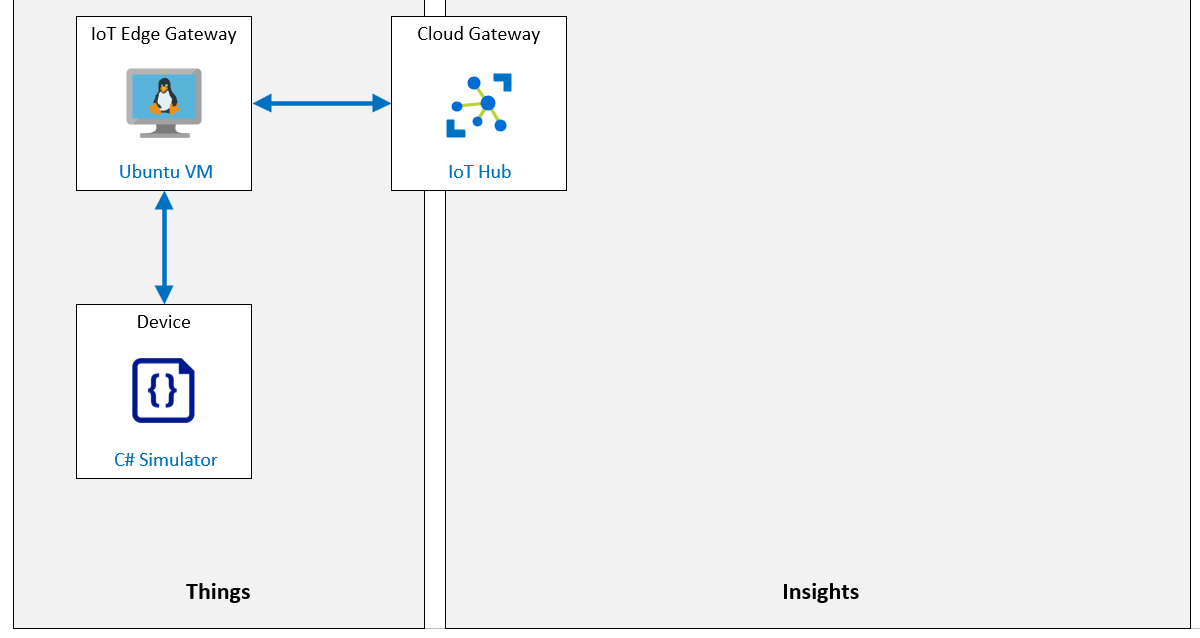
Run an IoT Edge device in restricted network and offline

Lab Scenario

**configuring IoT Edge to support an offline scenario**

[](https://microsoftlearning.github.io/AZ-220-Microsoft-Azure-IoT-Developer/Instructions/Labs/media/LAB_AK_14-architecture.png)

Exercise 6: Configure IoT Edge Device Time-to-Live and Message Storage

Configuring your IoT Edge Devices for extended offline scenarios includes **specifying the supported period of time that you may be offline**, often referred to as Time-to-Live, and **specifying your local storage settings**.

The **default value** for Time-to-Live (TTL) is 7200 (**7200 seconds, which is 2 hours**). This is plenty of time for quick interruptions, but there are cases when two hours may not be long enough, when a device or solution needs to function in Offline mode for an extended period of time. For the solution to operate without telemetry data loss when extended periods in a disconnected state can occur, you can configure the TTL property of the IoT Edge Hub module to a value up to 1,209,600 seconds (a **2 week** TTL period).

The IoT Edge Hub module ($edgeHub) is used to coordinate communications between the the Azure IoT Hub service and the IoT Edge Hub running on the gateway device. Within the Desired Properties for the Module Twin, the storeAndForwardConfiguration.timeToLiveSecs property specifies the time in seconds that IoT Edge Hub keeps messages when in a state disconnected from routing endpoints, such as the Azure IoT Hub service. The timeToLiveSecs property for the Edge Hub can be specified in the Deployment Manifest on a specific device as part of a single-device or at-scale deployment.

The IoT Edge Device will automatically store messages when in a disconnected / offline state. The storage location can be configured using a HostConfig object.

In this exercise, you will use the Azure Portal user interface for Azure IoT Hub to modify the timeToLiveSecs property for the Edge Hub ($edgeHub) module on the single IoT Edge Gateway device. You will also configure the storage location on the IoT Edge Device where the messages are to be stored.

**Task 1: Configure the $edgeHub Module Twin**

1. If necessary, log in to your Azure portal using your Azure account credentials.

If you have more than one Azure account, be sure that you are logged in with the account that is tied to the subscription that you will be using for this course.

1. On your **rg-az220** resource group tile, click **iot-az220-training-{your-id}**.
2. On left-side menu of your IoT hub blade, under **Automatic Device Management**, click **IoT Edge**.

This pane allows you to manage the IoT Edge devices connected to the IoT Hub.

1. Under **Device ID**, click **vm-az220-training-gw0002-{your-id}**.
2. Under **Modules**, click **$edgeHub**.

The Module Identity Details blade of the **Edge Hub** module provides access to the Module Identity Twin and other resources for your IoT Edge Device.

1. On the **Module Identity Details** blade, click **Module Identity Twin**.

This blade contains the module identity twin for vm-az220-training-gw0002-{your-id}/$edgeHub displayed as JSON in an editor pane.

1. Take a moment to review the contents of the $edgeHub module identity twin.

Notice that since this is a new device the desired properties are essentially empty.

1. Close the **Module Identity Twin** blade.
2. Navigate back to the **vm-az220-training-gw0002-{your-id}** blade.
3. At the top of the blade, click **Set Modules**.

The **Set modules on device** blade enables you to create and configure the IoT Edge Modules deployed to this IoT Edge Device.

1. On the **Set modules** blade, under **Iot Edge Modules**, click **Runtime Settings**.
2. On the **Runtime Settings** pane, locate the **Store and forward configuration - time to live (seconds)** field.
3. In the **Store and forward configuration - time to live (seconds)** textbox, enter **1209600**

This specifies a message Time-to-Live value of 2 weeks for the IoT Edge Device, which is the maximum time.

**Note**: There are several things to consider when configuring the **Message Time-to-Live** (TTL) for the Edge Hub ($edgeHub) module. When the IoT Edge Device is disconnected, the messages are stored on the local device. You need to calculate how much data will be stored during the TTL period, and make sure there is enough storage on the device for that much data. The amount of storage and TTL configured will need to meet the solutions requirements if you want to avoid the loss of important data.

If the device does not have enough storage, then you need to configure a shorter TTL. Once the age of a message reaches the TTL time limit, it will be deleted if it has not yet been sent to Azure IoT Hub.

The IoT Edge Device will automatically store messages when in a disconnected / offline state. The storage location can be configured using a HostConfig object.

1. Locate the **Create Options** field.

Notice that this field contains a HostConfig JSON object that can be configured. You will create a HostConfig property and an Environment Variable to configure the storage location for your Edge device.

1. In the HostConfig object, below the closing bracket of PortBindings property, add the following Binds property:

CodeCopy

"Binds": [

"/etc/iotedge/storage/:/iotedge/storage/"

]

**Note**: Be sure to separate the PortBindings property from the Binds property with a comma.

The resulting JSON in the **Create Options** textbox should look similar to the following:

CodeCopy

{

"HostConfig": {

"PortBindings": {

"443/tcp": [

{

"HostPort": "443"

}

],

"5671/tcp": [

{

"HostPort": "5671"

}

],

"8883/tcp": [

{

"HostPort": "8883"

}

]

},

"Binds": [

"/etc/iotedge/storage/:/iotedge/storage/"

]

}

}

This Binds value configures the /iotedge/storage/ directory in the Docker container for the Edge Hub Module to be mapped to the /etc/iotedge/storage/ host system directory on the physical IoT Edge Device.

The value is in the format of <HostStoragePath>:<ModuleStoragePath>. The <HostStoragePath> value is the host directory location on the IoT Edge Device. The <ModuleStoragePath> is the module storage path made available within the container. Both of these values must specify an absolute path.

1. Locate the **Environment Variables** field.

You need to add a new environment variable in order to complete the configuration of the message storage location.

1. Under **Environment Variables**, in the **Name** textbox, enter **storageFolder**
2. Under **Environment Variables**, in the **Value** textbox, enter **/iotedge/storage/**
3. At the bottom of the **Runtime Settings** pane, click **Save**.
4. On the **Set modules on device** blade, click **Review + create**.
5. Take a minute to review the contents of the deployment manifest.

Find your updates within the deployment manifest. You will need to look under both $edgeAgent and $edgeHub to find them.

1. At the bottom of the blade, click **Create**.

Once the change is saved, the **IoT Edge Device** will be notified of the change to the Module configuration and the new settings will be reconfigured on the device accordingly.

Once the changes have been passed to the Azure IoT Edge device, it will restart the **edgeHub** module with the new configuration.

Task 2: Update Directory Permissions

Before continuing, it is essential for you to ensure that the user profile for the IoT Edge Hub module has the required read, write, and execute permissions to the **/etc/iotedge/storage/** directory.

1. On your Azure portal dashboard, click **vm-az220-training-gw0002-{your-id}**.

This should open a blade for your IoT Edge virtual machine, and the Overview pane should be selected.

1. At the top of the **vm-az220-training-gw0002-{your-id}** blade, click **Connect**, and then click **SSH**.
2. Under **Connect via SSH with Client**, locate the **4. Run the example command below to connect to your VM.** field.

You will use an SSH command to connect to the virtual machine. You need a command that is formatted similar to ssh username@52.170.205.79.

1. Below and to the right of the **4. Run the example command below to connect to your VM.** field, click **Copy to clipboard**.
2. Paste the value into Notepad (or another text editor).

**Note**: If the sample command includes -i <private key path>, you must remove that portion of the command. You need a command that is formatted similar to ssh username@52.170.205.79.

1. On the Azure portal toolbar, click **Cloud Shell**.

Ensure that the environment is set to **Bash**.

1. At the Cloud Shell command prompt, paste in the ssh command from the clipboard, and then press **Enter**.
2. If you are prompted with **Are you sure you want to continue connecting?**, type yes and press Enter.

This prompt is a security confirmation since the certificate used to secure the connection to the VM is self-signed. The answer to this prompt will be remembered for subsequent connections, and is only prompted on the first connection.

1. When prompted to enter the password, enter the Administrator password that was entered when the VM was provisioned.

Once connected, the terminal prompt will be updated to display the name of the Linux VM that you are connected to. For example:

CodeCopy

username@vm-az220-training-gw0002-{your-id}:~$

1. To view the running IoT Edge modules, enter the following command:

CodeCopy

iotedge list

1. Take a moment to review the output of the iotedge list command:

You should see that the *edgeHub* has failed to start:

CodeCopy

NAME STATUS DESCRIPTION CONFIG

edgeAgent running Up 4 seconds mcr.microsoft.com/azureiotedge-agent:1.0

edgeHub failed Failed (139) 0 seconds ago mcr.microsoft.com/azureiotedge-hub:1.0

This is due to the fact that the *edgeHub* process does not have permission to write to the **/etc/iotedge/storage/** directory.

1. To confirm the issue with the directory permission, enter the following command:

CodeCopy

iotedge logs edgeHub

The terminal will output the current log - if you scroll through the log you will see the relevant entry that looks similar to the following:

CodeCopy

Unhandled Exception: System.AggregateException: One or more errors occurred. (Access to the path '/iotedge/storage/edgeHub' is denied.) ---> System.UnauthorizedAccessException: Access to the path '/iotedge/storage/edgeHub' is denied. ---> System.IO.IOException: Permission denied

1. To update the directory permissions, enter the following commands:

ShellCopy

sudo chown $( whoami ):iotedge /etc/iotedge/storage/

sudo chmod 775 /etc/iotedge/storage/

The first command sets the owner of the directory to the current user and the owning user group to **iotedge**. The second command enables full access to both the current user and members of the **iotedge** group. This will ensure that the *edgeHub* module is able to create directories and files within the **/etc/iotedge/storage/** directory.

**NOTE**: If you see an error stating **chown: cannot access ‘/etc/iotedge/storage/’: No such file or directory**, create the directory using the following command and then re-run the commands above:

ShellCopy

sudo mkdir /etc/iotedge/storage

1. To restart the *edgeHub* module, and then verify that it is started, enter the following commands:

CodeCopy

iotedge restart edgeHub

iotedge list

You should see that the *edgeHub* module is now running:

CodeCopy

NAME STATUS DESCRIPTION CONFIG

edgeAgent running Up 13 minutes mcr.microsoft.com/azureiotedge-agent:1.0

edgeHub running Up 6 seconds mcr.microsoft.com/azureiotedge-hub:1.0

You are now ready to connect an IoT device (the child/leaf) to this IoT Edge Gateway device.

Exercise 7: Connect Child IoT Device to IoT Edge Gateway

The process to authenticate regular IoT devices to IoT Hub with symmetric keys also applies to downstream (or child / leaf) devices. The only difference is that you need to add a pointer to the Gateway Device to route the connection or, in offline scenarios, to handle the authentication on behalf of IoT Hub.

**Note**: You will be using the connection string value for **sensor-th-0084** that you saved earlier in the lab. If you need a new copy of the connection string, it can be accessed from your Azure IoT Hub in the Azure portal. Open the **IoT devices** pane of your IoT Hub, click **sensor-th-0084**, copy the **Primary Connection String**, and then save it to a text file.

In this exercise, you will configure the downstream IoT device (child or leaf device) to connect to IoT Hub using Symmetric Keys. The devices will be configured to connect to IoT Hub and the parent IoT Edge Device using a Connection String that contains the Symmetric Key (in addition to the Gateway Hostname for the Parent IoT Edge Device).

1. Open the Windows **File Explorer** app, and then navigate to your **Downloads** folder.

Your Downloads folder should contain the X.509 certificate file that was downloaded when you configured the IoT Edge Gateway. You need to copy this certificate file to the root directory of your IoT device app.

1. In the **Downloads** folder, right-click **azure-iot-test-only.root.ca.cert.pem**, and then click **Copy**.

**Note**: If you already had an azure-iot-test-only.root.ca.cert.pem file in your Downloads folder, the file that you need may be named azure-iot-test-only.root.ca.cert (1).pem. You will need to rename it to azure-iot-test-only.root.ca.cert.pem once you’ve added it to the destination folder.

This file is the X.509 certificate file that you downloaded and will be adding to the lab 14 /Starter/ChildIoTDevice directory (where the source code for the Child IoT Device is located).

1. Navigate to the lab 14 Starter folder, and then paste the copied file into the **ChildIoTDevice** folder.
2. Ensure that the copied certificate file is named **azure-iot-test-only.root.ca.cert.pem**

If you already had an azure-iot-test-only.root.ca.cert.pem file in your Downloads folder, the file may have been named azure-iot-test-only.root.ca.cert (1).pem.

1. Open a new instance of Visual Studio Code.
2. On the **File** menu, click **Open Folder**.
3. In the **Open Folder** dialog, navigate to the lab 14 **Starter** folder, click **ChildIoTDevice**, and then click **Select Folder**.

You should now see the project files listed in the EXPLORER pane.

1. In the Visual Studio Code **EXPLORER** pane, click **Program.cs**.
2. In the **Program.cs** file, locate the declaration for the **connectionString** variable.
3. Replace the placeholder value with the Primary Connection String for the **sensor-th-0084** IoT Device.
4. Modify the Connection String value to include the **GatewayHostName** property as follows:

The **GatewayHostName** property should be set to the value of the Public IP Address for the IoT Edge Gateway virtual machine.

The updated Connection String will match the following format:

CodeCopy

HostName=<iot-hub-name>.azure-devices.net;DeviceId=DownstreamDevice1;SharedAccessKey=<iot-device-key>;GatewayHostName=<iot-edge-gateway-hostname>

Be sure to replace the placeholders shown above with the appropriate values:

* + **<IoT-Hub-Name>**: The Name of the Azure IoT Hub.
  + **<IoT-Device-Primary-Key>**: The Primary Key for the sensor-th-0084 IoT Device in Azure IoT Hub.
  + **<IoT-Edge-Gateway-Hostname>**: The Public IP Address of the **vm-az220-training-gw0002-{your-id}** virtual machine (Edge Gateway).

The **connectionString** variable assignment code should look similar to the following:

C#Copy

private readonly static string connectionString = "HostName=iot-az220-training-1119.azure-devices.net;DeviceId=sensor-th-0084;SharedAccessKey=ygNT/WqWs2d8AbVD9NAlxcoSS2rr628fI7YLPzmBdgE=;GatewayHostName=40.124.67.13";

1. On the **File** menu, click **Save**.
2. On the **View** menu, click **Terminal**.

Ensure that the **Terminal** command prompt lists the /Starter/ChildIoTDevice directory.

1. To build and run the **ChildIoTDevice** simulated device, enter the following command:

CodeCopy

dotnet run

**Note**: When the app installs the **X.509 certificate** on the local machine (so it can use it to authenticate with the IoT Edge Gateway), you may see a popup window that asks if you would like to install the certificate. Click **Yes** to allow the app to install the certificate.

1. Notice the output displayed in the Terminal.

Once the simulated device is running, the console output will display the events being sent to the Azure IoT Edge Gateway.

The terminal output will look similar to the following:

CodeCopy

IoT Hub C# Simulated Cave Device. Ctrl-C to exit.

User configured CA certificate path: azure-iot-test-only.root.ca.cert.pem

Attempting to install CA certificate: azure-iot-test-only.root.ca.cert.pem

Successfully added certificate: azure-iot-test-only.root.ca.cert.pem

11/27/2019 4:18:26 AM > Sending message: {"temperature":21.768769073192388,"humidity":79.89793652663843}

11/27/2019 4:18:27 AM > Sending message: {"temperature":28.317862208149332,"humidity":73.60970909409677}

11/27/2019 4:18:28 AM > Sending message: {"temperature":25.552859350830715,"humidity":72.7897707153064}

11/27/2019 4:18:29 AM > Sending message: {"temperature":32.81164186439088,"humidity":72.6606041624493}

1. Leave the simulated device running while you move on to the next Exercise.

Exercise 8: Test Device Connectivity and Offline Support

In this exercise, you will monitor events from the **sensor-th-0084** that are being sent to Azure IoT Hub through the **vm-az220-training-gw0002-{your-id}** IoT Edge Transparent Gateway. You will then interrupt connectivity between the **vm-az220-training-gw0002-{your-id}** and Azure IoT Hub to see that telemetry is still sent from the child IoT Device to the IoT Edge Gateway. After this, you will resume connectivity with Azure IoT Hub and monitor that the IoT Edge Gateway resumes sending telemetry to Azure IoT Hub.

1. If necessary, log in to your Azure portal using your Azure account credentials.

If you have more than one Azure account, be sure that you are logged in with the account that is tied to the subscription that you will be using for this course.

1. On the Azure portal toolbar, click **Cloud Shell**.

Ensure that the Environment dropdown is set to **Bash**.

1. At the Cloud Shell command prompt, to start monitoring the Events being received by the Azure IoT Hub, enter the following command:

CodeCopy

az iot hub monitor-events --hub-name iot-az220-training-{your-id}

Be sure to replace the {your-id} placeholder with your unique suffix for our Azure IoT Hub instance.

1. Notice that telemetry from the **sensor-th-0084** that is getting sent to Azure IoT Hub.

Keep in mind that the **sensor-th-0084** simulated device application is configured to send telemetry to the **vm-az220-training-gw0002-{your-id}** IoT Edge Transparent Gateway virtual machine, which is then sending the telemetry on to Azure IoT Hub.

The Cloud Shell should begin displaying event messages similar to the following:

CodeCopy

Starting event monitor, use ctrl-c to stop...

{

"event": {

"origin": "sensor-th-0084",

"payload": "{\"temperature\":20.30307372114764,\"humidity\":72.6844747889249}"

}

}

{

"event": {

"origin": "sensor-th-0084",

"payload": "{\"temperature\":31.73955729079412,\"humidity\":78.56052768349673}"

}

}

**Note**: Next, you will need to test the **Offline** capabilities. To do this, you need to make the **vm-az220-training-gw0002-{your-id}** device go offline. Since this is a Virtual Machine running in Azure, this can be simulated by adding an **Outbound rule** to the **Network security group** for the VM.

1. Within the **Azure portal**, navigate to your Dashboard, and then locate the **rg-az220vm** resource group tile.
2. In the list of resources, to open the **Network Security Group** for the **vm-az220-training-gw0002-{your-id}** virtual machine, click **vm-az220-training-gw0002-{your-id}NSG**.
3. On the **Network security group** blade, on the left side navigation pane under **Settings**, click **Outbound security rules**.
4. At the top of the blade, click **+ Add**.
5. On the **Add outbound security rule** pane, set the following field values:
   * Destination port ranges: **\***
   * Action: **Deny**
   * Name: **DenyAll**

A **Destination port range** of “**\***” will apply the rule to all ports.

1. At the bottom of the blade, click **Add**.
2. Go back to the **Cloud Shell** in the Azure portal.
3. If the az iot hub monitor-events command is still running, end it by pressing **Ctrl + C**.
4. At the Cloud Shell command prompt, to connect to the **vm-az220-training-gw0002-{your-id}** VM using ssh, enter the following command:

ShellCopy

ssh <username>@<ipaddress>

Be sure to replace the placeholders with the required values for the ssh command:

| Placeholder | Value to replace |
| --- | --- |
| <username> | The admin **Username** for the **IoTEdgeGateaway** virtual machine. This should be **vmadmin**. |
| <ipaddress> | The **Public IP Address** for the **vm-az220-training-gw0002-{your-id}** virtual machine. |

1. When prompted, enter the admin **Password** for the **vm-az220-training-gw0002-{your-id}**.

The command prompt will be update once you are connected to the **vm-az220-training-gw0002-{your-id}** VM via ssh.

1. To reset the IoT Edge Runtime, enter the following command:

ShellCopy

sudo systemctl restart iotedge

This will force the IoT Edge Runtime to disconnect from the Azure IoT Hub service, and then attempt to reconnect.

1. To verify that the *edgeHub* module has restarted correctly, enter the following command:

CodeCopy

iotedge list

If the *edgeHub* module failed to restart successfully, retry by entering the following commands:

CodeCopy

iotedge restart edgeHub

iotedge list

1. To end the ssh session with the **vm-az220-training-gw0002-{your-id}**, enter the following command:

CodeCopy

exit

1. At the Cloud Shell command prompt, to start monitoring the Events being received by the Azure IoT Hub, enter the following command

CodeCopy

az iot hub monitor-events --hub-name iot-az220-training-{your-id}

Be sure to replace the {your-id} placeholder with your unique suffix for our Azure IoT Hub instance.

1. Notice there are no longer any events being received by the **Azure IoT Hub**.
2. Switch to the Visual Studio Code window.
3. Open the **Terminal** where the **sensor-th-0084** simulated device application is running, and notice that it’s still sending device telemetry to the **vm-az220-training-gw0002-{your-id}**.

At this point the **vm-az220-training-gw0002-{your-id}** is disconnected from the Azure IoT Hub. It will continue to authenticate connections by the **sensor-th-0084**, and receive device telemetry from child device(s). During this time, the IoT Edge Gateway will be storing the event telemetry from the child devices on the IoT Edge Gateway device storage as configured.

1. Switch to you **Azure portal** window.
2. Navigate back to the **Network security group** blade for the **vm-az220-training-gw0002-{your-id}**.
3. On the left side navigation menu, under **Settings**, click **Outbound security rules**.
4. On the **Outbound security rules** pane, click **DenyAll**.
5. On the **DenyAll** pane, to remove this deny rule from the NSG, click **Delete**.
6. On the **Delete security rule** prompt, click **Yes**.

Once the **vm-az220-training-gw0002-{your-id}** IoT Edge Transparent Gateway is able to resume connectivity with Azure IoT Hub, it will sync the event telemetry from all connected child devices. This includes the saved telemetry that couldn’t be sent while disconnected, and all telemetry still being sent to the gateway.

**Note**: The IoT Edge Gateway device will take a couple minutes to reconnect to Azure IoT Hub and resume sending telemetry. After waiting a couple minutes, you will see events showing up in the az iot hub monitor-events command output again.

In this lab we have demonstrated that an Azure IoT Edge Gateway can utilize local storage to retain messages that can’t be sent due to an interruption in the connection to the IoT Hub. Once connection is reestablished, we saw that messages are then sent.

**Note**: Once you have finished with the lab, ensure you exit the device simulation application by pressing **CTRL+C** in the terminal.