Amazon Web Services: Getting started on Raspberry Pi 5

Colophon

Release	3
Build date	19/01/2026
Build version	8bfca70f8635

Legal disclaimer notice

TECHNICAL AND RELIABILITY DATA FOR RASPBERRY PI PRODUCTS (INCLUDING DATASHEETS) AS MODIFIED FROM TIME TO TIME ("RESOURCES") ARE PROVIDED BY RASPBERRY PI LTD ("RPL") "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW IN NO EVENT SHALL RPL BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THE RESOURCES, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

RPL reserves the right to make any enhancements, improvements, corrections or any other modifications to the RESOURCES or any products described in them at any time and without further notice.

The RESOURCES are intended for skilled users with suitable levels of design knowledge. Users are solely responsible for their selection and use of the RESOURCES and any application of the products described in them. User agrees to indemnify and hold RPL harmless against all liabilities, costs, damages or other losses arising out of their use of the RESOURCES.

RPL grants users permission to use the RESOURCES solely in conjunction with the Raspberry Pi products. All other use of the RESOURCES is prohibited. No licence is granted to any other RPL or other third party intellectual property right.

HIGH RISK ACTIVITIES. Raspberry Pi products are not designed, manufactured or intended for use in hazardous environments requiring fail safe performance, such as in the operation of nuclear facilities, aircraft navigation or communication systems, air traffic control, weapons systems or safety-critical applications (including life support systems and other medical devices), in which the failure of the products could lead directly to death, personal injury or severe physical or environmental damage ("High Risk Activities"). RPL specifically disclaims any express or implied warranty of fitness for High Risk Activities and accepts no liability for use or inclusions of Raspberry Pi products in High Risk Activities.

Raspberry Pi products are provided subject to RPL's Standard Terms. RPL's provision of the RESOURCES does not expand or otherwise modify RPL's Standard Terms including but not limited to the disclaimers and warranties expressed in them.

Document version history

Release	Date	Description
1	20 Nov 2025	Initial release
2	21 Nov 2025	Update GitHub repository name
3	16 Dec 2025	Copy edit

Scope of document

This document applies to the following Raspberry Pi products:

Single Board Computers / SBCs

Pi Zero			Pi Zero 2		Pi 1		Pi 2		Pi 3	Pi 4	Pi 5
-	W	H	W	H	A	B	A	B	B	-	-
											✓

Introduction

Applicable operating systems for this guide

This guide assumes you are using Raspberry Pi OS with a full desktop and browser. Other OS installations (e.g. Ubuntu) are also likely to work, but this is not guaranteed. Some instructions require you to use a browser to set up Amazon Web Services a full Raspberry Pi OS installation includes both Chromium and Firefox.

Overview

This document describes how to use Raspberry Pi 5 to connect to an Amazon Web Services (AWS) instance. Raspberry Pi is a general-purpose computing device, so rather than providing a specific AWS IoT application, this document describes the basics of obtaining certificates, adding them to a Raspberry Pi 5, and testing the device using AWS Device Compliance.

This example uses Python and the AWS Python bindings to provide proof of principle. It does not go into detail on how to write AWS applications, nor does it cover the use of the AWS Internet of Things (IoT) C/C++ SDK.

Hardware

Raspberry Pi 5 is a single-board computer (SBC) from Raspberry Pi Ltd. It is a general-purpose computing device that can leverage AWS APIs and libraries to connect to AWS IoT.

Product datasheet

Information on Raspberry Pi 5 can be found here:

Product brief

https://pip-assets.raspberrypi.com/categories/892-raspberry-pi-5/documents/RP-008348-DS-1-raspberry-pi-5-product-brief.pdf

Standard kit contents

Raspberry Pi 5 is usually sold as a standalone unit, but it can also be purchased in Desktop Kit form with a keyboard, a mouse, a case, a power supply, micro-HDMI cables, and a copy of *The Official Raspberry Pi Beginner's Guide*.

https://www.raspberrypi.com/products/raspberry-pi-5/

User-sourced items

Unless supplied in a Desktop Kit, the user will need to source their own power supply, HDMI cable, mouse, keyboard, and SD card. Raspberry Pi Ltd sells all of these items globally through our Approved Reseller network.

https://www.raspberrypi.com/resellers

Purchasable third-party items

None.

Additional hardware references

None.

Setting up your development environment

Tool installation (IDEs, toolchains, SDKs):

1 IDE-based
   a Raspberry Pi Ltd recommends Microsoft Visual Studio Code (VSCode) for software development on Raspberry Pi Ltd devices, but this is not mandatory — users can use whichever integrated development environment (IDE) or development system they prefer
   b If you are installing on a Raspberry Pi Ltd device, use the following:

```
Code
sudo apt update
sudo apt install code
```

   c On Windows or Linux devices, please follow the standard VSCode installation instructions: https://code.visualstudio.com/download
   d You may find it useful to install VSCode extensions for Python
2 CLI-based:
   a The example code described here is written in Python and can easily be run from the command line
3 Python 3 is required, along with the AWS Python bindings

Creating a project folder and a Python virtual environment

The latest versions of Python and Raspberry Pi OS require you to use virtual environments, into which the Python packages needed for the applications are installed.

To create a project folder and a Python virtual environment, do the following:

```
Code
mkdir Pi5AWSExample
cd Pi5AWSExample

# Create the virtual environment
python3 -m venv venv

# Activate it
source venv/bin/activate

# Update everything
pip install --upgrade pip

# Install AWS-specific libraries
pip install AWSIoTPythonSDK
```

Installing the example application

The example application is available from Raspberry Pi Ltd's GitHub repo. Use the following command to clone the code to a pre-existing folder named `Pi5AWSExample` :

```
Code
cd Pi5AWSExample
git clone https://github.com/raspberrypi/rpi-aws-examples.git
```

In a later section, we will edit the example Python code to customise it with the user's AWS 'thing' name, security keys, and topic names.

Additional software references

Technical support for Raspberry Pi Ltd devices can be found on the Raspberry Pi Ltd forums at https://www.raspberrypi.com/forums.

Setting up your hardware

Full instructions for setting up a Raspberry Pi 5 can be found on the Raspberry Pi Ltd website.

https://www.raspberrypi.com/documentation/computers/getting-started.html

Setting up your AWS account and permissions

If you do not have an existing AWS account, refer to the online AWS documentation at Set up AWS account. To get started, follow the steps outlined in the sections below:

- Sign up for an AWS account
- Create a user with administrative access
- Open the AWS IoT console

Pay special attention to the note sections.

AWS terminology

Getting to grips with Amazon Web Services terminology and permissions is vital to ensuring everything works as expected. The following section explains the various concepts and naming schemes involved.

ARNs, policies, permissions, and roles

ARNs

An Amazon Resource Name (ARN) is a unique identifier for an AWS resource, such as an EC2 instance, an S3 bucket, or an IAM user. It's a string that enables you to manage and control access to these resources within AWS services, including IAM policies, API calls, etc.

Policies

> In AWS, a policy is a document/object that defines permissions for an entity (user, group, or role) or resource. It dictates what actions an entity can perform on which AWS resources. These policies are crucial for managing access to AWS services and ensuring security within the AWS environment.
>
> — Amazon

How it works

Policies control access to AWS resources by specifying which actions are allowed or denied.

Policies are typically written in JSON format and consist of statements defining the effect (allow or deny), actions, resources, and conditions of permissions.

There are different types of policies, including:

- Identity-based policies, which are attached to users, groups, or roles and control what those identities can do
- Resource-based policies, which are attached to a specific resource (like an S3 bucket) and control who can access that resource

Permissions

> In AWS, a permission defines what actions a user, role, or service can perform on specific AWS resources. It's a grant of access, allowing or denying specific operations based on defined rules.
>
> — Amazon

How it works

Permissions are defined using identity and access management (IAM) policies, which specify which actions are allowed or denied. IAM is the AWS service responsible for managing permissions.

Roles

> An AWS IAM role is an AWS identity that you create to grant permissions to trusted entities to access AWS resources, without needing to create separate credentials for each entity. It's like a set of permissions that can be assumed by users, applications, or services, to temporarily access specific resources within your AWS environment.
>
> — Amazon

How it works

Instead of assigning long-term credentials to users or applications, you create a role with specific permissions and allow trusted entities to assume that role temporarily. Unlike users, roles do not have passwords or access keys associated with them. Instead, roles provide temporary security credentials to whoever has the ability to assume that role.

Things

An AWS 'thing' is a device or a virtual entity that is registered and managed within the IoT platform. It's a way to model and track physical devices — such as sensors or actuators — or even abstract entities within the AWS ecosystem. Each thing has associated certificates and policies that define its interactions with AWS IoT services.

So, a Raspberry Pi SBC could be an AWS IoT thing. You can use the AWS console to give a thing a name and generate a unique certificate for that thing. These certificates need to be downloaded on the thing itself, as they uniquely identify it.

Note

When you generate a certificate set on the console, you must download it straight away — you cannot go back and download them later.

When the certificate is generated, the AWS console also provides a set of four keys: one private, one public, and two Amazon Root certificates, which are Certificate Authority blocks. See https://docs.aws.amazon.com/privateca/latest/userguide/PcaTerms.html

During the creation of an AWS thing, you also assign a policy to it, which is actually attached to the certificate. This policy defines what the thing is able to do (see above).

Creating resources in AWS IoT

Making a thing object

Note

All of the following operations can be carried out on Raspberry Pi 5 itself.

Rather than outlining how to do this here, please refer to the AWS documentation on creating a thing object. https://docs.aws.amazon.com/iot/latest/developerguide/create-iot-resources.html#create-aws-thing

In short, you should create a new thing with the name `Pi5AWSExample` , select 'No shadow', and then select 'Auto-generate a new certificate'. You do not need to attach a policy.

Warning

During the creation process, you will be asked to download the certificate and keys. This is the only time that option is available, so make sure all of the keys are downloaded to your Raspberry Pi 5. You will need to add these keys to the example application.

Provisioning the device with credentials

As this is simple example software, there are a few things you need to manually do to get it up and running. The main thing is to add the credentials (keys) downloaded in the previous section to the example code. These items are hardcoded into the example, so they need to be updated to the user's specific keys.

1 Copy all of the downloaded keys to where the example application was cloned (these can have rather long file names consisting of seemingly random letters)

2 Edit the program, filling in the following entries with the updated information

Entry	Content
THING_NAME	Pi5AWSExample
PRIVATE_KEY_PATH	xxxxx-private.pem.key
CERTIFICATE_PATH	xxxxx-certificate.pem.crt
IOT_ENDPOINT	The endpoint is found in the test suite data (See the 'Run the example code' section below)

The 'xxxxx' prefix represents the lengthy hexadecimal file names of the downloaded keys.

Creating a Device Advisor test suite

From the AWS IoT console, select 'Device Advisor/Test Suites', then select 'Create test suite'. Choose 'AWS IoT Core qualification test suite' for the test suite type, and 'MQTT 3.1.1' as the protocol, then click 'Next'. You will now be presented with a page titled 'Create test suite', which contains the tests needed for the Device Advisor qualification. Click 'Test suite properties' and give the test suite an appropriate name — for our example, we use 'Device Advisor Suite'. Click 'Next'. You now need to select a role; this is how you grant permission for AWS to use your certificates and such during Device Advisor testing. For our example, we'll create a new role, so select 'Create new role'.

Creating a role

We need to specify access available to the Device Advisor.

For the 'Connect' option, enter the thing name that was specified when we created the Raspberry Pi 5 thing. In our example, this was `Pi5AWSExample` .

For 'Publish', 'Subscribe', 'Receive', and 'RetainPublish', we will enter a wildcard topic: `Pi5AWSExample*` , which allows any topics starting with `Pi5AWSExample` to be validated.

Building the example code

As the software is written in Python, there is no need for any compilation.

Running the example code

Start the test suite and run the example code on your Raspberry Pi 5. The SBC will then communicate with the test suite.

From the AWS IoT console, select ‘Device Advisor/Test Suites’, then select your newly created test suite by clicking on its name. On the subsequent screen, select ‘Actions’, then ‘Run test suite’.

Select the AWS thing you want to test — in our case, `Pi5AWSExample` . In the ‘Test endpoint’ section, select ‘Account-level endpoint’.

Important

You need to copy the endpoint information from the ‘Test endpoint’ section into the Python example application (see the IOT_ENDPOINT configuration item in the example above). This tells the example application where to find the Device Advisor tests. As this endpoint is fixed to the test suite, it does not change, so you only need to do this once.

Once the example code is updated with the endpoint, you can run the test suite by clicking ‘Run test’. This will start up the Device Advisor test suite. You now need to run the example code in order to communicate with the test suite.

Code

```
# cd to the location of your cloned example repository.
cd Pi5AWSExample/rpi-aws-examples
# Run the example
python3 Pi5AWSExample
```

After a few minutes — if all is working correctly — the Device Advisor test suite will display the results of the test.

Verifying messages in AWS IoT Core

You can examine the logging done during the test by selecting the test case log links next to each part of the test suite. These are on the qualification results page displayed while the test is running. These logs display all of the messages that were exchanged between Raspberry Pi 5 and AWS, and can be very useful if you find that some tests are not passing.

Debugging

As a Raspberry Pi 5 running Raspberry Pi OS is a full Linux-based system (rather than a dedicated and targeted device), debugging is considerably easier. All of the features are available and easily accessible when working with the Raspberry Pi OS desktop.

Device console output is simply viewed in the terminal windows used to start the example software, and it is easy to modify the example code in place (by adding commands like debugging print statements) to help with debugging. All of the standard Linux logging is available, and you can use VSCode's Python debugger, for example, to debug the Python example code.

Applications developed in C/C++ (not shown here) can also use IDEs like VSCode to carry out development and debugging on the device.

Troubleshooting

Incorrect permissions are almost always the reason why programs may not work as intended, or at all. Remember to double-check your roles, permissions, policies, and device names.

Please use the Raspberry Pi Ltd forums https://forums.raspberrypi.com for technical support from the community — this is usually the fastest way to get help. You can also get in touch with Raspberry Pi Ltd's applications team at applications@raspberrypi.com.

Conclusions

This document describes how to get a simple AWS application up and running and how to pass the Device Advisor certification tests. It shows that Raspberry Pi Ltd's SBCs are an effective way to provide IoT compute power that can easily leverage AWS to provide cloud-based back-end services.

The example application is extremely basic, and is there to show the principles of connecting, publishing, and subscribing. Users will need to develop their own applications targeted to their specific use case. Although the example application uses Python, AWS also provides a C/C++ SDK and other language bindings for application development, and the user should consider which option is most appropriate for them.

Although testing was carried out on a Raspberry Pi 5, there is no reason why earlier models running the same Raspberry Pi OS system would not work in the same way. Users should choose the device that is most appropriate for their application.

Contact Details for more information

Please contact applications@raspberrypi.com if you have any queries about this whitepaper.

Web: www.raspberrypi.com

Raspberry Pi

Raspberry Pi Ltd