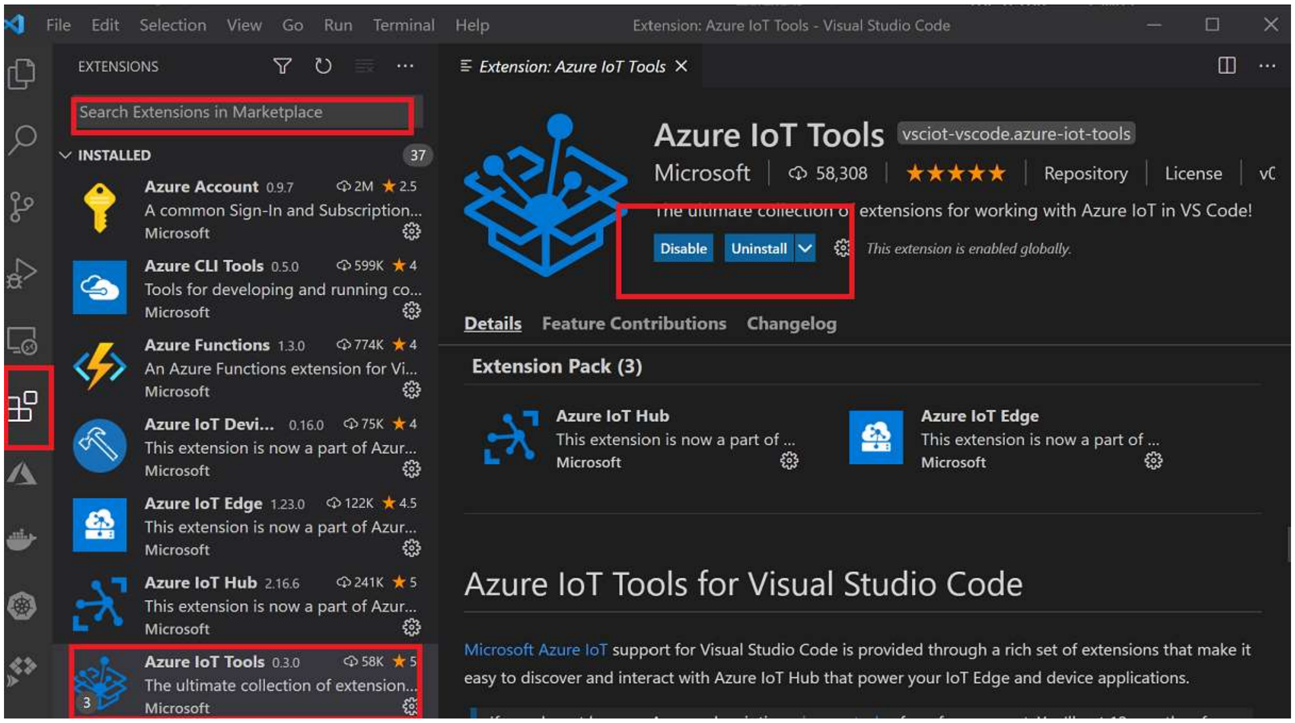
```
{
  "deviceId": "device01",
  "etag": "AAAAAAAAAAI=",
  "deviceEtag": "NTExMzkzMzYy",
  "status": "enabled",
  "statusUpdateTime": "0001-01-01T00:00:00Z",
  "connectionState": "Connected",
  "lastActivityTime": "0001-01-01T00:00:00Z",
  "cloudToDeviceMessageCount": 0,
  "authenticationType": "sas",
  "x509Thumbprint": {
    "primaryThumbprint": null,
    "secondaryThumbprint": null
  },
  "modelId": "",
  "version": 3,
  "tags": {
    "env": "dev",
    "location": "Tampa"
  },
  "properties": {
    "desired": {
      "$metadata": {
        "$lastUpdated": "2021-03-17T20:47:55.1963901Z"
      },
```

Task 2: Visual Studio Code

Next we will assing tags using Visual Studio Code.

1. Launch Visual Studio Code. If it is your fisrt time using VS Code with Azure IoT hub, you will need to follow the next step:

Click in the left menu **Extensions**, in search box type **Azure IoT Tools**, once you select the extension, you will have an **install** buttom on the right screen, click on it to start the installation

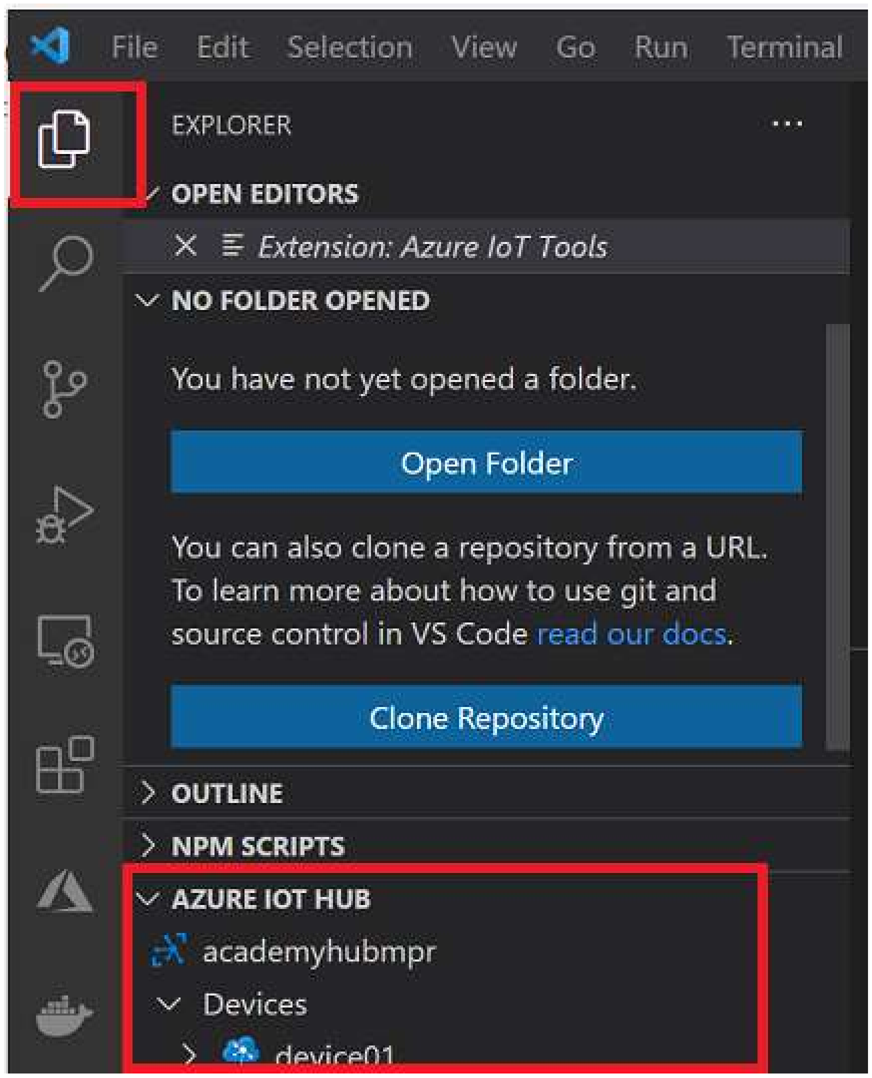

File Edit Selection View Go Run Terminal Help
Extension: Azure IoT Tools - Visual Studio Code
EXTENSIONS
Search Extensions in Marketplace
INSTALLED 37
Azure Account 0.9.7 2M 2.5
A common Sign-In and Subscription...
Microsoft
Azure CLI Tools 0.5.0 599K 4
Tools for developing and running co...
Microsoft
Azure Functions 1.3.0 774K 4
An Azure Functions extension for Vi...
Microsoft
Azure IoT Devi... 0.16.0 75K 4
This extension is now a part of Azur...
Microsoft
Azure IoT Edge 1.23.0 122K 4.5
This extension is now a part of Azur...
Microsoft
Azure IoT Hub 2.16.6 241K 5
This extension is now a part of Azur...
Microsoft
Azure IoT Tools 0.3.0 58K 5
The ultimate collection of extension...
Microsoft
Extension: Azure IoT Tools
Azure IoT Tools vsciot-vscode.azure-iot-tools
Microsoft | 58,308 | ★★★★★ | Repository | License
The ultimate collection of extensions for working with Azure IoT in VS Code!
Disable Uninstall
This extension is enabled globally.
Details Feature Contributions Changelog
Extension Pack (3)
Azure IoT Hub
This extension is now a part of ...
Microsoft
Azure IoT Edge
This extension is now a part of ...
Microsoft

Azure IoT Tools for Visual Studio Code

Microsoft Azure IoT support for Visual Studio Code is provided through a rich set of extensions that make it easy to discover and interact with Azure IoT Hub that power your IoT Edge and device applications.

2. After this step Go to the **View** menu select **Command Palette** in the search box type **Azure IoT Hub: Select IoT Hub** follow the steps to select Subscription, Resource group and finally the IoT Hub created for this training.

3. In the left menu select **Files** you should see at the bottom a new section for **Azure IoT Hub**, expanding this section you should see your IoT Hub and all the devices.



You have not yet opened a folder.

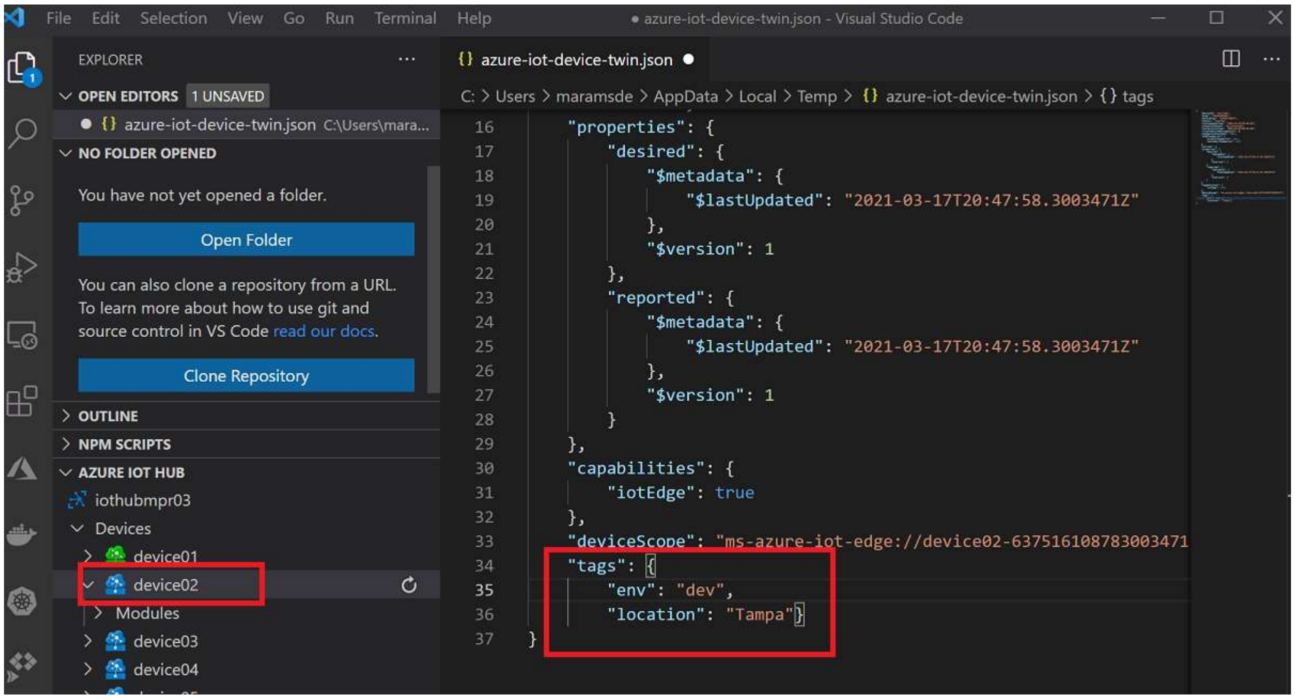
You can also clone a repository from a URL. To learn more about how to use git and source control in VS Code read our docs.

4. Right click on **device02** select **Edit Device Twin**

5. At the bottom of the new file, you will see a tags section, add there the following json

```
"tags": {
  "env": "dev",
  "location": "Tampa"
},
```

Your new file should look like the below screen:



File Edit Selection View Go Run Terminal Help
● azure-iot-device-twin.json - Visual Studio Code
EXPLORER
OPEN EDITORS 1 UNSAVED
● {} azure-iot-device-twin.json C:\Users\mara...
NO FOLDER OPENED
You have not yet opened a folder.
Open Folder
You can also clone a repository from a URL. To learn more about how to use git and source control in VS Code read our docs.
Clone Repository
OUTLINE
NPM SCRIPTS
AZURE IOT HUB
iothubmpr03
Devices
device01
device02
Modules
device03
device04
{} azure-iot-device-twin.json ●
C: > Users > maramsde > AppData > Local > Temp > {} azure-iot-device-twin.json > {} tags

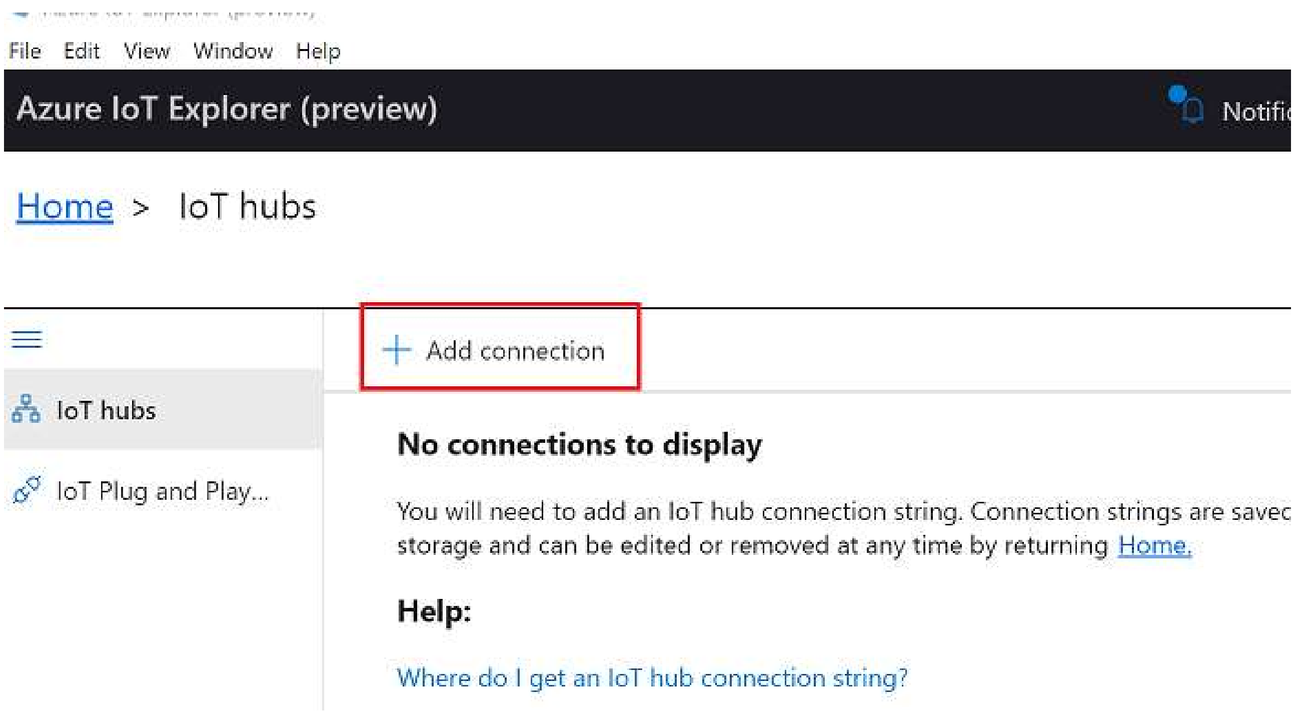
```
    "properties": {
        "desired": {
            "$metadata": {
                "$lastUpdated": "2021-03-17T20:47:58.3003471Z"
            },
            "$version": 1
        },
        "reported": {
            "$metadata": {
                "$lastUpdated": "2021-03-17T20:47:58.3003471Z"
            },
            "$version": 1
        }
    },
    "capabilities": {
        "iotEdge": true
    },
    "deviceScope": "ms-azure-iot-edge://device02-637516108783003471
    "tags": {
        "env": "dev",
        "location": "Tampa"}
}
```

6. Save your new file, **Ctrl+S**. After saving, right click any area of the new file and select **Update Device Twin**.

7. You should receive a message in the terminal specifiying **Device Twin updated successfully**. You can validate the changes in Azure Portal accessing the device twin of your device.

Task 3: IoT Hub Explorer

1. Launch Azure IoT explorer locally. Add a new connection



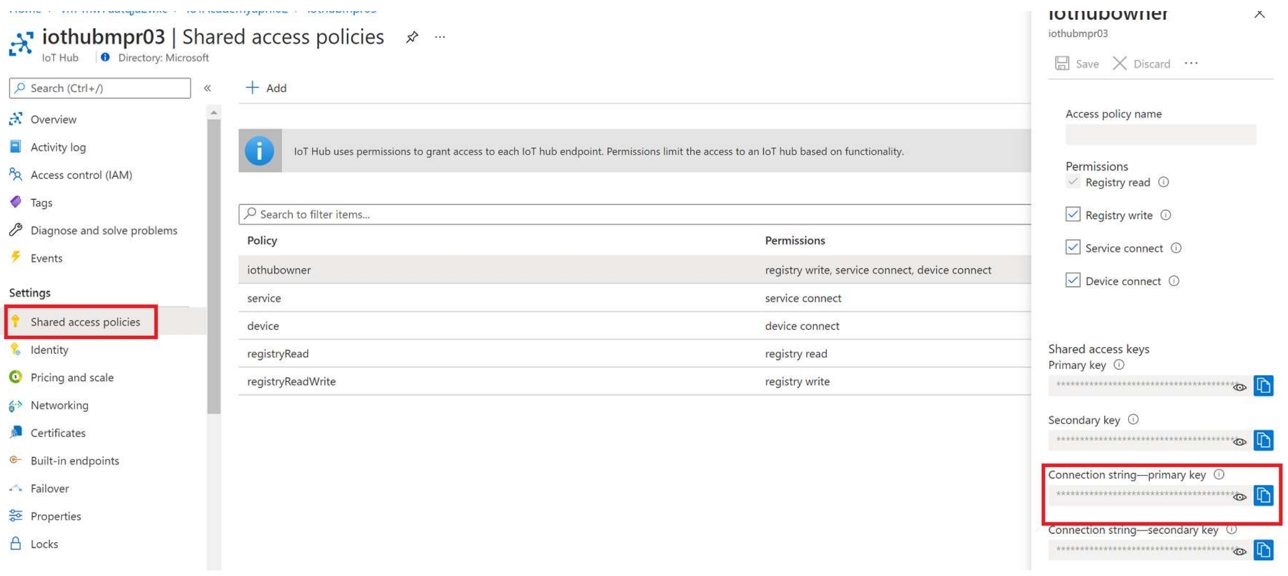
No connections to display

You will need to add an IoT hub connection string. Connection strings are savec storage and can be edited or removed at any time by returning Home.

Help:

Where do I get an IoT hub connection string?

2. In the new screen paste the IoT Hub **Primary connection string** you will find it in the **Shared access policies** section.



iothubmpr03 | Shared access policies

IoT Hub | Directory: Microsoft

Search (Ctrl+/)

Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Events

Settings

Shared access policies
Identity
Pricing and scale
Networking
Certificates
Built-in endpoints
Failover
Properties
Locks

+ Add

IoT Hub uses permissions to grant access to each IoT hub endpoint. Permissions limit the access to an IoT hub based on functionality.

Search to filter items...

Policy	Permissions
iothubowner	registry write, service connect, device connect
service	service connect
device	device connect
registryRead	registry read
registryReadWrite	registry write

iothubowner

iothubmpr03

Save Discard ···

Access policy name

Permissions

- Registry read
- [✓] Registry write
- [✓] Service connect
- [✓] Device connect

Shared access keys

Primary key

Secondary key

Connection string—primary key

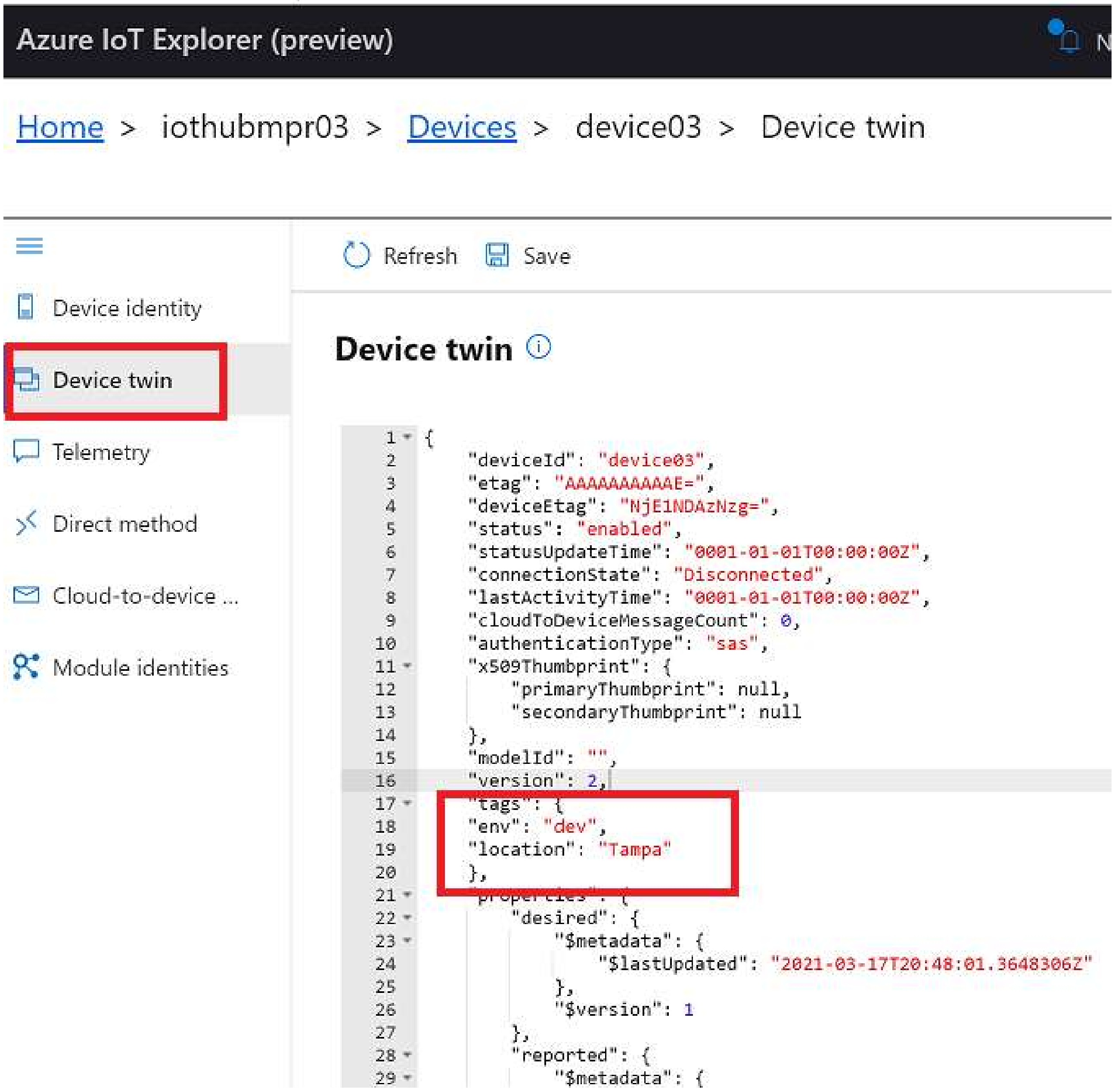
Connection string—secondary key

3. After pasting the connection string, click **Save**, next you will see the list of all your devices. Select **device03**

4. Select **Device Twin** and assign the new tags after the version section, then click **Save**

```
"tags": {
  "env": "dev",
  "location": "Tampa"
}
```

Your new twin should look like the below image:

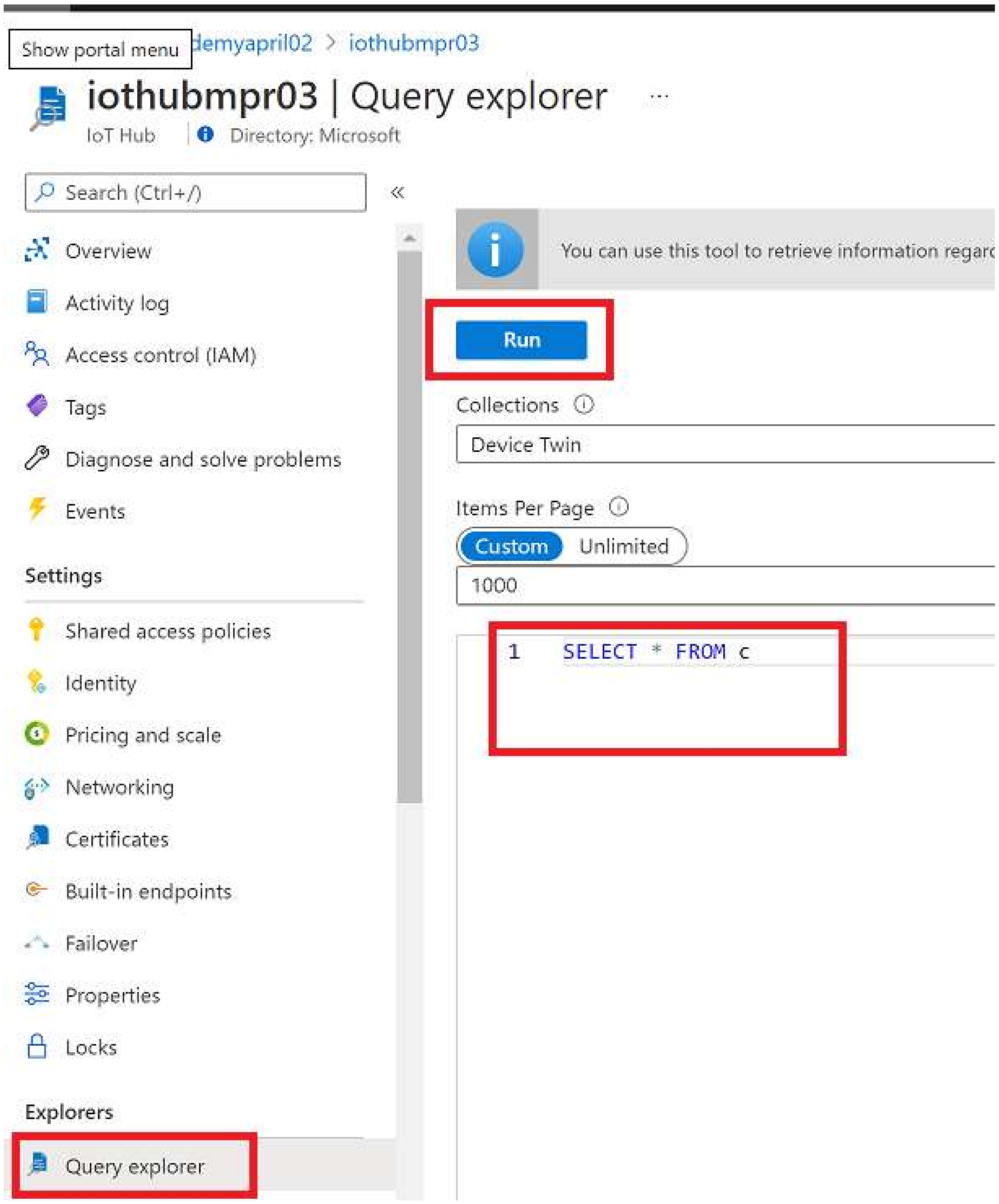


```
{
    "deviceId": "device03",
    "etag": "AAAAAAAAAAE=",
    "deviceEtag": "NjE1NDAzNzg=",
    "status": "enabled",
    "statusUpdateTime": "0001-01-01T00:00:00Z",
    "connectionState": "Disconnected",
    "lastActivityTime": "0001-01-01T00:00:00Z",
    "cloudToDeviceMessageCount": 0,
    "authenticationType": "sas",
    "x509Thumbprint": {
        "primaryThumbprint": null,
        "secondaryThumbprint": null
    },
    "modelId": "",
    "version": 2,
    "tags": {
    "env": "dev",
    "location": "Tampa"
    },
    "properties": {
        "desired": {
            "$metadata": {
                "$lastUpdated": "2021-03-17T20:48:01.3648306Z"
            },
            "$version": 1
        },
        "reported": {
            "$metadata": {
```

5. Now that you know some of the tools available complete the rest of the tags assignment for the devices with your prefer tool.

Exercise 3: Querying Devices

1. To query your devices, you can use the **Query explorer**. Go to your IoT Hub, **Explorers**, then click in **Query Explorer** now you should be able to write your queries on the right side panel.



You can use this tool to retrieve information regar

Collections: Device Twin

Items Per Page: Custom, 1000

```
SELECT * FROM c
```

2. Try different queries to identify devices by location or environment or both

```
SELECT * FROM devices where tags.location='Tampa'
```

```
SELECT * FROM devices where tags.location='Tampa' and tags.env='prod'
```

3. You can run queries based on grouping functions accorting to properties or status jobs.

```
SELECT properties.reported.telemetryConfig.status AS status,
    COUNT() AS numberOfDevices
  FROM devices
  GROUP BY properties.reported.telemetryConfig.status
```

Exercise 4: Creating a deployment at scale

Task 1: IoT Edge New Solution template

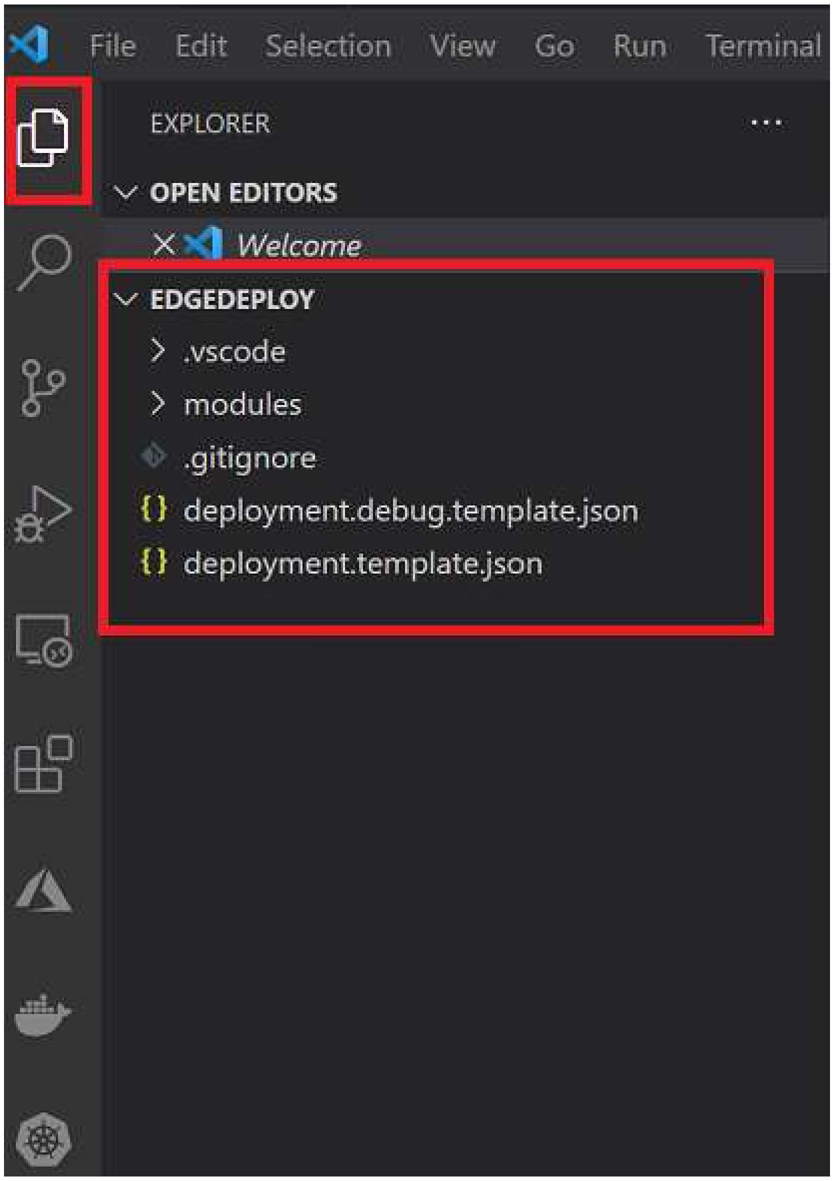
1. Go to Visual Studio Code, make sure you can see your IoT Hub at the bottom left of the screen, then click on **View** menu on the top bar, then **Command Pallette** then search for **Azure IoT Edge: New IoT Edge Solution**

2. Next step Create a folder for your deployment go to your C:\ drive, create a folder **iotedgedeploy**, select the folder just created.

3. Then the command paletter will ask you additional parameters for your solution

- Name your Solution: **edgedeploy**
- Select Language: **C# Module**
- Module name: **securitymodule**
- Docker image repository: Accept default option

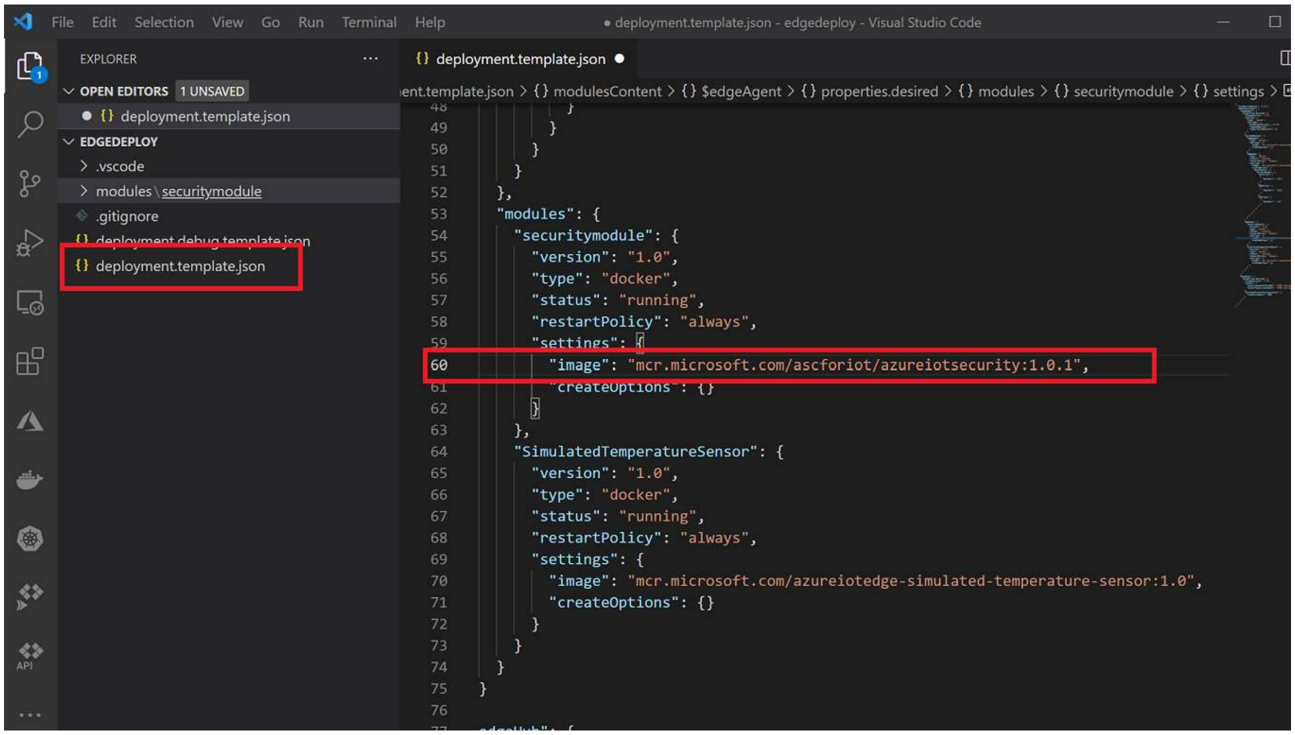
4. After a few minutes you will see a new window with the files ready for your solution as shown below:



5. Select the file **deployment.template.json** and replace the image URI for your deployment, use the following URI:

```
"image": "mcr.microsoft.com/ascforiot/azureiotsecurity:1.0.1",
```

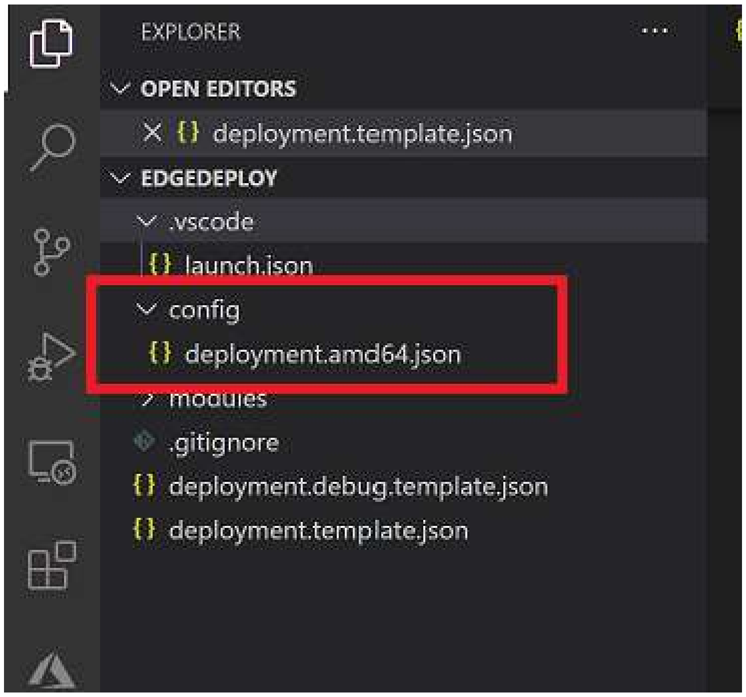
Your line 60, should look like the below screen:


File Edit Selection View Go Run Terminal Help
• deployment.template.json - edgedeploy - Visual Studio Code
EXPLORER
OPEN EDITORS 1 UNSAVED
deployment.template.json
EDGEDEPLOY
.vscode
modules \ securitymodule
.gitignore
deployment.debug.template.json
deployment.template.json
ent.template.json > {} modulesContent > {} $edgeAgent > {} properties.desired > {} modules > {} securitymodule > {} settings

```
          }
        }
      }
    },
    "modules": {
      "securitymodule": {
        "version": "1.0",
        "type": "docker",
        "status": "running",
        "restartPolicy": "always",
        "settings": {
          "image": "mcr.microsoft.com/ascforiot/azureiotsecurity:1.0.1",
          "createOptions": {}
        }
      },
      "SimulatedTemperatureSensor": {
        "version": "1.0",
        "type": "docker",
        "status": "running",
        "restartPolicy": "always",
        "settings": {
          "image": "mcr.microsoft.com/azureiotedge-simulated-temperature-sensor:1.0",
          "createOptions": {}
        }
      }
    }
  }
```

6. Save the File **Ctrl+S**

7. Right click in the **deployment.template.json** file, select **Generate IoT Edge Deployment Manifest** after generating the file, you should see the new deployment manifest in your **config** folfer



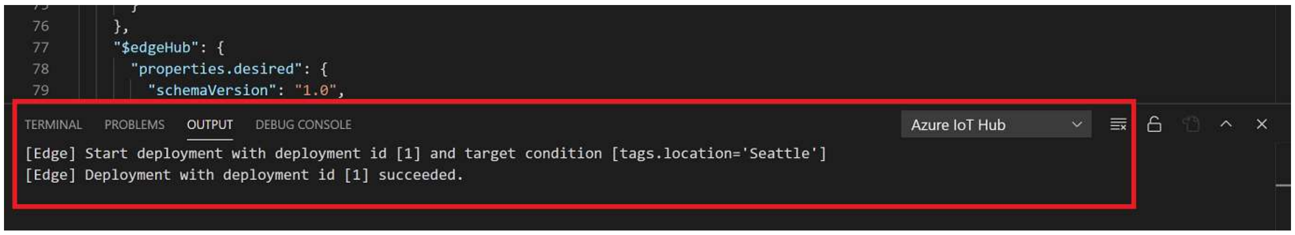
Task 2: Create deployment at Scale, VS Code

To start the deployment at scale we will select all the devices from **Seattle** to deploy our solution.

1. Go to the **View** Menu select **Command Palette** search for **Azure IoT Edge: Create Deployment at Scale**. This selection will trigger some parameters you will need to fill:

- Select the deployment manifest generated in previous step, search for it in the **config** folder.
- Enter deployment ID: **1**
- Enter a target condition: **tags.location='Seattle'**
- Select a priority for your deployment: **10**
- Enter to start deployment.

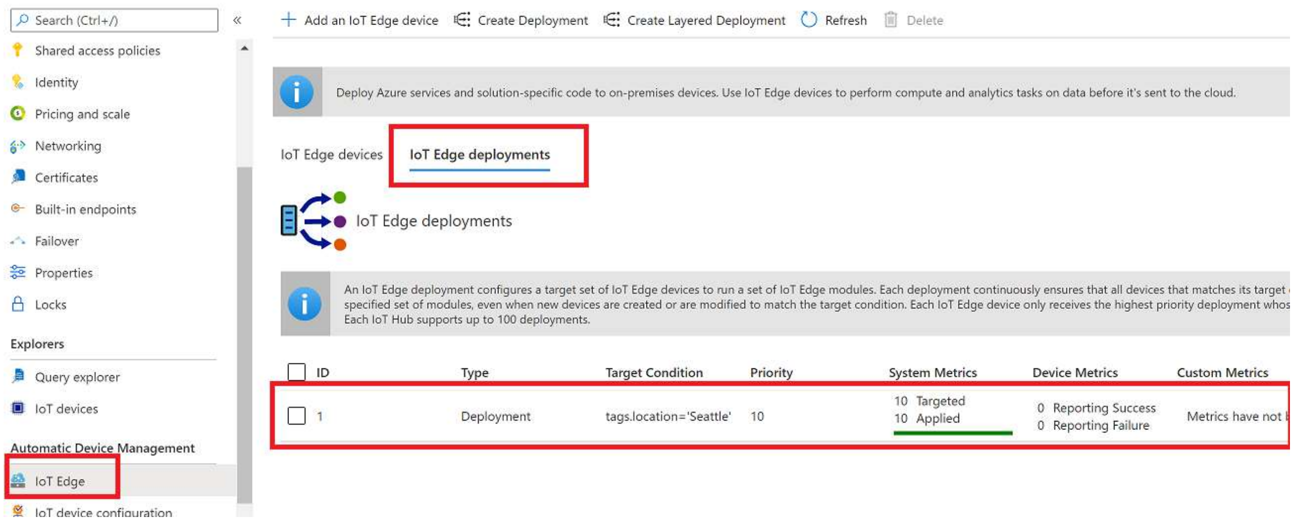
3. In a few minutes you should receive a message at the bottom of your screen with your deployment succeed.



```
    },
    "$edgeHub": {
      "properties.desired": {
        "schemaVersion": "1.0",
```

```
[Edge] Start deployment with deployment id [1] and target condition [tags.location='Seattle']
[Edge] Deployment with deployment id [1] succeeded.
```

4. Verify the status of your deployment directly in Azure Portal, open your **Azure IoT Hub**, select **IoT Edge** in **Automatic Device Management** Section, go to **IoT Edge Deployments**:



Deploy Azure services and solution-specific code to on-premises devices. Use IoT Edge devices to perform compute and analytics tasks on data before it's sent to the cloud.

IoT Edge deployments

An IoT Edge deployment configures a target set of IoT Edge devices to run a set of IoT Edge modules. Each deployment continuously ensures that all devices that matches its target
specified set of modules, even when new devices are created or are modified to match the target condition. Each IoT Edge device only receives the highest priority deployment whos
Each IoT Hub supports up to 100 deployments.

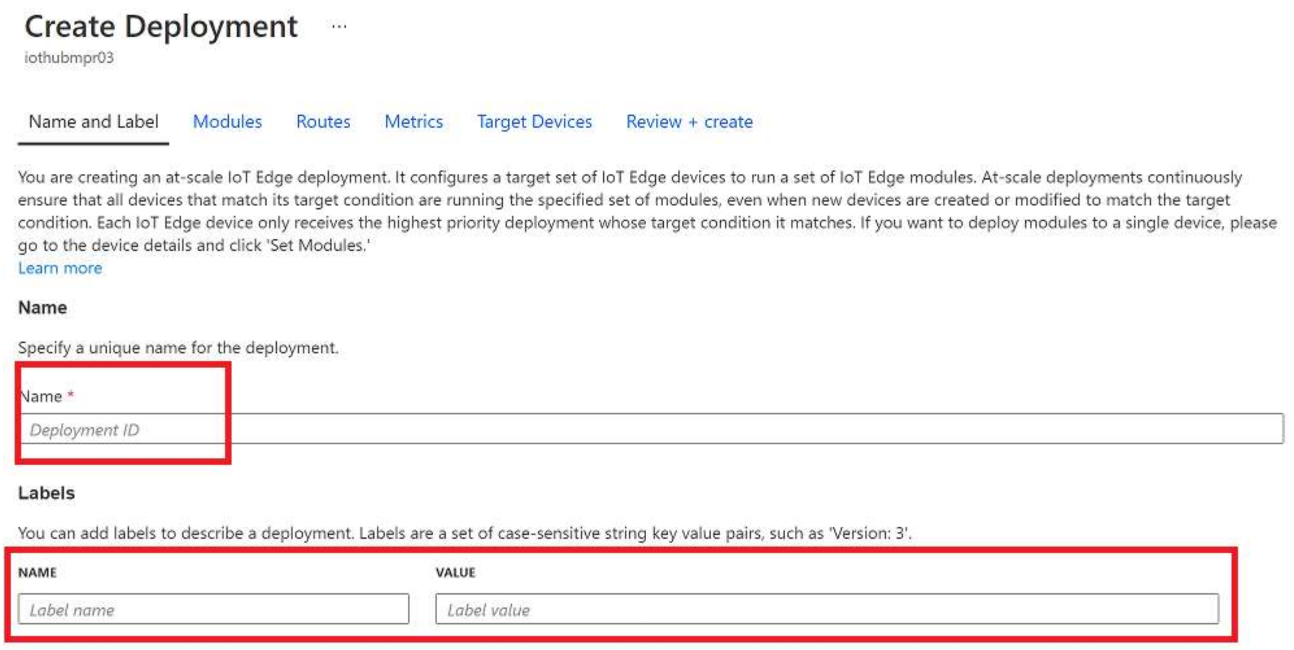
ID	Type	Target Condition	Priority	System Metrics	Device Metrics	Custom Metrics
1	Deployment	tags.location='Seattle'	10	10 Targeted 10 Applied	0 Reporting Success 0 Reporting Failure	Metrics have not

3. Validate the modules section in your Seattle devices, you should see the new modules ready for your devices.

Task 3: Create deployment at Scale, Azure Portal

Now let's make sure your Tampa devices are secure. We will create an additional deployment through the portal to deploy Azure Security Center for IoT to Tampa modules.

1. **Go to IoT Edge**, select **Create Deployments** in the new screen the fist tab will ask you to assign a deployment ID you can assign **2**


Create Deployment ...
iothubmpr03

Name and Label | Modules | Routes | Metrics | Target Devices | Review + create

You are creating an at-scale IoT Edge deployment. It configures a target set of IoT Edge devices to run a set of IoT Edge modules. At-scale deployments continuously ensure that all devices that match its target condition are running the specified set of modules, even when new devices are created or modified to match the target condition. Each IoT Edge device only receives the highest priority deployment whose target condition it matches. If you want to deploy modules to a single device, please go to the device details and click 'Set Modules.'
Learn more

Name

Specify a unique name for the deployment.

Name *
Deployment ID

Labels

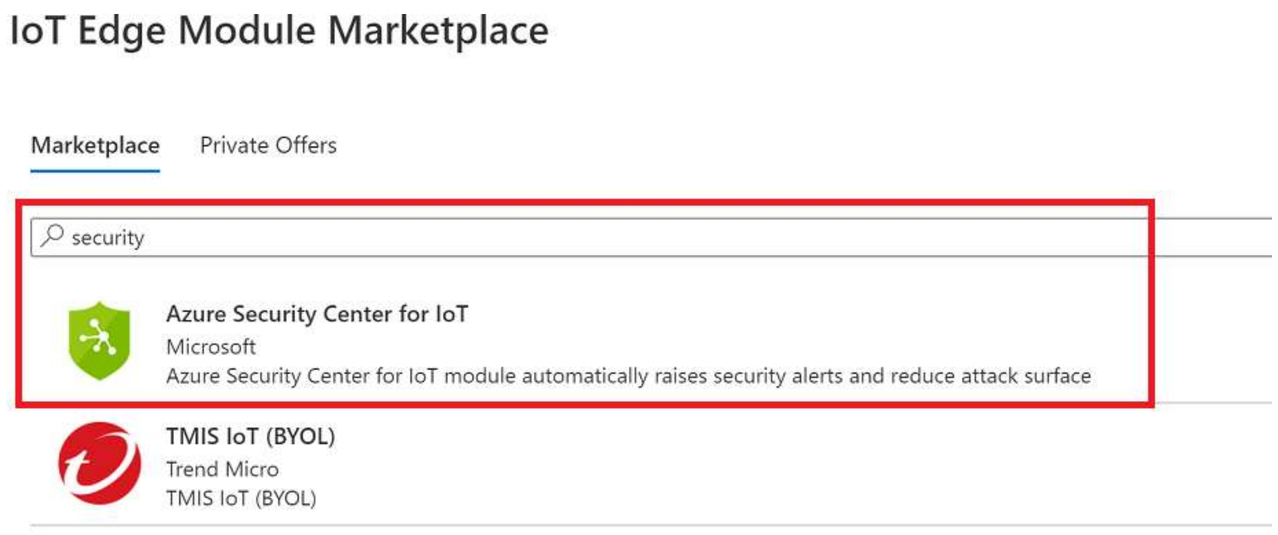
You can add labels to describe a deployment. Labels are a set of case-sensitive string key value pairs, such as 'Version: 3'.

NAME	VALUE
Label name	Label value

For Labels you can assign: Name: **Location** Value: **Tampa** Then click **Next: Modules**

2. In the Modules section: Name: **security** Address: **mcr.microsoft.com/ascforiot/azureiotsecurity:1.0.1**

3. Then go to **+Add** then select **+Marketplace Module**, search for **security** and select **Azure Security Center for IoT** as shown below:


IoT Edge Module Marketplace

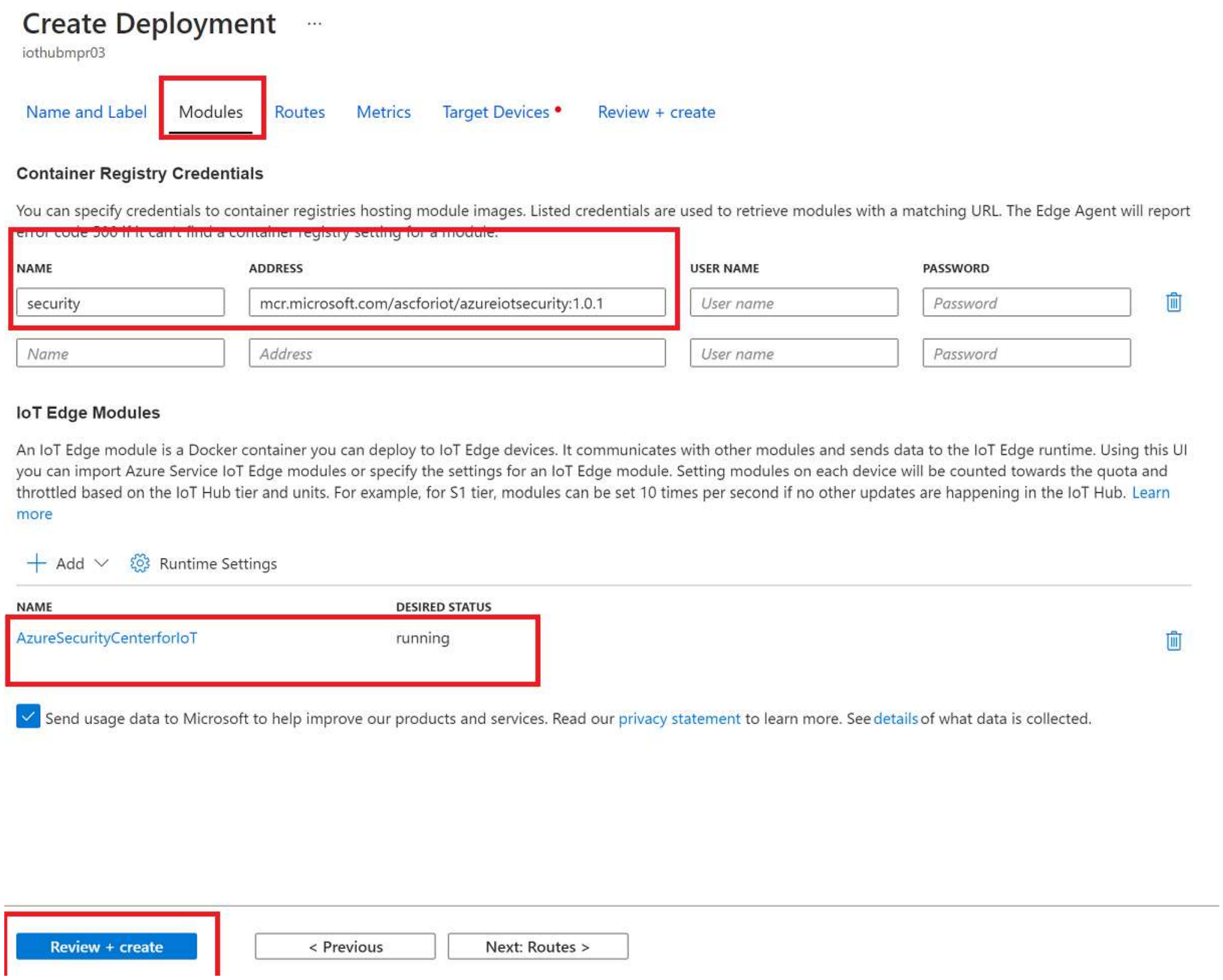
Marketplace | Private Offers

security

Azure Security Center for IoT
Microsoft
Azure Security Center for IoT module automatically raises security alerts and reduce attack surface

TMIS IoT (BYOL)
Trend Micro
TMIS IoT (BYOL)

4. Your modules tab should look like the below image:


Create Deployment ...

iothubmpr03

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
security	mcr.microsoft.com/ascforiot/azureiotsecurity:1.0.1	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

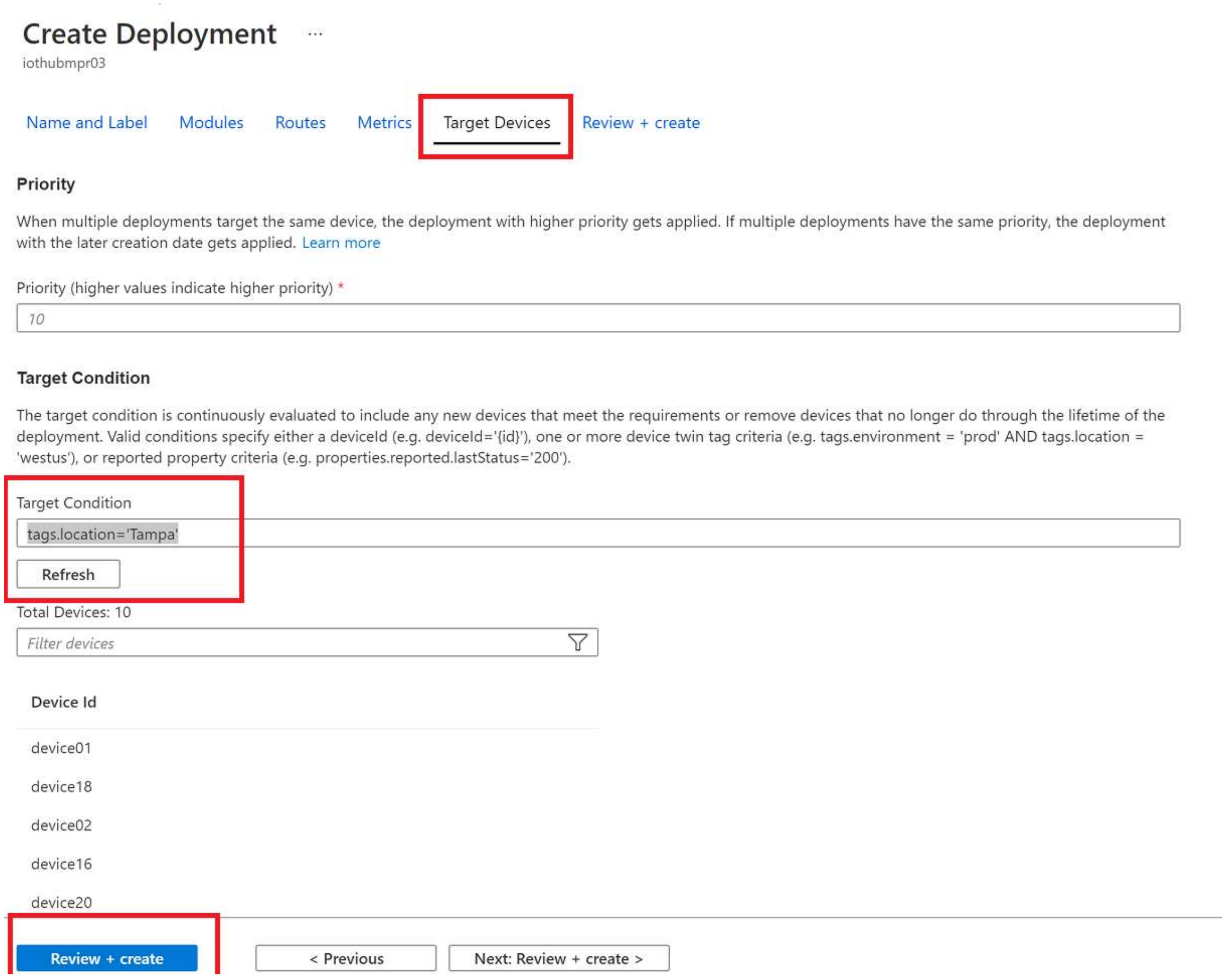
NAME	DESIRED STATUS
AzureSecurityCenterforIoT	running

✓ Send usage data to Microsoft to help improve our products and services. Read our privacy statement to learn more. See details of what data is collected.

5. Then select **Next: Routes**, leave the default option and move to **Metrics**

6. Again in metrics, leave default options and move to **Target Devices**

7. In the new tab, assign a priority for your deployment, i.e. **10**, then enter a **Target Condition** for your devices **tags.location='Tampa'** Refresh to see the devices targets based on your selection.



Create Deployment ···

iothubmpr03

Priority

When multiple deployments target the same device, the deployment with higher priority gets applied. If multiple deployments have the same priority, the deployment with the later creation date gets applied. Learn more

Priority (higher values indicate higher priority) *

10

Target Condition

The target condition is continuously evaluated to include any new devices that meet the requirements or remove devices that no longer do through the lifetime of the deployment. Valid conditions specify either a deviceId (e.g. deviceId='{id}'), one or more device twin tag criteria (e.g. tags.environment = 'prod' AND tags.location = 'westus'), or reported property criteria (e.g. properties.reported.lastStatus='200').

Target Condition

tags.location='Tampa'

Total Devices: 10

Device Id
device01
device18
device02
device16
device20

8. Click on **Review + Create**, once your validation passed and the deployment started you should see your device01 ready to receive the security module

9. At this point your deployment is ready to start, you can Monitor your deploymnet directly through **IoT Edge** then click **IoT Edge deployments**. Another way will be to check your device01 in the Module section of each device.

Exercise 5: Device Updates

1. Prepare your device

First, install the Device Update agent .deb packages in your edge device using Putty.

```
sudo apt-get install deviceupdate-agent deliveryoptimization-plugin-apt
```

Device Update for Azure IoT Hub software packages are subject to the following license terms:

Device update for IoT Hub license https://github.com/Azure/iot-hub-device-update/blob/main/LICENSE.md

Delivery optimization client license https://github.com/microsoft/do-client/blob/main/LICENSE

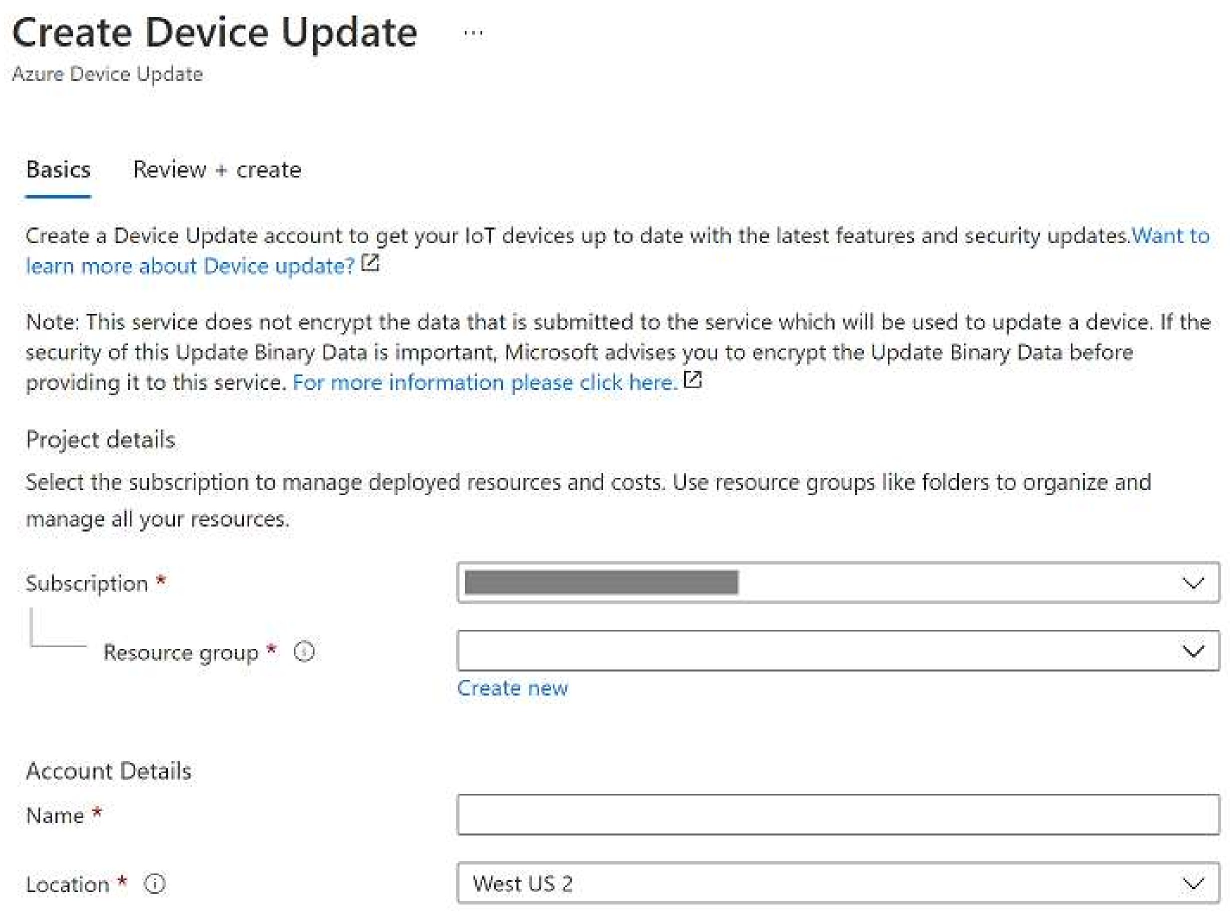
2. Create a Device update account.

Go to Azure Portal, create a new resource, in the search box type **"Device Update for IoT Hub"**

Click **Create**

Specify the **Azure Subscription** to be associated with your *Device Update Account and **Resource Group**

Specify a **Name** and **Location** for your Device Update Account. Then click **Create**



Create Device Update ...
Azure Device Update

Basics | Review + create

Create a Device Update account to get your IoT devices up to date with the latest features and security updates. Want to learn more about Device update?

Note: This service does not encrypt the data that is submitted to the service which will be used to update a device. If the security of this Update Binary Data is important, Microsoft advises you to encrypt the Update Binary Data before providing it to this service. For more information please click here.

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription *	[redacted]
Resource group *	Create new

Account Details

Name *	
Location *	West US 2

3.Create a device update instance

Once you are in your newly created account resource, go to the Instance Management **Instances** blade. Click **+ Create** and specify an instance **Name** and select your IoT Hub, the same IoT Hub you have been using during this training. Then Clikc **Create**

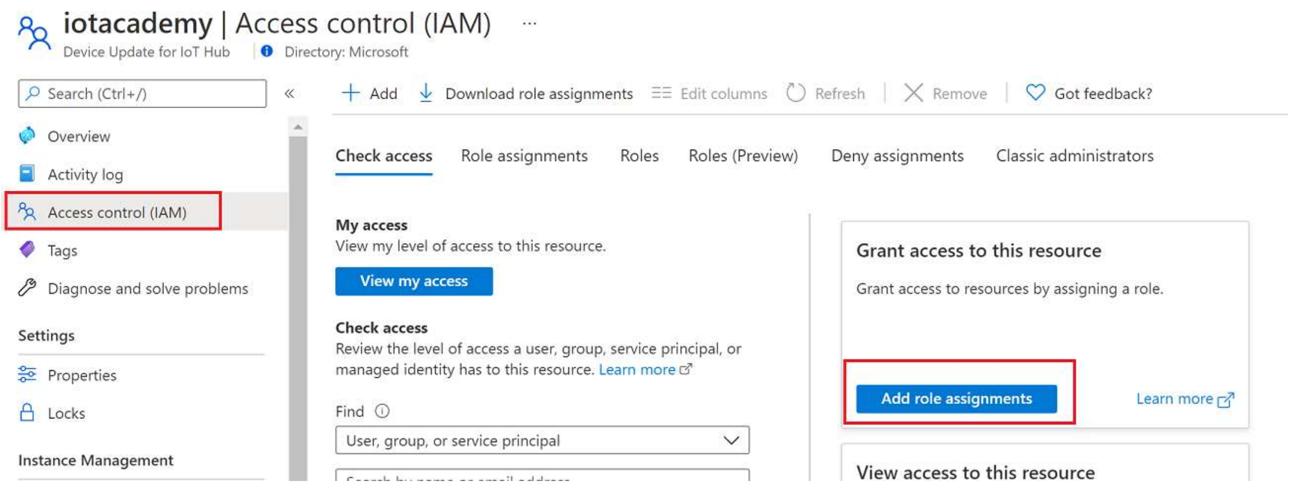
Your new instance will be in **Provisioning State: Creating** after 5-10min will change to **Succeeded**

Once the state is **Succeeded**, click on **Configure IoT Hub**. Select **I agree to make these changes** and then **Update**

4. Configure access control roles

Go to **Access control (IAM)** within the Device Update account

Click **Add role assignments**.


iotacademy | Access control (IAM)
Device Update for IoT Hub
Directory: Microsoft
Search (Ctrl+/)
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Settings
Properties
Locks
Instance Management
Add
Download role assignments
Edit columns
Refresh
Remove
Got feedback?
Check access
Role assignments
Roles
Roles (Preview)
Deny assignments
Classic administrators
My access
View my level of access to this resource.
View my access
Check access
Review the level of access a user, group, service principal, or managed identity has to this resource. Learn more
Find
User, group, or service principal
Grant access to this resource
Grant access to resources by assigning a role.
Add role assignments
Learn more
View access to this resource

Under **Select a Role**, select a Device Update role from the given options: **Device Update Administrator** and then select your name from the user list. Click **Save**

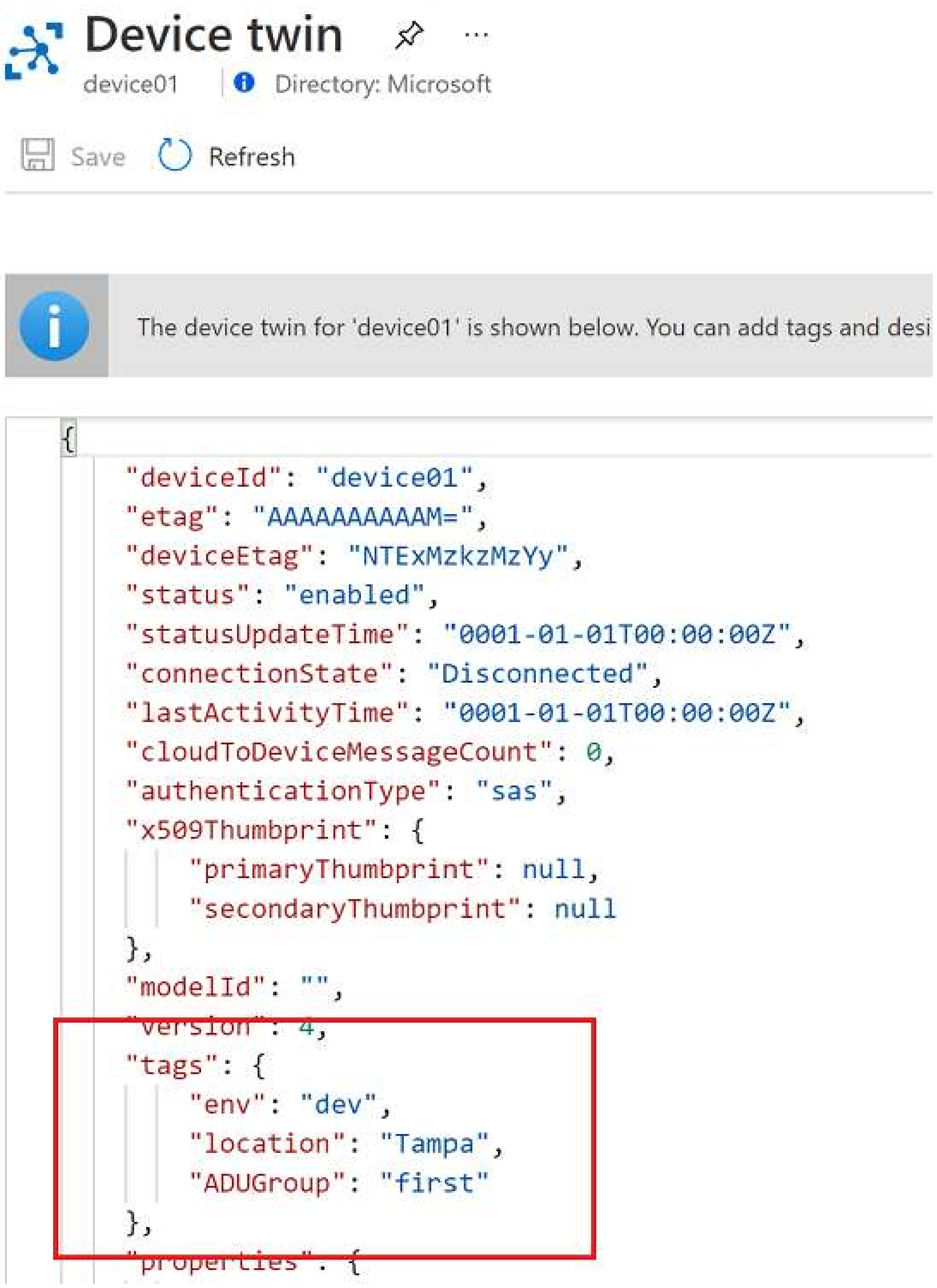
5. Add a tag to your device

In the Azure Portal, look for your IoT Hub, find your IoT Edge device and navigate to the Device Twin or Module Twin.

Add a new Device Update tag value as shown below.

```
"tags": {
            "ADUGroup": "<CustomTagValue>"
            },
```

Your device Twin should like the below image:



The device twin for 'device01' is shown below. You can add tags and desi

```
{
    "deviceId": "device01",
    "etag": "AAAAAAAAAAM=",
    "deviceEtag": "NTExMzkzMzYy",
    "status": "enabled",
    "statusUpdateTime": "0001-01-01T00:00:00Z",
    "connectionState": "Disconnected",
    "lastActivityTime": "0001-01-01T00:00:00Z",
    "cloudToDeviceMessageCount": 0,
    "authenticationType": "sas",
    "x509Thumbprint": {
        "primaryThumbprint": null,
        "secondaryThumbprint": null
    },
    "modelId": "",
    "version": 4,
    "tags": {
        "env": "dev",
        "location": "Tampa",
        "ADUGroup": "first"
    },
    "properties": {
```

6. Import update

Go to Device Update releases in GitHub and click the **Assets** drop-down.
https://github.com/Azure/iot-hub-device-update/releases

Download the **Edge.package.update.samples.zip** by clicking on it.

Extract the contents of the folder to discover an update sample and its corresponding import manifests.

In Azure portal, select the Device Updates option under **Automatic Device Management** from the left-hand navigation bar in your IoT Hub.

Select the **Updates** tab.

Select **+ Import New Update**

- Select the folder icon or text box under **Select an Import Manifest File**. You will see a file picker dialog. Select the **sample-1.0.1-aziot-edge-importManifest.json** import manifest from the folder you downloaded previously.
- Next, select the folder icon or text box under **Select one or more update files** You will see a file picker dialog. Select the **sample-1.0.1-aziot-edge-apt-manifest.json** apt manifest update file from the folder you downloaded previously. This update will update the aziot-identity-service and the aziot-edge packages to version 1.2.0~rc4-1 on your device.
- Select the folder icon or text box under **Select a storage container**. Then select the appropriate storage account or create one storage account during this step.
- Select **Submit** to start the import process.

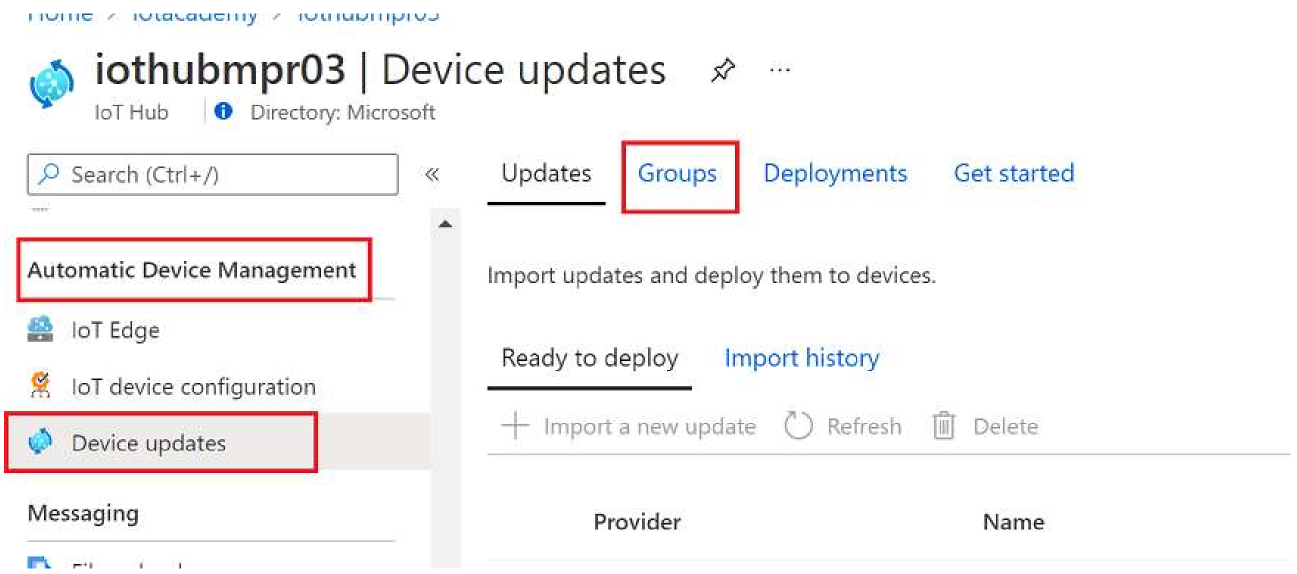
When the Status column indicates the import has succeeded, select the **Ready to Deploy** header. You should see your imported update in the list now.

7. Create update group

Go to the IoT Hub you previously connected to your Device Update instance.

Select the **Device Updates** option under A**utomatic Device Management** from the left-hand navigation bar.

Select the **Groups** tab at the top of the page.



Select the **Add** button to create a new group.

Select the IoT Hub tag you created in the previous step from the list. Select Create update group.

8. Deploy update

Once the group is created, you should see a new update available for your device group, with a link to the update in the Available updates column. You may need to Refresh once.

Click on the link to the available update.

Confirm the correct group is selected as the target group and schedule your deployment

Select Deploy update.

View the compliance chart. You should see the update is now in progress.

After your device is successfully updated, you should see your compliance chart and deployment details update to reflect the same.

You have now completed a successful end-to-end package update using Device Update for IoT Hub on an Ubuntu Server 18.04 x64 device.

Exercise 6: Clean up

Once you completed all the exercises, go to Azure Portal, look for the azure Resource Group you were using for this training and delete the resources group or the resources within the resource group, assuming this resouce group it is only used for the training and not for any other solutions.