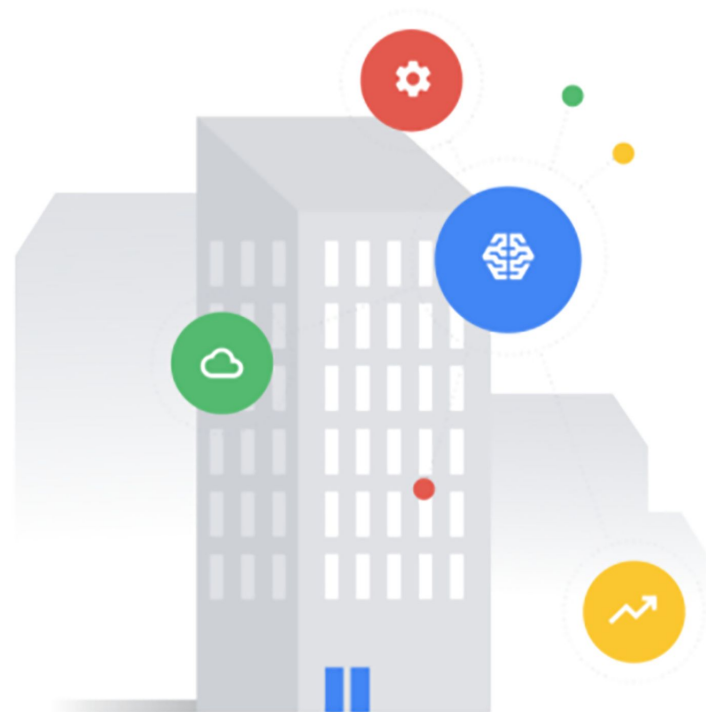




Lesson 5



Gateways



Before you get started

This onboarding deck has interactive features and activities that enable a self-guided learning experience. To help you get started, here are two tips for viewing and navigating through the deck.

1 View this deck in presentation mode.

- To enter presentation mode, you can either:
 - Click the **Present** or **Slideshow** button in the top-right corner of this page.
 - Press **Ctrl+F5** (Windows), **Cmd+Enter** (macOS), or **Ctrl+Search+5** (Chrome OS) on your keyboard.
- To exit presentation mode, press the **Esc** key on your keyboard.

2 Navigate by clicking the buttons and links.

- Click the **Back** or **Next** buttons to go backward or forward in the deck. Moving forward, you'll find them in the bottom corners of every slide.
- Click [blue text](#) to go to another slide in this deck or open a new page in your browser.
- For the best learning experience, using your keyboard or mouse wheel to navigate is discouraged.

Ready to get started?

Let's go!

Lesson 5

IoT gateway

[Back](#)

What you'll learn about:

- Connecting legacy devices to the cloud

By the end of this lesson, you'll be able to:

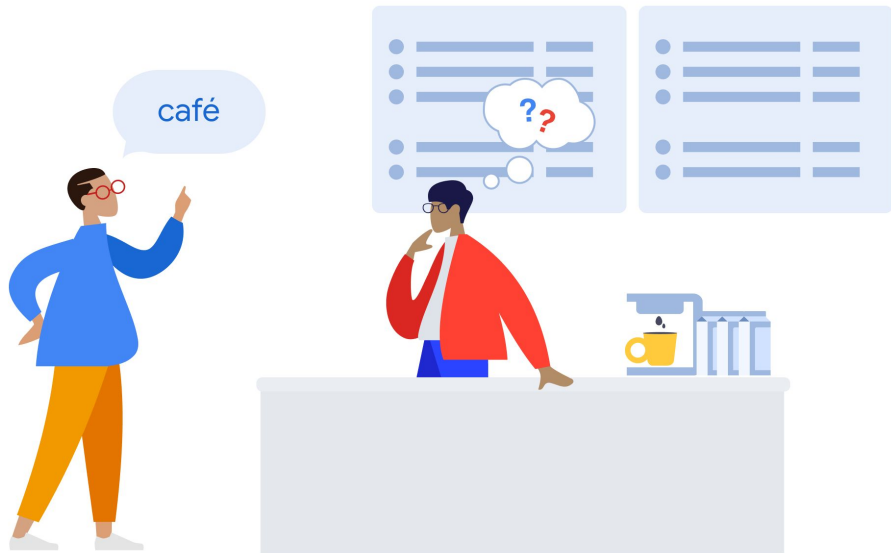
- Understand what a UDMI IoT gateway is, and differentiate it from a direct-connect device.
- Identify gateway and proxy devices in Google Cloud Project (GCP).
- Run pubsub as a gateway device.
- Observe gateway message traffic in GCP.
- Diagnose attachment problems in GCP after manually detaching a device.

[Next](#)

What is a gateway?

To better understand what a gateway is and why you might use one, imagine there's a coffee stand that happens to be in Thailand...

Antoine, a tourist visiting from Paris, can't wait to try the shop's espresso. The only problem is Antoine doesn't speak Thai. And the barista... doesn't speak French.



[Back](#)

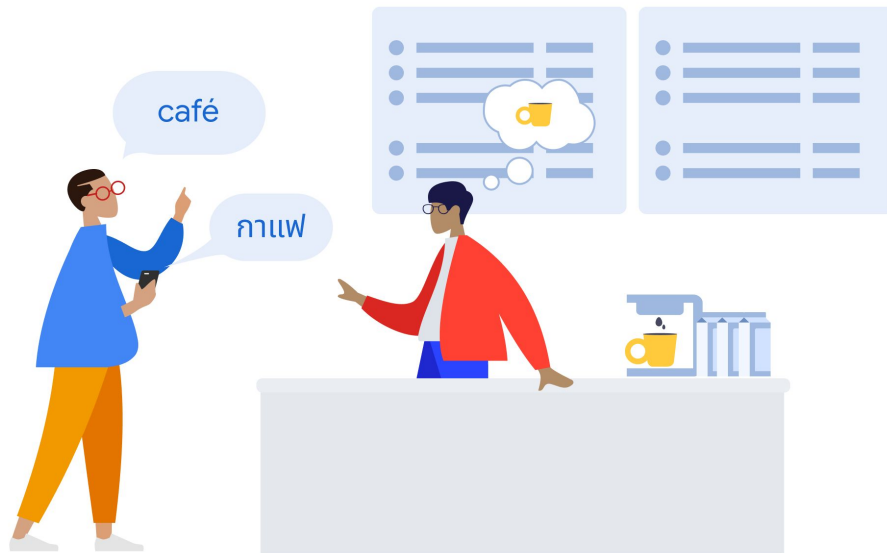
[Next](#)

What is a gateway? (continued)

Just as Antoine couldn't speak directly with the barista, some devices can't use MQTT to communicate directly with the cloud.

In those instances, a gateway, a directly-connected UDMI device, can be used to proxy the data to the cloud on behalf of the non-MQTT device.

This can work because, from the cloud's perspective, everything is just a "device" and can be interacted with normally. So the gateway acts as the bridge between the two languages.



[Back](#)

[Next](#)

Connection models and terminology

All devices are registered as devices in GCP IoT Core as a “digital twin” that mirrors their physical state. However, not all devices are able to connect directly to IoT Core. These devices require special handling.

Click to take a look at some primary connection models used in UDMI.

Direct devices

Gateway devices

Proxied devices



Back

Note: See the UDMI Project GitHub for more information about [connection models](#).

Next

Connection models and terminology

Remember, all devices are registered as devices in GCP IoT Core as a “digital twin.”

However, not all devices are able to connect directly to IoT Core. These devices require special handling.

Click to take a look at some primary connection models used in UDMI.

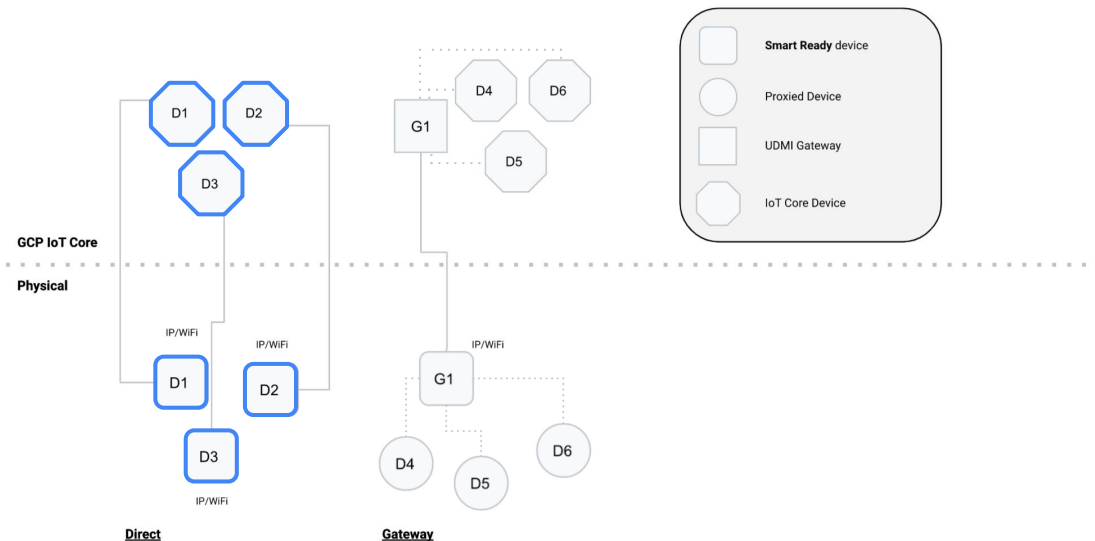
Direct devices

Gateway devices

Proxied devices

These are devices that can maintain an authenticated connection directly to GCP IoT Core. These devices:

- Speak directly to the cloud
- Properly implement a MQTT driver with proper UDMI message formatting and functionality (a capability present in many modern devices)
- Are connected over an IP network
- Have a unique private authentication key
- Are registered in GCP IoT Core



Back

Note: See the UDMI Project GitHub for more information about [connection models](#).

Next

Connection models and terminology

Remember, all devices are registered as devices in GCP IoT Core as a “digital twin.”

However, not all devices are able to connect directly to IoT Core. These devices require special handling.

Click to take a look at some primary connection models used in UDMI.

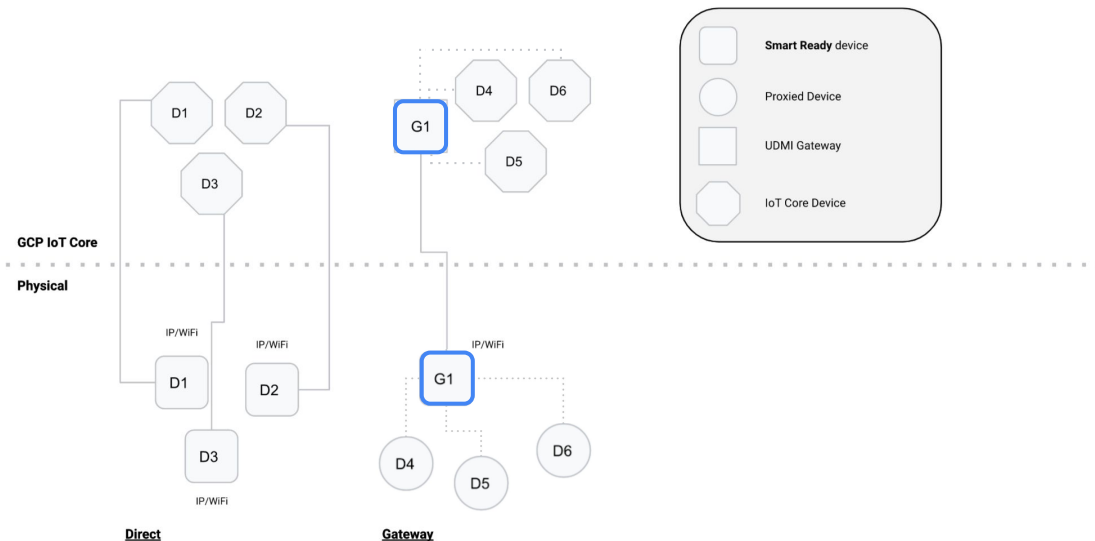
Direct devices

Gateway devices

Proxied devices

These are direct devices which also send data for other proxied devices. In addition to all the properties of a direct device, these devices:

- Are registered as gateways in GCP IoT Core
- Have proxied devices bound to them in the IoT Core Registry
- Attach other devices in order to proxy data



Back

Note: See the UDMI Project GitHub for more information about [connection models](#).

Next

Connection models and terminology

Remember, all devices are registered as devices in GCP IoT Core as a “digital twin.”

However, not all devices are able to connect directly to IoT Core. These devices require special handling.

Click to take a look at some primary connection models used in UDMI.

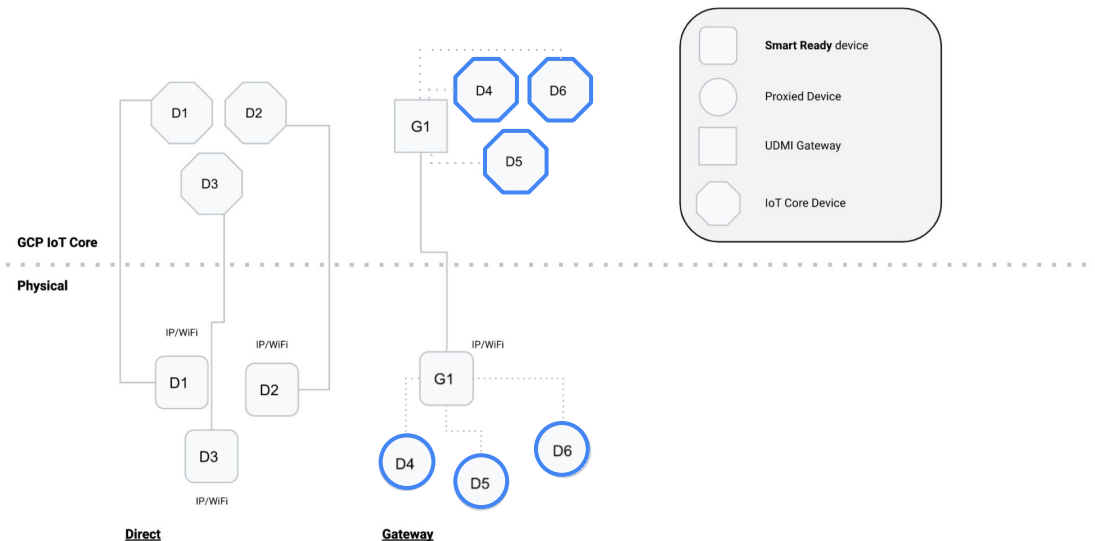
Direct devices

Gateway devices

Proxied devices

These are devices that are not direct devices and send data through a gateway. Proxied devices:

- Can be reached by an IP or fieldbus network, but don't support UDMI
- Are registered on GCP IoT Core and bound to a gateway
- Don't have authentication keys



Back

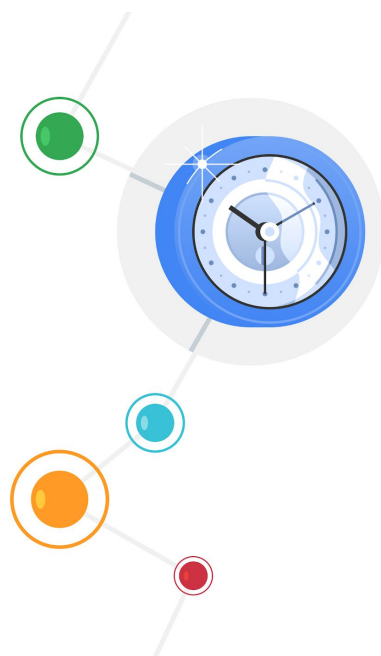
Note: See the UDMI Project GitHub for more information about [connection models](#).

Next

UDMI gateways

There are many kinds of gateways, including those that have nothing to do with UDMI (e.g., a network gateway).

UDMI gateways are special types of gateways, which send and receive data for proxied devices using UDMI. Devices which function as UDMI gateways may function as other kinds of gateways. The proxied devices could then be devices or entities which are found on different types of networks or serial buses.



[Back](#)

[Next](#)

UDMI gateway requirements

UDMI gateways must meet specific requirements.

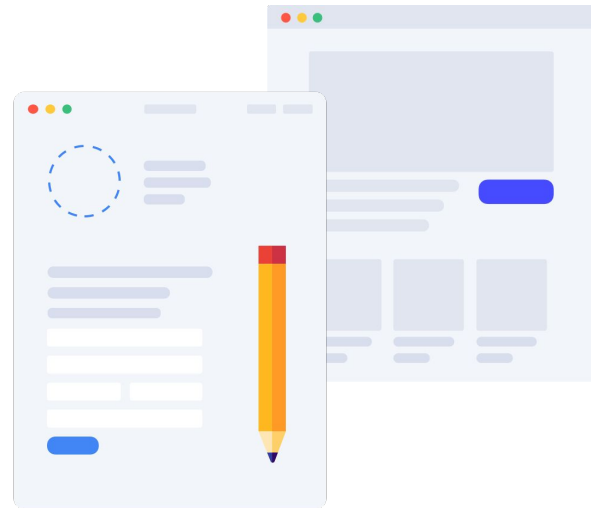
- The gateway and proxy devices must all be registered in GCP IoT Core.
- The proxy devices must be “bound” to the gateway in GCP IoT Core. This allows the gateway to attach to the device and send messages on the device.

There are also specific device behavior requirements for the gateway to send and receive data for other devices.

- The gateway maintains a single client connection to GCP IoT core which is utilized by all devices. Proxy devices do not have authentication keys or initiate separate connections.
- The gateway “attaches” (Cloud IoT terminology) its proxy device to itself to be able to send and receive messages on behalf of another device.

UDMI functionality is implemented by the gateway for the proxy devices. Here are a few of the kinds of functionalities implemented:

- Subscribing to the configuration topic
- Receiving and using config updates (e.g., for setting for sample rate, points, etc.)
- Publishing both state and telemetry
- Structuring of all payloads according the UDMI message format schema
- Writeback (where applicable)



[Back](#)

Note: When using the registrar, devices designated as gateways and proxied in the site model are registered accordingly on IoT Core, with UDMI gateway devices registered as gateways and proxy devices bound to the respective gateways.

[Next](#)

UDMI gateway architecture examples

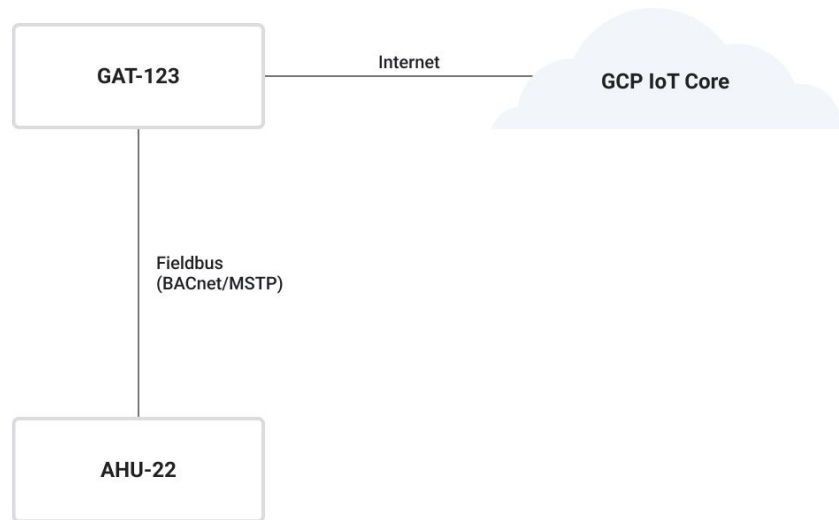
Here is a non-exhaustive list of devices that might require a gateway device.

- Devices which only connect over fieldbus connections, such as BACnet, Modbus, DALI
- Devices which don't support UDMI
- A single physical device reporting for multiple devices
- A headend reporting for other devices in a system
- A unitary controller reporting for itself and separately for the equipment it's controlling

To help get a better understanding of how UDMI gateways work, let's look at an example of one in use.

A local fieldbus network, which in this instance is BACnet/MSTP, has two devices: GAT-123 and AHU-22.

- GAT-123 is a UDMI gateway and is connected to the internet.
- AHU-22 is a controller, is not smart-ready, and is not connected to the internet. It is connected to GAT-123 over a fieldbus link.



[Back](#)

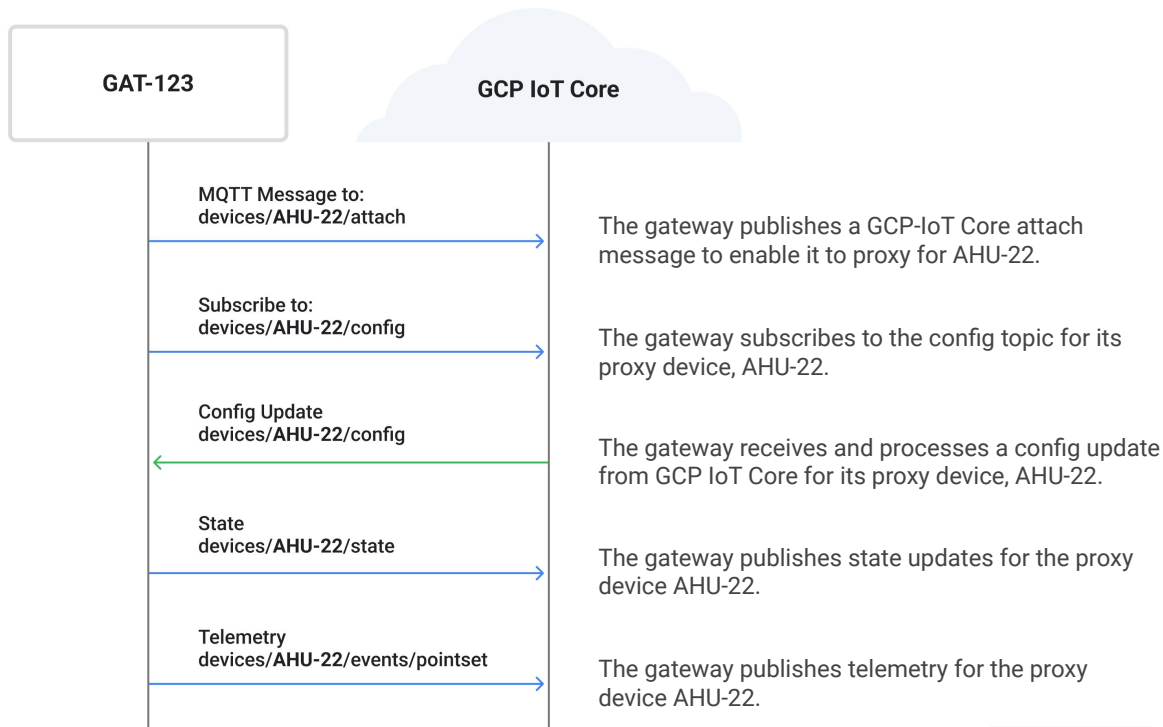
[Next](#)

UDMI gateway sequence example

Now, let's take a look at how these devices communicate with each other.

Before the gateway can proxy data for a device, it needs to communicate to the cloud that it's proxying for the device by publishing an "attach" message. The gateway must then subscribe to receive its config messages so that it can act on the device's behalf.

In this example, the gateway GAT-123 is connected to the device AHU-22 over a BACnet/MSTP fieldbus network. It receives telemetry from the device and then publishes that data to the cloud. Similarly, GAT-123 will receive config data from the cloud and translate it to the appropriate BACnet/MSTP communication to AHU-22.



[Back](#)

[Next](#)

Gateways and proxy devices in site model and config

In the site model, there are additional fields in the metadata for devices which are gateways or proxy devices.

The [Gateway block](#) in the metadata is used to describe the relationship between gateway and proxy devices, and provide additional information.

The [Localnet block](#) is used to describe the presence of the device on a local network. For example, this could be the device's BACnet ID.

[Back](#)

[Next](#)

UDMI gateway metadata

Here's an example of metadata for a UDMI gateway device.

For more information see:

<https://faucetsdn.github.io/udmi/docs/specs/gateway#metadata>

```
{
  "version": 1,
  "timestamp": "2018-08-26T21:39:29.364Z",
  "system": {
    ...
  },
  "pointset": {
    ...
  },
  "cloud": {
    "auth_type": "RS256"
  },
  "gateway": {
    "proxy_ids": ["AHU-22"]
  },
  "localnet": {
    "families": {
      "bacnet": {
        "id": "2"
      }
    }
  }
}
```

The system and pointset block metadata for the gateway device are similar to other devices. (The contents for system and pointset block are omitted for this example. Gateways can have their own data points, too.)

This block contains the cloud connection properties for the gateway and the type of authentication.

For a gateway, the proxy_ids field is a list of IoT Core device IDs that the gateway proxies for. In this example, the gateway will proxy for the device "AHU-22."

The **localnet block** defines local network properties for the device. In this instance, it provides the BACnet ID of the gateway device.

Back

Next

Proxy device metadata

Here's an example of metadata for a proxy device.

For more information see:

<https://faucetsdn.github.io/udmi/docs/specs/gateway#metadata-1>

Note: There is no cloud block for proxied devices, as these kinds of devices don't have any cloud authentication.

```
{
  "version": "1.3.14",
  "timestamp": "2018-08-26T21:39:29.364Z",
  "system": {
    ...
  },
  "pointset": {
    "points": {
      "return_air_temperature_sensor": {
        "ref": "AI3.present_value"
      },
      "cooling_valve_position": {
        "ref": "AO1.present_value"
      }
    }
  },
  "localnet": {
    "families": {
      "bacnet": {
        "id": "33"
      }
    }
  },
  "gateway": {
    "gateway_id": "DDC-1"
  }
}
```

The system metadata block includes information about the device as it does for a direct device.

The pointset block specifies the points the device has, again as it would for a direct connected device. This may additionally include fields such as "ref," which provides a reference for the point.

The [localnet block](#) defines the local network properties for the device. In this instance, it gives the BACnet ID of the device.

The gateway block provides the ID of the gateway which will proxy the device.

Back

Next

UDMI gateways and proxy config messages

Config messages for gateways and proxied devices are the same as the configuration for other devices, but include some additional gateway-specific fields.

The config messages delivered to gateway devices also include the “gateway” block, which tells the gateway which devices it is proxying for.

The config messages delivered to proxy devices may also additionally include the “localnet” block or “ref” field for points.

[Back](#)

[Next](#)

UDMI gateways

This is an example of a config for a gateway device.

For more information see:

<https://faucetsdn.github.io/udmi/docs/specs/gateway#config>

```
{
  "version": 1,
  "timestamp": "2018-08-26T21:39:29.364Z",
  "system": {
    "min_loglevel": 200
  },
  "pointset": {
    ...
  },
  "gateway": {
    "proxy_ids": ["AHU-22"]
  },
}
```

These are the system and pointset block config for the respective gateway device. (Gateways can have their own points, too!)

For a gateway, the proxy_ids field is a list of IoT Core device IDs the gateway proxies for. In this example, the gateway will proxy for the device "AHU-22."

Back

Next

Proxy devices

This is an example of a config for a proxy device.

For more see:

<https://faucetsdn.github.io/udmi/docs/specs/gateway#config-1>

```
{
  "version": "1.3.14",
  "timestamp": "2018-08-26T21:39:29.364Z",
  "system": {
    "min_loglevel": 200
  },
  "pointset": {
    "sample_limit_sec": 2,
    "sample_rate_sec": 500,
    "points": {
      "return_air_temperature_sensor": {
        "ref": "AI3.present_value"
      },
      "cooling_valve_position": {
        "ref": "AO1.present_value"
      }
    }
  },
  "localnet": {
    "families": {
      "bacnet": {
        "id": "33"
      }
    }
  }
}
```

This is the system configuration.

This is the pointset configuration, including the points for the device.

The points may include a ref field for the different points, which identifies the local reference/address for a point on the device.

In this example, the gateway will know that point **return_air_temperature_sensor** is the point with the address **AI3.present_value** on AHU-22.

These are the local network properties for the device. In this instance, the block is giving the BACnet ID for device "AHU-22" on the local BACnet network, which is "33."

The gateway may use this information to find and connect to this device.

Back

Next

UDMI gateways and proxy state messages

State messages from gateway devices have an additional [gateway](#) block, which is used to list its proxy devices and the respective status of each device.

For example, if there was a connection error, it would be indicated in this block.

Example

```
{
  ... // (rest of state message)
  "gateway": {
    "devices": {
      "AHU-22": {},
      "SNS-4": {
        "status": {
          "message": "Error connecting to device",
          "category": "gateway.proxy.connect",
          "timestamp": "2022-06-26T21:39:30.364Z",
          "level": 600
        }
      }
    }
  }
}
```

[Back](#)

[Next](#)

UDMI gateways and proxy devices in GCP

Devices registered as gateways are listed under the “Gateway” section on IOT Core.

The screenshot displays the Google Cloud IoT Core interface. On the left, a navigation menu includes 'Registry details', 'Devices', 'Gateways' (highlighted with a red box), and 'Monitoring'. The main content area is titled 'Gateways' and includes a '+ CREATE A GATEWAY' button and a 'DELETE' button. Below this, the 'Registry ID: ZZ-PER-FECTA' is shown, along with the location 'us-central1' and a note: 'Gateways handle communication for low-powered devices. [Learn more](#)'. A filter input field is present with the placeholder text 'Enter exact gateway ID'. Below the filter is a table with the following data:

<input type="checkbox"/>	Gateway ID	Communication	Last seen
<input type="checkbox"/>	GAT-123	Allowed	—

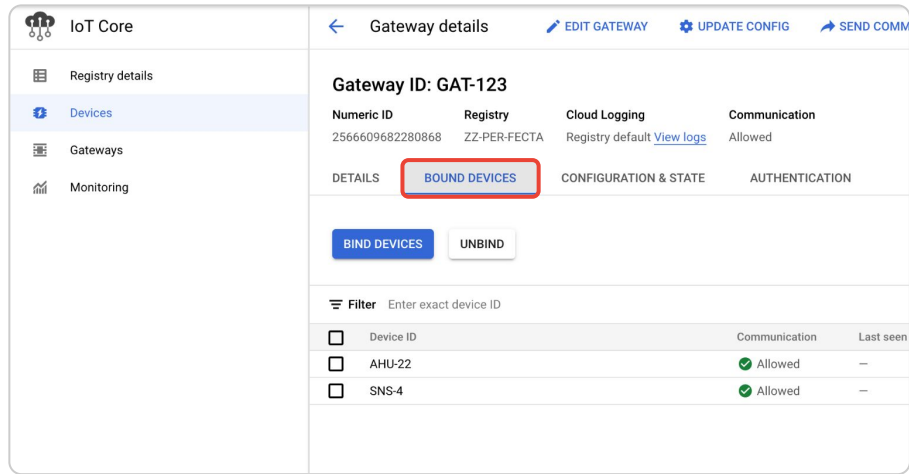
At the bottom of the main content area, there is a link: [Cloud IoT Core documentation](#).

Back

Next

UDMI gateways and proxy devices in GCP (continued)

When viewing a gateway device, the devices bound to it can be seen by clicking on the **BOUND DEVICES** tab.



The screenshot shows the IoT Core interface for a gateway named GAT-123. The left sidebar contains navigation options: Registry details, Devices (selected), Gateways, and Monitoring. The main content area displays the Gateway details for GAT-123, including its Numeric ID (2566609682280868), Registry (ZZ-PER-FECTA), Cloud Logging (Registry default), and Communication (Allowed). Below this, there are tabs for DETAILS, BOUND DEVICES (highlighted with a red box), CONFIGURATION & STATE, and AUTHENTICATION. There are buttons for BIND DEVICES and UNBIND. A filter section allows entering an exact device ID. A table below lists the bound devices:

<input type="checkbox"/>	Device ID	Communication	Last seen
<input type="checkbox"/>	AHU-22	✔ Allowed	—
<input type="checkbox"/>	SNS-4	✔ Allowed	—

Back

Next

Running pubber as a gateway or proxy device

Launching pubber for a proxy device will automatically launch the respective gateway device.

In the example site model (which you should have downloaded in [Lesson 3](#)), there is a gateway device (GAT-123) and two proxied devices (AHU-22 and SNS-4).

To launch pubber for a proxy device, use your project ID and the device ID for the proxy device in the launch command.

Example

```
bin/pubber sites/udmi_site_model udmi-learning AHU-22 123
```

[Back](#)

[Next](#)

Observing gateway message traffic

Messages from other devices which are published by a gateway have an additional Pub/Sub attribute named **“gatewayId.”**

This identifies the IoT ID of the gateway publishing on behalf of the device.

For example, with pubber running, look at the subscription to the `udmi_Target` topic (which was called `udmi_target_subscription` if earlier instructions were followed). It will show a new column named `“attribute.gatewayId”` which is populated with `“GAT-123,”` the ID of the gateway.

MESSAGES METRICS DETAILS

Click Pull to view messages and temporarily delay message delivery to other subscribers. Select Enable ACK messages and then click ACK next to the message to permanently prevent message delivery to other subscribers.

⚠ Some messages or columns were truncated due to size. To pull the full message, see this [documentation](#) for an alternative approach.

PULL Enable ack messages

Filter attribute.gatewayId: GAT-123 Filter messages

attribute.deviceRegistryLocation	attribute.gatewayId	attribute.projectId	attribute.subFolder	Ack ↑
us-central1	GAT-123	daq1-273309	pointset	Deadline exceeded
us-central1	GAT-123	daq1-273309	pointset	Deadline exceeded
us-central1	GAT-123	daq1-273309	system	Deadline exceeded
us-central1	GAT-123	daq1-273309	pointset	Deadline exceeded
us-central1	GAT-123	daq1-273309	pointset	Deadline exceeded

Back

Next

Diagnose attachment problems in GCP, after manually detaching a device

Within GCP IoT Core, there is a special topic `/devices/{gateway_ID}/errors` for gateway devices to subscribe to, on which they will receive messages on errors which are encountered by the gateway device.

For more information, see <https://cloud.google.com/iot/docs/how-tos/gateways/mqtt-bridge#troubleshooting>.

The [list of possible errors](#) is included in the GCP IoT Core documentation, as is the structure of messages published to this topic.

Gateway devices should subscribe to this topic and expose any messages, to provide useful information to diagnose issues.

[Back](#)

[Next](#)

Diagnose attachment problems in GCP, after manually detaching a device (continued)

Let's take a look at an example.

1. On GCP IoT Core, navigate to the device page for the "GAT-123" gateway device.
2. Click on **BOUND DEVICES**, select **AHU-22**, and click **UNBIND**. This will unbind AHU-22 from the gateway GAT-123, preventing the gateway from attaching to AHU-22 and publishing messages for it on GCP IoT Core.
3. Run pubber as device AHU-22. Again, be sure you are using your project ID in the command. For example:
4. Look for a log entry.

```
bin/pubber sites/udmi_site_model udmi-learning AHU-22 123  
  
GATEWAY_ATTACHMENT_ERROR for AHU-22: DeviceId AHU-22 is  
not associated with Gateway DeviceId *****
```

The screenshot shows the GCP IoT Core interface for a gateway named GAT-123. The 'BOUND DEVICES' tab is active, displaying a table of devices. The device AHU-22 is selected, and its status is 'Allowed'. The 'UNBIND' button is visible next to the device name.

Device ID	Communication	Last seen	Cloud Logging
<input checked="" type="checkbox"/> AHU-22	Allowed	Jan 1, 1970, 1:00:00 AM	Registry default
<input type="checkbox"/> SNS-4	Allowed	—	Registry default

Back

Next

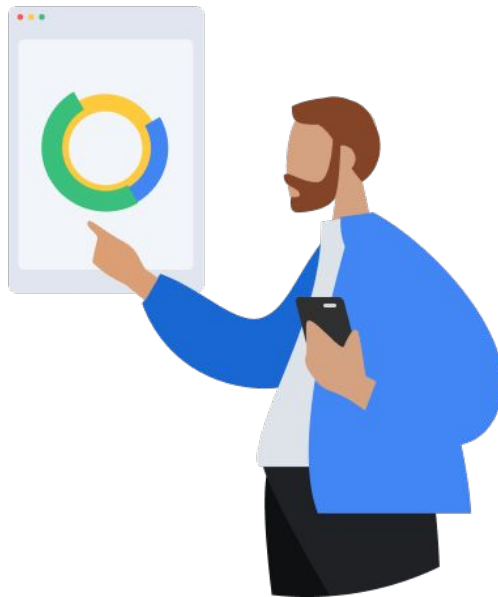
Lesson 5 summary

Let's review what you learned about:

- Connecting legacy devices to the cloud

Now you should be able to:

- Understand what a UDMI IoT gateway is, and differentiate it from a direct-connect device.
- Identify gateway and proxy devices in Google Cloud Project (GCP).
- Run pubsub as a gateway device.
- Observe gateway message traffic in GCP.
- Diagnose attachment problems in GCP after manually detaching a device.



[Back](#)

[Next](#)

You completed Lesson 5!

Now's a great time to take a quick break before starting Lesson 6.

Ready for Lesson 6?

Let's go!

Back

Press the **Esc** key on your keyboard to exit presentation mode.

Helpful resources

Bookmark these resources for future reference.

- [UDMI Project GitHub](#)
Contains specification for management and operation of IoT systems.
- [Git Documentation](#)
Contains various sources of information about Git contributed by Git community.