



Search



Home



My Network



Jobs



Messaging



Notifications



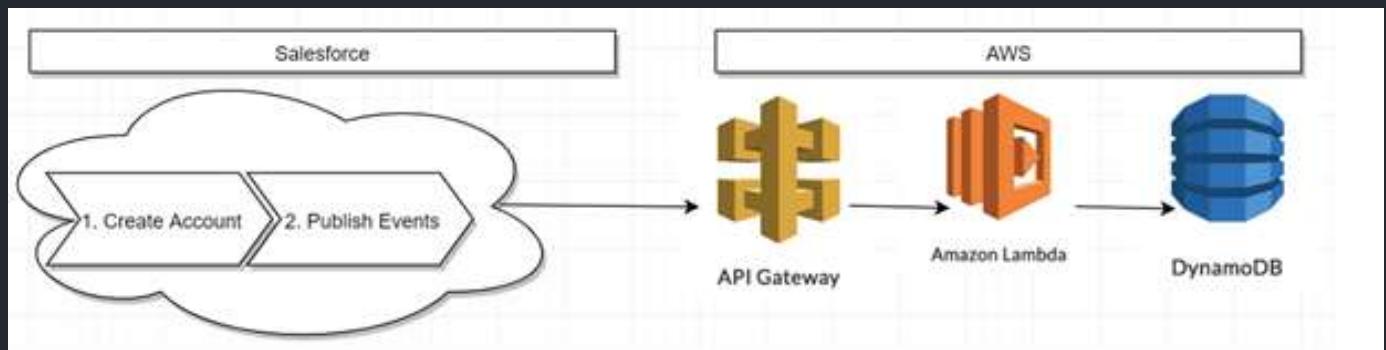
Salesforce AWS Integration using API gateway and Lambda



Rajesh S.

Staff Engineer @ PayPal | Ex-Salesforce | Akamai | Morgan Stanley

Published May 27, 2019

[+ Follow](#)

S

Prerequisite

1. AWS Free Tier Account ,
2. Salesforce Developer Account
3. Basic to advance knowledge of Salesforce Architecture and Coding Pattern.
4. Basic to Intermediate knowledge AWS lambda, Dynamo DB and server less processing.

Consideration

1. All AWS service creation has been kept simple .. i.e most part like AWS lambda has been configured to have admin access role .



Search



Home



My Network



Jobs



Messaging



Notifications

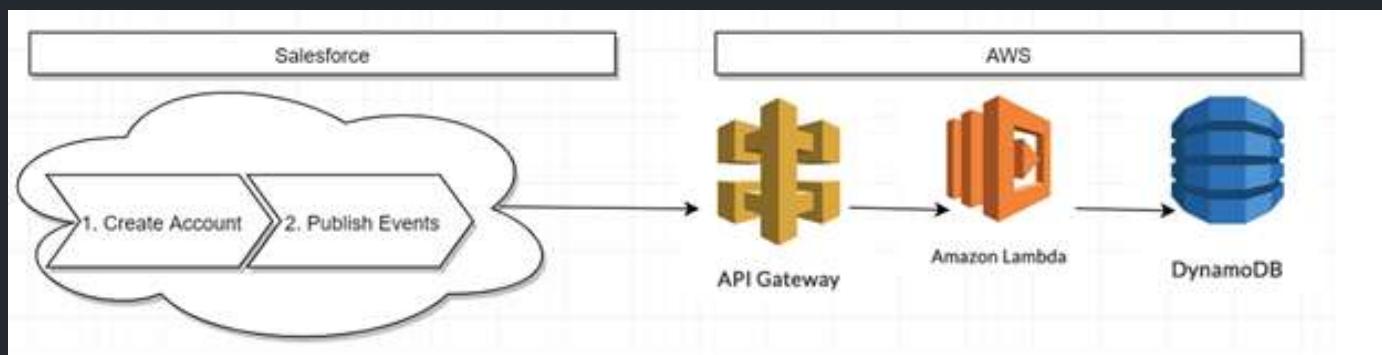


Me ▾

3. I will have separate article for security consideration b/w salesforce and AWS connection

Use Case Flow

1. Account is created in Salesforce
2. Platform Event is subscribed and in turns it calls amazon gateway and pass data.
3. Amazon Gateway receives the data and transform and pass to amazon lambda.
4. Amazon Lambda has logic to pass this data to dynamo DB and insert



Salesforce Changes

- a) Create Platform Even name “**SubscribetoAWSLambda**” (you can name whatever you want)
- b) Create two custom field in platform event “AccountId” & Name in above event created.(Refer below image for more detail)

Standard Fields

Action	Field Label	Field Name	Data Type	Controlling Field	Indexed
Edit Del	Created By	CreatedBy	Lookup(User)		
Edit Del	Created Date	CreatedDate	Date/Time		
Edit Del	ReplyId	ReplyId	External Lookup		

Custom Fields & Relationships

Action	Field Label	API Name	Data Type	Indexed	Controlling Field	Modified By
Edit Del	Account Id	AccountId__c	Text(255)			User User, 5/26/2019 12:11 PM
Edit Del	Name	Name__c	Text(255)			User User, 5/26/2019 12:12 PM

- c) Create process builder to publish data in above vents when account is created ,
- d) Create below class (Please note this class can be optimised for bulk operations and AWS can be read from custom Metadata/Custom Setting) . Please note that endpoint will be replaced when we create API gateway in later section.

```
public class AWSCallout
```

```
{
```

```
@future(callout = true)
```

```
public static void callAwsgateway(String recordId, String Name)
```

```
{
```

```
Http http = new Http();
```

```
HttpRequest request = new HttpRequest();
```

```
//Please set the endpoint of API gateway which we are going to create in later section
```

```
request.setEndpoint('awsapigateway');
```

```
request.setMethod('POST');
```



Search



Home



My Network



Jobs



Messaging



Notifications



```
// Set the body as a JSON object
```

```
request.setBody('{"id": "'+recordId+'","Name": "'+Name+'"}');
```

```
HttpResponse response = http.send(request);
```

```
System.debug('response==='+response.getBody());
```

```
// Parse the JSON response
```

```
if (response.getStatusCode() != 201) {
```

```
    System.debug('The status code returned was not expected: ' +
```

```
        response.getStatusCode() + ' ' + response.getStatus());
```

```
} else {
```

```
    System.debug(response.getBody());
```

```
}
```

```
}
```

```
}
```

e) Create Subscriber on platform Event (basically platform event Trigger)



Search



Home



My Network



Jobs



Messaging



Notifications



Me ▾

for (SubscribetoAWSLambda__e saws : Trigger.new)

{

 AWSCallout.callAwsgateway(saws.AccountId__c,saws.Name__c);

}

}

f) Do not forget to add the actual end point mentioned in Step c) to remote site setting.

AWS Changes

> Create below dynamodb table with "id" as Primary partition key

Filter by table name

Choose a table ... Actions

Name

Account

Create item Actions

Scan: [Table] Account: id

Scan [Table] Account: id Add filter Start search

	id	Name
	000000111	TestData
	0016D00000ANePGQA1	qewwdeaszd
	0016D00000ANePLQA1	DynamoExample
	22222222222222222222222222	RajeshTest
	id001	Name001
	id002	Name002
	id003	Name003
	id009	Name009

> Login to AWS and go to IAM and create custom role with Administrator access policy .I have used this role for AWS lambda , please note in real time situation we should not give admin access security should be limited here I have used just for demo purpose to avoid any access related issue .

Roles > customAdmin

Summary

Role ARN

Role description Allows Lambda functions to call AWS services on your behalf. | Edit

Instance Profile ARNs

Path /

Creation time 1 hour ago

Maximum CLI/API session duration 1 hour Edit

Permissions Trust relationships Tags Access Advisor Revoke sessions

Permissions policies (1 policy applied)

Attach policies

Policy name	Policy type
AdministratorAccess	AWS managed policy



Search



Home



My Network



Jobs



Messaging



Notifications



```
const AWS = require('aws-sdk');
const dynamodb = new AWS.DynamoDB({apiVersion: '2012-08-10'});

exports.handler = (event, context, callback) => {
  dynamodb.putItem({
    TableName: process.env.tableName,
    Item: {
      "id": {
        S: event.id
      },
      "Name": {
        S: event.Name
      }
    },
    }, function(err, data) {
      if (err) {
        console.log(err, err.stack);
        callback(null, {
          statusCode: '500',
          body: err
        });
      } else {
        callback(null, {
          statusCode: '200',
          body: 'Hello ' + event.id + event.Name + '!'
        });
      }
    });
};
```

Below image shows the setup for lambda function

Screenshot of the AWS Lambda function configuration page:

Environment variables

You can define environment variables as key-value pairs that are accessible from your function code. These are useful to store configuration settings which you want to keep separate from your code.

tableName	Account
Key	Value

Encryption configuration

Tags

You can use tags to group and filter your functions. A tag consists of a case-sensitive key-value pair. [Learn more](#)

Key	Value
-----	-------

Execution role

Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).

Use an existing role

Existing role

Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.

customAdmin

[View the customAdmin role on the IAM console.](#)

> Create below API gateway

Screenshot of the AWS API Gateway configuration page:

APIs > inserttoDynamoDB > Resources > /dynamodboperation (gz9ftx) > POST

Method Request (Client to API)

- Auth: NONE
- ARN: arn:aws:execute-api:us-east-1:152442172534:23yg525uh/*/POST/dynamodboperation

Integration Request (API to Lambda)

- Type: LAMBDA
- Region: us-east-1

Method Response (API to Client)

- HTTP Status: 200
- Models: application/json => Empty

Integration Response (Lambda to API)

- HTTP status pattern: -
- Output passthrough: Yes

> Create below model inside the gateway

The screenshot shows the AWS Lambda API Gateway interface. On the left sidebar, under the 'Models' section, there is a 'Create' button. Below it, three models are listed: 'Empty', 'Error', and 'inputMap'. The 'inputMap' model is selected. The main panel is titled 'Update Model' and contains fields for 'Model name' (set to 'inputMap'), 'Content type' (set to 'application/json'), and 'Model description'. A code editor shows the JSON schema for 'inputMap':

```
1  {
2      "type": "object",
3      "properties": {
4          "id": {"type": "string"},
5          "Name": {"type": "string"}
6      },
7      "title": "inputMap"
8 }
```

> Configure post method as below .

The screenshot shows the 'Method Execution' configuration for the POST method of the '/dynamodboperation' endpoint. The 'Actions' dropdown is set to 'POST'. The 'Authorization' setting is 'NONE'. The 'Request Validator' is configured to 'Validate body'. The 'API Key Required' setting is 'false'. The 'Content type' is 'application/json' and the 'Model name' is 'inputMap'. The 'Request Body' section includes an 'Add model' button.

> Deploy API gateway to stage (you can stage name whatever you want) I have named test.



Search



Home



My Network



Jobs



Messaging



Notifications



> Test the functionality by creating account in Salesforce which will flow to dynamo db table created earlier .

Note : Code used here is not optimised do not use for production this is just for demo purpose for production code need to handle bulk scenario without any hard coding.

Report this

Like Comment Share 26 · 8 Comments

Like

Comment

Share



Caterina Murdock-Thorum

2y ...

I'm interested in data exchange between AWS S3 (data lake) and AWS Lambda and Salesforce, but recipient of data, is Salesforce. The data would be queried by agent on need basis on a lookup tab in a Salesforce DB. I found your article helpful though I need the reverse flow.

Like Reply



Siddharth Raj

3y ...

great job Rajesh..... really helpful

Like Reply



Anirban Goswami

3y ...

truly fine stuff

Like Reply



Vivek Mehta

3y ...



Search



Home



My Network



Jobs



Messaging



Notifications



Me ▾

Like Reply

See more comments

More articles by this author



JS,LWC,Node JS

Last Name	Email	ACTION
Kumar	test@test.com	[button]

LWC With
Express,Node,MongoDB...
Jul 5, 2021

Published on LinkedIn

Bulk API in Salesforce using
JAVA Client
Jun 28, 2019

Published on LinkedIn

'Javascript Promises' in
Lightning Web Compon...
Jun 3, 2019

LinkedIn © 2022

[Accessibility](#)

[Privacy Policy](#)

[Copyright Policy](#)

[Community Guidelines](#)

[Settings](#)

[About](#)

[User Agreement](#)

[Cookie Policy](#)

[Brand Policy](#)

[Help Center](#)

[Language ▾](#)