AWS Lambda Learning part 2

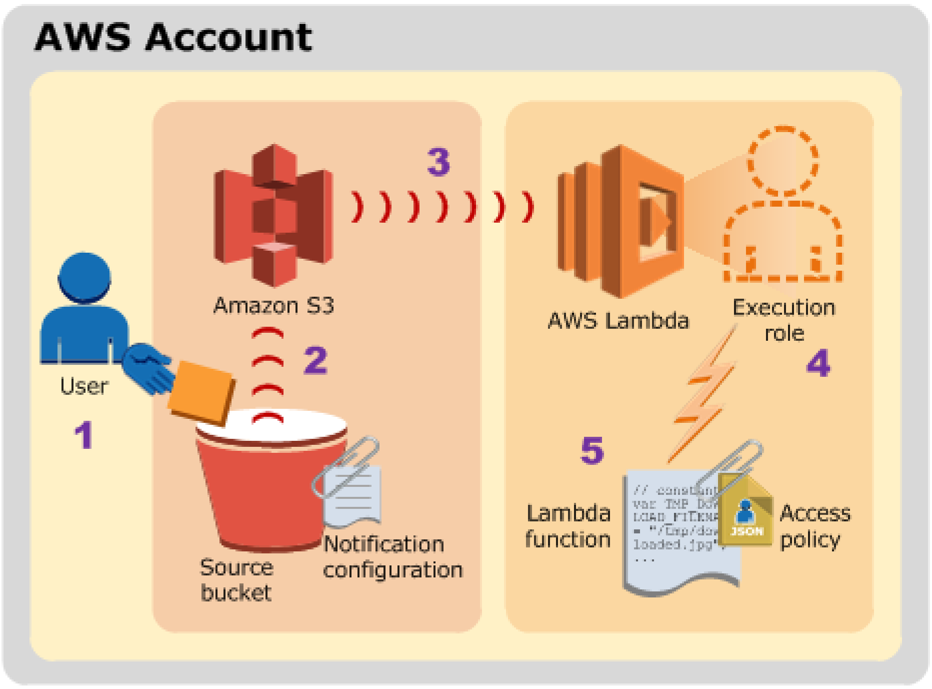
Reference: http://docs.aws.amazon.com/lambda/latest/dg/lambda-app.html

5. Building Applications with AWS Lambda

When building applications on AWS Lambda, including serverless applications, the core components are Lambda functions and event sources. An event source is the AWS service or custom application that publishes events, and a Lambda function is the custom code that processes the events. There are some scenarios listed below:

- File processing

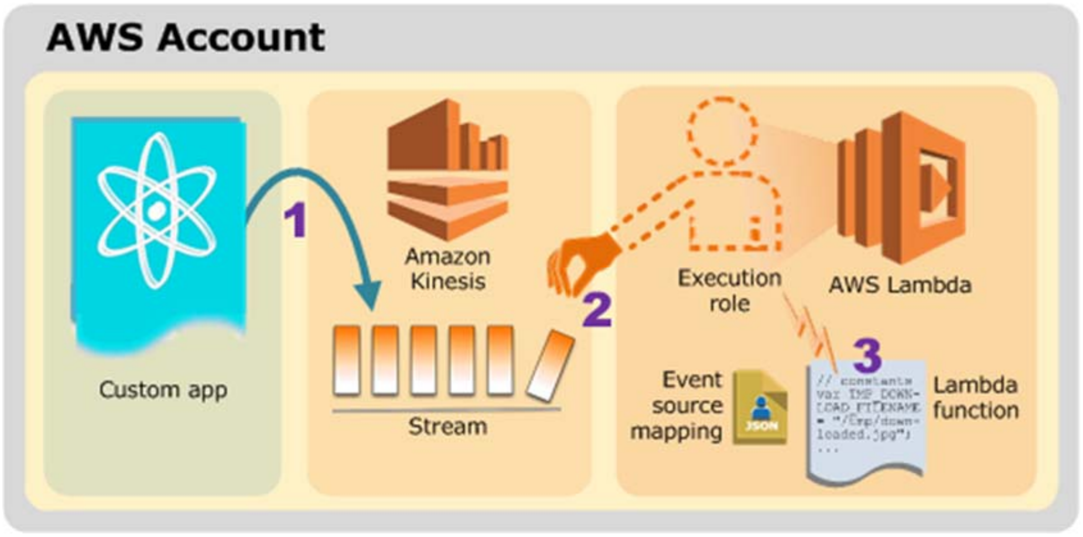
  Suppose we have a photo sharing application. People use our application to upload photos, and the application stores these user photos in an Amazon S3 bucket. Then, our application creates a thumbnail version of each user's photos and displays them on the user's profile page. In this scenario, we may choose to create a Lambda function that creates a thumbnail automatically. Amazon S3 is one of the supported AWS event sources that can publish object-created events and invoke our Lambda function. Our Lambda function code can read the photo object from the S3 bucket, create a thumbnail version, and then save it in another S3 bucket.



  a) User uploads an object to an S3 bucket (object-created event).
  b) Amazon S3 detects the object-created event.
  c) Amazon S3 invokes a Lambda function that is specified in the bucket notification configuration.
  d) AWS Lambda executes the Lambda function by assuming the execution role that we specified at the time we created the Lambda function.
  e) The Lambda function executes.

- Data and analytics

Suppose we are building an analytics application and storing raw data in a DynamoDB table. When we write, update, or delete items in a table, DynamoDB streams can publish item update events to a stream associated with the table. In this case, the event data provides the item key, event name (such as insert, update, and delete), and other relevant details. We can write a Lambda function to generate custom metrics by aggregating raw data.



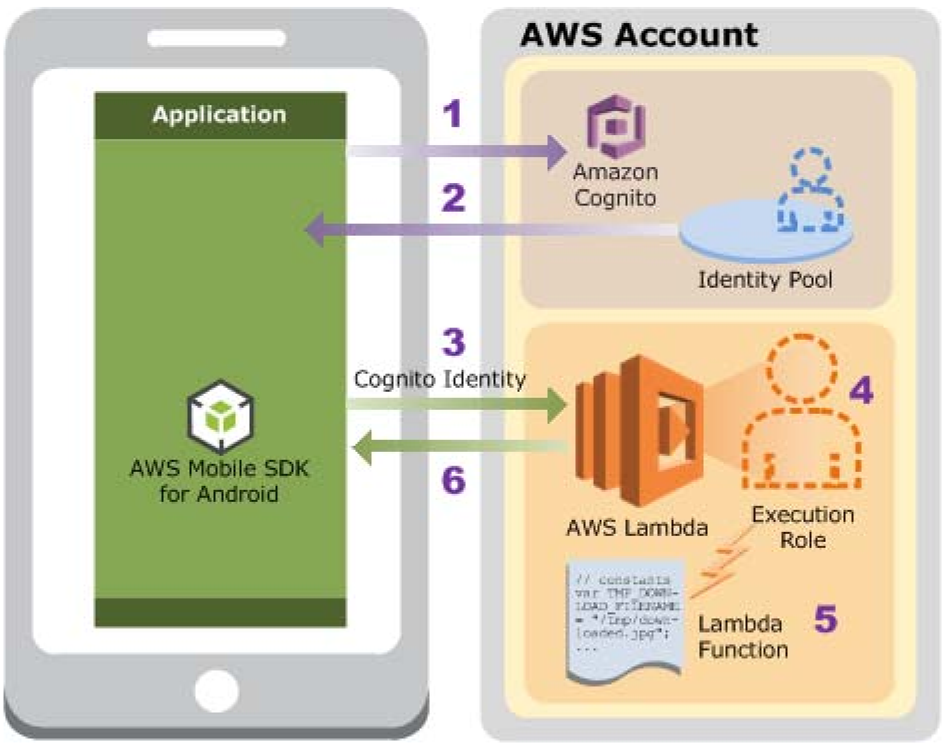
a) Custom app writes records to the stream.

b) AWS Lambda polls the stream and, when it detects new records in the stream, invokes our Lambda function.

c) AWS Lambda executes the Lambda function by assuming the execution role we specified at the time we created the Lambda function.

- Websites
  Suppose we are creating a website and we want to host the backend logic on Lambda. We can invoke our Lambda function over HTTP using Amazon API Gateway as the HTTP endpoint. Now, our web client can invoke the API, and then API Gateway can route the request to Lambda.

- Mobile applications
  Suppose we have a custom mobile application that produces events. We can create a Lambda function to process events published by our custom application. For example, in this scenario we can configure a Lambda function to process the clicks within our custom mobile application.



a) The mobile application sends a request to Amazon Cognito with an identity pool ID in the request (we create the identity pool as part the setup).

b) Amazon Cognito returns temporary security credentials back to the application.

c) Amazon Cognito assumes the role associated with the identity pool to generate temporary credentials. What the application can do using the credentials is limited to the permissions defined in the permissions policy associated with the role Amazon Cognito used in obtaining the temporary credential.

d) AWS Lambda assumes the execution role to execute our Lambda function on our behalf.

e) The Lambda function executes.

f) AWS Lambda returns results to the mobile application, assuming the app invoked the Lambda function using the RequestResponse invocation type (synchronous invocation).