

Contents

IAM > Roles > LambdaAccessRole > Permissions.....	1
AWSLambdaBasicExecutionRole	1
AmazonSNSFullAccess.....	1
AmazonS3FullAccess	2
CloudWatchFullAccess	2
Config policy in role.....	4
AWS access levels.....	5
Source code: ReadS3Object.py	6

IAM > Roles > LambdaAccessRole > Permissions

AWSLambdaBasicExecutionRole

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Resource": "*"
    }
  ]
}
```

AmazonSNSFullAccess

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```

        "Action": [
            "sns:*"
        ],
        "Effect": "Allow",
        "Resource": "*"
    }
]
}

```

AmazonS3FullAccess

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "s3-object-lambda:*"
            ],
            "Resource": "*"
        }
    ]
}

```

CloudWatchFullAccess

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "autoscaling:Describe*",
                "cloudwatch:*",
                "logs:*",
                "sns:*",
                "iam:GetPolicy",

```

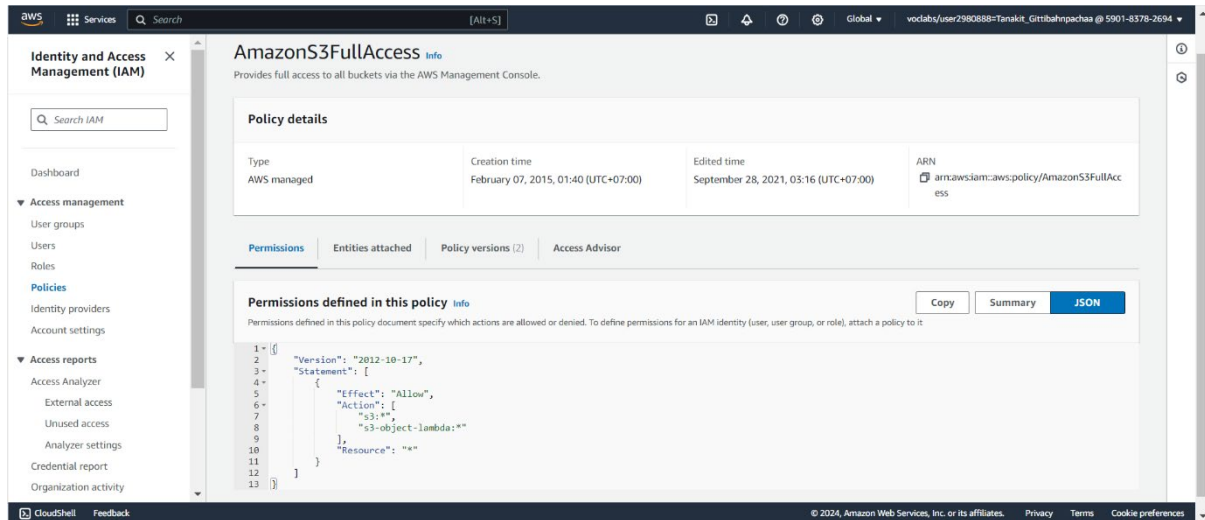
```

        "iam:GetPolicyVersion",
        "iam:GetRole",
        "oam:ListSinks"
    ],
    "Resource": "*"
},
{
    "Effect": "Allow",
    "Action": "iam:CreateServiceLinkedRole",
    "Resource": "arn:aws:iam::*:role/aws-service-
role/events.amazonaws.com/AWSServiceRoleForCloudWatchEvents*",
    "Condition": {
        "StringLike": {
            "iam:AWSServiceName": "events.amazonaws.com"
        }
    }
},
{
    "Effect": "Allow",
    "Action": [
        "oam:ListAttachedLinks"
    ],
    "Resource": "arn:aws:oam::*:sink/*"
}
]
}

```

Config policy in role

ไม่สามารถตั้งค่า Action ที่จะกระทำต่อ Resources ได้ เนื่องจาก IAM Policies เป็นประเภท AWS managed



AmazonS3FullAccess Info

Provides full access to all buckets via the AWS Management Console.

Policy details

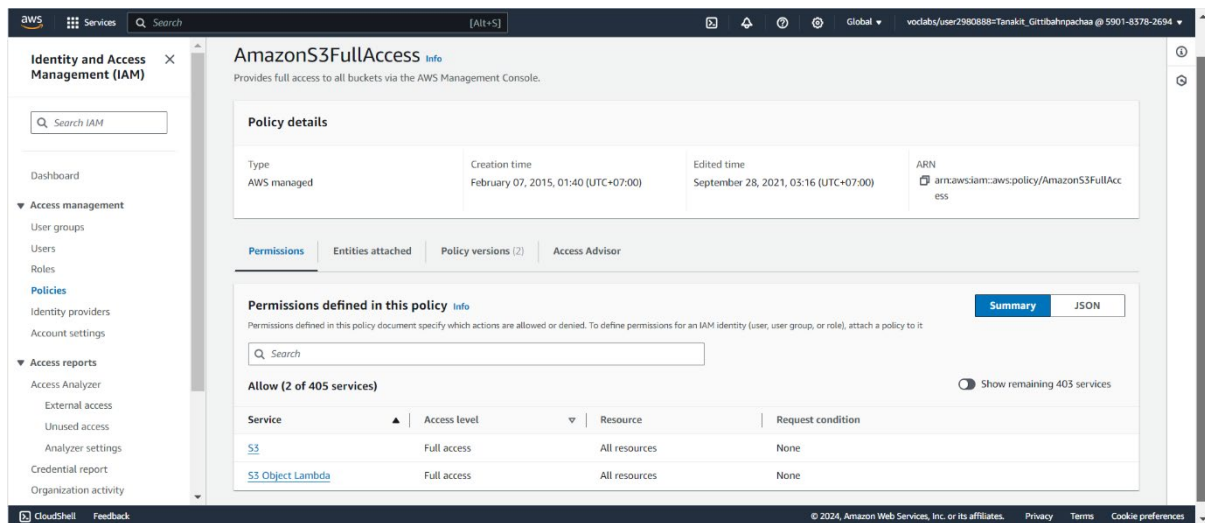
Type	Creation time	Edited time	ARN
AWS managed	February 07, 2015, 01:40 (UTC+07:00)	September 28, 2021, 03:16 (UTC+07:00)	arn:aws:iam::aws:policy/AmazonS3FullAccess

Permissions | Entities attached | Policy versions (2) | Access Advisor

Permissions defined in this policy Info

Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it.

```
1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Allow",
6       "Action": [
7         "s3:*",
8         "s3-object-lambda:*"
9       ],
10      "Resource": "*"
11    }
12  ]
13 }
```



AmazonS3FullAccess Info

Provides full access to all buckets via the AWS Management Console.

Policy details

Type	Creation time	Edited time	ARN
AWS managed	February 07, 2015, 01:40 (UTC+07:00)	September 28, 2021, 03:16 (UTC+07:00)	arn:aws:iam::aws:policy/AmazonS3FullAccess

Permissions | Entities attached | Policy versions (2) | Access Advisor

Permissions defined in this policy Info

Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it.

Search

Allow (2 of 405 services) ☐ Show remaining 403 services

Service	Access level	Resource	Request condition
S3	Full access	All resources	None
S3 Object Lambda	Full access	All resources	None

AWS access levels

AWS ได้กำหนดระดับการเข้าถึงสำหรับการดำเนินการในบริการต่างๆ ดังนี้:

List: สิทธิในการดูรายการทรัพยากรภายในบริการเพื่อตรวจสอบว่าออบเจกต์นั้นมีอยู่หรือไม่ การดำเนินการในระดับนี้สามารถดูรายการออบเจกต์ได้ แต่จะไม่สามารถดูเนื้อหาของทรัพยากรนั้น เช่น คำสั่ง ListBucket ใน Amazon S3 มีระดับการเข้าถึง คือ List

Read: สิทธิในการอ่านแต่ไม่สามารถแก้ไขเนื้อหาและแอตทริบิวต์ของทรัพยากรในบริการนั้นได้ เช่น คำสั่ง GetObject และ GetBucketLocation ใน Amazon S3 มีระดับการเข้าถึง คือ Read

Tagging: สิทธิในการดำเนินการที่เปลี่ยนแปลงสถานะของแท็กทรัพยากรเท่านั้น เช่น คำสั่ง TagRole และ UntagRole ใน IAM มีระดับการเข้าถึง คือ Tagging เนื่องจากอนุญาตให้ทำแท็กหรือถอดแท็กบทบาทได้เท่านั้น อย่างไรก็ตาม คำสั่ง CreateRole อนุญาตให้ทำแท็กบทบาทได้ในขณะสร้างบทบาทนั้น เนื่องจากคำสั่งนี้ไม่ได้เพียงแค่เพิ่มแท็ก จึงมีระดับการเข้าถึง คือ Write

Write: สิทธิในการสร้าง ลบ หรือแก้ไขทรัพยากรในบริการนั้น เช่น คำสั่ง CreateBucket, DeleteBucket และ PutObject ใน Amazon S3 มีระดับการเข้าถึง คือ Write คำสั่งระดับ Write อาจอนุญาตให้แก้ไขแท็กทรัพยากรได้ด้วย อย่างไรก็ตาม คำสั่งที่อนุญาตให้เปลี่ยนแปลงเฉพาะแท็กเท่านั้นจะมีระดับการเข้าถึง คือ Tagging

Permissions management: สิทธิในการให้หรือแก้ไขสิทธิ์การเข้าถึงทรัพยากรในบริการนั้น เช่น ส่วนใหญ่ของคำสั่ง IAM และ AWS Organizations รวมถึงคำสั่งอย่าง PutBucketPolicy และ DeleteBucketPolicy ใน Amazon S3 ล้วนมีระดับการเข้าถึง คือ Permissions management

Source code: ReadS3Object.py

```
import boto3

def lambda_handler(event, context):
    # Get the S3 bucket and file key from the event
    bucket = event['Records'][0]['s3']['bucket']['name']
    key = event['Records'][0]['s3']['object']['key']

    # Download the file from S3
    s3 = boto3.client('s3')
    response = s3.get_object(Bucket=bucket, Key=key)
    text = response['Body'].read()

    # Count the number of words
    word_count = len(text.split())

    # Create an SNS client and publish the word count
    sns = boto3.client('sns')
    sns.publish(
        TopicArn='YOUR_SNS_TOPIC_ARN',
        Message=f'The word count for in the {key} is {word_count}.',
        Subject='Word Count Result'
    )
    return {
        'statusCode': 200,
        'body': f'Word count: {word_count}'
    }
```

`import boto3` โหลดไลบรารี boto3 เพื่อใช้งาน AWS SDK สำหรับ Python ซึ่งจะช่วยในการเชื่อมต่อและจัดการกับบริการต่างๆ ของ AWS ได้ เช่น S3 และ SNS

`def lambda_handler(event, context):` เป็นฟังก์ชันหลักของ Lambda ถูกเรียกใช้เมื่อมีการใช้งาน Lambda function โดยมีพารามิเตอร์สองตัวคือ event เป็น dictionary ที่บรรจุข้อมูล event ที่ trigger ให้ฟังก์ชันทำงาน ในที่นี้คือ ฟังก์ชันทำงานเมื่อมีการอัปโหลดไฟล์ไปยัง S3 bucket และ context เป็น object ที่ให้ข้อมูลเกี่ยวกับ Lambda execution environment

`bucket = event['Records'][0]['s3']['bucket']['name']` นำชื่อของ bucket ที่เกี่ยวข้อง กับ event ที่ส่งเข้ามาจาก S3 มาเก็บไว้ในตัวแปร bucket

`key = event['Records'][0]['s3']['object']['key']` นำ key (path) ของไฟล์ใน bucket ที่เกี่ยวข้องกับ event ที่ส่งเข้ามาจาก S3 มาเก็บไว้ในตัวแปร key

เช่น ถ้าเก็บไฟล์ "image.jpg" ในโฟลเดอร์ "photos" ใน bucket "my-bucket" จะมี key (path) เป็น "photos/image.jpg"

`s3 = boto3.client('s3')` สร้าง client object สำหรับใช้งาน S3

client object เป็น object ที่ใช้ในการติดต่อสื่อสารกับบริการอื่นผ่าน API

`response = s3.get_object(Bucket=bucket, Key=key)` ดึงข้อมูลของไฟล์จาก S3 โดยใช้ bucket และ key ที่ได้จากขั้นตอนก่อนหน้านี้

`text = response['Body'].read()` อ่านข้อมูลส่วน Body (เฉพาะเนื้อหา) ของไฟล์ที่ดึงมาจาก S3

`word_count = len(text.split())` นับจำนวนคำในข้อความที่อ่านได้จากไฟล์โดยใช้ `split()` เพื่อแยกคำโดยการเว้นวรรคแล้วนับจำนวนคำทั้งหมด

`sns = boto3.client('sns')` สร้าง client object สำหรับใช้งาน SNS

`sns.publish(TopicArn='YOUR_SNS_TOPIC_ARN', Message=f'The word count for in the {key} is {word_count}.', Subject='Word Count Result')` ส่งข้อความไปยัง SNS topic ที่ระบุไว้ โดยข้อความจะระบุจำนวนคำที่นับได้และชื่อของไฟล์ที่ถูกลบ (SNS topic จะส่งต่อข้อความไปยัง subscribers ต่อไป)

`return {'statusCode': 200, 'body': f'Word count: {word_count}'}'` ส่ง response กลับไปที่ caller โดยระบุว่าการทำงานเสร็จสมบูรณ์และระบุจำนวนคำที่นับได้ในรูปแบบของ JSON object ซึ่งมี statusCode เป็น 200 และ body เป็นข้อความที่ระบุจำนวนคำที่นับได้

caller ในที่นี้คือ Lambda service เมื่อ Lambda function ทำงานเสร็จ จะส่งผลลัพธ์กลับไปยัง Lambda service

