

•

•

•

Search

Courses

Projects

Posts

Topics

Questions

Code Blocks

Start Here

Posts / Prepare AWS IAM User, Role, and Policies for Zappa and Serverless Python

aws

Lambda x Zappa

Setup IAM User & Roles & Policies

# Prepare AWS IAM User, Role, and Policies for Zappa and Serverless Python

November 9, 2018 · [Justin Mitchel](#)

This is [Zappa](#) specific guide. Zappa is amazing for creating a serverless architecture. We'll be exploring *much* more about Zappa in the future so be sure to check back soon.

Zappa in AWS needs all of these permissions to function correctly. If you know AWS well and how to limit your resources, you can do so but what you see below is the bare minimum that zappa needs to create.

If all fails, open a new AWS account (seriously), give the IAM User Full Programmic Admin previliges, use `awscli` to login that user and create/deploy a Zappa project. It should work without issues. If these permissions are given, Zappa will automotically create some of the items you see below. The reason this guide exists is to **limit** what the AWS IAM User (or User Group) can do as a security measure.

Oh and I promise you, you'll hit snags along the way. Please comment below so this guide can help others.

## Requirements

- Create an AWS account, if you don't have one

## 1. Create AWS Policy for AWS Role

- Go to [IAM](#)
- Click "Policies"
- Click "Create policy"
- Click "JSON", add in:

Your email

Get Started for Free >>

Trending

[Install Python & Django on Windows](#)

[Install Python 3.6, Virtualenv, & Django on Mac](#)

[Install Tensorflow GPU on Windows using CUDA and cuDNN](#)

[Install OpenCV 3 for Python on Windows](#)

[How to Create a Custom Django User Model](#)

Featured

[Deploying Django on Docker to Heroku with OpenCV](#)

[Django on Docker - A Simple Introduction](#)

[Simple Docker Container Tutorial](#)

[Build a Python Jupyter Notebook Server with Docker & Heroku](#)

[Django Tutorial - ManyToManyField via a Comma Separated String in Admin & Forms](#)

Discover

[Install Python & Django on Windows](#)

https://www.codingforentrepreneurs.com/blog/aws-iam-user-role-policies-zappa-serverless-python

1/7

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Action": [

"logs:\*"

],

"Resource": "arn:aws:logs:\*:\*:\*"

},

{

"Effect": "Allow",

"Action": [

"lambda:GetFunctionConfiguration",

"lambda:UpdateFunctionConfiguration",

"lambda:InvokeFunction"

],

"Resource": [

"\*"

]

},

{

"Effect": "Allow",

"Action": [

"xray:PutTraceSegments",

"xray:PutTelemetryRecords"

],

"Resource": [

"\*"

]

},

{

"Effect": "Allow",

"Action": [

"ec2:AttachNetworkInterface",

"ec2:CreateNetworkInterface",

"ec2>DeleteNetworkInterface",

"ec2:DescribeInstances",

"ec2:DescribeSecurityGroups",

"ec2:DescribeNetworkInterfaces",

"ec2:DetachNetworkInterface",

"ec2:ModifyNetworkInterfaceAttribute",

"ec2:ResetNetworkInterfaceAttribute"

],

"Resource": "\*"

},

{

"Effect": "Allow",

"Action": [

"s3:\*"

],

"Resource": "arn:aws:s3:\*:\*:\*"

},

{

"Effect": "Allow",

"Action": [

"kinesis:\*"

],

"Resource": "arn:aws:kinesis:\*:\*:\*"

},

{

"Effect": "Allow",

"Action": [

"sns:\*"

],

"Resource": "arn:aws:sns:\*:\*:\*"

},

{

"Effect": "Allow",

"Action": [

"sqs:\*"

]

},

]

}

Copy

[Common Regular Expressions for Django URLs](#)

[Create a Timelapse with Python & OpenCV](#)

[REST API Basics with the Django REST Framework](#)

[Create a Blank Django Project](#)

https://www.codingforentrepreneurs.com/blog/aws-iam-user-role-policies-zappa-serverless-python

2/7

```
    ],
    "Resource": "arn:aws:sqs:*:*:*"
  },
  {
    "Effect": "Allow",
    "Action": [
      "dynamodb:*"
    ],
    "Resource": "arn:aws:dynamodb:*:*:*"
  },
  {
    "Effect": "Allow",
    "Action": [
      "route53:*"
    ],
    "Resource": "*"
  }
]
```

- 5. Name the policy something like: **ZappaRolePolicy**
- 6. Select **Create Policy**

## 2. Create AWS IAM ROLE

- 1. Go to [IAM](#)
- 2. Click "Roles"
- 3. Click "Create Role"
- 4. For "Select Type of trusted entity" select "AWS service"
- 5. For "Choose the service that will use this role" select "Lambda"
- 6. Click "Next: Permissions"
- 7. Find the newly created policy from above **ZappaRolePolicy** and select it
- 8. Click **Next: Review**
- 9. Give a role name such as **ZappaDjangoRole**
- 10. Add/change **Role description** as needed.
- 11. Click **Create role**
- 12. Click on your newly created role (such as **ZappaDjangoRole**) in [IAM roles](#)
- 13. Click **Trust Relationships**
- 14. Click **Edit trust relationship** and add the following:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "lambda.amazonaws.com",
          "apigateway.amazonaws.com",
          "events.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

- 15. Click Update Trust Policy
- 16. Copy the **Role ARN** such as **arn:aws:iam::908424941159:role/ZappaDjangoRole**

## 3. Group-Level Permissions Policy for Lambda Role and Zappa Related Events

- 1. Go to [IAM](#)
- 2. Click "Policies"

3. Click "Create policy"
4. Click "JSON", add in:

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Action": [

"iam:GetRole",

"iam:PutRolePolicy"

],

"Resource": [

"\*"

]

},

{

"Effect": "Allow",

"Action": [

"iam:PassRole"

],

"Resource": [

"arn:aws:iam::[ROLE\_ID]:role/[ROLENAME]"

]

},

{

"Effect": "Allow",

"Action": [

"apigateway:DELETE",

"apigateway:GET",

"apigateway:PATCH",

"apigateway:POST",

"apigateway:PUT",

"cloudformation:CreateStack",

"cloudformation>DeleteStack",

"cloudformation:DescribeStackResource",

"cloudformation:DescribeStacks",

"cloudformation>ListStackResources",

"cloudformation:UpdateStack",

"events:DeleteRule",

"events:DescribeRule",

"events>ListRules",

"events>ListRuleNamesByTarget",

"events>ListTargetsByRule",

"events:PutRule",

"events:PutTargets",

"events:RemoveTargets",

"lambda:\*",

"lambda:AddPermission",

"lambda:CreateFunction",

"lambda>DeleteFunction",

"lambda:GetFunction",

"lambda:GetFunctionConfiguration",

"lambda>ListVersionsByFunction",

"lambda:UpdateFunctionCode",

"logs:DescribeLogStreams",

"logs:FilterLogEvents",

"route53>ListHostedZones",

"route53:ChangeResourceRecordSets",

"route53:GetHostedZone"

],

"Resource": [

"\*"

]

}

]

}

Copy

5. Replace `arn:aws:iam::[ROLE_ID]:role/[ROLENAME]` with your above Role ARN

6. Add the name, such as `ZappaUserGeneralPolicy`

7. Click `Create policy`

https://www.codingforentrepreneurs.com/blog/aws-iam-user-role-policies-zappa-serverless-python

4/7

If unsuccessful, check your json above.

## 4. Create an S3 Bucket

This is where zappa will upload your code deployments (aka the zipped folder of your code) for Lambda to use.

1. Go to [S3](#)
2. Click + **Create Bucket**
3. Add **Bucket name** mine was **my-awesome-zappa-bucket** (yours will have to be different due to aws thankfully making this unique enforced)
4. Choose region, I used **us-west-2** (*US West (oregon)*)

Be sure to use the region you want to use. I've tested **us-west-2** (*US West (oregon)*) and **us-east-1**(*US East (N. Virginia)*) and both work well. Make note of the url-encoded name (like **us-west-2**. Often, you'll find this in the **region=** in your url. The region, at this point, is important to note. You can use the table on [this page](#) as reference if you can't find the Region's name.

5. Keep all defaults (so just click next until it's created.)
6. Be sure to remember your bucket name for the next step.

## 5. Group-Level Permissions Policy for S3 for Zappa

1. Go to [IAM](#)
2. Click "Policies"
3. Click **Create policy**
4. Click "Create policy"
5. Click "JSON", add in:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::[YOUR_BUCKET_NAME]"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:DeleteObject",
        "s3:GetObject",
        "s3:PutObject",
        "s3:CreateMultipartUpload",
        "s3:AbortMultipartUpload",
        "s3:ListMultipartUploadParts",
        "s3:ListBucketMultipartUploads"
      ],
      "Resource": [
        "arn:aws:s3:::[YOUR_BUCKET_NAME]/*"
      ]
    }
  ]
}
```

Copy

6. Replace **[YOUR\_BUCKET\_NAME]** with the name you choose for your bucket in the previous step (mine was **my-awesome-zappa-bucket**).
7. Add your policy name, such as **ZappaUserS3Policy**



## 6. Create AWS IAM Group

1. Go to [IAM](#)
2. Click "Groups"
3. Click "Create New Group"
4. Set a name for your group, I used **ZappaGroup**
5. Find your policy **ZappaUserS3Policy** and **ZappaUserGeneralPolicy**

**do not** add the **ZappaRolePolicy**

Creating a group is optional. You can just add these polices to the user only (below) if you prefer. I tend to want to work with groups that way I can add/remove users if needed. This is standard practice because then I can worry more about managing Groups and their policies than individual users. This becomes espeically important if your team is going to grow (or shrink).

## 7. Create AWS IAM User.

1. Go to [IAM](#)
2. Click "Users"
3. Click "Add user"
4. Add no policies to the user directly, the group has them.
5. Set a user name, I used **ZappaUser**
6. Check access type: **Programmatic access**
7. Click "Next: Permissions"
8. Add user to the **ZappaGroup** created above.
9. Click "Next: Review"
10. Click "Create user"
11. Make note of your **Access key id** and **Secret access key** and ensure it's safely kept. We'll be using it in the next guide.

## 8. Grab IAM User ARN

1. Click on 'Users'
2. Click **ZappaUser** (or your previously created user)
3. Grab the **User ARN**, something like **arn:aws:iam::123342844315:user/ZappaUser**, we'll be using it in the next guide.

## 9. Proceed to the [Django with Zappa on AWS Serverless](#) guide.

# Comments

Write your comment/question

 Comment



[wechatsmallapp.anlian](#) 6 months ago [Thread](#)  
"s3:CreateMultipartUpload" in "Group-Level Permissions Policy for S3" may not need, with that I got AccessDenied error when do "zappa deploy"




[jmitchel3](#) 6 months ago  
So without it it works?

M

[michael4338](#) 6 months ago

yea



[jmitchel3](#) 6 months ago

Interesting.

S

[shadowRa88it](#) 5 months, 1 week ago

I get access denied for any s3 actions I try to do.

N

[nk.shukla2k18](#) 2 months, 1 week ago

Hi justin I am getting below error when deploying

botocore.exceptions.ClientError: An error occurred (InvalidSignatureException) when calling the CreateFunction operation: Signature expired: 20191104T121023Z is now earlier than 20191104T121521Z (20191104T122021Z - 5 min.)

Write your reply

Reply

