Task 2: Creating a Kinesis data stream

In this task , we will create a Kinesis data stream by providing the name, capacity and other configuration.

1. Make sure you are inthe **US East (N. Virginia) us-east-1** Region.
2. Navigate to **Kinesis** by clicking on the **Services** menu, under the **Analytics** section.
3. Under **Get Started**, select **Kinesis Data Streams** and click on **Create data stream**button.

A picture containing text, screenshot, font

Description automatically generated

1. Under **Data stream name**, enter the name **whiz-data-stream**.
2. Under Capacity mode : select **Provisioned**
3. Under **Provisioned shards**, enter **1**.

A screenshot of a computer

Description automatically generated

1. Click on **Create data stream**button.

A screenshot of a computer

Description automatically generated with low confidence

1. Once the data stream is created, click to open it.
2. Click on the **Configuration** tab.
3. Scroll down to **Encryption** and click on **Edit**.
4. Check **Enable server-side encryption** and use the default encryption key type, i.e **Use AWS managed CMK**.
5. Click on **Save changes**.

A screenshot of a computer

Description automatically generated with medium confidence

1. You have used AWS KMS to encrypt your data.

A screenshot of a computer

Description automatically generated with medium confidence

Task 3: Creating an S3 Bucket

In this task, we are going to create an S3 bucket by providing details about name, versioning and encryption.

1. Make sure you are inthe **US East (N. Virginia) us-east-1** Region.
2. Navigate to **S3** by clicking on the **Services** menu, under the **Storage** section.
3. Click on **Create bucket**button.
4. In the General Configuration,
   * **Bucket name** :Enter **whiz-datasource<RANDOM-NUMBER>**.
   * **Note:** S3 Bucket names are globally unique, choose a name that is available.
5. **Region**:Select **US East (N. Virginia) us-east-1** (i.e same region as the Kinesis data stream).
6. In the **Bucket Versioning,** Check the option **Enable.**
7. In the Default encryption,

* **Server-side encryption**: Select **Enable**
* **Encryption key type**: Leave the key type as **Amazon S3-managed key (SSE-S3)**.

1. Click on **Create bucket** button.

A screenshot of a computer

Description automatically generated with medium confidence

Task 4: Creating producer Lambda function

Let us create 3 lambda functions. One function for the producer and the other two for consumers.

1. Make sure you are inthe **US East (N. Virginia) us-east-1** Region.
2. Navigate to **Lambda** by clicking on the **Services** menu, under the **Compute** section.
3. Click on the **Create function**button.
   * Choose **Author from scratch**
   * **Function name**: Enter **producer**
   * **Runtime:** Select **Node.js 14.x**

A screenshot of a computer

Description automatically generated with low confidence

* + Under the **Permissions** section, click on **Change default execution role** and then choose **use an existing role**.
  + **Existing role:** Select **lambda\_Role\_<RANDOM\_NUMBER>**

A screenshot of a computer

Description automatically generated with medium confidence

* + Click on **Create function** button.

1. **Configuration Page:** On this page, we need to configure our lambda function.
2. If you scroll down a little bit, you can see the **Code source** section. Here we need to write a NodeJs function which reads the file in S3 and sends the data to kinesis data stream.
3. Remove the existing code in AWS lambda **index.js**. Copy the below code and paste it into your lambda index.js file.

|  |
| --- |
| const AWS = require('aws-sdk');  AWS.config.update({      region: 'us-east-1'  })  const s3 = new AWS.S3();  const kinesis = new AWS.Kinesis();  exports.handler = async (event) => {      console.log(JSON.stringify(event));      const bucketName = event.Records[0].s3.bucket.name;      const keyName = event.Records[0].s3.object.key;      const params = {          Bucket: bucketName,          Key: keyName      }      await s3.getObject(params).promise().then(async (data) => {          const dataString = data.Body.toString();          const payload = {              data: dataString          }          await sendToKinesis(payload, keyName);      }, error => {          console.error(error);      })  };  async function sendToKinesis(payload, partitionKey) {      const params = {          Data: JSON.stringify(payload),          PartitionKey: partitionKey,          StreamName: 'whiz-data-stream'      }      await kinesis.putRecord(params).promise().then(response => {          console.log(response);      }, error => {          console.error(error);      })  } |

1. You need to change the **StreamName** in the index.js file based on your Kinesis data stream name under the function **sendToKinesis**.

A screen shot of a computer program

Description automatically generated with low confidence

1. Save the function by clicking on the **Deploy** button.

Task 5: Creating an event notification

1. Navigate to **S3** by clicking on the **Services** menu, under the **Storage** section.
2. Click on the created S3 bucket and navigate to the **Properties** tab.
3. Scroll down to **Event notifications** and click on **Create event notification**.
4. Under **Create event notification**,
   * **Event name** : Enter **upload-event**
   * Leave the **prefix** as it is.
   * Suffix : Enter **.txt**
5. Under **Event types**, select **All object create events**.
6. Under **Destination**, select **Lambda function**.
7. Under **Specify Lambda function**, select **Choose from your Lambda functions** and choose the **producer** from the list and click on **Save changes** button.

A screenshot of a computer screen

Description automatically generated with low confidence

1. That means whenever an object is created, producer lambda function is triggered.

A screenshot of a computer

Description automatically generated with low confidence

Task 6: Creating consumer Lambda functions

Let us create lambda functions for consumers.

1. Make sure you are inthe **US East (N. Virginia) us-east-1** Region.
2. Navigate to **Lambda** by clicking on the **Services** menu, under the **Compute** section.

Consumer -1

1. Click on the **Create function** button.
   * Choose **Author from scratch**
   * **Function name**: Enter **consumer1**
   * **Runtime**: Select **Node.js 14.x**

A screenshot of a computer

Description automatically generated with medium confidence

* + Under the **Permissions** section, click on **Change default execution role** and then choose **Use an existing role**.
  + **Existing role:** Select **lambda\_Role\_<RANDOM\_NUMBER>**
  + Click on **Create function** button.

1. **Configuration Page:** On this page, we need to configure our lambda function.
2. If you scroll down a little bit, you can see the **Code source** section. Here we need to write a NodeJs function which reads the file in the data stream and processes the data, here we are logging out the data.
3. Remove the existing code in AWS lambda **index.js**. Copy the below code and paste it into your lambda index.js file. Save the function by clicking on the**Deploy** button.

|  |
| --- |
| exports.handler = async (event) => {      console.log(JSON.stringify(event));      for (const record of event.Records) {          const data = JSON.parse(Buffer.from(record.kinesis.data, 'base64'));          console.log('consumer #1', data);      }  }; |

1. On the same page, go to **Configuration** tab and click on **Triggers**
2. Under **Triggers,**  Kinesis trigger will be in a **Disabled**state, select the trigger and click on **Edit** button.**(If trigger is not present , Move to Step 11)**
3. **Check** the **Activate** trigger and click on the **Save** button
4. Now you can see the **Kinesis: whiz-data-stream (Enabled).**
5. In case the trigger is not present already , click on **Add Trigger** button.
   * Select a source dropdown: Select **Kinesis**
   * Kinesis stream : Select **whiz-data-stream**
   * **Check** the **Activate trigger** checkbox.
   * Click on **Add** button.  
     A screenshot of a computer

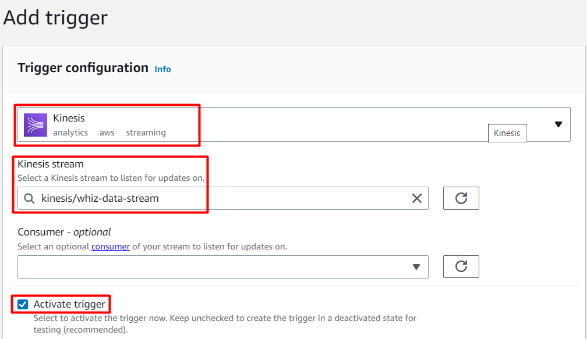
     Description automatically generated with medium confidence

Consumer - 2

1. Click on the **Create function** button.
   * Choose **Author from scratch**
   * **Function name**: Enter **consumer2**
   * **Runtime**: Select **Node.js 14.x**
   * Under the **Permissions** section, click on **Change default execution role** and then choose **use an existing role**.
   * **Existing role:** Select **lambda\_Role\_<RANDOM\_NUMBER>**
   * Click on **Create function** button.
2. **Configuration Page:** On this page, we need to configure our lambda function.
3. If you scroll down a little bit, you can see the **Code source** section. Here we need to write a NodeJs function which reads the file in the data stream and processes the data, here we are logging out the data.
4. Remove the existing code in AWS lambda **index.js**. Copy the below code and paste it into your lambda index.js file.

|  |
| --- |
| exports.handler = async (event) => {      console.log(JSON.stringify(event));      for (const record of event.Records) {          const data = JSON.parse(Buffer.from(record.kinesis.data, 'base64'));          console.log('consumer #2', data);      }  }; |

|  |
| --- |
|  |

1. Save the function by clicking on the **Deploy**button.
2. On the same page, go to **Configuration** tab and click on **Triggers**
3. Under **Triggers,** Kinesis trigger will be in **Disabled**state select the trigger and click on **Edit** button,**( If trigger is not present, move to Step 21.)**
4. **Check** the **Activate trigger** and click on **save** button
5. Now you can see the **Kinesis: whiz-data-stream (Enabled).**
6. In case the trigger is not present already , click on **Add Trigger** button.
   * Select a source dropdown: Select**Kinesis**
   * Kinesis stream : Select **whiz-data-stream**
   * **Check**the **Activate trigger** checkbox.
   * Click on **Add**button.  
     

Task 7: Creating and uploading a test file to S3 bucket

1. Open any text editor on your computer.
2. Copy and paste the following data and save the file in **txt**, in my case **test.txt**

|  |
| --- |
| Hello  This is Whizlabs...  Check out our courses  Bye bye!!! |

1. Navigate to **S3** by clicking on the **Services** menu, under the **Storage** section.
2. Click on the bucket we created earlier.
3. Under the **Objects** tab, click on **Upload**.
4. In the **Files and folders**, click **Add files**.
5. Navigate and select the **test.txt** file created earlier in the task.

A screenshot of a computer

Description automatically generated with medium confidence

1. Once you select the file, click on **Upload** button.
2. Now, click **Close** to close the Upload: status page.

Task 8: Testing the configuration

1. Now, we have uploaded the file to the S3 bucket. Since we have configured the event notification, the producer lambda function should get triggered.
2. Let us test the configuration by checking the logs of lambda functions.
3. Make sure you are inthe **US East (N. Virginia) us-east-1** Region.
4. Navigate to **CloudWatch** by clicking on the **Services** menu, under the **Management & Governance** section.
5. On the left panel, click on **Logs** and select **Log groups**.
6. Here you can see all 3 functions logs.

A screenshot of a computer

Description automatically generated with medium confidence

Producer

1. In the filter log groups, search for the **producer**.

A screenshot of a computer

Description automatically generated with medium confidence

1. Click and open the log group.
2. In the **Log streams**, you will find the latest event, in our case we’ll have only 1 log event.
3. Click and open the log stream. This means our lambda function is triggered successfully.

A screenshot of a computer

Description automatically generated with low confidence

1. Click and expand the event. We can see the event that triggers the lambda function from our S3 bucket.

A screenshot of a computer program

Description automatically generated with low confidence

1. Then the lambda function sends the data to Kinesis and returns a successful message.

A picture containing text, receipt, font, screenshot

Description automatically generated

1. In the producer lambda function, we can see that the S3 is configured as a trigger for the producer lambda function.

A screenshot of a computer

Description automatically generated with medium confidence

Consumer 1

1. Now return to the log groups main menu.
2. In the filter log groups, search for the **consumer1**.

A screenshot of a computer

Description automatically generated with medium confidence

1. Click and open the log group.
2. In the **Log streams**, you will find the latest event, in our case we’ll have only 1 log event.
3. Click and open the log stream. This means the lambda function got executed.

A screenshot of a computer

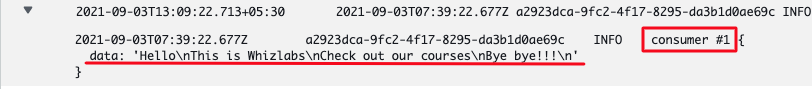
Description automatically generated with low confidence

1. We can see the event object from the Kinesis.

A screenshot of a computer code

Description automatically generated with low confidence

1. We can see from the event that the data is encrypted and encoded.
2. Our lambda function has extracted the data and read it out.



Consumer 2

1. Now return to the log groups main menu.
2. In the filter log groups, search for the **consumer2**.

A screenshot of a computer

Description automatically generated with medium confidence

1. Click and open the log group.
2. In the **Log streams**, you will find the latest event, in our case we’ll have only 1 log event.
3. Click and open the log stream. This means the lambda function got executed.

A screenshot of a search box

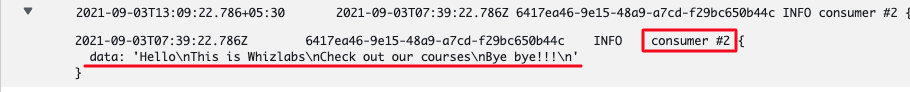
Description automatically generated with low confidence

1. We can see the event object from the Kinesis.

A screen shot of a computer

Description automatically generated with low confidence

1. We can see from the event that the data is encrypted and encoded.
2. Our lambda function has extracted the data and read it out.



Task 9: Delete AWS Resources

Deleting Kinesis Data Streams

1. Make sure you are in the **US East** (**N.Virginia) us-east-1** Region.
2. Navigate to **Kinesis** by clicking on the **Services** menu, under the **Analytics** section.
3. On the left panel, click on the **Data streams**.
4. Select the created data stream and click on the **Actions** button.
5. Select **Delete** from the drop-down.
6. Confirm by typing **Delete** and click .



**Completion and Conclusion**

1. You have created an IAM Role for the Lambda functions.
2. You have created a S3 Bucket and created an event notification.
3. You have created lambda functions for a producer and two consumers.
4. You have added the triggers for the consumer lambda functions.
5. You have uploaded a text file to the S3 bucket.
6. You have tested the configuration by checking the cloudwatch logs of all the 3 lambda functions.

**End Lab**