AWS CLI, SDK, IAM Roles & Policies

Notes by Christina Grys

Amazon Web Services, Command Line Interface, Software Development Kit, Identity and Access Management

- **Review this section** for further coding credentials and authentications when performing in AWS console.

AWS EC2 Instance Metadata

- AWS EC2 Instance Metadata is powerful but one of the least known features to developers
- It allows AWS EC2 instances to "learn about themselves" without using an IAM Role for that purpose.
- The URL is http://169.254.169.254/latest/meta-data
- You can retrieve the IAM Role name from the metadata, but you CANNOT retrieve the IAM policy.
- Metadata = Info about the EC2 instance
- Userdata = launch script of the EC2 instance

MFA with CLI

- To use MFA with the CLI, you must create a temporary session
- To do so, you must run the STS GetSessionToken API call

- Aws sts get-session-token –serial-number arn-of-the-mfa-device –token-code code-from-token –duration-seconds 3600

```
{
    "Credentials": {
        "SecretAccessKey": "secret-access-key",
        "SessionToken": "temporary-session-token",
        "Expiration": "expiration-date-time",
        "AccessKeyId": "access-key-id"
    }
}
```

AWS SDK Overview

- What if you want to perform actions on AWS directly from your applications code? (Without using the CLI)
- You can use an SDK (software development kit)
- Official SDKs are:
- Java, .NET, Node.js, PHP, Python (named boto3/botocore), Go, Ruby, C++

- We have to use the AWS SDK when coding against AWS services such as DynamoDB

- Fun fact: AWS CLI uses the Python SDK (boto3)
- Exam expects you to know when you should use an SDK
- Good to know – if you don't specify or configure a default region, then us-east-I will be chosen by default

AWS Limits (Quotas)

API Rate Limits

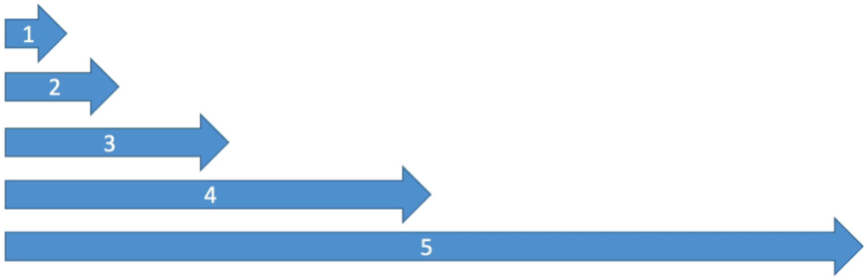
- **DescribeInstances** API for EC2 has limit of 100 calls per seconds
- **GetObject** on S3 has a limit of 5500 GET per second per prefix
- For Intermittent Errors: implement Exponential Backoff
- For Consistent Errors: request an API throttling limit increase

Service Quotas (Service Limits)

- Running On-Demand Standard Instances: I I 52 vCPU
- You can request a service limit increase by opening a ticket
- You can request a service quota increase by using the Service Quotas API

Exponential Backoff (any AWS service)

- If you get **ThrottlingException** intermittently, use exponential backoff
- Retry mechanism already included in AWS SDK API calls
- Must implement yourself if using the AWS API as-is or in specific cases
    - **Must only implement the retries on 5xx server errors and throttling**
    - Do not implement on the 4xx client errors



AWS CLI Credentials Provider Chain

- The CLI will look for credentials **IN THIS ORDER:**

1. **Command Line Options**: --region, --output, and –profile
2. **Environment variables**: AWS_ACCESS_KEY_ID,AWS_SECRET_ACCESS_KEY, and AWS_SESSION_TOKEN
3. **CLI credentials file**: aws configure

~/.aws/credentials on Linux / Mac & C:\Users\user\.aws\credentials on Windows

4. **CLI configuration file**: - aws configure
   ~/.aws/config on Linux / maxOS & C:\Users\USERNAME\.aws\config on Windows

5. **Container credentials**: for ECS tasks
6. **Instance profile credentials**: for EC2 Instance Profile

AWS SDK Default Credentials Provider Chain

The Java SDK (example) will look for credentials **in this order:**

1. **Java system properties:** aws.accessKeyId and aws.secretKey
2. **Environment variables**: AWS_ACCESS_KEY_ID and AWS_SECRET_ACCESS_KEY
3. **The default credential profile file**: ex at: ~/.aws/credentials, shared by many SDK
4. **Amazon ECS container credentials**: for ECS containers
5. **Instance profile credentials**: used on EC2 instances

AWS Credentials Scenario

- An application deployed on an EC2 instance is using environment variable with credentials from an IAM user to call the Amazon S3 API.
- The IAM user has S3FullAccess permissions
- The application only uses one S3 bucket, so according to be best practices:
  - An IAM Role & EC2 Instance Profile was created for the EC2 instance
  - The Role was assigned the minimum permissions to access that one S3 bucket

The IAM Instance Profile was assigned to the EC2 instance, but it still had access to all S3 buckets. Why?
→ The credentials chain is still giving priorities to the environment variables

AWS Credentials Best Practices

** NEVER STORE AWS CREDENTIALS IN YOUR CODE**
Best practice is for credentials to be inherited from the credentials chain

If using working within AWS, use IAM Roles

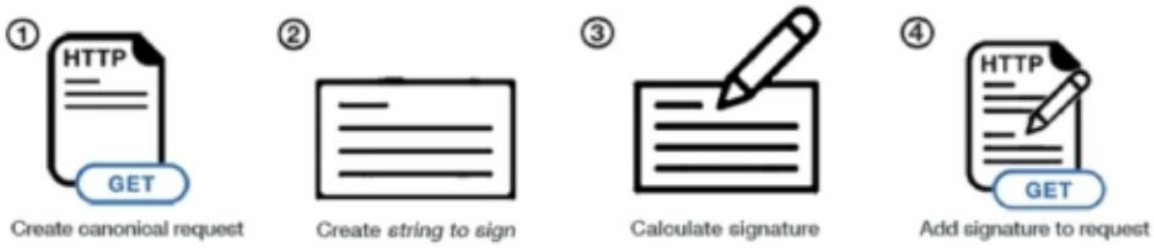
- ⇨ EC2 Instances Roles for EC2 Instances
- ⇨ ECS Roles for ECS tasks
- ⇨ Lambda Roles for Lambda functions

- If working outside of AWS, use environment variables / named profiles

Signing AWS API requests

- When you call the AWS HTTP API, you sign the request so that AWS can identify you, using your AWS credentials (access key & secret key)
- Note: some requests to Amazon S3 don't need to be signed
- If you use the SDK or CLI, the HTTP requests are signed for you

- You should sign an AWS HTTP request using Signature v4 (SigV4)



SigV4 Request examples

- HTTP Header Option

```
GET https://iam.amazonaws.com/?Action=ListUsers&Version=2010-05-08 HTTP/1.1
Authorization: AWS4-HMAC-SHA256 Credential=AKIDEXAMPLE/20150830/us-east-1/iam/aws4_request,
SignedHeaders=content-type;host;x-amz-date,
Signature=5d672d79c15b13162d9279b0855cfba6789a8edb4c82c400e06b5924a6f2b5d7
content-type: application/x-www-form-urlencoded; charset=utf-8
host: iam.amazonaws.com
x-amz-date: 20150830T123600Z
```

- Query String option (ex: S3 pre-signed URLs)

```
GET https://iam.amazonaws.com?Action=ListUsers&Version=2010-05-08&
X-Amz-Algorithm=AWS4-HMAC-SHA256&
X-Amz-Credential=AKIDEXAMPLE%2F20150830%2Fus-east-1%2Fiam%2Faws4_request&
X-Amz-Date=20150830T123600Z&X-Amz-Expires=60&X-Amz-SignedHeaders=content-type%3Bhost&
X-Amz-Signature=37ac2f4fde00b0ac9bd9eadeb459b1bbee224158d66e7ae5fcadb70b2d181d02 HTTP/1.1
content-type: application/x-www-form-urlencoded; charset=utf-8
host: iam.amazonaws.com
```