

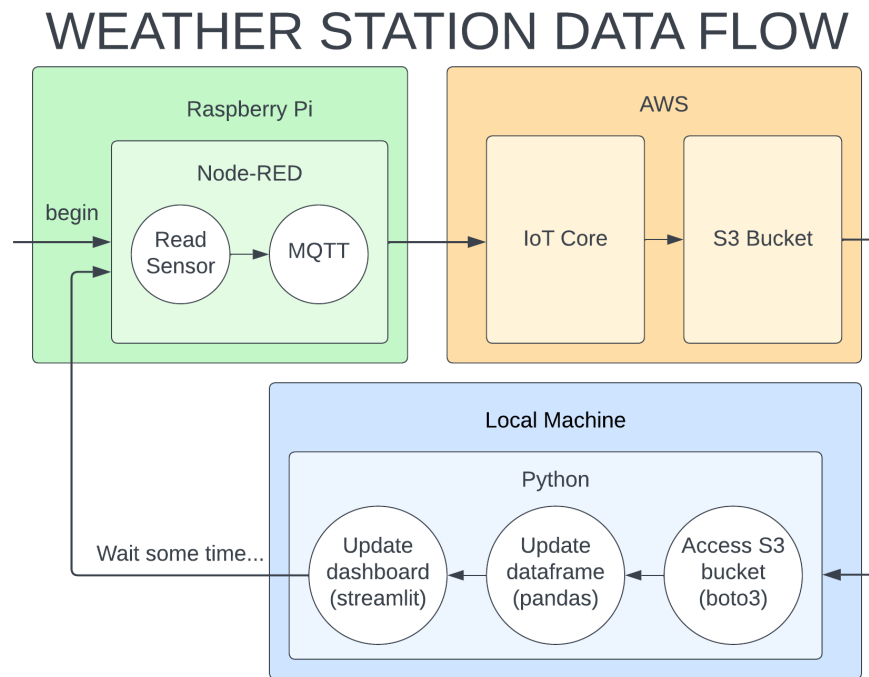


CPE 4040: DATA COLLECTION AND ANALYSIS

Lab 7: A Data Project Using NodeRED, AWS IoT, and Streamlit

Learning Objectives:

1. To build an end-to-end data project that collects sensor data and creates a web dashboard.
2. To learn how to interface sensor data with AWS IoT using Node-RED on a Raspberry Pi.
3. To learn how to create and store data in the AWS S3 buckets.
4. To develop a Streamlit application that can display the sensor data in a web dashboard.



AWS S3 (Simple Storage Service):

Amazon S3 is an *object* storage service that allows users to store files and any metadata that describe the files. A *bucket* is a container that stores the objects. To store an object in Amazon S3, you create a bucket and then upload the object to a bucket.

- [What is Amazon S3?](#)
- [Creating, configuring, and working with Amazon S3 buckets](#)

Streamlit:

Streamlit is an open-source app framework written in Python. It allows programmers to easily create web application for data visualization.

- [Streamlit Website](#)
- [Streamlit Tutorial](#)

Hardware and Software Requirement:

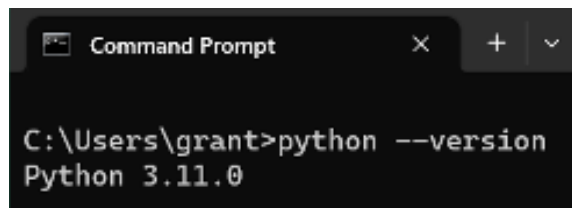
1. Temperature sensor (DHT-11, DHT-22, or DHT-20)
2. Resistors and bread board
3. Python scripts (available in D2L Assignment)

Lab Procedure:

1. Plug the power adapter into the Raspberry Pi and power it up. Next, open Remote Desktop Connection or SSH connection on your laptop and connect to the Raspberry Pi.

Section 1: Hardware and Software Setup

2. Open a terminal window on your computer and type “*python --version*” to make sure you have Python 3 installed. If not, go to <https://www.python.org/downloads/> and download the proper release.



```
Command Prompt
C:\Users\grant>python --version
Python 3.11.0
```

3. Install the following packages on your computer, if they are not already installed.

Note: You will research on how to install those packages. The TA is available for support.

- `Boto3`: an AWS Python Software Development Kit (AWS) to create, configure, and manage AWS services. We use Boto3 to interact with AWS S3.
 - `streamlit`: will also install pandas and NumPy, if they are not installed.
 - `plotly`: a plotting library used by Streamlit.
4. Go to <https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html> and follow the instruction to download the AWS CLI (Command Line Interface) to your computer. After downloading and installing, type “*aws --version*” in your terminal to make sure it is installed properly. You may need to open a new terminal window for the command to be recognized.
 5. Follow the instructions from Lab 4 to connect your temperature sensor to the Raspberry Pi.
 6. Make sure the sensor is working properly using the *digitalOut.py* code from Lab 4.

Section 2: Configuring AWS IoT Core

Note: This section repeats the steps from Lab 6, so it should be familiar. The only difference is

we are now creating a thing for the temperature sensor, so keep that in mind when naming your thing and policies.

7. Log into your Amazon AWS account. Once you are logged in, navigate to the AWS Management Console. From there, search and open **IoT Core** under **Services**.
8. In the AWS IoT Core page, select **Connect One Device** from the **Connect** section in the menu pane. Follow the instructions to configure your Thing as you did in Lab 6. Ensure you unzip the file containing the certificates onto your Pi, and verify that your SDK is set to Python.
9. In the AWS IoT Core page, under **Security** click on **Policies**, and select the policy named <Your thing's name>-Policy. Click the **Edit Active Version** button, and configure **Policy Effect** to **Allow**, and configure **Policy Action** and **Policy Resource** to the wildcard (*). Remember to check the box in **Policy Version Status** to save these changes, then you can click the orange Save as new version button.

10. To attach the policy to your certificate, go to “Certificates” (under “Security”) in the AWS IoT Core page. Click the certificate you created and choose **Attach policy** from the **Actions** menu. Select the policy from the list and click **Attach policies**. Now AWS IoT is ready to receive data from the device.
11. Go back to the Raspberry Pi and create a folder named “cert” in your application folder for this lab. Transfer the device certificate, private key, and root CA files into the “cert” folder. Before transferring, you can rename the files as follows:
 - Device certificate: RaspberryPi-cert.pem

- Private key: RaspberryPi-private.pem.key
- Root CA: RootCA1.pem

Section 3: Setting up AWS S3 Bucket

- Go to the AWS Management Console (<https://console.aws.amazon.com/>). Using the **Search** bar, look for and select **S3**.
- On the S3 page, click **Create bucket**.



- Enter a bucket name. **Note:** It must be a globally unique name with no uppercase letters or spaces. Under **Object Ownership** click **ACLs enabled**. Under **Block Public Access settings for this bucket**, uncheck the box **Block all public access**. Now, click **Create bucket**, then click the button to acknowledge your bucket will be public, and confirm.

General configuration

Bucket name

Bucket name must be globally unique and must not contain spaces or uppercase letters. [See rules for bucket naming](#)

AWS Region

US East (N. Virginia) us-east-1

Copy settings from existing bucket - *optional*
Only the bucket settings in the following configuration are copied.

[Choose bucket](#)

Object Ownership Info

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

☐ ACLs disabled (recommended)

All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

☒ ACLs enabled

Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

Block Public Access settings for this bucket

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

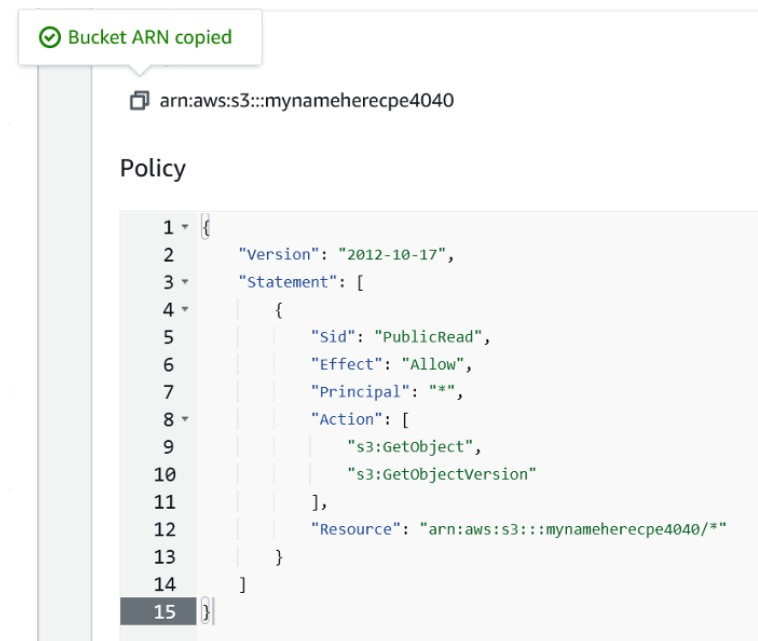
☐ **Block all public access**

Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

- Select the newly created bucket. Click on the **Permissions** tab. Next, click the edit button next to **Bucket policy**. Copy the following text into the policy text box. Click the copy button next to Bucket ARN and paste it into the resource field of the policy. Make sure to preserve

the “/” at the end. Then, **Save changes**.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PublicRead",
      "Effect": "Allow",
      "Principal": "*",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion"
      ],
      "Resource": "arn:aws:s3:::<Paste-Your-Bucket-Name-Here>/*"
    }
  ]
}
```



16. Scroll down to the CORS section and click the edit button. Paste the following text in the text box and **Save changes**.

```
[
  {
    "AllowedHeaders": [
      "Authorization"
    ],
    "AllowedMethods": [
      "GET"
    ],
    "AllowedOrigins": [
      "*"
    ],
    "ExposeHeaders": [],
    "MaxAgeSeconds": 3000
  }
]
```

17. Under **Access Control List**, click **Edit**. Click the check boxes to give the **List** and **Read** permissions to **Everyone (Public Access)**. Then click the box to acknowledge your bucket is public and **Save changes**.

Grantee	Objects	Bucket ACL
Bucket owner (your AWS account) Canonical ID: 33070124fe3c4ef026d0652fe3a8bc9643be449871b8a805bd63e6ae86803d13	<input checked="" type="checkbox"/> List <input checked="" type="checkbox"/> Write	<input checked="" type="checkbox"/> Read <input checked="" type="checkbox"/> Write
Everyone (public access) Group: http://acs.amazonaws.com/groups/global/AllUsers	<input checked="" type="checkbox"/> List <input type="checkbox"/> Write	<input checked="" type="checkbox"/> Read <input type="checkbox"/> Write
Authenticated users group (anyone with an AWS account) Group: http://acs.amazonaws.com/groups/global/AuthenticatedUsers	<input type="checkbox"/> List <input type="checkbox"/> Write	<input type="checkbox"/> Read <input type="checkbox"/> Write
S3 log delivery group Group: http://acs.amazonaws.com/groups/s3/LogDelivery	<input type="checkbox"/> List <input type="checkbox"/> Write	<input type="checkbox"/> Read <input type="checkbox"/> Write

Warning: When you grant access to the Everyone or Authenticated users group grantees, anyone in the world can access the objects in this bucket.
[Learn more](#)

☒ I understand the effects of these changes on my objects and buckets.

18. Navigate to **IoT Core** in AWS. On the sidebar menu, click **Rules** under **Message Routing**. Then click **Create rule**. Give the rule a name then click **Next**.

Rule properties

Rule name
to_s3_bucket
Enter an alphanumeric string that can also contain underscore (_) characters, but no spaces.

Rule description - optional
Enter a description to provide additional details about the rule to others.
A description of your new rule

Tags - optional

Cancel Next

19. In **SQL Statement** text box, enter the following query:

```
SELECT *, timestamp() AS timestamp FROM '<Your-incoming-IoT-Topic-Here>'
```

This will automatically add a timestamp to the JSON payload we send from NodeRED later. Make sure to change the topic name to something appropriate (i.e. “sensor”). Then click **Next**.

20. From the dropdown menu under **Rule actions**, search for and select **S3 bucket**. Next to Bucket name, click the **Browse S3** button and select the bucket you created. Under **Key**, enter a name. This “key” refers to a value in a “key-value” pair. You can choose any value for the key, but make sure to take note of what you choose.



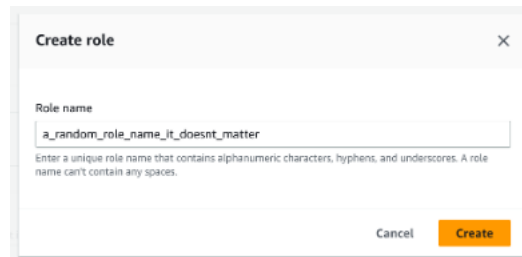
Rule actions
Select one or more actions to happen when the above rule is matched by an inbound message. Actions define additional activities that occur when messages arrive, like storing them in a database, invoking cloud functions, or sending notifications. You can add up to 10 actions.

Action 1
▼ S3 bucket
Store a message in an Amazon S3 bucket
Remove

Bucket name: info
S3 URL: s3://mynameherecpe4040
View Browse S3

Key
The S3 key for this message.
s3Key

21. Go to **IAM Role** and select **Create new role** to begin creating a new role. Enter a name for the new role, then click **Next** and **Create** to complete the process.

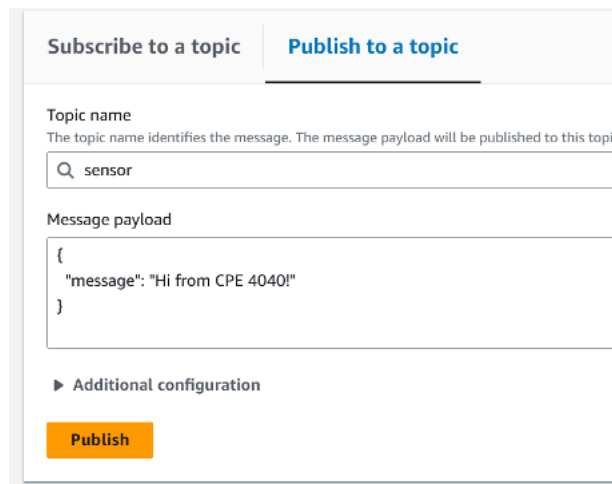


Create role X

Role name
a_random_role_name_it_doesnt_matter
Enter a unique role name that contains alphanumeric characters, hyphens, and underscores. A role name can't contain any spaces.

Cancel Create

22. In the sidebar menu, click **MQTT test client** under **Test**. Click the **Publish to a topic** tab. Enter the topic name you entered previously (i.e. ‘sensor’), and enter a custom message in the message field. Then click **Publish**.



Subscribe to a topic **Publish to a topic**

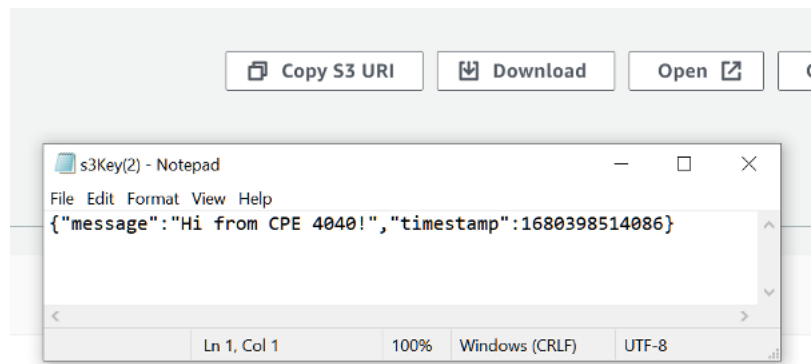
Topic name
The topic name identifies the message. The message payload will be published to this topic.
Q sensor

Message payload
{
"message": "Hi from CPE 4040!"
}

► Additional configuration

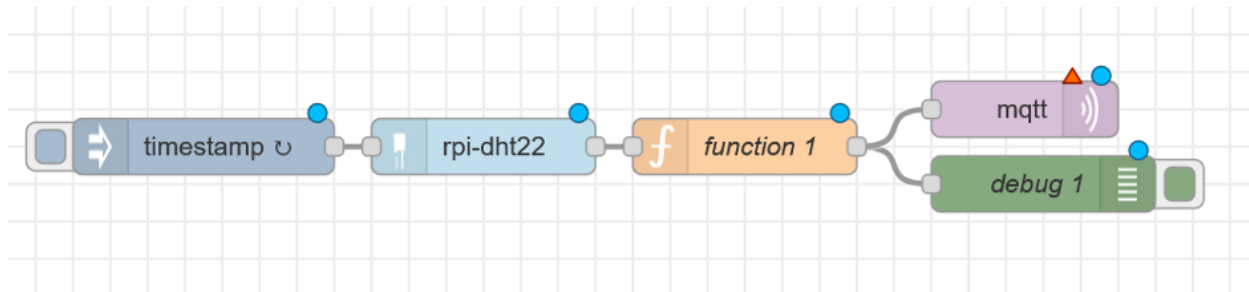
Publish

23. In a separate tab, navigate back to the S3 page. Click on your bucket, then click on the Key object that you created. In the upper right click the **Download** button and open the file that is downloaded. You should see the message that you just sent, along with a timestamp.



Section 4: Set Up NodeRED Flow to Send Sensor Data to S3 Bucket via MQTT

24. On your Raspberry Pi, start NodeRED and open the flow editor in a web browser. Please refer back to the Lab 5 assignment for detailed instructions.
25. In the Palette Manager, install **node-red-contrib-dht-sensor**. If you are using DHT-20, you will install **node-red-contrib-aht20** instead.
26. Create a flow by adding an **inject** node, **rpi-dht22** (or **aht20**) node, **function** node, **mqtt out** node, and **debug** node and connect them as shown.



27. Edit the inject node to activate every 5 seconds.

Edit inject node

Delete Cancel Done

Properties

Name Name

msg. payload = timestamp

msg. topic = a_z

+ add inject now

☐ Inject once after 0.1 seconds, then

Repeat interval

every 5 seconds

☐ Enabled

28. Edit the **rpi-dht** node to use GPIO pin 12. For **ah20**, nothing needs to be done.

Edit rpi-dht22 node

Delete Cancel Done

Properties

Topic rpi-dht22

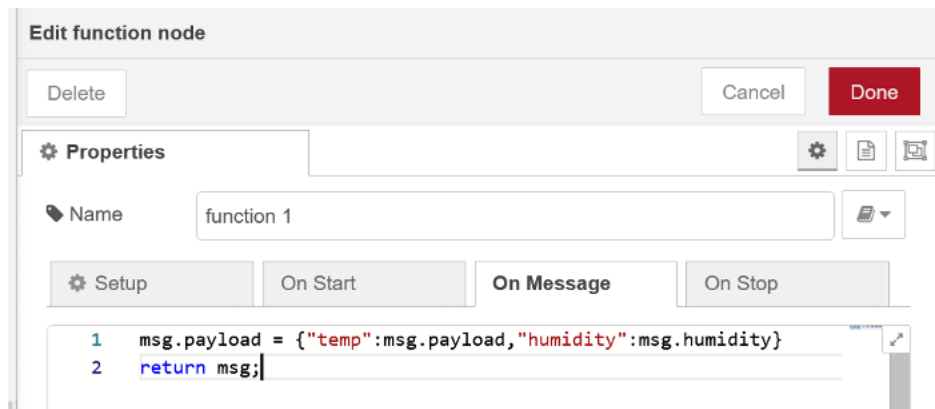
Sensor model DHT22

Pin numbering BCM GPIO

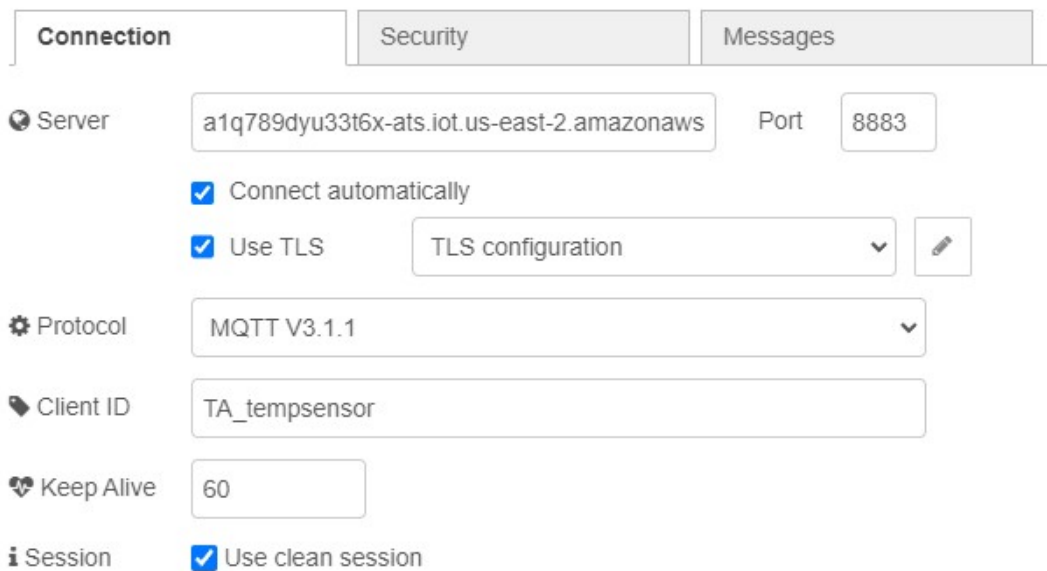
Pin number 12

Name Name

29. Edit the function node and add the following code.



30. Edit the MQTT out node. Click the edit button next to server to add a new MQTT broker. Add the URL from AWS IoT -> Settings -> Device Data endpoint (refer back to Lab 6, Step 16). Add the device name in the Client ID field. Change Port to 8883.



31. Click button to Use TLS. Edit the configuration. Check the “Use key and certificates from local files” and copy the filepath to each of the certificate files from your Pi into the textboxes.

Properties

☒ Use key and certificates from local files

Certificate: /home/raspberry/Lab7/cert/RaspberryPI-ci

Private Key: /home/raspberry/Lab7/cert/RaspberryPI-pi

Passphrase: private key passphrase (optional)

CA Certificate: /home/raspberry/Lab7/cert/RootCA1.pem

☒ Verify server certificate

Server Name: for use with SNI

ALPN Protocol: for use with ALPN

Name: Name

32. Add a topic in the topic field. Make sure to use the same topic as you configured in the S3 bucket.

Edit mqtt out node

Delete Cancel Done

Properties

Server: weather@a1yqbxumcgt31w-ats.iot.us-eas

Topic: sensor

QoS: [dropdown] Retain: [dropdown]

Name: Name

Tip: Leave topic, qos or retain blank if you want to set them via msg properties.

33. Deploy the flow and make sure all nodes are connected.

Section 5: Sending Data in S3 Bucket to Streamlit

34. Go to the AWS Management Console (<https://console.aws.amazon.com/>). Using the **Search** bar, look for and select **IAM**, then click **Users** in the IAM sidebar menu. From here, click **Add Users**, enter a name for the user, and click **Next**.

Specify user details


User details

User name

testuser

The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = , . @ _ - (hyphen)

☐ Provide user access to the AWS Management Console - *optional*
If you're providing console access to a person, it's a [best practice](#) to manage their access in IAM Identity Center.

 If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate them after you create this IAM user. [Learn more](#)

CancelNext

35. Go to **Permissions Boundary**, select the checkbox, then search for **AmazonS3ReadOnlyAccess** in the filter box. Once selected, click "Next" and "Create User" to complete the process.


▼ **Permissions boundary** - *optional*

Set a permissions boundary to control the maximum permissions for this user. Use this advanced feature used to deleg

☒ Use a permissions boundary to control the maximum permissions
You can select one of the existing permissions policies to define the boundary.

Permissions policies (1/1068)

Select policy to set the permissions boundary.

 Filter distributions by text, property or value


1 match


AmazonS3ReadOnly


×

Clear filters

Policy name





 AmazonS3ReadOnlyAccess

36. Select your newly created user, and click the **Security Credentials** tab. Under **Access Keys**, click **Create access key**.

testuser

Summary

ARN

 arn:aws:iam::122870662461:user/testuser

Created

April 03, 2023, 18:13 (UTC-04:00)

Permissions

Groups

Tags

Security credentials

Access Advisor

Access keys (0)

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. [Learn more](#)

Create access key

No access keys

As a best practice, avoid using long-term credentials like access keys. Instead, use tools which provide short term credentials. [Learn more](#)

Create access key

37. From **Access key best practices & alternatives** page, choose the “Local code” option and click **Next** then **Create access key**. Copy the resulting **Access key** and **Secret access key** into a text file. **Important:** You will not be able to see the secret access key again, make sure you copy it now.

38. In your Windows/Mac command terminal, type “**aws configure**”

39. When prompted, enter your access and secret access keys. You may skip the other prompts by pressing Enter without typing anything. This creates a credentials file that will be used by the script. To make sure it was created, go to C:\Users\[user]\.aws to see the file. On Mac, it should be in /Users/[user]/.aws.

Note: Make sure that no additional spaces or characters are added when pasting in these keys. It could cause some annoying errors to debug later!

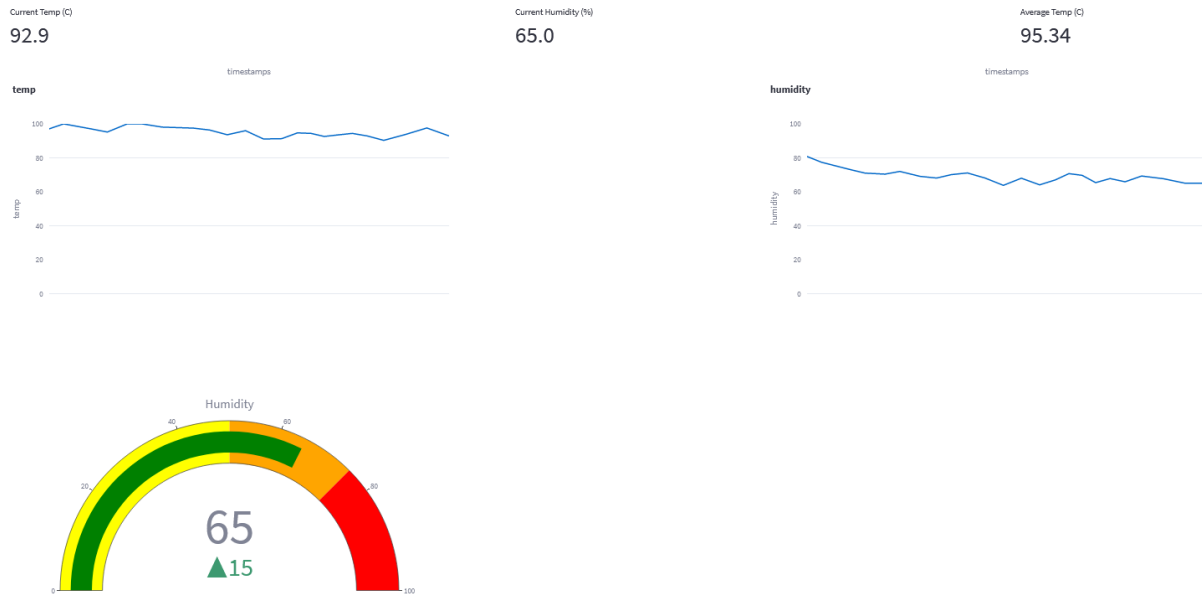
40. Download the **app.py** script from D2L.

41. Open **app.py** in a text editor and edit the following lines with the name of your S3 bucket and the name of your object key.

```
AWS_S3_BUCKET = "MY_BUCKET" #edit this
AWS_S3_KEY_PREFIX = "MY_KEY_PREFIX" #edit this NOTE: refers to s3 object key
#not access key
```

42. To view your dashboard, run the script with the command '**streamlit run app.py**' and open the URL provided in a web browser. Once you've done this, you should see two line charts and a gauge on your dashboard, similar to ones shown below.

CPE 4040 Weather Station



Lab Report

In the report, you will show screenshots of setup and results during critical steps in:

- Hardware and Software Setup
- Configuring AWS IoT Core
- Setting up AWS S3 Bucket
- Setting Up NodeRED Flow
- Sending Data in S3 Bucket to Streamlit

Attach your modified **app.py** code as well.

Use the lab report template and submit the report via D2L Assignment, in PDF format.